

**INTELLIGENT HEALTHCARE DATA ANALYTICS COUPLED WITH  
SENSOR ASSESSMENT FOR NON-ALCOHOLIC FATTY LIVER  
DISEASE (NAFLD)**

by  
**Ridhi Deo**

**A Dissertation**

*Submitted to the Faculty of Purdue University  
In Partial Fulfillment of the Requirements for the degree of*

**Doctor of Philosophy**



School of Engineering Technology  
West Lafayette, Indiana  
May 2022

**THE PURDUE UNIVERSITY GRADUATE SCHOOL**  
**STATEMENT OF COMMITTEE APPROVAL**

**Dr. Suranjan Panigrahi, Chair**

School of Engineering Technology

**Dr. Edward Liechty**

Professor Emeritus of Pediatrics

Indiana University School of Medicine

**Dr. Jennifer Freeman**

School of Health Sciences

**Dr. Frederick C. Berry**

School of Engineering Technology

**Approved by:**

Dr. Kathryne A. Newton

*Dedicated to my Grandpa – Kishore Palsikar. I cannot fully express how much your encouragement and support mean to me. You fostered my love for reading – and I am forever grateful to you for this gift. Thank you for always seeing the best in me, I could not have done this without you.*

## ACKNOWLEDGMENTS

Through the process of researching and writing this dissertation, I was lucky to receive help from several individuals. With great joy and respect, I would like to acknowledge these individuals for taking time, effort and patience with helping me get here.

First, I want to thank Dr. Suranjan Panigrahi, my advisor and mentor, who spent many hours of his time in ensuring I was headed in the right direction before I started my PhD journey, through my Ph.D., and in choosing my next steps. I am grateful for your consistent support, direction, advice and overall mentorship. I could not have done this without your support and encouragement. Thank you for all your help – I am forever grateful.

Thank you to Dr. Edward Liechty, for taking time out from his research and practice, even during Covid -19. The perspective and feedback you provided helped us frame our work to be useful in the healthcare domain. I would also like to thank Dr. Jennifer Freeman, not only for her feedback as a committee member, but also for allowing us to share her lab space and resources. I received a great education and training in a very short time through yourself, and your graduate students – thank you. Thank you to Dr. Frederick Berry, for his support and feedback. Some of the questions you asked made me think more clearly through my work and I thank you very much for that. Thank you to Dr. Arlene Rothwell and Dr. Anusha Hettiyadura, from the Mass Spectrometry lab for training me patiently. Your kind help was timely and invaluable.

A huge thank you to Dr. Patrick Connolly and Prof. Abrar Hammoud for supporting me by providing research and teaching assistantships all through my Ph.D. journey – I am grateful for your kindness, and I feel lucky to have had an opportunity to work with you both.

I would like to thank my lab-mates, Jiexiong Xu and Jonah Yap for helping me with lab experiments, providing valuable insights from your own experiments and for being ready to chat about any bottle-necks. I appreciate the time we spent discussing and brainstorming together. To my friends– Trevor Mamer, Matthew Scott, Katie Leyba, Ananya Ipsita, and Manav Wadhawan – thank you for talking through the many ups and downs of person and professional lives. I am grateful for our friendship.

Other than my professors and colleagues, my family in India and my family in the USA have played a huge role as my support system. It is with utmost gratitude and love that I express my thanks to all of them. To my parents – although we live miles apart, your support means the

world to me. To my mom – the time we spent on the phone always made me feel like home. Your kind words of support and affection kept me going, Ma. Thank you! To my dad – you are my inspiration, and you will always be the best Dr. Deo. To my little brother, Rishabh, – you are the greatest blessing in my life. To my uncle, Aniketh Ramname – your support all through my graduate school has kept me going. Thank you for putting your faith in me and helping me see through this. I could not have done this without any of you.

Finally, I would like to thank my partner, Karthik Sethuraman, for always being on my side through the ups and downs of graduate school. You are my biggest pillar of support and I consider myself incredibly lucky to be with you. Thank you for taking time out to read my work, for meaningfully critiquing it (sometimes critique is hard to take but it was helpful every single time!) and for always looking out for the best for me. Needless to say, this dissertation would not have been possible without your support.

## TABLE OF CONTENTS

LIST OF TABLES .....	9
LIST OF FIGURES .....	11
GLOSSARY .....	13
LIST OF ABBREVIATIONS.....	14
ABSTRACT.....	15
CHAPTER 1. GENERAL LITERATURE REVIEW .....	21
1.1 Artificial Intelligence: benefits and challenges for healthcare .....	21
1.1.1 Challenges in processing big data.....	22
1.1.2 Challenges specific to healthcare datasets and the need to address them.....	22
1.1.3 Regulation of healthcare AI:.....	23
1.1.4 Interpretability of black-box models .....	24
1.2 Liver and toxicology .....	25
1.2.1 Liver functionality and importance .....	25
1.3 Heavy metals.....	26
1.3.1 Heavy metal contamination and pathways into the human system .....	27
1.3.2 Lead: .....	27
1.3.3 Arsenic.....	29
1.4 Heavy metals and NAFLD.....	32
CHAPTER 2. HEPATIC STEATOSIS (HS) PREDICTION USING MACHINE LEARNING (PAPER 1) .....	34
2.1 Introduction.....	34
2.2 Literature review .....	35
2.2.1 NAFLD background and epidemiology .....	35
2.2.2 Etiology of NAFLD.....	36
2.2.3 Biomarkers and tools for NAFLD detection .....	38
2.2.4 Machine learning (ML)-based NAFLD detection .....	39
2.3 Objectives .....	41
2.4 Methods.....	41
2.4.1 Objective 1A - methods .....	41

2.5	Results and discussion .....	54
2.5.1	Objective 1A – results and discussion .....	55
2.5.2	Objective 1B - Results and Discussion.....	55
2.6	Summary & conclusion.....	58
2.7	Recommendations for future work .....	59
2.8	Figures – objective 1A .....	60
2.9	Tables – objective 1A .....	62
2.10	Figures – objective 1B.....	63
2.11	Tables - objective 1B.....	65
2.12	References .....	67
APPENDIX A. – CODE FOR OBJECTIVE 1A .....		74
APPENDIX B. - CODE FOR OBJECTIVE 1B .....		94
CHAPTER 3. EXPLAINABLE ARTIFICIAL INTELLIGENCE (XAI) APPLIED TO HS- SCREENING MODELS (PAPER 2).....		136
3.1	Abstract .....	136
3.2	Introduction.....	136
3.3	Literature review .....	137
3.3.1	Background and importance of XAI in healthcare research.....	137
3.3.2	Tools and techniques for XAI.....	138
3.4	Methods.....	141
3.4.1	Data.....	141
3.4.2	Model & explainable AI tool selection.....	141
3.5	Results and discussion .....	146
3.5.1	Analysis of the models for male population .....	148
3.5.2	Analysis of the models for female population .....	153
3.5.3	Top predictors of HS .....	159
3.5.4	Comparison of results in male vs female populations .....	159
3.6	Summary and conclusions .....	160
3.7	Recommendations for future work .....	161
3.8	Figures.....	162
3.9	Tables.....	171

3.10	References .....	180
APPENDIX C. P2 - CODE.....		183
CHAPTER 4. ASSESSMENT OF HS PREDICTION MODELS USING HEAVY METAL EXPOSURE DATA (PAPER 3).....		204
4.1	Introduction.....	204
4.2	Literature review .....	204
4.3	Methods.....	205
4.4	Results and discussion .....	209
4.5	Summary and conclusions .....	211
4.6	Recommendations for future work .....	211
4.7	Tables.....	212
4.8	References.....	214
APPENDIX D. P3 - CODE FOR OBJECTIVE 3 .....		216
CHAPTER 5. ASSESSMENT OF A COMMERCIALLY AVAILABLE SENSOR FOR ARSENIC DETECTION IN WATER (PAPER 4) .....		270
5.1	Introduction.....	270
5.2	Methods.....	271
5.3	Results & discussion .....	276
5.4	Summary & conclusions .....	281
5.5	Recommendations for future work .....	282
5.6	Figures.....	283
5.7	Tables.....	292
5.8	References.....	295
APPENDIX E. P4 - CODE.....		297
GENERAL CONCLUSIONS.....		307
GENERAL REFERENCES.....		309
APPENDIX F. NON-ALCOHOLIC FATTY LIVER DISEASE (NAFLD) FROM MULTIPLE SCIENTIFIC PERSPECTIVES AND CONTEMPORARY REVIEWS .....		316
APPENDIX G – IRB INFORMATION .....		335



## LIST OF TABLES

Table 2.1: Model performance summary - objective 1A.....	62
Table 2.2: Class balanced datasets using under-sampling.....	65
Table 2.3: Class balanced datasets using SMOTE.....	65
Table 2.4: Model Performance Summary for HS Screening using Under-Sampling.....	65
Table 2.5: Model Performance Summary for HS Screening using SMOTE.....	66
Table 2.6: Best performing (sensitivity only) models for HS Screening using SMOTE .....	66
Table 3.1: Clinically defined normal values for male and female populations .....	171
Table 3.2: Male quadratic SVM – partial dependency result analysis .....	171
Table 3.3: Male gaussian I SVM – partial dependency result analysis .....	172
Table 3.4: Male gaussian II SVM – partial dependency result analysis .....	173
Table 3.5: Female quadratic SVM – partial dependency result analysis .....	174
Table 3.6: Female gaussian I SVM – partial dependency result analysis.....	175
Table 3.7: Female gaussian II SVM – partial dependency result analysis .....	176
Table 3.8: Male-specific model observations .....	177
Table 3.9: Female-specific model observations.....	178
Table 3.10: Comparison of best performing models in male vs in female populations .....	179
Table 3.11: Mean of predictor performances – male-specific models.....	179
Table 3.12: Mean of predictor performances – female-specific models .....	179
Table 4.1: Dataset sizes after applying SMOTE – with heavy metal exposure parameters .....	212
Table 4.2: Dataset sizes after applying SMOTE – without heavy metal exposure parameters ..	212
Table 4.3: Best performing models using heavy metal exposure data - male populations.....	212
Table 4.4: Best performing models excluding heavy metal exposure data - male populations..	213
Table 4.5: Best performing models using heavy metal exposure data - female populations.....	213
Table 4.6: Best performing models excluding heavy metal exposure data - female populations .....	213
Table 5.1: Visual snapshot - within concentration analysis (4 replicates/concentration).....	292
Table 5.2: Visual snapshot - pairwise concentration analysis .....	292
Table 5.3: Mean values of the hue data – all replicates .....	293

Table 5.4: Mean and SD of replicates – hue data .....	293
Table 5.5: Mean values of the saturation data – all replicates .....	293
Table 5.6: Mean and SD of replicates – saturation data .....	294
Table 5.7: Confusion matrix of Euclidean distances .....	294

## LIST OF FIGURES

Figure 2.1: Global NAFLD prevalence [7].....	60
Figure 2.2: Progression of NAFLD [18].....	61
Figure 2.3: Summary of the methods used for data cleaning and model training .....	61
Figure 2.4: Flowchart of methods used in Objective 1B .....	63
Figure 2.5: Logic used for creating synthetic male HS data with SMOTE .....	64
Figure 2.6: Logic used for creating synthetic female HS data with SMOTE .....	64
Figure 3.1: Typical machine learning models complexity vs interpretability .....	162
Figure 3.2: Partial prediction plot with ambiguity zone ( $0 \pm 0.15$ ) .....	162
Figure 3.3: Methods used for explainability analysis .....	163
Figure 3.4: Quadratic SVM - Partial dependency plots for each predictor - male population ...	164
Figure 3.5: Gaussian SVM – Scale 1 - Partial dependency plots for each predictor - male population .....	165
Figure 3.6: Gaussian SVM – Scale 2 - Partial dependency plots for each predictor - male population .....	166
Figure 3.7: Quadratic SVM - Partial dependency plots for each predictor - female population	167
Figure 3.8: Gaussian SVM – Scale 1 - Partial dependency plots for each predictor - female population .....	168
Figure 3.9: Gaussian SVM – Scale 2 - Partial dependency plots for each predictor - female population .....	169
Figure 3.10: Individual predictor performance - male population.....	170
Figure 3.11: Individual predictor performance - female population.....	170
Figure 5.1: Image of the photography box (purchased from a commercial source) used to capture images with consistent lighting.....	283
Figure 5.2: Calibration scale provided by the manufacturer* .....	283
Figure 5.3: Results of the Arsenic calibration curve using ICP-Mass spec.....	284
Figure 5.4: Flowchart indicating the process used for visual analysis .....	284
Figure 5.5: Flowchart indicating the process used for image analysis and pattern recognition .	285
Figure 5.6: Graphical lay out for the digital images of test kit samples arranged in row for each concertation (rows) and each replicate (columns) [9].....	286

Figure 5.7: A: 0 ppb (Arsenic in water) test results for four replicates (R1,R2,R3,R4) ..... B: Manufacturer's result .....	287
Figure 5.8: A: 10 ppb (Arsenic in water) test results for 4 replicates (R1,R2,R3, R4)     B: Manufacturer's result .....	287
Figure 5.9: A: 50 ppb (Arsenic in water) test results for four replicates (R1,R2,R3, R4)     B: Manufacturer's result .....	287
Figure 5.10: A. 100 ppb (Arsenic in water) test results for four replicates (R1, R2, R3, R4)     B: Manufacturer's result .....	288
Figure 5.11: A. 200 ppb (Arsenic in water) test results for four replicates (R1,R2,R3, R4)     B: Manufacturer's result .....	288
Figure 5.12: 0 ppb vs 10 ppb (Arsenic in water) side by side comparison .....	288
Figure 5.13: 10 ppb vs 50 ppb (Arsenic in water) side by side comparison .....	289
Figure 5.14: 50 ppb vs 100 ppb (Arsenic in water) side by side comparison .....	289
Figure 5.15: 100 ppb vs 200 ppb (Arsenic in water) side by side comparison .....	289
Figure 5.16: Boxplot of mean hue data, commercial kit testing (four replicates per concentration) .....	290
Figure 5.17: Box plot of mean saturation data, commercial kit testing (four replicates per concentration) .....	290
Figure 5.18: Mean hue vs mean saturation scatter plot .....	291

## GLOSSARY

**Artificial Intelligence:** The umbrella term used to describe a range to tools like Machine Learning (ML), Artificial Neural Networks (ANN) etc.

**Machine Learning (ML):** A technique used to train mathematical models using a set of training data and test them with a set of different, test data points. ML is typically used for prediction problems like classifying data into groups etc.

**Training a model:** Using a set of input observations to a mathematical model to “train” the model to fit the data

**Testing a model:** Using new data on a trained model to predict a specific output

## LIST OF ABBREVIATIONS

AI:	Artificial Intelligence
ALT:	Alanine aminotransferase
As:	Arsenic
ASP:	Alkaline phosphatase
AST:	Aspartate aminotransferase
Cd:	Cadmium
GGT:	Gamma-glutamyl transferase
Hg:	Mercury
HS:	Hepatic Steatosis
IR:	Insulin Resistance
KNN:	K-nearest neighbors
ML:	Machine Learning
NAFLD:	Non- Alcoholic Fatty Liver Disease
Pb:	Lead
PDP:	Partial Dependency Plots
SVM:	Support vector machines
TAFLD:	Toxicant Associated Fatty Liver Disease
TASH:	Toxicant Associated Steatohepatitis
XAI:	Explainable Artificial Intelligence

## **ABSTRACT**

This research was conducted to develop and evaluate a screening tool for Hepatic Steatosis (or fatty liver) detection using machine learning based models. The developed models are intended to be used as a potential clinical decision support tool for identifying patients with Non-Alcoholic Fatty Liver Disease (NAFLD). Two versions of a HS prediction tool are discussed in Paper 1, Objectives 1A, and 1B, respectively.

Explainability analysis of the developed models is also a major component of this work, discussed in Paper 2. Models from Paper 1 are analyzed further for interpretability and the results are then compared with current clinical literature. Insights from the explainability analysis are used to identify best models that follow the clinical literature logically. Most contributing features within each model are also identified in this work.

Another aspect of NAFLD management is related to the chronic exposure to heavy metals in the environment (such as: Arsenic, Lead, Cadmium etc.). The heavy metal exposure component is explored in two ways in this dissertation. In paper 3, another version of the ML-based screening tool is explored by including heavy metal exposure data. The results from the model (with heavy metal data) are then compared with models that exclude the heavy metal exposure data. The results and their implications are discussed in paper 3.

Arsenic is a major hepatotoxin and the chronic exposure can lead to severe liver injury. In Paper 4, a commercially available Arsenic detection kit was examined for Arsenic detection in water at a household level. The kit was evaluated following a short experimental plan and the obtained results are discussed. Finally, the obtained images were quantified digitally using a customized image analysis and pattern recognition algorithm. The methods used for quantification and the obtained results are also discussed.

## ORGANIZATION OF THIS DISSERTATION

The dissertation starts with a general introduction, a list of research objectives and a general literature review. Three conference articles (all published in conference proceedings), one under-review journal article and one future publication are included in this dissertation, with each research objective organized as an individual chapter. The general literature review is relevant to the overall theme and content in this dissertation. The general literature review contains literature related to each of the three objectives with further cross-references included to specific objectives. Each objective then has a specific literature review section relevant to the topic.

Although all objectives are linked to the goal, each objective is stand-alone. Output of every objective can be used to create one or more scholarly publications. Attached appendices and supporting information are available at the end of each chapter.

*Note: The structure of this thesis is not in the conventional format. This thesis is structured using a graduate school approved - article format.*



## GENERAL INTRODUCTION

This Ph.D. dissertation focuses on addressing an important disease called non-alcoholic fatty liver disease (NAFLD) using health informatics and machine learning models. This research was conducted in the Integrated Sensing and Smart Solutions laboratory of Purdue University. Before this research was undertaken, a core concept for a holistic system-based representation of liver diseases and its association with human lifestyle, food, water, environment was discussed [1]. From this framework, a specific issue of NAFLD using intelligent modeling and health informatics is discussed below.

Non-alcoholic fatty liver disease (NAFLD) is a progressive liver condition in individuals with low to moderate alcohol consumption. An estimated 80 million people are affected with NAFLD in the USA, as of 2016 [2]. The estimates further report a 25% prevalence of NAFLD in adults around the world [2].

The disease is chronic and can be broken down into broadly four stages, ranging from simple fatty liver to liver cirrhosis (or even liver carcinoma) [3]. In the final stages, a liver transplant becomes necessary for survival. Based on 2018 data from the Organ Procurement and Transplant Network (OPTN), NAFLD was increasingly found to occur in liver transplant waitlist candidates [4].

NAFLD is associated with several risk factors like obesity, diabetes (type-II), insulin resistance and, hyperlipidemia [5]. Aside from the typical risk factors, environmental risk factors like heavy metal exposure are known to worsen liver injury [6]–[9]. Chronic exposure to Arsenic, lead, mercury, and Cadmium both directly and indirectly (via contaminated food and water) could worsen liver functionality. Recently, the terms “Toxicant” associated fatty liver disease (TAFLD) and “Toxicant” associated steatohepatitis (TASH) were coined to identify liver injury caused specifically due to toxicants [10], [11]. TAFLD and TASH are similar in pathology to NAFLD and NASH, respectively [10], [11].

To assess liver injury in general, clinicians use Liver function tests (LFTs) as initial investigation tools. Liver function tests include measurements of the following liver biochemicals: alanine aminotransferase (ALT), aspartate aminotransferase (AST), alkaline phosphatase (ASP), Albumin, Gamma-glutamyl transferase (GGT) and Bilirubin. In general, a ratio of  $AST/ALT > 1$  is an indicator of hepatocellular injury [12]. The ratio is useful to detect overall liver injury, but it

is not specific for NAFLD detection [12]. Further, use of LFTs alone for NAFLD detection can be misleading. Research shows that use of liver function tests alone has resulted in a steady underestimation of NAFLD prevalence [2], [13]– [17]. Based on the LFT results, if there is suspicion of a liver disease, imaging tests like qualitative or quantitative ultrasound, Magnetic Resonance Imaging (MRI), or computer tomography (CT) are used for further investigation.

Although imaging tests can be a useful tool for identifying fatty liver, ultrasound tests lack the sensitivity for NAFLD detection [18]. MRI and CT can also be used, but MRIs are expensive and unavailable in many locations. CT on the other hand, involves ionizing radiation, which can be risky for certain individuals. Finally, although liver biopsy is the benchmark for confirmed diagnosis, it is subject to sampling error, is invasive, risky, and expensive [14]. While tools are available to detect liver injury in general, there is no specific biomarker being used in clinical settings for NAFLD detection. The gold standard for NAFLD diagnosis is still liver biopsy [5].

Considering the above reasons, NAFLD detection is currently limited and subject to availability of medical imaging tools. Further, current clinical guidelines do not recommend screening for NAFLD, even for high-risk populations (obesity and diabetes), due to a lack of definitive and inexpensive detection tools [5].

With increasing data and computational power, researchers have used machine learning (ML) and other artificial intelligence techniques and applied it to health conditions. Considering the current lack of screening for HS (even in high-risk groups), the use of a ML-based decision support system would be a novel, complementary approach. Researchers have used ML-based liver disease prediction by using images (MRI, ultrasound, CT), and biopsy data in the past [19]– [21]. However, these are expensive and not accessible to everyone, as discussed earlier. The potential of machine learning to more commonly available multivariate data, has not been implemented before, as per our literature survey. In this dissertation, the potential of minimally invasive data like demographics, previous disease conditions, lipid information, and certain lifestyle factors in identifying NAFLD is investigated. One of the objectives of this thesis is to create ML models that are suited to be used as decision support systems for clinicians, instead of standalone models, due to the complexity of the disease and organ system.

Another dimension to the complexity of liver diseases is added due to exposure to heavy metals or other pollutants. The relationship of these (heavy metals and pollutants) with liver injury has been investigated and well-documented in the past [9], [10], [22], [23]. Heavy metal exposure

is found to leads to various types of skin diseases, kidney malfunctions, and liver problems [24]–[26]. Majority of the heavy metals are documented carcinogens [27]. Global and national agencies like the World Health Organization (WHO), United States Environmental and Food Protection Agency (US-EPA) and, the Food and Drug Administration (FDA) publish their standards for safe drinking water and food [25], [28]. In accordance with the published standards, local state and county agencies monitor the quality of food and water. However, at a household level, only limited options are available for water quality monitoring. If the water in a specific pipeline or area is polluted, chronic consumption of polluted water can lead to bioaccumulation of the toxins in the body, leading to liver dysfunction [29]. Arsenic in particular is a well-documented hepatotoxin that leads to significant bioaccumulation in the liver over time [9].

An easy-to-use sensor for heavy metal detection at a household-level could be useful in proactive monitoring of individuals' water quality and can therefore lead to reduced risk for liver and other organ disorders [30]. The overall research thrust of our lab is related to developing sensor systems for “Water Linked Health and Wellness” [31]. A combination of the sensor system and the screening models are postulated to help in a proactive approach to NAFLD identification and management. The increasing prevalence of the chronic NAFLD condition, combined with the silent symptoms of the disease can be challenging to identify early. Lack of early identification can then lead to a sudden onset of advanced symptoms in the later disease stages. Screening and proactive disease management using the developed sensor system and the ML based screening tools are targeted in this work. Both the developed tools are aimed for use in low resource settings.

The overall research goals of our lab are: A) Use of big-data and advanced computational techniques to detect and manage disease conditions B) Assess, experiment, and evaluate cost-effective sensors for detection of heavy metals in water. Therefore, the goal of this dissertation is to investigate the capability of health-informatics (including machine learning techniques) for identifying specific (early) stages of a progressive liver disease (Hepatic Steatosis) and to explore the potential of a sensor techniques in determining the level of Arsenic contamination in water- a contributing medium for liver disease.

## OBJECTIVES

The goal of this research is to tackle NAFLD from two perspectives: Early screening for Hepatic Steatosis using ML-based model (and related model explainability) and detection of Arsenic in contaminated water. Based on the above introduction, and the research goals, this thesis has the following objectives:

1. Develop models for predicting Hepatic Steatosis (HS) using machine learning tools and specific datasets:
  - a. Physiological data only.
  - b. Liver biochemistry & physiological data.
2. Understand the model predictions using explainable artificial intelligence (XAI) to enable model interpretability.
3. Further evaluate the effect of specific heavy metal exposure (contaminants) on the performance of ML-based HS prediction models.
4. Explore and assess specific Arsenic detection technique by testing and evaluating an existing commercial sensor and assessing its performance.

## **CHAPTER 1. GENERAL LITERATURE REVIEW**

### **1.1 Artificial Intelligence: benefits and challenges for healthcare**

The increasing collection and storage of data via personal devices, sensors, and other digital sources has led to a spike in data availability. The combination of large amounts of data with powerful computation has given rise to multiple domains such as data analytics, data science, informatics, database management, data mining & statistical analyses etc. These domains are applied to solve problems in various fields such as manufacturing, finance, technology, communications, transportation, education, and healthcare.

Data analytics has also been applied to medical/healthcare data to solve specific clinical problems. Due to the large amounts of medical data being collected by hospitals, sensors, and wearable devices, etc. the electronic health records (EHR) are now available for analyses. EHR and other forms of medical data (image data (like MRI, Ultrasound etc.), time-series data (like EEG, ECG, EKG etc.), doctor's notes and annotations etc.) are now being used to predict or detect specific health conditions. Disease progression modeling, disease/condition predictions, specific motion detection (fall detection, gait detection, sleep analysis etc.) have been researched previously. Further, Human Activity Detection (HAR) using wearable sensors and video-based detection are being increasingly researched recently.

The motivation and need for healthcare related analytics arise from a combination of scarcity of clinical resources, clinical manpower and access to medical devices/technologies. Use of data and technology for healthcare is also being applied in the mobile-Health or m-Health domain. Using machine learning (ML) or other artificial intelligence techniques (AI), m-health services can now be provided remotely, using tele-medicine or tele-health. Personalized recommendations, clinical services and health predictions can all be made remotely.

Although the computational power and motivation for clinical analytics is high, it has several challenges. Processing a large amount of data, making sense of the trends in big data, handling challenges with poor quality of healthcare data (missing information, class imbalance etc.), data privacy, design and implementation, interpretability of the ML/AI models etc. are some of the major challenges. These need to be tackled carefully as the impact of health-related predictions is very high. Some of these challenges are explained in detail below.

### **1.1.1 Challenges in processing big data**

Large number of features/parameters and many observations/samples exist in big datasets. Such high-dimensional data is challenging to process for several reasons. First, large datasets often contain noise (undesirable features or observations), missing information, sparsity, and irregularity. Solutions need to be designed on a case-by-case basis to tackle the above-mentioned issues. The challenges are often unique based on the problem statement and the dataset in question. Therefore, customized solutions are often required. This step is commonly termed as data ‘pre-processing’. Development of custom solutions to pre-process the data is time-consuming and difficult but is also necessary to avoid future biases and poorly trained AI/ML models.

Second, the selection of relevant features for a particular use-case and processing them to better train the model is called ‘feature selection’ and ‘feature-engineering’, respectively. Feature selection is important to make sure the model performs optimally and does not learn from undesired parameters (noise). Any confounding/redundant parameters are also eliminated in this step. Feature-engineering can be then used to process the features further. For example, converting a continuous feature into discrete, segmenting a time-series dataset into segments, or applying specific scaling or normalizations to any feature. Such data transformation techniques are used to further improve the model performance and eliminate any irregularities/outliers.

### **1.1.2 Challenges specific to healthcare datasets and the need to address them**

In addition to the challenges presented by big data in general, there are certain challenges that are inherent to healthcare datasets. Class-imbalance is one of the common challenges with any specific disease related data. Typically, a higher percentage of the population does not have a certain disease/condition, while a smaller percentage of the same population has the disease. For example, a study reporting on incidence of cancers in the USA found that intrahepatic bile duct cancer was the most common cancer with an incidence rate of 1.49 per 100,000 persons [32]. Direct use of such population health data, without adjusting or rebalancing the disease vs no-disease classes could lead to a mis-trained AI/ML model with severe implications. Therefore, class-balancing is an important aspect as part of training an AI/ML model for healthcare applications.

Noise in time-series data (such as EEG, ECG etc.) is inherent to healthcare datasets for especially related to neurology, cardiology, etc. Various noise artifacts are part of the data, some due to the patient's motion (general motion, scratching etc.), some others as baseline noise, line noise, etc. The use of such datasets requires very specific and intensive data cleaning and processing before the data can be meaningfully used. More details about the challenges specific to healthcare are elaborated in a 2017 book related to big data in healthcare [33].

### **1.1.3 Regulation of healthcare AI:**

As such, the domain of developing prediction models for human health is also very highly regulated. The Food and Drug Administration (FDA) in the United States launched a 'Digital Health' division in 2019 with 'new regulatory standards for AI based technologies' [34], [35]. This division regulates any standalone algorithms as that can be used as 'medical devices. Further, the International Medical Device Regulators Forum (IMDRF) also treats software that can be used for one or more medical needs without any hardware requirements as 'Software as a Medical Device (SaMD)' [34]. The FDA also treats such medical standalone software as SaMD. AI-based algorithms that are made for treatment, cure, prediction, mitigation and/or prevention of any disease/condition need to be approved by both the FDA and IMDRF [34].

Most AI-based models are black-box in nature. Black-box models use input parameters to predict or produce output(s), but the internal working of the model is not explained. These black-box models are quick solution providers, used to replace certain tests that could take a long time and resources [36]. FDA regulates any such black-box models [35].

Although the extensive regulations surrounding healthcare analytics/ healthcare AI could be challenging, such regulations are required to ensure that the AI models are thoroughly tested and validated before being implemented in the clinical system [33], [34]. However, the laws are different based on different regions and some countries, such as India, do not yet have any regulatory framework to patent algorithms [34]. Algorithms are the base of any AI system, and they need to be regulated especially when implemented in fields like healthcare.

Finally, data privacy is essential in all domains where data is collected, analyzed, stored, and used for recommendations/predictions etc. But in healthcare, patient data and its use in an AI model can have severe impact, if it is not handled with care. Preserving the privacy of patient data during collection, storage, analyses, and transfer of patient related information is critical, as

mandated by the Health Information Portability and Accountability Act's (HIPPA) [36]. While adhering with these rules can be a challenge, they are critical and need to be integrated into the first steps of any model development pipeline.

#### **1.1.4 Interpretability of black-box models**

Use of AI for medical applications has been of interest for both researchers and for clinicians. The commonly used AI/ML models are black box in nature, as explained earlier. The reasoning behind a black-box model in producing a specific output is not given. This lack of explanation makes the models un-interpretable. Typically, as the model complexity increases, the model performance increases but its interpretability decreases.

Potential biases in a black-box model, combined with a lack of trust and understanding of the model's internal working are especially detrimental when applied to healthcare. Such a lack of trust has been expressed previously by clinical practitioners [37]. Therefore, developing tools for explaining the predictions of AI-based models is warranted. This need has led to an increasing interest in explainable artificial intelligence (XAI), specifically in medical applications, and a surge in research has been observed since 2015 [38].

Although previous research was mostly focused on the use of ML and other AI based models as black boxes, more research is recently being conducted in understanding how the model behaves. There are two ways to understand model behavior: 1) Explainability 2) Interpretability. Although these two terms are often used interchangeably, they are different. Interpretability analysis is used to understand how a model is interpreting a certain feature. Explainability is used to understand how a certain input effects the model's output.

As mentioned earlier, an increase in the research articles related to explainable AI in medicine was noted since 2015 [38], [39]. XAI and its applications have several benefits including but not limited to identifying potential bias in the data, gaining better insights about the use of input parameters, improving model understanding, interpretability and overall building higher trust in the model's predictions [37], [40], [41]. However, when XAI is applied to health care, it poses several unique challenges.

The first challenge is that unlike in other physical systems where the underlying behaviors can be quantified using mathematical equations, such behaviors cannot be obtained in most healthcare applications [37]. That is, the exact relationship, in a cause-and-effect manner, cannot



be obtained for multiple healthcare applications [37]. In fact, for the same medical condition, the diagnosis and treatment provided by different clinicians could vary [37]. The subjectivity among clinicians and lack of the knowledge regarding the exact behavior are a major challenge in developing XAI methods for healthcare applications.

Secondly, the availability of sufficient data, longitudinal parameters, and quality of the data, lack of structure within data pose another challenge when applying XAI to healthcare/medicine [38], [39], [41]. Further, an inherent challenge with any XAI tool is that the tool estimates the provided interpretations and could therefore have errors in estimation itself [42]. Potential errors in estimation, coupled with other errors in the underlying AI model could significantly impact the interpretation of the model's predictions [42]. Finally, the regulation laws regarding software as a medical device by the Food & Drug Administration (FDA) [43], General Data Protection Regulation (GDPR) in Europe [44], have extensive requirements for use of AI models and explainability tools. Based on these challenges, multiple researchers recommend the use of current XAI tools in healthcare to augment the decision of clinicians, instead of using such tools in a standalone manner [37], [39], [42]. In the future, more robust XAI techniques, combined with richer datasets could lead to significant breakthroughs in this domain, but remain an open challenge for now [45].

Although XAI has its challenges, its potential to improve medical decision making has led to the use of XAI in some healthcare/medical models. The XAI methods in medicine have been developed for intraoperative decision support [45], for predicting acute critical illness using EHRs [46], for simulation-based training in surgery [47] and also for prediction of deteriorating Hepatitis [48]. Recent literature related to such methods is outlined in the literature review section specific to Paper 2.

## **1.2 Liver and toxicology**

### **1.2.1 Liver functionality and importance**

The human liver is the major organ for metabolism and synthesis of carbohydrate and lipids in the body. The synthesis of all food and liquids makes the human liver highly prone to toxicity [9], [22]. Further, the liver functionality depends on several factors. These factors can be internal agents or external agents. Viruses, toxins (like pollutants or heavy metals) and drugs are the major

external factors that can impact the liver's ability to function. On the other hand, genetic conditions or cancers can also impact the liver's functionality internally.

Viral infections can lead to various hepatitis conditions (A, B or C) while toxins can cause chronic conditions due to their bioaccumulation in the liver [8], [24], [49]. While certain trace metals like Zinc, Copper, Manganese etc., are essential for metabolism and homeostasis [50], certain other metals like Arsenic, Lead, Mercury and Cadmium can severely deter the liver's functionality [9], [22], [51]– [54].

Chronic liver disease (CLD) can occur due to a variety of reasons. Some of the chronic liver conditions are Alcoholic Liver Disease (ALD), Non-Alcoholic Fatty Liver Disease (NAFLD), Chronic Viral Hepatitis, Genetic conditions (Alpha-1 antitrypsin deficiency, Hereditary hemochromatosis, Wilson disease), Autoimmune hepatitis, Primary biliary cirrhosis, primary sclerosing cholangitis and other drug, vascular conditions [55].

### **1.3 Heavy metals**

Metals are unique in comparison with other toxic substances in multiple ways. Unlike other pollutants that are man-made (plastic, manufactured chemicals, pharmaceutical products, cosmetics, etc.), metals occur naturally in the environment [49]. Metals are also used in various industrial processes, leading to the creation of synthetic heavy metals [56]. Although some metals are essential to the proper functioning of several biological species, their presence is required in very low amounts [56]. Essential metals like copper, Iron, and zinc are required in trace amounts for the proper functioning of enzymes and other cellular operations [56]. Certain other metals, however, are non-essential and in fact, toxic to biological organisms in any quantity (e.g.: lead) [56].

Heavy metals differ from other metals based on their high atomic density (relative) and their insolubility [49]. Heavy metals exist both naturally and in synthetic forms. However, the difference in their chemical species varies their toxicity [49]. It is not possible to create or destroy metals completely [49]. They are non-biodegradable which makes them accumulate inside biological organisms (bioaccumulation) [49]. They can also travel across biological systems from water to seafood and eventually to the human system. Fish, rice, and other foodstuffs were found to contain heavy metals when farmed in a contaminated environment [8]. Environmental disasters and spills related to metals, particularly heavy metals, can also cause acute toxicity.

The use of heavy metals in industries leads to unavoidable human exposure (even though it is in trace and regulated amounts) [49]. Some examples of industries with occupational exposure hazards are pharmaceutical, manufacturing, packaging, agricultural, and construction industries [56]. The extent of metal usage makes their concentrations in the environment vary (air, water, soil) [56]. The levels of heavy metals in air, soil, and water differ by area; point-source areas such as those with mining activity, anthropological activity, foundries, smelters, etc. are prone to high heavy-metal contamination [56].

### **1.3.1 Heavy metal contamination and pathways into the human system**

Heavy metals exist in different forms (organic, inorganic, etc.) based on their stability and form (solid, liquid, gas etc.). The metabolism of heavy metals by the body (toxic mechanisms), their rate of dispersion into the body (Toxicokinetics) and their routes of exposure are also different. In some cases, the same heavy metal can be exposed to the human body via multiple pathways like ingestion, dermal exposure, absorption etc. Mechanisms of some of the common heavy metals (Lead and Arsenic) are elaborated below.

### **1.3.2 Lead:**

#### *Routes of exposure*

Lead has multiple routes of exposure to the human environment via air, water, soil, and consumer products [25]. Further, lead exposure routes can also be related to past uses of lead [25]. The use of fossil fuels (use of leaded gasoline in the past), lead-based paints, ceramics, corroding plumbing materials inside households, and industrial activities are some of the common routes of exposure to lead [25]. Industrial sources, contaminated past lead smelters, mining, and refining activities also lead to increased lead concentrations in the environment (soil, water) [25]. Lead travels from the soil into water, based on the type of lead compound and soil properties [25]. Industrial contamination also releases lead particles in the air, which can travel and fall onto the soil, polluting both soil and air [25]. Five categories of potential lead sources are air, dust, soil, water, household materials.

1. *Air*: Lead smelters, metals processing, piston-engine aircraft operations using leaded aviation fuel, waste incinerators, utilities, and lead-acid battery manufacturers.

2. *Dust*: Lead paint dust from old homes, home development activities that release dust with lead in it.
3. *Soil*: Past use of leaded fuel could be present in soil (travel via dust, settled into the soil), exterior building paints from past lead use in paints can become flaky and deposit on soil, industrial sources.
4. *Water*: Contamination from nearby industrial sources and corrosion of plumbing fittings and other plumbing material.
5. *Household materials*: Painted toys, furniture, jewelry, cosmetics, certain types of food, and liquid containers.

### *Toxic mechanisms*

Lead toxicity was found to cause cell death by reducing the antioxidant defense mechanism and increasing ROS production, leading to oxidative stress and eventual cell death [57]. Lead toxicity also disrupts the protein, lipid, and DNA pathways in the body via protein oxidation (altering the function), lipid peroxidation (disrupting membrane), and nucleic acid oxidation (cancer/mutation), respectively [57].

The health consequences of lead toxicity were found to majorly affect the central and peripheral nervous systems compared to any other organ systems in the human body [57]. Some of the established health hazards caused by lead are encephalopathy, paralysis, coma, neurological problems with fetuses and growing children, anemia, renal dysfunction, hypertension, cardiovascular diseases, ischemic coronary heart disease, cerebrovascular accidents and peripheral vascular disease, reproductive disorders in both men and women, lead storage and mobilization from bones [57].

### *Toxicokinetics*

Lead exposure can be absorbed by the respiratory tract (inhalation), GI tract (ingestion), and by touching (dermal exposure). Dermal exposure is much less lethal compared to oral or inhalation routes. Children can absorb lead at a higher rate (40-50%) compared to adults (3-10%) when exposed to water-soluble lead [58].

Once absorbed by the body, lead distribution primarily occurs via absorption by bones through the bloodstream (In adults, 94% of body burden is in the bones, compared to 73% in children [58]. In the bloodstream, lead travels through the red blood cells. The metabolism of lead (inorganic) leads to complex formation with proteins and ligands [58]. The liver actively metabolizes organic lead compounds [58]. Independent of the exposure route, lead is excreted mainly by urine and feces [58]. Smaller elimination routes are via hair, nails, breast milk, sweat, and saliva [58]. Elimination time for lead is dependent on the retention rate. Elimination time for lead ranges from 1 week to 2 years, however, lead from bones is excreted at a much slower rate of 1-2 decades [58].

#### *Sensitive populations*

Children, pregnant women, and adults at risk of occupational exposure are the most sensitive populations for lead exposure [25]. In children, lead absorption occurs at a higher rate than in adults, leading to higher brain and nervous system damage [25]. During pregnancy, lactation, menopause, and osteoporosis increase the exchange rate between blood and bones, leading to a higher rate of lead in the blood for these populations. It can also be transferred between the mother and the fetus or the mother and the baby (via breast milk) [25]. Occupational exposure due to the breathing of dust or air particles with lead is also a health concern [25].

### **1.3.3 Arsenic**

#### *Routes of exposure*

Arsenic (As) has multiple routes of exposure through ingestion, inhalation, and dermal exposure. In human beings, the primary route of exposure to Arsenic is via consumption of contaminated food and water [24], [59]. Geological characteristics of soil and drinking water quality can lead to health complications in the exposed populations via the transfer of heavy metals [24]. In food, the order of Arsenic concentration from highest to lowest is as follows: seafood 16.7 mg/kg in marine fish, 3.5 mg/kg in mussels, and more than 100 mg/kg in certain crustaceans), followed by meats, cereals, vegetables, fruit, and dairy products [59].

Inhalation, particularly via cigarette smoking and as an occupational exposure in miners is another route [24]. Arsenic exposure through particulate matter in air occurs through the inorganic

compound, arsenic trioxide [24]. These levels of arsenic in the air vary by area based on anthropogenic activities and industrial zones [24]. Although exposure via skin contact is also possible with Arsenic, it is not a common route of exposure when compared to exposure via contaminated water and food [24].

Arsenic sources of exposure are both natural (deposits of Arsenic, volcanic activity, erosion, Arsenic in water bodies like aquifers) and manufactured (like mining, anthropological activities, etc.) [24]. The primary source of Arsenic exposure currently is mining, which leads to mineral dissolution into water and soil. Soil concentration of Arsenic ranges from 1 to 40 ppm (mean: 5 ppm) [24]. Volcanic areas have a soil Arsenic concentration of 20 ppm [24]. The natural presence of Arsenic in drinking water due to geological features is also a major contributor to Arsenic pollution [24]. The second-largest source of exposure is the use of pesticides and by-products from industrial activities [24].

Arsenic exists in three different forms: Organic Arsenic, inorganic Arsenic, and arsine gas [59]. The compounds in each form are listed below.

1. Organic Arsenic compounds: Arsanilic acid, Methylarsonic acid, Dimethylarsinic acid (cacodylic acid), and Arsenobetaine [59].
2. Inorganic Arsenic compounds: Arsenic Trioxide, Sodium Arsenite, Arsenic Trichloride, Arsenic Pentoxide, Arsenic acid, and Arsenates (like Lead Arsenate, Calcium Arsenate) [59]. Inorganic Arsenic is the most toxic form and a confirmed carcinogen [9]. Organic Arsenic present in seafood is considered less harmful [60].
3. Inhalation of Arsine gas in significant quantities can be fatal [61].

### *Toxic mechanisms*

Arsenic has multiple complex metabolic pathways inside living organisms, which are further dependent on the chemical species [60]. Arsenic is stored and metabolized primarily in the liver, where it is prepared for elimination via urine [24]. The major detoxification pathway of inorganic Arsenic from the human body is via demethylation. However, intermediate metabolites released during demethylation were found to have toxic effects and cause DNA damage [60]. A high methylation index was also associated with skin lesions and skin cancer [60].

Neurotoxicity due to Arsenic exposure impacts the peripheral and central nervous systems, particularly the glial component of the central nervous system [60]. Mitochondria are highly prone

to neurotoxicity by Arsenic [60]. Further, a class of Arsenic compounds: arseno-lipids, arseno-hydrocarbons, was found to be toxic to human neurons and was able to cross an in vitro brain barrier [60]. Hence, compounds of this class might have the potential for neurodevelopmental toxicity [60].

The Arsenic (III) compounds are believed to be the most toxic and carcinogenic form of Arsenic [60]. DNA repair system inhibition, interference with redox regulation, and ROS (reactive oxygen species) production were also found to be linked with carcinogenic mechanisms [60]. As (III) compounds have a strong affinity for SH-groups, contributing to acute toxicity [60]. Further, Arsenic (III)-compounds inhibit cytosolic SH-enzymes such as glutathione reductase [60].

Long- term Arsenic exposure was found to be associated with lasting epigenetic changes, potentially causing heritable gene expression changes (histone modification, RNA interference, and DNA methylation) [60]. However, organic Arsenical arsenobetaine was found to be excreted unchanged and is therefore not classified as carcinogenic [60].

### *Toxicokinetics*

Arsenic exposure can advance into the human body via ingestion (food, water), inhalation (air, Arsine gas), skin penetration (via touch). The toxicokinetic mechanism of Arsenic is dependent on the chemical species, duration of exposure, and physiochemical properties of the exposed compound.

Ingestion of inorganic Arsenic leads to absorption of 70-90% of Arsenic into the gastrointestinal (GI) tract [24]. From the GI tract, it spreads mainly to the liver, kidneys, lungs, and bladder via the bloodstream [24]. The highest accumulation occurs in the liver during this phase. A portion of the absorbed Arsenic is excreted through urine. Cells in the body disperse Arsenic through the phosphate transport system, aquaporins and transporters of hexose permeases [24]. pKa and intestinal microbiota also contribute to the absorption and toxicity of As. The affinity of Arsenic (III) compounds with SH-groups results in high Arsenic deposition in hair, skin, and nails [24]. Once in the body, the majority of Arsenic metabolism occurs in the liver (especially in mammals) [24]. The processing in the liver facilitates the elimination of Arsenic via urination from the body [24]. The majority of eliminated Arsenic is in demethylated forms (60-80%), while the remainder is inorganic (10-30%) and monomethylated (10- 20%) forms [24].

### *Sensitive populations*

Certain populations are more likely to be exposed to Arsenic based on geographical location and occupation. Individuals working in wood preservation work, metal manufacturing, glass production, and electronics industries are most susceptible aside from individuals living in areas with high natural levels of Arsenic [62]. Children are also more susceptible than adults based on health impacts [62]. Further, pregnant women and unborn babies could also be harmed due to Arsenic exposure [62].

## **1.4 Heavy metals and NAFLD**

Heavy metals and other toxicants are an additional burden for patients suffering with chronic liver conditions like NAFLD. Upon examination of liver biopsies of workers exposed to vinyl chloride, it was found that 80% prevalence of steatohepatitis existed in workers with occupational exposure [11]. It is critical to note that these workers were specifically identified to not have any other identifiable risk factors, other than occupational exposure [11]. Several other studies have also found a positive association between exposure to toxicants and increased prevalence of NAFLD [9], [10], [63]– [65]. In fact, in cases of high chronic exposure, these conditions are labelled as Toxicant Associated Fatty Liver Disease (TAFLD) and Toxicant-Associated Steatohepatitis (TASH). As mentioned earlier in the introduction section, TAFLD and TASH are similar in pathobiology to NAFLD and NASH, respectively [10], [11].

Chronic exposure to heavy metals is dangerous and can lead to several other health conditions as well. US-EPA, WHO and other agencies around the world establish the acceptable levels of heavy metals and other pollutants in drinking water. Local agencies then monitor and ensure compliance of public water sources with the established standards. However, private sources of water like wells, springs or surface water sources are not monitored. Contamination of drinking water sources by heavy metals leeching into could occur via faults in “household plumbing, service lines, mining operations, petroleum refineries, electronics manufacturers, municipal waste disposal, cement plants, and natural mineral deposits” [66]. Chronic exposure to such contaminated drinking water can potentially initiate liver dysfunction or exacerbate existing liver injuries.



Recent studies have shown that an association between NAFLD and exposure to heavy metals like Mercury [63], Arsenic [9], and Lead [67]. Several animal models were explored to understand the relationship between abnormal liver biochemistry and heavy metal exposure. For instance, chronic Lead exposure in adult mice was found to lead to both hepatotoxicity and change in multiple signaling pathways, fatty acid metabolism, and drug metabolism [68]. Further, the same study also reported an increase in the levels of three liver biochemicals (AST, ALT, and ALP) with an increase in Lead exposure [68]. A similar finding was reported using Common Carp from the Topolnitsa reservoir [69]. Similarly, it was found that chronic Arsenic exposure (> 9 months) led to induced hepatic steatosis in mice when fed with drinking water contaminated with Arsenic (3.2 mg/L) [70].

In human studies, researchers reported abnormal levels of liver biochemicals when individuals were exposed to heavy metals. For example, a clear ‘demographic and mechanistic overlap’ was reported between Arsenic exposure and NAFLD in individuals [9]. Higher incidences of obesity and NAFLD were also reported from states within the USA where Arsenic was contaminating the drinking water [71].

In summary, chronic heavy metal exposure is reported to be linked with abnormal liver biochemistry, with NAFLD and, with liver damage in general. The liver cannot metabolize heavy metals and therefore the screening, diagnosis, and early intervention for individuals living with chronic heavy metal exposure is critical.

## **CHAPTER 2.      HEPATIC STEATOSIS (HS) PREDICTION USING MACHINE LEARNING (PAPER 1)**

*A portion of the work in this paper was published in two peer-reviewed conference proceedings. 1) 3rd International Conference on Computational Biology and Bioinformatics (ICCB 2019), and 2) IEEE – Engineering in Medical & Biological society Conference (EMBC 2021).*

### **2.1 Introduction**

Despite the increasing prevalence of NAFLD, there is no clinical procedure for screening, yet. Further, the exact cause for Non-Alcoholic Fatty Liver Disease (NAFLD) is unknown and currently there are no specific biomarkers that can identify the disease with specificity. However, multiple factors contribute to the disease condition and are established as risk factors (obesity, hyperlipidemia, diabetes etc.) [1]. Due to the complexity of the disease condition and the lack of clear detection options, the American Association for Liver Diseases (AASLD), does not recommend screening for NAFLD, even within the high-risk categories (obesity and diabetes) [1].

Therefore, there are two potential ways in which NAFLD can be detected in asymptomatic adults, currently. They are: 1) Annual liver functionality tests (LFTs) (only in certain high-resource settings) 2) Incidental detection when the adult is getting treatment/ diagnosis for a different condition. In both the above ways, the health care practitioner recommends a suspected NAFLD patient for further investigation via additional testing/screening based on the liver functionality and related physiological parameters of an individual (ultrasound, MRI, etc.). However, this recommendation is subjective among healthcare practitioners. Note that in this condition, the practitioner makes the decision solely based on available individual data (liver functionality, physiological data, etc.). Therefore, a lack of systematic screening of NAFLD currently exists.

Use of healthcare data to detect disease conditions is currently increasing due to the combination of increased data availability and increased computational power. In this research, the gap in screening of NAFLD is identified and machine learning based models are developed. Background relevant to the disease condition, its prevalence, etiology, and risk factors are elaborated in the literature review section below. Specific gaps are highlighted, and research objectives are presented at the end of the literature review.

## **2.2 Literature review**

Non-Alcoholic Fatty Liver Disease (NAFLD) is one of the major causes for chronic liver condition globally, with increasing prevalence in the recent years. It occurs in individuals who consume limited or no alcohol. However, as explained earlier, the exact cause for NAFLD is not known [2]. While there is no known cause, several risk factors have been identified for this disease [1]. Briefly, obesity, diabetes, dyslipidemia, metabolic syndrome, insulin resistance, polycystic ovary syndrome (PCOS), and chronic heavy metal exposure are all linked to NAFLD risk [1], [3]–[6]. These risk factors and their relationships with NAFLD are elaborated in detail in the sections below along with the etiology and epidemiology of the condition.

### **2.2.1 NAFLD background and epidemiology**

The prevalence of NAFLD has been increasing globally. A global meta-analysis of epidemiological NAFLD data conducted in 2016 reported a 10% increase in global prevalence from 2005 to 2010 [7]. Other researchers have also found similar spikes in NAFLD prevalence in the recent past [1], [8], [9]. The prevalence was found to vary by region and is the highest in the Middle East (31.8%) and South America (30.4%) [7]. The lowest prevalence was reported in Africa at 13.48% [7]. In the USA alone, the prevalence rose from 5.5% to 11% in two decades (1988-08) [7]. As of 2016, an estimated 80 million individuals have NAFLD in the USA [9].

Within Asia, the prevalence across regions was found to be significantly different based on a 2013 study [10]. Within a group of regions in Asia (India, Sri Lanka, Malaysia, Singapore, Indonesia, Korea, Japan, and Taiwan) the prevalence varied between 15 to 45% [10]. China has a reported prevalence of 20%, Japan has 17% and Hong Kong has 27%, with all three regions having increasing prevalence between 2003 to 2013 [10]. Within highly populated regions like China and India, such high prevalence percentages imply that many people are impacted by the disease.

The incidence of NAFLD in diabetics (specially type -2 diabetics) was also found to be rising sharply [1], [9]. It was found to be 55.5% (with patients from 20 different countries) in a 2019 meta-analysis [9]. Of the patients with NAFLD and diabetes, 17% had the advanced form of NAFLD (fibrosis) [9]. In Iran, Saudi Arabia and Turkey combined, a 59.20% of NAFLD prevalence was reported within type-2 diabetics [9]. Similar numbers were reported in India and

Pakistan at 57.46% [9]. However, Europe was reported to have the highest prevalence at 68.82% within the type-2 diabetics [9].

Other disease conditions, like insulin resistance, hypertension and dyslipidemia are also linked with increasing NAFLD incidences [8]. Although obesity is a major risk factor for NAFLD [1], it was also found in lean populations in Asia and in the United States [10], [11]. Increasing age was another factor in increasing NAFLD prevalence, especially in the 40–50-year age range [8]. Different ethnicities were found to be linked to NAFLD, with Hispanic population having the highest prevalence, when compared with non-Hispanic Caucasians and African Americans [12]. Sex-based NAFLD prevalence was found to be significantly different by several researchers [13]–[17]. Researchers consistently report a higher NAFLD risk for men, when compared with women [13]–[17]. Differences between menopausal women vs pre-menopausal women have also been noted, with postulations of specific hormones inducing a protective effect in women [14], [15]. Though the prevalence of NAFLD is higher in men vs in women, the advancement of NAFLD from simple fatty liver to fibrosis was reported to be faster in women than in men [15]. Due to the complexity of the condition and its association with several factors, it is important to review what is known about the etiology of NAFLD.

### **2.2.2 Etiology of NAFLD**

NAFLD is a chronic condition and can be broadly classified into four stages. In the first stage, the liver starts building up fatty deposits but there are no signs of inflammation [18]. In this stage, the liver shows no symptoms, and the disease could continue to progress undetected. This stage is also called as simple fatty liver or Non- Alcoholic Fatty Liver (NAFL) [18]. In the second stage, signs of inflammation begin to appear as the liver attempts to repair the damaged tissue [18]. If the tissue is not repaired quickly enough to the point of excessive inflammation, liver scarring may occur [18]. At the time of liver scarring, the condition is categorized as stage 3 [18]. However, the liver can continue to function well in this advanced stage [18]. Over a period, excessive presence of scar tissue and low amounts of normal tissue can lead into the final stage called liver cirrhosis [18]. At this point, the liver struggles to function normal and symptoms such as yellowing of the skin, eyes, and a dull ache in the lower ribs appear [18]. The progression can also be seen diagrammatically in Figure 2.2.

The interaction between multiple factors like inflammation, IR, diabetes, obesity, general diet, and lifestyle can have an impact of the progression of the disease. It is known to commonly co-exist with obesity, dyslipidemia, and insulin resistance [19]. Based on multiple studies from various regions (Italy, China, UK) and through meta-analyses, it has been found that NAFLD prevalence is much higher in population with either obesity and/or with type-2 diabetes [9], [19]. In both high-risk groups (obesity and diabetes type -2), the prevalence of NAFLD is roughly double that of the prevalence in normal population [9], [19]. While obesity is a significant risk factor for NAFLD, recent studies have also shown that it can occur in subjects with BMI <25 Kg/m<sup>2</sup> as well [11], [20]. Studies from various regions indicate prevalence rates as follows in those with BMI < 25 kg/m<sup>2</sup>: India (20%) [21], Japan (15.2%) [22], China (15%) [23], Greece (12%) [24], and South Korea (12.6%) [20], [25]. It was found that non-obese subjects with NAFLD are more insulin-resistant than those without NAFLD [26].

Insulin-resistance (IR) was found to be the standalone parameter to determine high risk of NAFLD [27]. While NAFLD can occur in those with IR and in those with hyperinsulinemia, there is a high presence of circulating free fatty acids (FFA) in those with IR [28]. The presence of FFAs in the blood can then lead to an uptake of these FFAs by hepatic cells which further leads to increased gluconeogenesis and decreased storage of glycogen [28]. Further, liver IR was found to be linked to FFA levels in the liver, but not to the levels of visceral fat [29]. Explanations about lean individuals (particularly Asians) with high IR in the liver could be related to the previous finding [30], [31].

Another condition that is often correlated with NAFLD prevalence is Metabolic Syndrome (MS). Several researchers have detailed studies regarding the relationship of NAFLD with MS [27], [28], [32], [33]. Definitions of MS differ slightly between different agencies like the WHO, European Group for the Study of Insulin Resistance (EGIR), National Cholesterol Education Program (NCEP) Adult Treatment Panel III (ATP III) [34], [35]. A definition from ATP III identifies MS when three of the following five conditions are met:

1. High abdominal obesity
2. High triglycerides
3. Low HDL
4. High blood pressure
5. High fasting glucose

While MS and NAFLD have common risk factors, there is no clear indication of which condition occurs first [27], [28], [36]. Further, a difference in the prevalence rates of NAFLD in patients with MS is found based on race and ethnicity [28].

In addition to the disease conditions discussed above, hormones could also play a role in NAFLD prevalence and progression. Several hormones in the body are related to the promotion of obesity and inflammation. Two such hormones derived that are related to NAFLD are a) glucagon-like peptide 1 (GLP-1) and b) Ghrelin [32]. The role of GLP-1 is the activation of reward centers of the brain when fructose and other macronutrients are consumed. On the other hand, Ghrelin concentrations promote hunger.

Reduced secretion of GLP – 1 along with reduced receptors for the same have been found in NAFLD patients [32]. This reduction damages the glucose and lipid metabolism in the liver [32]. The concentration of acylated/deacylated Ghrelin in NAFLD patients was found to be elevated on the other hand [32]. The simultaneous decrease in GLP-1 and increase in Ghrelin could be severely damaging for NAFLD progression.

Overall, of the various chronic liver conditions, NAFLD was found to be one of the major etiologies. Among chronic liver conditions in young adults in United States, 22% of cases were attributed to NAFLD [37]. Similarly, 39.7% of chronic liver cases among adults in India were attributed to NAFLD [38]. NASH, the second stage of the NAFLD condition was found to be a common and increasing etiology of end-stage liver disease in the US [39]. Finally, NAFLD and NASH are growing all around the world. Early detection of the condition and disease management are not only critical but also urgent.

### **2.2.3 Biomarkers and tools for NAFLD detection**

Majority of the findings related to NAFLD biomarkers are still in the research phase. As indicated earlier, there are no established, specific biomarkers for NAFLD currently. However, some researchers have identified potential biomarkers for NAFLD and NASH. Some of those are discussed here.

Adipose tissue was previously thought of as a passive energy storage unit. However, more recent studies have identified the ability of the adipose tissue in synthesizing and releasing hormones and cytokines [40]. Therefore, increasing researchers are now investigating these in the context of NAFLD. Four adipokines: “Leptin, adiponectin, ghrelin, interleukin-6, and tumor

necrosis factor- $\alpha$ ” were found to be associated with NASH [41], [42]. Research is also continuing to investigate the relationship of fatty acid-binding proteins: adipokine binding protein (A-FABP), retinol-binding protein (RBP4), and lipocalin-2 due to their association with obesity, IR, and MS [43]. Fibroblast growth factor 21 (FGF21) is a hormone released by the liver. It has been found to be related with “lowering blood glucose, lipids, and insulin levels, reversing hepatic steatosis, and increasing insulin sensitivity” in individuals [44]. These features have led to investigation of FGF21 as a potential early biomarker for NAFLD [44].

While the research to identify biomarkers for NAFLD is ongoing, it is important to find ways to detect NAFLD in patients in the early stages. Early identification is critical for an increasingly prevalent chronic condition like NAFLD. Due to the complex etiology of the disease and the relationship of multiple other conditions with NAFLD, the use of machine-learning based tools for early detection would be a timely solution.

#### **2.2.4 Machine learning (ML)-based NAFLD detection**

Use of ML/AI tools for healthcare applications is a fast-growing domain. The availability of large amounts of healthcare data and increasing computational power are enabling this domain. Upon surveying the recent literature for ML/AI tools in the context of NAFLD, research related to the following three areas was found: 1) liver fat quantification, 2) fibrosis pattern detection, and 3) assessment of the severity of the liver disease. These research tools use multi-modal data to train the ML/AI models. Overall, four different modalities of data were used, per our literature search. They are:

- i. Imaging modalities (Ultrasound, MRI)
- ii. Omics data (genetics, transcriptomics, metabolomics, etc.)
- iii. Images of liver biopsies (on a microscopic slide)
- iv. Physiological parameters (BMI, age, etc.) in combination with one of the other modalities

Images provide a large amount of data and have the potential to be used for identifying damage in the liver or quantifying the extent of the damage. While there are three different liver imaging tools (Ultrasound, MRI, and Computer tomography (CT)), Ultrasound (US) based images were most used by researchers for training ML/AI models. Further, US tests are of two types: 1) Conventional US (CUS) 2) Quantitative US (QUS). While CUS is available more commonly than

QUS, the use of QUS was found to be more accurate (68.3%) in quantifying steatosis when compared to CUS (51.7%) [45]. Other studies using QUS also reported similar findings [46]– [48].

While some researchers have used ML and other AI tools in the context of NAFLD, most of these tools require images for detection or quantification of NAFLD [45]– [56].

One research study also combined the use of image-data and physiological-data to extract rules for Fatty Liver Disease (FLD) detection using artificial neural networks (ANN) [57]. Although the use of US images shows potential, particularly for quantification of the extent of liver damage, the use of US for screening purposes can be harder to implement. Medical imaging exams are more expensive and less accessible when compared to other, minimally invasive options like blood tests. In low-resource settings, where NAFLD is prevalent, the access to medical imaging tests is not reliable. Under these constraints, the use of physiological parameters that can be obtained using minimally invasive tests (like oral glucose test, blood test etc.) are better suited for screening purposes.

There is a need to develop tools to help doctors in making better recommendation so that patients with HS can be screened at an early stage. To address this need, mathematical models are developed in this research. Recent developments in Machine Learning (ML) technology as described earlier are utilized in this work to screen for HS. The overall goal of this project is to address the need of early HS detection by developing medical decision support tools using machine learning and existing data.

The hypotheses used in this work are as follows:

1. Six physiological parameters: age, sex, BMI, triglycerides, HDL, and total cholesterol relate to NAFLD/HS occurrence in individuals and can therefore be used to predict HS using ML.
2. The liver functionality parameters (ALT, AST, ASP) in addition to some physiological parameters (Age, BMI, HDL, plasma-glucose) can improve HS prediction.

To address the goal described above, two different objectives were developed and tested. The methodologies for each objective were independently developed and analyzed.



## **2.3 Objectives**

1. Objective 1A: Develop and explore the performance capability of a ML model in predicting HS in using only physiological parameters.
2. Objective 1B: Develop and evaluate a HS screening model to be used as a clinical decision support tool using physiological and liver biochemistry parameters.

## **2.4 Methods**

### **2.4.1 Objective 1A - methods**

The NHANES III data were processed relevant to the hypothesis of this objective. In the sections below, detailed data processing steps, statistical model selection, model development and pseudo code used in the model are provided.

#### **a. Dataset description**

The dataset from the third National Health and Nutrition Examination Survey (NHANES III) was used in this research [58]. It contains data from the USA (N = 33,994) for individuals older than 2 months. The data are organized in four categories: “NHANES III Household Adult, NHANES III Household Youth, NHANES III Examination, and NHANES III Laboratory” [58]. Additional files regarding the “Hepatic/Gallbladder Ultrasound and Hepatic Steatosis (HGUHS)” were also used from NHANES III [59]. For hepatic steatosis (HS) assessments, the liver was grouped as “normal”, “mild”, “moderate”, or “severe hepatic steatosis” [59]. The HS variable was recoded by the data providers (NHANES III) using ultrasound (US) exams and double radiologist reviews to determine the presence or absence of HS in ages between 20 - 74 [59].

The output variable used in this research was hepatic steatosis (HS). The NHANES III Household Youth data (age range 2 months - 16 years) were specifically excluded, and only ages greater than 20 years were included in this research. Alcohol related variables from the dataset were processed to exclude individuals who consumed > 7 drinks per week, for women and > 14 drinks per week, for men. The data used here were from NHANES III. While the NHANES III website provides sample weights [60], they were not used in this research. Other researchers who used NHANES data have also chosen not to use sample weight adjustment [61]– [63]. Therefore,

in this research, it is assumed that the impact of not using sample weights is minimal. The use of the NHANES III dataset for the current study was approved by Purdue Institutional Review Board (IRB) (PROPEL # 17975020). A copy of the IRB approval is in Appendix G.

### **b. Data processing**

Data from NHANES III were first read into SAS software [64] as four different files. Data features relevant to this research were retained and the other features were discarded. See code in Appendix A.1 for details. Each observation in the dataset is provided (by NHANES III) with a unique sequential number (SEQN). These SEQN numbers were used to combine the data from the four different files into one file using a customized SAS code (Appendix A.2). Observations with missing information pertaining to alcohol consumption were eliminated using the code in Appendix A.3. The reduced dataset was then exported out of SAS and imported into MATLAB [65].

Additional processing was performed on the data in MATLAB using a customized code [65]. The imported data had a size of 17,704 x 11. After importing, any missing values were re-labelled to indicate “Not a Number (NaN)” for ease of data representation in MATLAB. After eliminating observations with any missing data and applying alcohol related exclusions, the dataset size reduced to 8,703 samples. Six predictor variables (predictors) were selected and used in this study. The predictors were: Age, sex, BMI, triglycerides, HDL and, total cholesterol. These parameters were selected based on previous literature and their relationship with NAFLD. A summary of the methods used in this paper are summarized in Figure 2.3. Additional details about the MATLAB code are explained in the sections to follow.

### **c. Model selection**

The following ML model families were selected based on the binary nature of the output variable and the size of the dataset in this study. The details regarding each of these model families are further elaborated in the following sections.

- i. Support Vector Machines (SVM)
- ii. Bagged trees
- iii. Boosted trees

Within these model families, a total of five models were trained and tested (Fine gaussian SVM, medium gaussian SVM, bagged trees, gentle boosted tree and ADA boosted tree).

### ***Support Vector Machines (SVM)***

SVM are a family of supervised ML techniques which separate output classes using an optimal-hyper plane. The hyper-plane may be linear or non-linear. SVM are particularly useful when the output class is binary (for example in this study: HS/no-HS). SVM are also appropriate to be used when the datasets are mixed in nature [66], [67]. Mixed datasets include data that is a continuous parameter (For example: Age), and data that is discrete or categorical in nature (For example: Sex). Considering that the six selected input features are a mixture of continuous and discrete parameters, the use of SVM is well-suited for this study.

The governing equations for SVMs are different based on the separability of the data. For linearly separable data, the equations are as shown in (1) and (2). A set of inputs are indicated by  $i'_j$ , their corresponding output categories by  $o'_j$ , and dimension by  $d$ . Since the SVMs are being used for binary classification,  $o_j = \pm 1$ .  $\alpha$  is a vector with coefficients that are orthogonal to those of the separating hyperplane and  $c$  is a constant [66]–[68]. The equation of a hyperplane to separate one category of inputs from the other would then be indicated by equation (1).

$$f(i) = i' \alpha + c = 0 \quad (1)$$

To optimize the separating hyperplane, the following constraint is used (shown in [2]) [66]–[68]. This constraint ensures that the distance between the different output classes is maximized.

$$o_j f(i_j) \geq 1 \quad (2)$$

In case of data that cannot be linearly separated, a soft margin is used. The mathematical description of soft margin is shown in equation (3). It is also called the ‘Kernel Trick’. In the equation,  $\hat{c}$  indicates the estimate of the bias,  $\hat{\beta}_j$  indicates the  $j^{th}$  estimate of the vector  $\hat{\beta}$  and  $G(i_j, i_k)$  provides a result from the ‘Gram Matrix’ which is calculated using an inner product of  $\phi(i_j), \phi(i_k)$  where  $\phi$  is the kernel function [66]–[68]. The kernel function varies based on the specific kernel used.

$$\hat{f}(i) = \sum_{j=1}^n \hat{\beta}_j o_j G(i_j, i_k) + \hat{c} \quad (3)$$

$\beta_j$  is subject to an additional constraint shown by equation 4 [66]– [68]

$$0 < \beta_j < \hat{c} \quad (4)$$

In this work, a gaussian kernel was implemented. The equation specific to a gaussian kernel is shown in equation (5)

$$G(i_j, i_k) = \exp\left(-\|i_j - i_k\|^2\right) \quad (5)$$

### ***Bagged trees***

Bagged and boosted tree methods are ensemble algorithms that involve growing a group of decision trees and aggregating their results. The trees are also called “learners”. A split criteria needs to be used for growing a decision tree. In this work, “Gini’s Diversity Index (GDI)” was used [69]. GDI splits the nodes of any decision tree based on a condition. The condition is mathematically described in equation (6). ‘ $i$ ’ is the total number of groups for any decision and  $f(i)$  represents the groups that match with group  $i$  and arrive at the node.

$$GDI \text{ of a node} = 1 - \sum_i f^2(i) \quad (6)$$

Bagged trees algorithm is also called random forest, because it groups or bags random predictor selections at each split. The number of splits is capped at one less than the number of observations in this algorithm. The number of predictors is decided using the square root of the total number of predictors. This formula allows bagged trees algorithm to have deeper trees when compared to other tree-based algorithms [69].

### ***Boosted trees***

Boosted trees are like bagged trees as both algorithms are ensemble methods based on generating decision trees. The boosted tree methods also use GDI to split nodes in its trees, but in this case, the number of splits is limited to 10. The limit leads to multiple shallow trees, as opposed

to deep trees in bagged trees algorithm. Two types of boosted trees were implemented in this research: 1) ADA boost 2) Gentle boost

The ADA boost algorithm trains the trees (or learners) one after the other, in a sequential pattern. The weighted classification error ( $\varepsilon_i$ ) for each tree, 't' is computed using the equation (7) [69]. The error is summed up over 'n' observations. The prediction made by each tree is represented by  $h_t$  and the weight of each observation is represented by  $d_n^{(t)}$ . The variable I is used as an indicator.

$$\varepsilon_i = \sum_{n=1}^N d_n^{(t)} I(t_n \neq h_t(p_n)) \quad [7]$$

After training is completed, predictions are made using equations (8) & (9) [69]

$$f(p) = \sum_{t=1}^T \alpha_t h_t(p) \quad [8]$$

$$\alpha_t = 0.5 \log \left( \frac{1 - \varepsilon_t}{\varepsilon_t} \right) \quad [9]$$

$\alpha_t$  is the weight associated with the weak trees (learners) in the group of trees [69]?

Gentle boost algorithm uses a combination of ADA boost and Logit Boost. The first part of gentle boost is same as that of ADA boost. That is the loss function is the same in both gentle book and ADA boost. But in gentle boost, the weak learners are fitted into a regression model. The output of the regression model forces the observations to be classified into one of two groups: [-1, +1]. The mean squared error for Gentle boost is as shown in equation (10). Again, the variable  $d_n^{(t)}$  represents the weight of the observation at step 't'. and  $h_t(p_n)$  represents the prediction of the regression model at response  $o_n$  [69].

$$\varepsilon_i = \sum_{n=1}^N d_n^{(t)} (o_n - h_t(p_n))^2 \quad (10)$$

Both the bagged and boosted trees are suitable for use with classifying multi-dimensional datasets and were therefore used in this research.

#### **d. Model development**

As discussed in the general literature review section of this dissertation, there are multiple challenges in developing ML models, specifically in the healthcare domain. Two such challenges were encountered during the development of this objective. The first challenge is inherent to all healthcare datasets called ‘Class Imbalance’. The second challenge is small datasets. Both challenges were tackled in this project and the methods used are described in the section below.

#### **e. Challenges**

##### *Class imbalance:*

Health care datasets are inherently class imbalanced, i.e., most observations in any given population dataset fall in the “normal” or no-disease category, whereas a minority of observations fall in the “disease” category. Training ML models using imbalanced data can lead to skewed or biased results in the favor of no-disease [70]. The idea of screening for disease cases can therefore be completely missed if trained with imbalanced data. Use of balanced training and test datasets is therefore recommended [70].

In this research, after the initial data processing, the class distribution was as follows: 2,008 observations with HS (disease category) and 6,695 observations with no-HS (no disease category). To balance the datasets, a combination of two separate statistical techniques was implemented in this research. Minority class data (disease category) were synthetically generated and combined with existing data by implementing the Synthetic Minority Oversampling Technique (SMOTE) [71]. However, typical SMOTE implementation is implemented in datasets with all discrete or all continuous features [71]. The distance metric used within the SMOTE algorithm is the bottleneck for mixed features. Since the data used in this research had mixed features, as explained earlier, a typical SMOTE implementation was not appropriate.

Therefore, in this research, a different distance metric called “Gower’s Distance” [72] was implemented in combination with the SMOTE algorithm. A pseudo code of the implemented algorithm is presented in the section below. Detailed code is available in Appendix A.

#### f. Pseudo code for objective 1A

*Start of code*

Input: Original dataset of size: 8,703 x 6 observations

Output: Synthetically generated disease class samples

1. The original dataset was split into sub-datasets based on the output category (disease (HS), no-disease (No-HS)). See code in Appendix A.4 for details.
  - a. Disease class (HS): 2,008 x 6
  - b. No-Disease class (No- HS): 6,695 x 6
2. The disease class sub-dataset was used as input to synthetically generate more disease class data. The following parameters were used:
  - a.  $N = 2$  (i.e., 200% data was synthetically generated) (
  - b.  $K = 2$  (number of nearest neighbors used)
  - c.  $T = 2,008$  (number of diseased samples)
  - d. Sample  $(m, n) = 2,008 \times 6$  (2D array for original disease samples)
  - e. New Index = 4,016 (Empty variable to keep a count of new samples generated)
  - f. Synthetic  $(o, n) = 4,016 \times 6$  (2D array for synthetic samples)
3. Using the above variables, Gower's Distance [72] was computed between each observation ' $i$ ' for each other observation ' $j$ ' and every feature/attribute (' $attr$ ') using equation (11) below [72].

$$GD(i) = \frac{\sum_{attr}(1 - |sample(i, attr) - sample(j, attr)|) + a}{number\ of\ continuous\ attributes + (1 - d)} \quad (11)$$

where,  $a = +1$ , when binary attributes  $i$  and  $j$  match, else 0

$d = +1$ , when binary attributes  $i$  and  $j$  do not match, else 0

4. After computing the distances, they were sorted in ascending order to find the neighbors (nearest distances). Distance of an observation from itself was ignored.
5. The distances were used to generate synthetic data using SMOTE algorithm [71], applied separately to continuous and discrete features.
  - a. For continuous features, equation (12) was implemented

$$Synthetic(n, attr) = sample(i, attr) + dif * gap \quad (12)$$

where,

*dif = a randomly picked difference between sample (i, attr) and one of its two nearest neighbors (since  $K = 2$ )*

*gap = any random number between 0 and 1*

b. For binary features, equation (13) was implemented

*Synthetic (n, attr)=majority of sample(i, attr) and two nearest neighbors (13)*

6. After completing the synthetic generation at 200%, 4,016 synthetic disease class samples were combined with the original 2,008 disease class data. The new disease class dataset was now at 6,024 observations. The no-disease class dataset was at 6,695 samples (same as before)
7. The data were then divided into 70:30 ratio in a class-balanced manner to obtain training and test datasets
8. Five chosen ML models were then trained using the training datasets, including 5-fold cross validation. Detailed code is available in Appendix A.5
9. The trained models were tested using the separate test dataset and their performances were analyzed. Results regarding model performance are provided in the Results (Objective 1A) portion.

*End of code*

#### **2.4.2 Objective 1B - methods**

NAFLD is impacted by multiple risk factors as explained earlier. While there are no specific biomarkers for NAFLD yet, there is a liver function test (LFT) available to assess the general health of the liver. In current clinical practice, LFTs are used to understand liver functionality and identify if an injury to the liver is hepatitic or cholestatic in nature [73]. It measures multiple liver enzymes and other hormone levels, and these values are examined by clinicians who then identify the nature of the injury [73]. Specific liver enzyme ratios are also identified and elaborated in clinical literature to narrow down a suspect or cause for the liver injury [73].

In this research, specific liver biochemistry data from the liver function tests was utilized to train ML models. These models were developed and assessed with the potential to be used as decision support tools for HS screening. The model development is broken down in specific stages and elaborated in the sections to follow.



As explained in the literature review portion (NAFLD background and epidemiology), the condition is known to impact men and women in different ways [12]– [14]. While research in this domain is ongoing, it is unclear exactly why the sex disparity exists [12]– [14]. Researchers have identified hormones to be potentially linked with the pathobiology of the disease, with some research suggesting a protective effect of estrogen against NAFLD [13]. Since hormonal data (or other such parameters) were not used in this research, HS screening models were developed separately for male and female populations, considering the difference in pathobiology of NAFLD.

In this research, the following features were chosen to be used as inputs to train the ML models: Age, BMI, HDL (high density lipids), plasma glucose, AST (aspartate aminotransferase), ALT (alanine aminotransferase), and ASP (aspartate transaminase). These parameters were chosen based on clinical literature and their link with fatty liver disease [9], [73], [74]. The sex feature was used to separate out male and female populations into sub-datasets but was not used thereafter. The output feature of HS was used, same as that in Objective 1A.

#### **a. Data processing**

Adult data from the NHANES III dataset [58], [59] were used in this research objective as well. Data was again extracted using SAS and then further processed using MATLAB. Initial data processing like retaining parameters of interest and merging multiple datasets into one were conducted in SAS [64]. The SAS code is presented in detail in appendices B.1 and B.2.

After initial processing in SAS, the data was exported into MATLAB [65] for further processing. In this step, any observations with missing data were deleted. Alcohol related exclusions were applied. Men who consumed  $> 2$  alcoholic drinks/day and women who consumed  $> 1$  drink per day were excluded. The size of the available data was considered, and next steps were identified. After applying the necessary exclusions and eliminating missing data, the dataset size reduced from 20,050 to 9,619 samples.

To aid the model with better learning, data normalization was applied to four features: ALT, AST, BMI, and Plasma glucose. Four normalized features were derived from this process, called: ALT%, AST%, BMI% and Plasma glucose%. The formula used for normalization is shown in equations (14) – (17). Normal values for ALT, AST, BMI and Glucose were used from existing literature [75]– [78].

$$ALT_i\% = \frac{ALT_i - ALT_{ULN}}{ALT_{ULN}} \times 100 \quad (14)$$

$$AST_i\% = \frac{AST_i - AST_{ULN}}{AST_{ULN}} \times 100 \quad (15)$$

$$BMI_i\% = \frac{BMI_i - 25}{25} \times 100 \quad (16)$$

$$Plasma\ glucose_i\% = \frac{Plasma\ glucose_i - 120}{120} \times 100 \quad (17)$$

Where:

$i = i^{th}$  sample in the dataset

$ULN$  = Upper limit of normal

$ALT_{ULN}$ : 33U/L - male, 25 U/L – female [75], [76]

$AST_{ULN}$ : 30 U/L – male, 20 U/L – female [75], [76]

After normalizing the data using the clinical normal values, the negative values ( $< 0$ ) in the derived variables were recoded as zero (indicating normal) and the positive values were retained as is – indicating a deviation from normal. The normalization was applied separately to male and female datasets. The clinically defined parameters were different between the sexes, and they were used as such in this research. The derived features along with the other input features were used in training the ML models.

## **b. Model selection**

In addition to the three model families discussed in objective 1A (SVM, Bagged Trees, and Boosted Trees), two additional ML families were explored in this research. The two additional families were: K-nearest neighbors (KNN) and Logistic Regression.

### ***K-Nearest Neighbors (KNN)***

KNN is a distance-based supervised ML algorithm. It is a commonly used technique that works with classification or regression problems. In this case, the output is a binary classification problem, therefore a KNN-classifier was implemented. The distance metric used in this research was Euclidean Distance. If ' $I$ ' is an input matrix, treated in terms of row vectors  $i_1, i_2, i_3, \dots i_a$  and ' $O$ ' is an output matrix treated as row vectors of  $o_1, o_2, o_3, \dots o_a$ . Then the pair-wise Euclidean distances between different input and output points are defined using equation (18) [79].

$$ed_{mn}^2 = (i_m - o_n)(i_m - o_n)' \quad (18)$$

The distance between different observations is sorted and then used by the KNN algorithm to classify observations into one of the output classes.

### ***Logistic regression***

Logistic regression is a form of linear regression, applied to classification problems. It is the simplest ML algorithm and is often used as a benchmark method to compare with other, more complex ML algorithm performances. Like simple linear regression, logistic regression associates weights with each input feature (or independent variable) to fit the data and find the output value (or dependent variable) accurately. However, unlike linear regression where the output is a continuous, numerical value, the output of logistic regression is categorical in nature. In this research, binary logistic regression was used. Binary logistic regression uses the sigmoidal or logit function to then map the output variable from numerical range to a binary value. The sigmoidal function expression is shown in equation (19). Here,  $s(x)$  represents the sigmoidal value of any input 'x'. The output of the sigmoidal function always lies between 0 and 1.

$$s(x) = \frac{1}{1+e^{-x}} \quad (19)$$

The overall equation for logistic regression involving 'i' input and 'o' predicted output can be described as shown in equation (20).  $b_0$  is the term representing bias and  $b_1$  represents the weight of the input feature. While logistic regression is a linear method, the output predictions are transformed using a logit function.

$$o = \frac{e^{b_0+b_1i}}{1+e^{b_0+b_1i}} \quad (20)$$

Five ML families in total were evaluated in this research objective, as explained earlier. Details related to how class imbalance was handled, and the following results are outlined below.

### c. **Class imbalance**

Similar to that in objective 1A, class imbalance was found to be a recurring challenge with in pursuing this research objective as well. The data distribution between the classes was as follows (combined data from both sexes):

- i) Observations with HS: 2, 956
- ii) Observations without HS: 9,959

To avoid class imbalance, two different approaches were implemented and compared in this research. The first approach called ‘under-sampling’ is discussed here. The second approach was to synthetically generate minority samples using SMOTE (similar to objective 1A). Methods and results from both these approaches are discussed here. Objective 1B – Under Sampling refers to the first approach and Objective 1B – SMOTE refers to the second approach used to tackle class imbalance. The flowchart of the methods used is shown in Figure 2.4 below for clarity.

#### ***Under-sampling***

This approach is different from synthetic data generation. In this method, existing data is used to balance out the classes. As a result, using this approach, only real-world data is used to train and test the ML models. In this technique, data from the majority class (no-HS) is randomly sampled to match the size of the minority class (HS). This way, the class sizes are balanced but the overall size of the dataset is reduced significantly. In this case, the under sampling was applied separately to male and female datasets. The results from under sampling are shown in below in Tables - objective 1B

Table 2.2. Code is available in appendices B.4A and B.4B (for male and female, respectively)

#### ***Synthetic data generation using SMOTE***

Although the under-sampling approach only uses real-world data, it can severely reduce the size of the dataset. Therefore, an alternate approach of generating synthetic data using SMOTE was also implemented and the results were compared with those from under sampling. 200% synthetic data generation was implemented for each male and female dataset. Size of the dataset

increases in this approach due to imputing the data using synthetic methods. The sizes of the specific datasets are shown in

Table 2.3. Detailed steps in the process of creating synthetic data are explained using flowcharts in Figure 2.5 Figure 2.6. Code is available in appendices B.6A and B.6B (for male and female, respectively)

#### **d. Model development**

The male and female processed datasets were divided into separate class-balanced training and test in a 70:30 ratio, respectively (refer to Figure 2.5 Figure 2.6). The models were trained on 17 ML-based models (belonging to the five model families) using 11 features (7 selected features + 4 derived features). The trained models were then used for testing the results using separate test datasets. Code is available for Objective 1B – under sampling in Appendix B.4. and P1.B.4.B Code for objective 1B – SMOTE is available in Appendix B.6.A and P1.B.6.B. In this case, five models of support vector machines (SVM) family were found to perform better among the 17 ML models. Code related to training and testing is the same for both under-sampling and SMOTE approaches. This code is available in Appendix B.4. A summary of the best performing models is provided in Table 2.4 below.

Overall, compared to objective 1A, additional data processing was applied in this research objective for two reasons:

- 1) To enhance the use of liver biochemistry by incorporating clinically defined “normal” values
- 2) To incorporate the differences in NAFLD pathobiology between male and female populations, these datasets were processed separately. This encourages the model to learn from these two datasets independently
- 3) To compare the results by implementing two types of data imputation (to handle class imbalance)

The relevant model performances and the discussion of these results is presented in the Results and Discussion – 1B section.

*The results and discussion have been evaluated independently for Objectives 1A and 1B.*

## 2.5 Results and discussion

Three families of machine learning models were trained and tested on the dataset described in section 2.4.c. Each model was run independently, and each run was repeated 10-times for robustness. Then, the model's performances were measured on the following four parameters:

1. Training Accuracy
2. Test Accuracy
3. Test Sensitivity
4. Test Specificity

Training accuracy is a measure of the model's accuracy in predicting HS vs no-HS while using 70% of the entire data (stored and used as training data). The 10- fold cross-validation (commonly known as k-fold validation) is used to compute the training accuracy as follows. The training dataset is split up into 10-sub datasets. Each ' $k^{th}$ ' sub dataset is used to compute the accuracy by measuring the number correctly classified observations in the kth dataset divided by the total observations in the  $k^{th}$  dataset. The accuracy over each of the 'k' folds is computed and averaged to obtain the training accuracy. The training accuracies of the best performing models in objective 1A are reported in Table 2.1.

Test accuracy is similarly computed but using 30% of the entire dataset (stored separately and labelled as test dataset). It is important to note that none of the data from the test dataset were used in the training or cross-validation process. The test accuracy can be defined as the ratio of correctly classified observations divided by the total number of observations in the test dataset.

Test sensitivity is computed using test data only. The model's ability to identify the observations with HS correctly is measured using sensitivity. In more general terms, model sensitivity can be defined as the ability of the model to correctly predict true positives. Test sensitivity is computed by the ratio of number of true positives divided by the number of disease class observations.

Test specificity on the other hand measures the model's ability to predict true negatives. In this case, test sensitivity is used to inform the models prediction ability in identifying observations with no-HS. Test specificity is computed by the ratio of number of true negatives divided by the total number of non-disease observations.

In general, when screening models are used in clinical settings, the primary idea is to find those candidates with the disease so that they can be provided with the appropriate

treatment/further testing. Therefore, in this work, the focus is more on sensitivity compared to specificity. While both the parameters are measured and the results are provided, the emphasis of model performance is focused on sensitivity and the general test accuracy as well.

### **2.5.1 Objective 1A – results and discussion**

Of the five models, a maximum accuracy of approximately 79% and a maximum sensitivity of 77% was found. Overall, the model sensitivities ranged from 69 – 77% whereas the specificities ranged from 72 – 82%. The test accuracies ranged from 71 – 79%. Detailed results are presented in Table 2.1.

The best performing model (in terms of highest test accuracy) was found to be the ‘Gentle Boosted Tree’ with a test accuracy of 79%. It also had a high specificity of almost 82%. The sensitivity of this model was approximately 76%. While the gentle boosted tree model was the best in terms of both test accuracy and specificity, the best model when considering sensitivity was ‘bagged trees. The bagged trees model had a sensitivity of approximately 72%. It had a specificity of 79% and test accuracy of almost 78%.

Overall, the tree-based models were found to perform well in predicting HS using the six predictor variables: age, sex, BMI, triglycerides, HDL, and total cholesterol. Further work is needed to improve the performance of the models either by using other AI tools or by processing the features to aid with learning. These models also need to be validated on a much larger dataset with demographics that are representative of the population. The data used here was from NHANES III. While the NHANES III dataset website provides sample weights, they were not used in this research. Additional analyses with weighted data and/or larger datasets are required before the models can be implemented in clinical settings.

### **2.5.2 Objective 1B - Results and Discussion**

#### ***Results (objective 1B – under-sampling)***

Five families of ML models (17 models total) were assessed in this research objective. Using the first approach of under-sampling, the majority class data were randomly sampled to match the size of the minority class data. This method significantly reduces the size of the dataset, but in this case, the data were all real-world and no synthetic data generation was required in this

step. The under sampled dataset (size: 2,076 Male; 2,424 Female) was then divided into training and test datasets in a 70:30 ratio, respectively. Dataset sizes before and after under sampling are provided in

Table 2.2.

In this research objective, the models were developed separately for each sex to accommodate the potential differences in pathobiology, as explained earlier. Five best performing models for each sex were identified and the results are in

Table 2.4. As with objective 1A, the four parameters, training accuracy, test accuracy, test sensitivity and test specificity were used as model performance indicators.

Overall, the best performing male-specific models demonstrated test accuracy ranging from 66 – 69%, sensitivity ranging from 63 – 72% and specificity ranging from 61 – 72%. The best model in terms of test accuracy was found to be Gaussian SVM scale II (69%), whereas the best model in terms of sensitivity was found to be Gaussian SVM scale I (72%). The sensitivity of the prediction models is emphasized in this work, as this metric identifies the model's ability to find those with the disease. Therefore, after conducting additional validation with larger, diverse datasets, the model with highest sensitivity would be recommended. In this case, the Gaussian SVM I model resulted in the highest sensitivity.

Similarly, the performance of the female-specific models was as follows. The test accuracy ranged between 69 – 71%. The sensitivity ranged from 67-71% and specificity from 68 – 75%. The model with highest test accuracy (69%) was observed in the Quadratic SVM model and the highest sensitivity (71%) was found in the Gaussian SVM I model. While the Quadratic SVM model also had a high specificity of 75%, the emphasis of this objective is on sensitivity as explained earlier.

Overall, in terms of high sensitivities, the Gaussian Scale I SVM model was found to be the best performing for both Male and Female – specific models. Additional investigation is needed to understand the model performance. Future direction might include improving the Gaussian Scale I model further using other statistical techniques/methods.

### ***Results (objective 1B – SMOTE)***

A second approach to handling class imbalance for data in objective 1B was implemented. SMOTE was used to synthetically generate HS data and augment the original datasets instead of



under sampling. The data were then split into training, test and the performances were measured. A summary of the logic used for SMOTE implementation in male and female datasets can be seen in Figure 2.5 and Figure 2.6.

Creating synthetic data and imputing the existing datasets increased the male dataset size from 2,076 to 6,291. Similarly, the female dataset size increased from 2,424 to 9,439. Use of SMOTE instead of under-sampling to handle class imbalance, increases the size of the original dataset. Increased dataset sizes provide more training data for the model to learn from. After the data imputation, the rest of the ML-based training and testing was conducted.

The model performance summaries using SMOTE are shown in Table 2.5. The best performing models were identified as those with high performance across all four metrics (training accuracy, test accuracy, sensitivity, and specificity). These models perform well over-all the measured metrics and are considered to be best performing in this work. For male-specific models the results indicate that the test accuracy ranges between 71- 77%, sensitivity ranges between 70 – 76% and specificity ranges between 71 – 79%. Best performing model inters of test accuracy was found to be the Bagged Trees model with 77% accuracy, 76% sensitivity and 79% specificity.

Similarly for female-specific models, the ranges of performance were as follows: Test accuracy 73 – 82%, sensitivity 74 – 81% and specificity 71 – 79%. Best performance in terms of test accuracy was found in the Bagged trees model as well with 82% test accuracy, 81% sensitivity and 82% specificity.

Note that the best performing model across all metrics was found to be Bagged Trees for both male and female-specific models. However, the performance of the female-specific Bagged Trees model was higher than its male counterpart by about 4.4% in test accuracy and 5.5% in sensitivity. It is postulated that the increase in accuracy for the female-specific models could be driven by the larger dataset size for the female population (9,439) vs the male population (6,291). More testing with much larger datasets is required to confirm this postulation and is out of scope for this research.

Since the focus of screening models is to maximize sensitivity, the models that perform well with respect to sensitivity alone were identified. Some of these models might have very poor specificity performances but they are presented here due to their high sensitivity results. The models that perform well when only sensitivity is considered are in Table 6. The weighted KNN model resulted in a sensitivity of approximately 83% for male-specific and 86% for female-

specific models. Other models like Fine KNN and Fine Gaussian SVM also showed high sensitivity results. When creating screening tools, often the focus is on sensitivity – to identify positive cases and recommend them for further testing. In screening for HS, the models that perform well with respect to sensitivity might be useful and therefore are included in this section.

On comparing the results of under-sampling with those of SMOTE, a clear increase in performance with SMOTE is observed. Test accuracy performances of the models using SMOTE were 8% and 11% higher in male and female specific models, respectively, when compared with those using under sampling. Similarly, the sensitivity performance was 11% (Male) and 15% (Female) higher with SMOTE vs under-sampling approach. Sensitivity was 5% (Male) and 7% (Female) higher as well.

This increase in performance across all metrics can be attributed to the increase in dataset sizes significantly when using SMOTE instead of under sampling. Larger dataset sizes are typically considered to be better for training ML models as more data provides more opportunity to learn. In summary, while both the techniques were separately implemented to handle class imbalance, the SMOTE approach resulted in higher performance across Male and Female -specific models. Additional testing and validation of these screening tools is recommended. The tools developed in this work show promising results and could be used in the future as potential decision support tools for clinicians to screen for HS.

## **2.6 Summary & conclusion**

In this research, models were developed to predict Hepatic Steatosis (HS) using ML-techniques. In developing decision support systems based on ML for HS prediction, a hierarchical approach was used in exploring different input parameters. Models developed as part of objective 1A used only six physiological parameters. The models in objective 1B used seven physiological and liver biochemistry parameters.

The maximum accuracies of models using physiological, and liver biochemical parameters were 77.2% and 81.6% for male and female, respectively (using the SMOTE approach). Maximum sensitivities were 75.8% and 81.3% for male and female, respectively. Maximum specificities were 78.6% and 81.9% for male and female, respectively. When sensitives alone are considered, the models developed in the SMOTE approach (obj 1B) out-perform other models developed in using only physiological parameters (obj 1A) and under sampling approach (obj 1B). The

weighted KNN model resulted in sensitivities of 82.6% (male) and 86% (female) using only seven input features.

Finally, the models developed in this work need more validation on a much larger and much more diverse dataset. These models need to be tested for robustness before they can be implemented in clinical settings. However, the work in this dissertation shows promising results and a potential for the use of such screening tools, especially when no specific clinical screening is available. Early detection of NAFLD can potentially lead to remission or halt the progression of the disease. Therefore, low-cost, early detection tools are crucial to handle the increasing NAFLD condition.

Based on the work described above, the following are concluded:

1. Physiological parameters alone can predict HS using 79% accuracy, 76% sensitivity and 82% specificity
2. Models with only seven parameters (vital and liver biochemical) led screening models with sensitivities of 82.6% for male-specific models and 86% for female specific models. It is logical to use both physiological and liver biochemical parameters to maximize the sensitivity and therefore, screening capability of these models.

## **2.7 Recommendations for future work**

Based on the reported work, the following are recommended as potential future work.

1. Testing and validation of the developed models using larger and diverse datasets is recommended.
2. Utilizing the sample weights provided by NHANES and developing models using weighted observations are recommended.

## 2.8 Figures – objective 1A

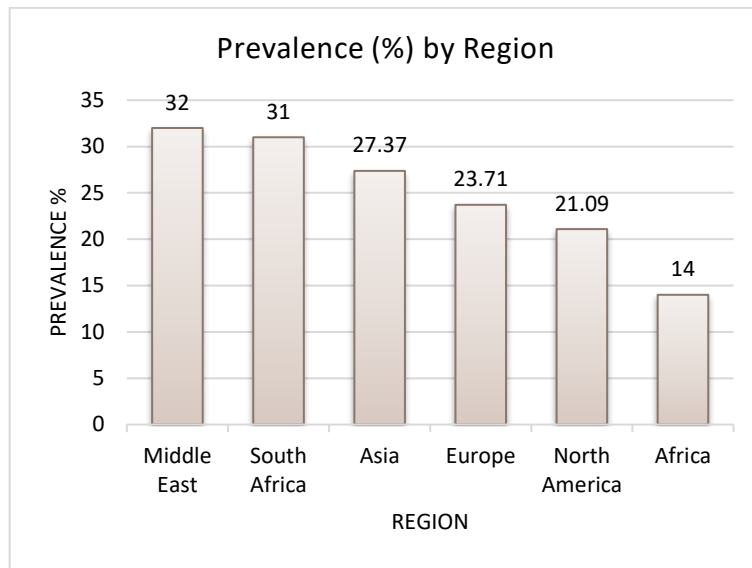


Figure 2.1: Global NAFLD prevalence [7]

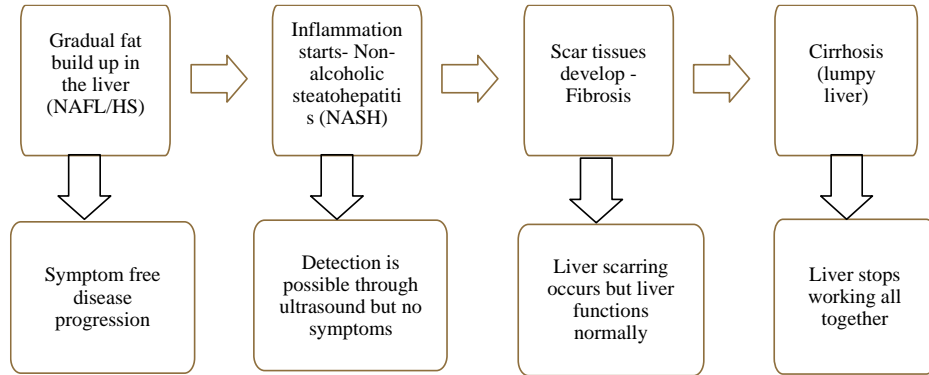


Figure 2.2: Progression of NAFLD [18]

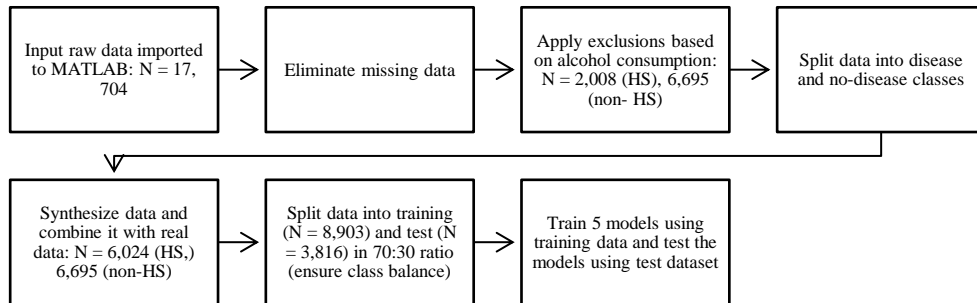


Figure 2.3: Summary of the methods used for data cleaning and model training

## 2.9 Tables – objective 1A

Table 2.1: Model performance summary - objective 1A

Models	Accuracy (%)	Sensitivity (%)	Specificity (%)
Fine Gaussian SVM	76.58	75.76	77.35
Medium Gaussian SVM	71.06	69.24	72.72
Bagged Trees	77.96	76.62	79.14
Gentle Boosted Trees	79.03	75.88	81.86
ADA Boosted Trees	71.24	70.58	71.83

## 2.10 Figures – objective 1B

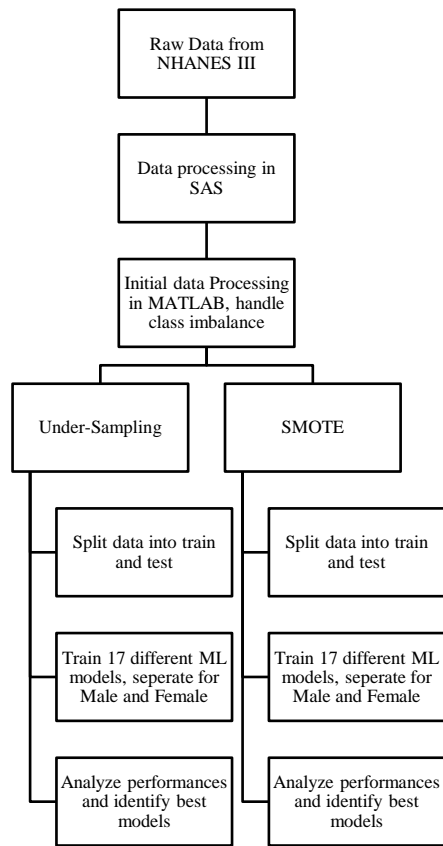


Figure 2.4: Flowchart of methods used in Objective 1B

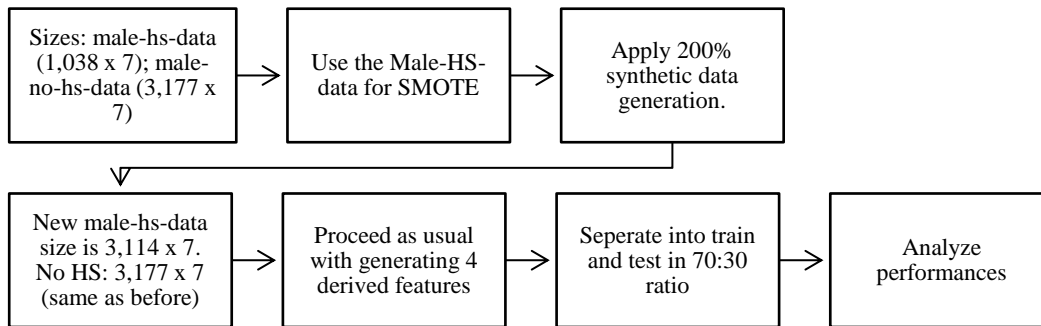


Figure 2.5: Logic used for creating synthetic male HS data with SMOTE

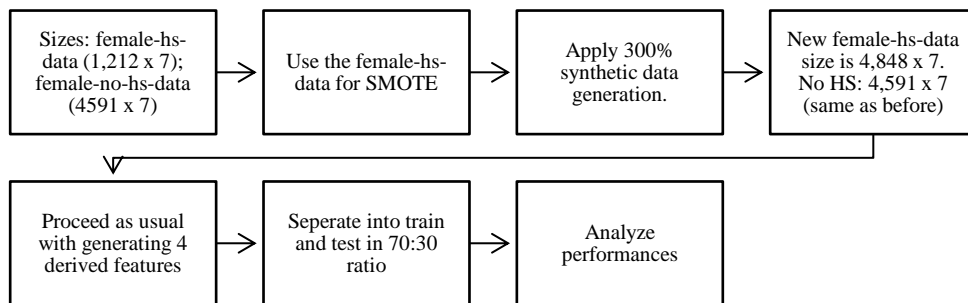


Figure 2.6: Logic used for creating synthetic female HS data with SMOTE



## 2.11 Tables - objective 1B

Table 2.2: Class balanced datasets using under-sampling

Sex	HS	No-HS (before under sampling)	No-HS (after under sampling)	Total (after under sampling)
Male	1,038	3,177	1,038	2,076
Female	1,212	4,591	1,212	2,424

Table 2.3: Class balanced datasets using SMOTE

Sex	HS (before SMOTE)	HS (after SMOTE)	No-HS	Total
Male	1,038	3,114	3,177	6,291
Female	1,212	4,848	4,591	9,439

Table 2.4: Model Performance Summary for HS Screening using Under-Sampling

SVM Models	Performance Metrics							
	Training Accuracy (%) $\pm$ SD		Test Accuracy (%) $\pm$ SD		Test Sensitivity (%) $\pm$ SD		Test Specificity (%) $\pm$ SD	
Sex	M <sup>a</sup>	F <sup>b</sup>	M	F	M	F	M	F
Linear	69.360 $\pm$ 0.007	71.385 $\pm$ 0.011	68.553 $\pm$ 0.011	70.865 $\pm$ 0.016	65.852 $\pm$ 0.022	68.104 $\pm$ 0.026	71.254 $\pm$ 0.026	73.626 $\pm$ 0.023
Quadratic	68.954 $\pm$ 0.009	71.362 $\pm$ 0.010	68.778 $\pm$ 0.014	71.002 $\pm$ 0.014	65.562 $\pm$ 0.02	66.620 $\pm$ 0.027	71.993 $\pm$ 0.021	75.384 $\pm$ 0.027
Gaussian scale 1	66.334 $\pm$ 0.010	69.504 $\pm$ 0.013	66.559 $\pm$ 0.011	69.148 $\pm$ 0.011	72.122 $\pm$ 0.023	70.659 $\pm$ 0.02	60.996 $\pm$ 0.028	67.637 $\pm$ 0.021
Gaussian scale 2	69.133 $\pm$ 0.009	71.462 $\pm$ 0.011	68.987 $\pm$ 0.018	70.659 $\pm$ 0.012	66.463 $\pm$ 0.015	67.032 $\pm$ 0.02	71.511 $\pm$ 0.027	74.285 $\pm$ 0.015
Gaussian scale 3	68.954 $\pm$ 0.007	71.409 $\pm$ 0.009	68.794 $\pm$ 0.011	70.178 $\pm$ 0.009	63.826 $\pm$ 0.021	67.225 $\pm$ 0.015	73.762 $\pm$ 0.026	73.131 $\pm$ 0.02

<sup>a</sup>Male, <sup>b</sup>Female, SD = Standard Deviation

Table 2.5: Model Performance Summary for HS Screening using SMOTE

Models	Performance Metrics							
	Training Accuracy (%) ± SD		Test Accuracy (%) ± SD		Test Sensitivity (%) ± SD		Test Specificity (%) ± SD	
Sex	M <sup>a</sup>	F <sup>b</sup>	M	F	M	F	M	F
Bagged Trees	76.4 ± 0.006	80.2 ± 0.005	77.2 ± 0.009	81.6 ± 0.010	75.8 ± 0.010	81.3 ± 0.018	78.6 ± 0.011	81.9 ± 0.011
Boosted Trees	72.5 ± 0.009	75.0 ± 0.004	72.7 ± 0.009	75.3 ± 0.008	72.5 ± 0.016	75.3 ± 0.014	72.9 ± 0.011	75.4 ± 0.020
Medium KNN	71.2 ± 0.006	72.9 ± 0.005	71.8 ± 0.010	74.4 ± 0.008	72.7 ± 0.016	75.6 ± 0.016	70.9 ± 0.010	73.0 ± 0.011
Cubic KNN	70.8 ± 0.005	72.5 ± 0.003	71.0 ± 0.009	73.4 ± 0.010	71.4 ± 0.017	74.3 ± 0.016	70.7 ± 0.011	72.5 ± 0.013
Cosine KNN	71.1 ± 0.004	73.0 ± 0.005	71.4 ± 0.006	74.2 ± 0.007	69.5 ± 0.011	73.6 ± 0.014	73.3 ± 0.012	74.8 ± 0.016

Table 2.6: Best performing (sensitivity only) models for HS Screening using SMOTE

Models	Performance Metrics							
	Training Accuracy (%) ± SD		Test Accuracy (%) ± SD		Test Sensitivity (%) ± SD		Test Specificity (%) ± SD	
Sex	M <sup>a</sup>	F <sup>b</sup>	M	F	M	F	M	F
Fine Gaussian SVM	73.0 ± 0.003	74.1 ± 0.003	73.4 ± 0.008	75.4 ± 0.008	78.1 ± 0.013	78.4 ± 0.012	68.9 ± 0.013	72.3 ± 0.016
Fine KNN	73.4 ± 0.005	75.1 ± 0.003	73.8 ± 0.007	76.6 ± 0.004	82.1 ± 0.015	85.7 ± 0.007	65.6 ± 0.009	67.0 ± 0.009
Weighted KNN	74.1 ± 0.004	76.0 ± 0.005	74.5 ± 0.008	77.3 ± 0.007	82.6 ± 0.009	86.0 ± 0.013	66.5 ± 0.011	68.1 ± 0.010

## 2.12 References

- [1] N. Chalasani *et al.*, “The diagnosis and management of nonalcoholic fatty liver disease: Practice guidance from the American Association for the Study of Liver Diseases: Hepatology,” *Hepatology*, vol. 67, no. 1, pp. 328–357, Jan. 2018, doi: 10.1002/hep.29367.
- [2] “Symptoms & Causes of NAFLD & NASH | NIDDK.” <https://www.niddk.nih.gov/health-information/liver-disease/nafl-d-nash/symptoms-causes> (accessed Oct. 19, 2020).
- [3] G. E. Arteel, “Hepatotoxicity,” in *Arsenic*, John Wiley & Sons, Ltd, 2015, pp. 249–265. doi: 10.1002/9781118876992.ch11.
- [4] B. Wahlang *et al.*, “Toxicant-associated Steatohepatitis,” *Toxicol. Pathol.*, vol. 41, no. 2, pp. 343–360, Feb. 2013, doi: 10.1177/0192623312468517.
- [5] M. Cave *et al.*, “Toxicant-associated steatohepatitis in vinyl chloride workers,” *Hepatology*, vol. 51, no. 2, pp. 474–481, Feb. 2010, doi: 10.1002/hep.23321.
- [6] D. N. G. Mazumder *et al.*, “Chronic arsenic toxicity from drinking tubewell water in rural West Bengal,” p. 8.
- [7] Z. M. Younossi, A. B. Koenig, D. Abdelatif, Y. Fazel, L. Henry, and M. Wymer, “Global epidemiology of nonalcoholic fatty liver disease-Meta-analytic assessment of prevalence, incidence, and outcomes,” *Hepatology*, vol. 64, no. 1, pp. 73–84, Jul. 2016, doi: 10.1002/hep.28431.
- [8] U. Iqbal, B. Perumpail, D. Akhtar, D. Kim, and A. Ahmed, “The Epidemiology, Risk Profiling and Diagnostic Challenges of Nonalcoholic Fatty Liver Disease,” *Medicines*, vol. 6, no. 1, p. 41, Mar. 2019, doi: 10.3390/medicines6010041.
- [9] Z. M. Younossi *et al.*, “The global epidemiology of NAFLD and NASH in patients with type 2 diabetes: A systematic review and meta-analysis,” *J. Hepatol.*, vol. 71, no. 4, pp. 793–801, Oct. 2019, doi: 10.1016/j.jhep.2019.06.021.
- [10] G. C. Farrell, V. W.-S. Wong, and S. Chitturi, “NAFLD in Asia—as common and important as in the West,” *Nat. Rev. Gastroenterol. Hepatol.*, vol. 10, no. 5, pp. 307–318, May 2013, doi: 10.1038/nrgastro.2013.34.
- [11] Z. M. Younossi *et al.*, “Nonalcoholic Fatty Liver Disease in Lean Individuals in the United States,” *Medicine (Baltimore)*, vol. 91, no. 6, pp. 319–327, Nov. 2012, doi: 10.1097/MD.0b013e3182779d49.
- [12] A. Lonardo *et al.*, “Sex Differences in Nonalcoholic Fatty Liver Disease: State of the Art and Identification of Research Gaps,” *Hepatology*, vol. 70, no. 4, pp. 1457–1469, Oct. 2019, doi: 10.1002/hep.30626.

- [13] S. Ballestri, F. Nascimbeni, E. Baldelli, A. Marrazzo, D. Romagnoli, and A. Lonardo, "NAFLD as a Sexual Dimorphic Disease: Role of Sex and Reproductive Status in the Development and Progression of Nonalcoholic Fatty Liver Disease and Inherent Cardiovascular Risk," *Adv. Ther.*, vol. 34, no. 6, pp. 1291–1326, 2017, doi: 10.1007/s12325-017-0556-1.
- [14] J. D. Yang *et al.*, "Sex and menopause impact severity of fibrosis among patients with nonalcoholic steatohepatitis," *Hepatology*, vol. 59, no. 4, pp. 1406–1414, Apr. 2014, doi: 10.1002/hep.26761.
- [15] M. Balakrishnan *et al.*, "Women Have a Lower Risk of Nonalcoholic Fatty Liver Disease but a Higher Risk of Progression Vs Men: A Systematic Review and Meta-analysis," *Clin. Gastroenterol. Hepatol.*, p. S1542356520306121, Apr. 2020, doi: 10.1016/j.cgh.2020.04.067.
- [16] M. Gambarin–Gelwan, S. V. Kinkhabwala, T. D. Schiano, C. Bodian, H. Yeh, and W. Futterweit, "Prevalence of Nonalcoholic Fatty Liver Disease in Women With Polycystic Ovary Syndrome," *Clin. Gastroenterol. Hepatol.*, vol. 5, no. 4, pp. 496–501, Apr. 2007, doi: 10.1016/j.cgh.2006.10.010.
- [17] C. E. Kelley, "Review of nonalcoholic fatty liver disease in women with polycystic ovary syndrome," *World J. Gastroenterol.*, vol. 20, no. 39, p. 14172, 2014, doi: 10.3748/wjg.v20.i39.14172.
- [18] D. L. Wyness, "The four stages of Non-Alcoholic Fatty Liver Disease (NAFLD)," *liver-health-uk*, Oct. 09, 2017. <https://www.liverhealthuk.com/post/the-four-stages-of-naflD> (accessed Mar. 11, 2022).
- [19] T. Marjot, A. Moolla, J. F. Cobbald, L. Hodson, and J. W. Tomlinson, "Nonalcoholic Fatty Liver Disease in Adults: Current Concepts in Etiology, Outcomes, and Management," *Endocr. Rev.*, vol. 41, no. 1, pp. 66–117, Feb. 2020, doi: 10.1210/endrev/bnz009.
- [20] J. Wattacheril and A. J. Sanyal, "Lean NAFLD: an Underrecognized Outlier," *Curr. Hepatol. Rep.*, vol. 15, no. 2, pp. 134–139, Jun. 2016, doi: 10.1007/s11901-016-0302-1.
- [21] G. Bhat and C. S. Baba, "Insulin resistance and metabolic syndrome in nonobese Indian patients with non-alcoholic fatty liver disease," *Trop. Gastrology*, vol. 34, no. 1, pp. 18–24, Mar. 2013, doi: 10.7869/tg.2012.86.
- [22] K. Nishioji *et al.*, "Prevalence of and risk factors for non-alcoholic fatty liver disease in a non-obese Japanese population, 2011–2012," *J. Gastroenterol.*, vol. 50, no. 1, pp. 95–108, Jan. 2015, doi: 10.1007/s00535-014-0948-9.
- [23] R.-N. Feng *et al.*, "Lean-non-alcoholic fatty liver disease increases risk for metabolic disorders in a normal weight Chinese population," *World J. Gastroenterol.*, vol. 20, no. 47, pp. 17932–17940, Dec. 2014, doi: 10.3748/wjg.v20.i47.17932.

- [24] E. Margariti, M. Deutsch, S. Manolakopoulos, and G. V. Papatheodoridis, “Non-alcoholic fatty liver disease may develop in individuals with normal body mass index,” *Ann. Gastroenterol.*, vol. 25, no. 1, pp. 45–51, 2012.
- [25] H. J. Kim *et al.*, “Metabolic Significance of Nonalcoholic Fatty Liver Disease in Nonobese, Nondiabetic Adults,” *Arch. Intern. Med.*, vol. 164, no. 19, p. 2169, Oct. 2004, doi: 10.1001/archinte.164.19.2169.
- [26] K. Cusi, “Nonalcoholic steatohepatitis in nonobese patients: Not so different after all,” *Hepatology*, vol. 65, no. 1, pp. 4–7, Jan. 2017, doi: 10.1002/hep.28839.
- [27] A. Lonardo, S. Ballestri, G. Marchesini, P. Angulo, and P. Loria, “Nonalcoholic fatty liver disease: A precursor of the metabolic syndrome,” *Dig. Liver Dis.*, vol. 47, no. 3, pp. 181–190, Mar. 2015, doi: 10.1016/j.dld.2014.09.020.
- [28] R. M. Carr, A. Oranu, and V. Khungar, “Nonalcoholic Fatty Liver Disease,” *Gastroenterol. Clin. North Am.*, vol. 45, no. 4, pp. 639–652, Dec. 2016, doi: 10.1016/j.gtc.2016.07.003.
- [29] E. Fabbrini *et al.*, “Intrahepatic fat, not visceral fat, is linked with metabolic complications of obesity,” *Proc. Natl. Acad. Sci.*, vol. 106, no. 36, pp. 15430–15435, Sep. 2009, doi: 10.1073/pnas.0904944106.
- [30] K. Azuma *et al.*, “Higher liver fat content among Japanese in Japan compared with non-Hispanic whites in the United States,” *Metabolism*, vol. 58, no. 8, pp. 1200–1207, Aug. 2009, doi: 10.1016/j.metabol.2009.03.021.
- [31] B. Sears and M. Perry, “The role of fatty acids in insulin resistance,” *Lipids Health Dis.*, vol. 14, no. 1, p. 121, Dec. 2015, doi: 10.1186/s12944-015-0123-1.
- [32] S. Petta *et al.*, “Pathophysiology of Non-Alcoholic Fatty Liver Disease,” *Int. J. Mol. Sci.*, vol. 17, no. 12, p. 2082, Dec. 2016, doi: 10.3390/ijms17122082.
- [33] P.-C. Chen, K.-L. Chien, H.-C. Hsu, T.-C. Su, F.-C. Sung, and Y.-T. Lee, “Metabolic syndrome and C-reactive protein in stroke prediction: a prospective study in Taiwan,” *Metabolism*, vol. 58, no. 6, pp. 772–778, Jun. 2009, doi: 10.1016/j.metabol.2009.01.006.
- [34] P. L. Huang, “A comprehensive definition for metabolic syndrome,” *Dis. Model. Mech.*, vol. 2, no. 5–6, pp. 231–237, May 2009, doi: 10.1242/dmm.001180.
- [35] S. M. Grundy, H. B. Brewer, J. I. Cleeman, S. C. Smith, and C. Lenfant, “Definition of Metabolic Syndrome: Report of the National Heart, Lung, and Blood Institute/American Heart Association Conference on Scientific Issues Related to Definition,” *Circulation*, vol. 109, no. 3, pp. 433–438, Jan. 2004, doi: 10.1161/01.CIR.0000111245.75752.C6.

- [36] A. Chowdhury and Z. M. Younossi, “Global Epidemiology and Risk Factors for Nonalcoholic Fatty Liver Disease,” in *Alcoholic and Non-Alcoholic Fatty Liver Disease: Bench to Bedside*, N. Chalasani and G. Szabo, Eds. Cham: Springer International Publishing, 2016, pp. 21–40. doi: 10.1007/978-3-319-20538-0\_2.
- [37] I. Doycheva *et al.*, “Increasing Burden of Chronic Liver Disease Among Adolescents and Young Adults in the USA: A Silent Epidemic,” *Dig. Dis. Sci.*, vol. 62, no. 5, pp. 1373–1380, May 2017, doi: 10.1007/s10620-017-4492-3.
- [38] G. Choudhuri, S. Chaudhari, D. Pawar, and D. S. Roy, “Etiological Patterns, Liver Fibrosis Stages and Prescribing Patterns of Hepato-Protective Agents in Indian Patients with Chronic Liver Disease,” *J. Assoc. Physicians India*, vol. 66, no. 12, pp. 58–63, Dec. 2018.
- [39] G. Cholankeril *et al.*, “Liver Transplantation for Nonalcoholic Steatohepatitis in the US: Temporal Trends and Outcomes,” *Dig. Dis. Sci.*, vol. 62, no. 10, pp. 2915–2922, Oct. 2017, doi: 10.1007/s10620-017-4684-x.
- [40] F. A. Cimini *et al.*, “Relationship between adipose tissue dysfunction, vitamin D deficiency and the pathogenesis of non-alcoholic fatty liver disease,” *World J. Gastroenterol.*, vol. 23, no. 19, pp. 3407–3417, May 2017, doi: 10.3748/wjg.v23.i19.3407.
- [41] E. Vilar-Gomez and N. Chalasani, “Non-invasive assessment of non-alcoholic fatty liver disease: Clinical prediction rules and blood-based biomarkers,” *J. Hepatol.*, vol. 68, no. 2, pp. 305–315, Feb. 2018, doi: 10.1016/j.jhep.2017.11.013.
- [42] M. V. Machado, J. Coutinho, F. Carepa, A. Costa, H. Proença, and H. Cortez-Pinto, “How adiponectin, leptin, and ghrelin orchestrate together and correlate with the severity of nonalcoholic fatty liver disease,” *Eur. J. Gastroenterol. Hepatol.*, vol. 24, no. 10, p. 1166, Oct. 2012, doi: 10.1097/MEG.0b013e32835609b0.
- [43] J.-B. Suh, S. M. Kim, G.-J. Cho, and K. M. Choi, “Serum AFBP levels are elevated in patients with nonalcoholic fatty liver disease,” *Scand. J. Gastroenterol.*, vol. 49, no. 8, pp. 979–985, Aug. 2014, doi: 10.3109/00365521.2013.836754.
- [44] X. Gong *et al.*, “Membraneless reproducible MoS2 field-effect transistor biosensor for high sensitive and selective detection of FGF21,” *Sci. China Mater.*, vol. 62, no. 10, pp. 1479–1487, Oct. 2019, doi: 10.1007/s40843-019-9444-y.
- [45] J. S. Paige *et al.*, “A Pilot Comparative Study of Quantitative Ultrasound, Conventional Ultrasound, and MRI for Predicting Histology-Determined Steatosis Grade in Adult Nonalcoholic Fatty Liver Disease,” *Am. J. Roentgenol.*, vol. 208, no. 5, pp. W168–W177, May 2017, doi: 10.2214/AJR.16.16726.

- [46] S. C. Lin *et al.*, “Noninvasive Diagnosis of Nonalcoholic Fatty Liver Disease and Quantification of Liver Fat Using a New Quantitative Ultrasound Technique,” *Clin. Gastroenterol. Hepatol.*, vol. 13, no. 7, pp. 1337–1345.e6, Jul. 2015, doi: 10.1016/j.cgh.2014.11.027.
- [47] A. Han *et al.*, “Inter-platform reproducibility of ultrasonic attenuation and backscatter coefficients in assessing NAFLD,” *Eur. Radiol.*, vol. 29, no. 9, pp. 4699–4708, Sep. 2019, doi: 10.1007/s00330-019-06035-9.
- [48] W. Cao, X. An, L. Cong, C. Lyu, Q. Zhou, and R. Guo, “Application of Deep Learning in Quantitative Analysis of 2-Dimensional Ultrasound Imaging of Nonalcoholic Fatty Liver Disease,” *J. Ultrasound Med.*, vol. 39, no. 1, pp. 51–59, Jan. 2020, doi: 10.1002/jum.15070.
- [49] C.-C. Wu *et al.*, “Prediction of fatty liver disease using machine learning algorithms,” *Comput. Methods Programs Biomed.*, vol. 170, pp. 23–29, Mar. 2019, doi: 10.1016/j.cmpb.2018.12.032.
- [50] M. Biswas *et al.*, “Symtosis: A liver ultrasound tissue characterization and risk stratification in optimized deep learning paradigm,” *Comput. Methods Programs Biomed.*, vol. 155, pp. 165–177, Mar. 2018, doi: 10.1016/j.cmpb.2017.12.016.
- [51] P. Sorino *et al.*, “Selecting the best machine learning algorithm to support the diagnosis of Non-Alcoholic Fatty Liver Disease: A meta learner study,” *PLOS ONE*, vol. 15, no. 10, p. e0240867, Oct. 2020, doi: 10.1371/journal.pone.0240867.
- [52] G. I. Rajathi and G. W. Jiji, “Chronic Liver Disease Classification Using Hybrid Whale Optimization with Simulated Annealing and Ensemble Classifier,” *Symmetry*, vol. 11, no. 1, p. 33, Jan. 2019, doi: 10.3390/sym11010033.
- [53] T. Renukadevi and S. Karunakaran, “Optimizing deep belief network parameters using grasshopper algorithm for liver disease classification,” *Int. J. Imaging Syst. Technol.*, vol. 30, no. 1, pp. 168–184, Mar. 2020, doi: 10.1002/ima.22375.
- [54] K. B. Kim, G. H. Kim, D. H. Song, H. J. Park, and C. W. Kim, “Automatic segmentation of liver/kidney area with double-layered fuzzy C-means and the utility of hepatorenal index for fatty liver severity classification,” *J. Intell. Fuzzy Syst.*, vol. 39, no. 1, pp. 925–936, Jul. 2020, doi: 10.3233/JIFS-191850.
- [55] Y. Huo *et al.*, “Fully automatic liver attenuation estimation combining CNN segmentation and morphological operations,” *Med. Phys.*, vol. 46, no. 8, pp. 3508–3519, Aug. 2019, doi: 10.1002/mp.13675.
- [56] A. E. Bohte, J. R. van Werven, S. Bipat, and J. Stoker, “The diagnostic accuracy of US, CT, MRI and 1H-MRS for the evaluation of hepatic steatosis compared with liver biopsy: a meta-analysis,” *Eur. Radiol.*, vol. 21, no. 1, pp. 87–97, Jan. 2011, doi: 10.1007/s00330-010-1905-5.

- [57] M. Shahabi, H. Hassanpour, and H. Mashayekhi, “Rule extraction for fatty liver detection using neural networks,” *Neural Comput. Appl.*, vol. 31, no. 4, pp. 979–989, Apr. 2019, doi: 10.1007/s00521-017-3130-5.
- [58] Centers for Disease Control and Prevention (CDC). National Center for Health Statistics (NCHS). National Health and Nutrition Examination Survey Data, “NHANES III (1988-1994) - Data Files.” <https://wwwn.cdc.gov/nchs/nhanes/nhanes3/datafiles.aspx#core> (accessed Apr. 23, 2021).
- [59] “NHANES 1988-1994: Hepatic/Gallbladder Ultrasound and Hepatic Steatosis Data Documentation, Codebook, and Frequencies.” [https://wwwn.cdc.gov/nchs/data/nhanes3/34a/HGUHS.htm#Data\\_Processing\\_and\\_Editing](https://wwwn.cdc.gov/nchs/data/nhanes3/34a/HGUHS.htm#Data_Processing_and_Editing) (accessed Jan. 30, 2019).
- [60] “NHANES Tutorials - Module 3 - Weighting.” <https://wwwn.cdc.gov/nchs/nhanes/tutorials/module3.aspx> (accessed Apr. 03, 2022).
- [61] J. Luo and M. Hendryx, “Metal mixtures and kidney function: An application of machine learning to NHANES data,” *EnvIron. Res.*, vol. 191, p. 110126, Dec. 2020, doi: 10.1016/j.envres.2020.110126.
- [62] F. López-Martínez, E. R. Núñez-Valdez, R. G. Crespo, and V. García-Díaz, “An artificial neural network approach for predicting hypertension using NHANES data,” *Sci. Rep.*, vol. 10, no. 1, p. 10620, Dec. 2020, doi: 10.1038/s41598-020-67640-z.
- [63] G. A. Klados *et al.*, “Machine Learning Model for Predicting CVD Risk on NHANES Data,” in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, Mexico, Nov. 2021, pp. 1749–1752. doi: 10.1109/EMBC46164.2021.9630119.
- [64] SAS Inc., “*The data analysis for this paper was generated using SAS software. Copyright ©2019 SAS Institute Inc.*” 2019.
- [65] MATLAB, *MATLAB v9.9.0 (R2020b)*. Natick, Massachusetts: The MathWorks Inc., 2020.
- [66] N. Cristianini, J. Shawe-Taylor, and others, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.
- [67] T. Hastie and R. Tibshirani, *J. Friedman The Elements of Statistical Learning. Chapter 6*. Springer Verlag, New York, 2001.
- [68] “Support Vector Machine Classification - MATLAB & Simulink.” <https://www.mathworks.com/help/stats/support-vector-machine-classification.html> (accessed May 03, 2021).
- [69] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.



- [70] Q. Wei and R. L. Dunbrack, “The Role of Balanced Training and Testing Data Sets for Binary Classifiers in Bioinformatics,” *PLoS ONE*, vol. 8, no. 7, p. e67863, Jul. 2013, doi: 10.1371/journal.pone.0067863.
- [71] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic Minority Over-sampling Technique,” *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.
- [72] L. Guzman, “Data sampling improvement by developing SMOTE technique in SAS,” in *Proceedings of the SAS Global Forum 2015 Conference*, 2015, pp. 3483–2015.
- [73] P. Hall and J. Cash, “What is the real function of the liver ‘function’ tests?,” *Ulster Med. J.*, vol. 81, no. 1, pp. 30–36, Jan. 2012.
- [74] A. Dasgupta and A. Wahed, “Chapter 10 - Liver Diseases and Liver Function Tests,” in *Clinical Chemistry, Immunology and Laboratory Quality Control*, A. Dasgupta and A. Wahed, Eds. San Diego: Elsevier, 2014, pp. 177–195. doi: 10.1016/B978-0-12-407821-5.00010-3.
- [75] P. Y. Kwo, S. M. Cohen, and J. K. Lim, “ACG Clinical Guideline: Evaluation of Abnormal Liver Chemistries,” *Am. J. Gastroenterol.*, vol. 112, no. 1, pp. 18–35, Jan. 2017, doi: 10.1038/ajg.2016.517.
- [76] “Understand Liver Enzyme Test Results — American Liver Foundation.” <https://liverfoundation.org/understand-liver-enzyme-test-results-2/> (accessed Mar. 30, 2021).
- [77] “Risk Factors for Diabetes | NIDDK,” *National Institute of Diabetes and Digestive and Kidney Diseases*. <https://www.niddk.nih.gov/health-information/professionals/clinical-tools-patient-management/diabetes/game-plan-preventing-type-2-diabetes/prediabetes-screening-how-why/risk-factors-diabetes> (accessed Mar. 30, 2021).
- [78] “Hyperglycemia in diabetes - Diagnosis and treatment - Mayo Clinic.” <https://www.mayoclinic.org/diseases-conditions/hyperglycemia/diagnosis-treatment/drc-20373635> (accessed Mar. 30, 2021).
- [79] “Classification Using Nearest Neighbors - MATLAB & Simulink.” <https://www.mathworks.com/help/stats/classification-using-nearest-neighbors.html> (accessed Mar. 23, 2022).

## APPENDIX A. – CODE FOR OBJECTIVE 1A

### 1. SAS CODE TO KEEP VARIABLES OF INTEREST AND DISCARD THE REST

```
%%%%%%%%%
% Created on: 02/19/2019
% Input: Raw data from NHANES
% Output: Data with only variables of interest, specific to objective 1A
% Author: Ridhi Deo
% File name: obj1a_sas_1.sas
% Description: Used eliminate the variables that are not required and to only keep the variables
of interest from the raw datasets. This program was developed using SAS 2019 [64].
%%%%%%%%%

% set the data path and choose the variables to keep. Variable codes are as provided by
% NHANESIII
LIBNAME NH3 "Raw data path";
data NH3.adult_reduced;
set NH3.adult;
keep SEQN HSAGEIR HSSEX;
proc sort; by seqn; run;

data NH3.lab_reduced;
set NH3.lab;
keep SEQN TGP HDP TCP;
proc sort; by seqn; run;

data NH3.exam_reduced;
set NH3.exam;
keep SEQN BMPBMI MAPE1 MAPE2 MAPE4;
proc sort; by seqn; run;

data NH3.HGUHS_reduced;
set NH3.HGUHS;
keep SEQN GUPHSPFR;

proc sort; by seqn; run;

proc contents data = NH3.adult_reduced;
run;

proc contents data = NH3.lab_reduced;
run;
proc contents data = NH3.exam_reduced;
run;
```

```
proc contents data = NH3.HGUHS_reduced;
run;
```

**%% Note: The data output from this program has been archive in the Purdue Research Repository**

## 2. SAS CODE TO MERGE DATASETS

```
%%%%%%%%%
% Created on: 02/19/2019
% Input: Processed data with only variables of interest, specific to objective 1A
% Output: Multiple datasets of interest merged into one dataset
% Author: Ridhi Deo
% File name: obj1a_sas_2.sas
% Description: Used to combine different datasets of interest into one. This program was
developed using SAS 2019 [64].
%%%%%%%%%
```

```
LIBNAME NH3 "Input data path";
```

```
% Sorting data by the sequential number
proc sort data=NH3.adult_reduced;
    by SEQN;
proc sort data=NH3.lab_reduced;
    by SEQN;
proc sort data=NH3.exam_reduced;
    by SEQN;
proc sort data=NH3.hguhs_reduced;
    by SEQN;
```

```
%Merging data using the sequential number
data NH3.merged;
merge NH3.adult_reduced
      NH3.lab_reduced
      NH3.exam_reduced
      NH3.hguhs_reduced;
    by SEQN;
```

```
proc contents data = NH3.merged varnum;
proc means data=NH3.merged N Nmiss min max maxdec=2;
run;
```

**%% Note: The data output from this program has been archive in the Purdue Research Repository**

### 3. SAS CODE TO REMOVE OBSERVATIONS WITH MISSING DATA

```
%%%%%%%%%
% Created on: 02/19/2019
% Input: Merged dataset
% Output: Merged dataset without missing data
% Author: Ridhi Deo
%File name: obj1a_sas_3.sas
%Description: This code was written to eliminate any observations containing missing data
related to alcohol information. This program was developed using SAS 2019 [64].
%%%%%%%%%
```

```
LIBNAME NH3 " Raw data path";
data NH3.merged_deletedNaNs;
set NH3.merged;
if nmiss(MAPE1) > 0 then delete;

proc means data=NH3.merged_deletedNaNs N Nmiss min max maxdec=2;
run;
```

### 4. MATLAB CODE TO PROCESS AND CREATE DISEASE AND NO-DISEASE DATASETS

```
%%%%%%%%%
% Created on: 01/31/2019
% Input: Processed data exported from SAS
% Output: Processed datasets for disease (HS yes) and no disease (HS no)
% Author: Ridhi Deo
% File name: Obj1a_matlab_1.m (MATLAB R2018b [65])
% Description: This code was written to further clean and process the input dataset (exported
from SAS). Then the data was divided into two sub-datasets based on HS yes or no.
%%%%%%%%%
```

```
%% Clear Screen
clc;
clear all;
%% Read the merged data text file into MATLAB
data = readtable(Raw data path);
```

```
%% Changing the values in the variables to 0s and 1s for clear representation of data
data.TGP(data.TGP == 8888) = NaN;
data.TCP(data.TCP == 8888) = NaN;
data.HDP(data.HDP == 8888) = NaN;
data.BMPBMI(data.BMPBMI == 8888) = NaN;
data.MAPE1(data.MAPE1 == 8) = NaN; % 8 = blank but applicable as per NHANES
data.MAPE1(data.MAPE1 == 2) = 0; % 2 = No as per NHANES %1 is yes..leaving it as is
data.MAPE1(data.MAPE1 == 9) = NaN; % 9 = don't know as per NHANES
```

```

data.MAPE2(data.MAPE2 == 8) = NaN; % 8 = blank but applicable as per NHANES
data.MAPE2(data.MAPE2 == 2) = 0; % 2 = No as per NHANES %1 is yes..leaving it as is
data.MAPE2(data.MAPE2 == 9) = NaN; % 9 = don't know as per NHANES
data.MAPE4(data.MAPE4 == 999) = NaN; % 999 = don't know as per NHANES
data.MAPE4(data.MAPE4 == 888) = NaN; % 888 = blank but applicable as per NHANES
data.GUPHSPFR(data.GUPHSPFR == 8) = NaN; %No image as per NHANES
data.GUPHSPFR(data.GUPHSPFR == 7) = NaN; %Image is present, but ungradable as per NHANES
data.GUPHSPFR(data.GUPHSPFR == 1) = 0; % 1 is Normal-Mild as per NHANES. Changing it to 0 to indicate no risk
data.GUPHSPFR(data.GUPHSPFR == 2) = 1; % 2 is Moderate - Severe as per NHANES. Changing it to 1 to indicate risk

```

```

%% Changing the names of the variables to make it easy to understand

```

```

data.Properties.VariableNames{'SEQN'} = 'Sequential_Number';
data.Properties.VariableNames{'HSSEX'} = 'Sex';
data.Properties.VariableNames{'HSAGEIR'} = 'Age';
data.Properties.VariableNames{'TGP'} = 'Triglycerides';
data.Properties.VariableNames{'TCP'} = 'Total_Cholesterol';
data.Properties.VariableNames{'HDP'} = 'HDL';
data.Properties.VariableNames{'BMPBMI'} = 'BMI';
data.Properties.VariableNames{'MAPE1'} = 'Alcohol_12_life';
data.Properties.VariableNames{'MAPE2'} = 'Alcohol_12_last_year';
data.Properties.VariableNames{'MAPE4'} = 'Drinks_per_day';
data.Properties.VariableNames{'GUPHSPFR'} = 'Fatty_Liver';

```

```

%% Filling in missing data for the 12 drinks per year column with information from 12 drinks in life column

```

```

% If a person has not had 12 drinks in their lifetime, the response on the
% variable 12 drinks in past year are missing
% To fix that, individuals who have not had 12 drinks in their life will
% have 0s on the column 12 drinks in past year
% Same logic applies - making all those who have not had 12 drinks in their life 0 in the column
drinks per day

```

```

for i = 1: size(data,1)

```

```

    if (data.Alcohol_12_life(i) == 0)
        data.Alcohol_12_last_year(i) = 0;
        data.Drinks_per_day(i) = 0;
    end

```

```

end

```

```

for i = 1: size(data,1)

```

```

    if (data.Alcohol_12_last_year(i) == 0)

```

```

        data.Drinks_per_day(i) = 0;
    end

end

%% Deleting data samples with missing information wrt Fatty_Liver column
idx_FL = find(isnan(data.Fatty_Liver));
data(idx_FL,:) = [];

%% Deleting datasamples with missing information wrt alcohol
idx_alc_life = find(isnan(data.Alcohol_12_life));
data(idx_alc_life,:) = [];

%% Deleting datasamples with missing information wrt triglycerides
idx_Trig = find(isnan(data.Triglycerides));
data(idx_Trig,:) = [];

%% Deleting data samples with > 1 drink per day for women and > 2 drinks per day for men
% As per the exclusion criteria followed by Long et al (They used drinks per week)
k = 1;
for i = 1: size(data,1)
    if(data.Sex(i) == 1 && data.Drinks_per_day(i) > 2)
        idx(k) = i;
        k = k + 1;
    end
end

data(idx,:) = [];

m = 1;
for i = 1: size(data,1)
    if(data.Sex(i) == 2 && data.Drinks_per_day(i) > 1)
        idx1(m,1) = i;
        m = m+1;
    end
end

data(idx1,:) = [];

%% Changing Sex to 0 and 1
data.Sex(data.Sex == 2) = 0; % Changing women to 0
% Men remain as 1 (1 is high risk, 0 is low, generally)

%% Deleting data samples with missing information wrt alcohol per day
idx_Drinks = find(isnan(data.Drinks_per_day));

```

```

data(idx_Drinks,:) = [];
%% Deleting datasamples with missing BMI
idx_BMI = find(isnan(data.BMI));
data(idx_BMI,:) = [];

%% Creating a dataset which is the subset of the above dataset to compare with FHS fatty
liver study

data_FHS = data(:,[1:7,11]);

%% Normalizing Age, BMI,drinks_per_day
data_FHS.Age = normalize(data_FHS.Age,'zscore'); %Normalizing to range between 0
and 1
data_FHS.BMI = normalize(data_FHS.BMI,'zscore'); %Normalizing to range between 0
and 1
data_FHS.Drinks_per_day = normalize(data_FHS.Drinks_per_day,'zscore');
%Normalizing to range between 0 and 1
data_FHS.HDL = normalize(data_FHS.HDL , 'zscore');
data_FHS.Total_Cholesterol = normalize(data_FHS.Total_Cholesterol , 'zscore');
data_FHS.Triglycerides = normalize(data_FHS.Triglycerides,'zscore');

all_disease = array2table(zeros(sum(data_FHS.Fatty_Liver == 1),8));
all_no_disease = array2table(zeros( size(data_FHS,1) - sum(data_FHS.Fatty_Liver == 1),
8));

%% Saved the disease and no-disease datasets into my hard drive as disease_dataset.mat
and
% no_disease_dataset.mat, respectively

```

## 5. CODE TO CREATE SYNTHETIC DATA AND TO TRAIN, TEST ML MODELS

```

%%%%%%%%%%
% Created on: 04/23/2019
% Input: Processed disease and no disease datasets from obj1a_matlab_1.m
% Output: Results from the ML models
% Author: Ridhi Deo
%File name: obj1a_matlab_2.m (R2018b [65]) )
%Description: This code is a parent code that was used to create synthetic data using the
SMOTE and Gower's distance metrics. Data was then divided into 70:30 training:test ratio,
respectively. Five machine learning models were trained and tested: Fine Gaussian SVM,
Medium Gaussian SVM, Bagged Trees, Gentle Boosted Trees and ADA Boosted Trees.
%This code is written to internally call on the obj1a_matlab_2a.m code.
%The obj1a_matlab_2a.m code further calls on: obj1a_matlab_2b.m,
obj1a_matlab_2c.m, %obj1a_matlab_2d.m, obj1a_matlab_2e.m, and obj1a_matlab_2f.m.
%%%%%%%%%%

```

```

clc;
clear all;
% Loading the disease dataset
disease_sample = load(Raw data path);
disease_sample = disease_sample.disease_dataset;
disease_sample(:,[1,8:12]) = [];
T = size(disease_sample,1); %Measure of number of diseased samples
N = 2; % Equivalent of 200% synthetic sample generation
k = 2; % Setting number of nearest neighbours to 2
num_attrs = size(disease_sample,2); %Number of variables
new_index = 0; % Variable to keep a count of newly generated synthetic samples
    synthetic_sample.N{2} = zeros(T,num_attrs); %Since we are generating 200% synthetic,
    this value is N{2}
nn_array = zeros(T,k+1); %Tp keep a list of nearest neighbours for each sample
R = zeros(5,1); %Range of continous variables is represented by the array R
    temp_range = table2array(disease_sample(:,2:6)); %Temporary conversions to array - for
    computational ease. This is essential disease_sample
temp_dist = table2array(disease_sample);
distance = zeros(T,T); %Preallocating a matrix to store all the gower's distances
for i = 1:5 %Calculating ranges
    R(i,1) = (max(temp_range(:,i)) - min(temp_range(:,i)));
end
for i = 1:T %Computing Gower's distance and populating the distance matrix
    for j = 1:T
        if(temp_dist(i,1) == temp_dist(j,1))
            a = 1;
            d = 0;
        else
            d = 1;
            a = 0;
        end

        for m = 1:5
            part1(m) = 1 - (abs(temp_dist(i,m+1) - temp_dist(j,m+1))/R(m,1));
        end

        part2(i,j) = sum(part1,2);
        part3(i,j) = (part2(i,j) + a)/(5 + (1-d));
        distance(i,j) = 1 - part3(i,j);
    end
    [nn_values(i,:),nn_array(i,:)] = mink(distance(i,:),k+1); %Finding the 5 nearest neighbors
    because the nearest one is with the sample itself
    % And to remove the column of zeros and still have 2 NN, I am obtaining 3
    % to start with
end

```



```

nn_array(:,1) = []; %Nearest neighbor indices
nn_values(:,1) = []; %Nearest neighbor values
while (N~=0) %To perform 200% synthetic sampling
    for i = 1:size(temp_dist,1)
        for attr=2:size(temp_dist,2)
            nn = randi([1 k],1);
            dif = temp_dist(nn_array(i,nn),attr) - temp_dist(i,attr);
            gap = 0 + rand(1,1);
            synthetic_sample.N{N}(i,attr) = temp_dist(i,attr) + (gap*dif); %Synthetic continuous
            attributes
        end
        %Synthetic binary attribute
        if(temp_dist(i,1) + sum(temp_dist(nn_array(i,1:k),1)) < 2)
            synthetic_sample.N{N}(i,1) = 0;
        else
            synthetic_sample.N{N}(i,1) = 1;
        end
    end
    N = N-1; %To avoid infinite loops
end

total_synthetic_samples = [synthetic_sample.N{1};synthetic_sample.N{2}];

synthetic_original_disease = [total_synthetic_samples; temp_dist];
synthetic_original_disease(:,7) = ones(size(synthetic_original_disease,1),1);
    non_disease_dataset = load('file path');

non_disease_dataset = non_disease_dataset.no_disease_dataset;
non_disease_dataset(:,[1,9:12]) = [];
non_disease_dataset = table2array(non_disease_dataset);

full_data = [synthetic_original_disease; non_disease_dataset];
    full_data(:,2:6) = normalize(full_data(:,2:6), 'range'); %Scaling continuous variables
    between 0 and 1
synthetic_original_disease = full_data(find(full_data(:,7) == 1), :);
non_disease_dataset = full_data(find(full_data(:,7) == 0), :);

Q = size(synthetic_original_disease,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,~,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_disease = synthetic_original_disease(trainInd,:);
test_disease = synthetic_original_disease(testInd,:);

```

```

% train_disease.Properties.VariableNames = {'Var2', 'Var3', 'Var4', 'Var5', 'Var6',
'Var7','Var8','Var9','Var10','Var11','Var12'};
% test_disease.Properties.VariableNames = {'Var2', 'Var3', 'Var4', 'Var5', 'Var6',
'Var7','Var8','Var9','Var10','Var11','Var12'};

Q = size(non_disease_dataset,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,valInd,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_no_disease = non_disease_dataset(trainInd,:);
test_no_disease = non_disease_dataset(testInd,:);

training = [train_disease; train_no_disease];
test = [test_disease; test_no_disease];
test = test(randperm(size(test,1)),:);
training = training(randperm(size(training,1)),:);

test = array2table(test);
training = array2table(training);
test.Properties.VariableNames = {'Var1','Var2', 'Var3', 'Var4', 'Var5', 'Var6', 'Var7'};
training.Properties.VariableNames = {'Var1','Var2', 'Var3', 'Var4', 'Var5', 'Var6', 'Var7'};

[test_results, train_results] = smote_testing(test, training)

```

## 6. CODE TO TRAIN AND TEST ML MODELS

```

%%%%%%%%%%
% Created on: 05/01/2019
% Input: Internally called from obj1a_matlab_2.m
% Output: Further calls other matlab functions
% Author: Ridhi Deo
% File name: obj1a_matlab_2a.m (R2018b [65]) )
% Description: This code is called internally from obj1a_matlab_2.m. This code is used to train
and test five machine learning models and compute their performances. It outputs the
performances back to its parent code: obj1a_matlab_2.m
%%%%%%%%%%

```

```

function [test_results, train_results] = smote_testing(test, training)

```

```

%% Fine Gaussian SVM
[fine_gauss, val_acc_fine_gauss] = smote_fine_gauss(training);
yfit_1 = fine_gauss.predictFcn(test(:,1:end-1));
yfit_1(:,2) = test.Var7;
g1 = yfit_1(:,2)'; %Known values - Ground Truth
g2 = yfit_1(:,1)'; % predicted values

```

```

figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Fine Gaussian SVM')
[X,Y,T,AUC] = perfcure(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Fine Gaussian SVM')
txt = ['AUC for Fine Gaussian is ',num2str(AUC)];
text(0.5,0.9,txt)
clear X Y T AUC;
cp_1 = classperf(g1,g2);
cp_1_accuracy = cp_1.CorrectRate;

%% Medium Gaussian SVM
[med_gauss, val_acc_med_gauss] = smote_med_gauss(training);
yfit_2 = med_gauss.predictFcn(test(:,1:end-1));
yfit_2(:,2) = test.Var7;
g1 = yfit_2(:,2)'; %Known values - Ground Truth
g2 = yfit_2(:,1)'; % predicted values
figure
plotconfusion(g1,g2), title('Medium Gaussian SVM')
[X,Y,T,AUC] = perfcure(g1,g2,'1');
figure
plot(X,Y), title('Medium Gaussian SVM');
txt = ['AUC for Medium Gaussian is ',num2str(AUC)];
text(0.5,0.9,txt)
clear X Y T AUC;
cp_2 = classperf(g1,g2);
cp_2_accuracy = cp_2.CorrectRate;

%% Ensemble - Bagged Trees
[bagged_trees, val_acc_bagged_trees] = smote_bagged_trees(training);
yfit_3 = bagged_trees.predictFcn(test(:,1:end-1));
yfit_3(:,2) = test.Var7;
g1 = yfit_3(:,2)'; %Known values - Ground Truth
g2 = yfit_3(:,1)'; % predicted values
figure
plotconfusion(g1,g2), title('Bagged Trees')
[X,Y,T,AUC] = perfcure(g1,g2,'1');
figure
plot(X,Y), title('Bagged Trees');
txt = ['AUC for Bagged Trees is ',num2str(AUC)];
text(0.5,0.9,txt)
clear X Y T AUC;
cp_3 = classperf(g1,g2);
cp_3_accuracy = cp_3.CorrectRate;

%% Ensemble RUS boosted Trees

```

```

[RUS, val_acc_RUS_boosted] = smote_RUS(training);
yfit_4 = RUS.predictFcn(test(:,1:end-1));
yfit_4(:,2) = test.Var7;
g1 = yfit_4(:,2)'; %Known values - Ground Truth
g2 = yfit_4(:,1)'; % predicted values
figure
plotconfusion(g1,g2), title('Ensemble RUS boosted Trees')
[X,Y,T,AUC] = perfcurve(g1,g2,'1');
figure
plot(X,Y), title('Ensemble RUS boosted Trees');
txt = ['AUC for RUS is ',num2str(AUC)];
text(0.5,0.9,txt)
clear X Y T AUC;
cp_4 = classperf(g1,g2);
cp_4_accuracy = cp_4.CorrectRate;

```

%% Ensemble Gentle boost

```

[gentle, val_acc_gentle] = smote_gentle(training);
yfit_5 = gentle.predictFcn(test(:,1:end-1));
yfit_5(:,2) = test.Var7;
g1 = yfit_5(:,2)'; %Known values - Ground Truth
g2 = yfit_5(:,1)'; % predicted values
figure
plotconfusion(g1,g2), title('Ensemble Gentle boost')
[X,Y,T,AUC] = perfcurve(g1,g2,'1');
figure
plot(X,Y), title('Ensemble Gentle boost');
txt = ['AUC for Gentle boost is ',num2str(AUC)];
text(0.5,0.9,txt)
clear X Y T AUC;
cp_5 = classperf(g1,g2);
cp_5_accuracy = cp_5.CorrectRate;

```

%% Ensemble ADA boost

```

[ADA, val_acc_ADA] = smote_ADA(training);
yfit_6 = ADA.predictFcn(test(:,1:end-1));
yfit_6(:,2) = test.Var7;
g1 = yfit_6(:,2)'; %Known values - Ground Truth
g2 = yfit_6(:,1)'; % predicted values
figure
plotconfusion(g1,g2), title('Ensemble ADA boost')
[X,Y,T,AUC] = perfcurve(g1,g2,'1');

```

```

figure
plot(X,Y), title('Ensemble ADA boost');
txt = ['AUC for ADA boost is ',num2str(AUC)];
text(0.5,0.9,txt)
clear X Y T AUC;
cp_6 = classperf(g1,g2);
cp_6_accuracy = cp_6.CorrectRate;

%% Tabulated results
test_results = table(cp_1_accuracy, cp_2_accuracy, cp_3_accuracy, cp_4_accuracy,
cp_5_accuracy, cp_6_accuracy);
test_results.Properties.VariableNames = {'Fine_Gaussian', 'Medium_Gaussian', 'Bagged_Trees',
'RUS', 'Gentle_Boost', 'ADA_boost'};

train_results = table(val_acc_fine_gauss, val_acc_med_gauss, val_acc_bagged_trees,
val_acc_RUS_boosted, val_acc_gentle, val_acc_ADA);
train_results.Properties.VariableNames = {'Fine_Gaussian', 'Medium_Gaussian', 'Bagged_Trees',
'RUS', 'Gentle_Boost', 'ADA_boost'};

```

#### **A. CODE TO TRAIN FINE GAUSSIAN SVM MODEL**

```

%%%%%%%%%%
% Created on: 05/01/2019
% Input: Internally called from obj1a_matlab_2a
% Output: Trained Fine Gaussian SVM model
% Author: MATLAB Auto Generation implemented by Ridhi Deo
% File name: obj1a_matlab_2b.m (R2018b [65]) )
% Description: Used to train the model fine gaussian SVM. Outputs the trained model back to
obj1a_matlab_2a.m
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = smote_fine_gauss(trainingData)

% Auto-generated by MATLAB on 01-May-2019 11:12:40

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

```

```

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 0.61, ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
trainedClassifier.ClassificationSVM = classificationSVM;
    trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2018b.';
    trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
    original training data. \nAdditional variables are ignored. \n \nFor more information, see
    <a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
    "appclassification_exportmodeltoworkspace")">How to predict using an exported
    model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

```

```
% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');
```

## **B. CODE TO TRAIN MEDIUM GAUSSIAN SVM**

```
%%%%%%%%%%
% Created on: 05/01/2019
% Input: Internally called from obj1a_matlab_2a
% Output: Trained Medium Gaussian SVM model
% Author: MATLAB Auto Generation implemented by Ridhi Deo
% File name: obj1a_matlab_2c (R2018b [65]) )
% Description: Used to train the model medium gaussian SVM. Outputs the trained model back
to obj1a_matlab_2a
%%%%%%%%%%
```

```
function [trainedClassifier, validationAccuracy] = smote_med_gauss(trainingData)
```

```
% Auto-generated by MATLAB on 01-May-2019 11:16:26
```

```
% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 2.4, ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));
```

```

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
trainedClassifier.ClassificationSVM = classificationSVM;
    trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2018b.';
    trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
    original training data. \nAdditional variables are ignored. \n \nFor more information, see
    <a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
    "appclassification_exportmodeltoworkspace")">How to predict using an exported
    model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## C. CODE TO TRAIN BAGGED TREES

```

%%%%%%%%%%
% Created on: 05/01/2019
% Input: Internally called from obj1a_matlab_2a
% Output: Trained bagged trees model
% Author: MATLAB Auto Generation implemented by Ridhi Deo
%File name: obj1a_matlab_2d (R2018b [65]) )
%Description: Used to train the model bagged trees. Outputs the trained model back to
obj1a_matlab_2a
%%%%%%%%%%

```



```

function [trainedClassifier, validationAccuracy] = smote_bagged_trees(trainingData)

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateTree(...
    'MaxNumSplits', 8902);
classificationEnsemble = fitcensemble(...
    predictors, ...
    response, ...
    'Method', 'Bag', ...
    'NumLearningCycles', 30, ...
    'Learners', template, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
ensemblePredictFcn = @(x) predict(classificationEnsemble, x);
trainedClassifier.predictFcn = @(x) ensemblePredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
trainedClassifier.ClassificationEnsemble = classificationEnsemble;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2018b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
"trainedModel". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;

```

```

predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationEnsemble, 'KFold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## D. CODE TO TRAIN ADA BOOSTED TREES

```

%%%%%%%%%%
% Created on: 05/01/2019
% Input: Internally called from obj1a_matlab_2a
% Output: Trained ADA model
% Author: MATLAB Auto Generation implemented by Ridhi Deo
% File name: obj1a_matlab_2e (R2018b [65]) )
% Description: Used to train the ADA model. Outputs the trained model back to
obj1a_matlab_2a
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = smote_ADA(trainingData)

% Auto-generated by MATLAB on 01-May-2019 11:20:29

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateTree(...
    'MaxNumSplits', 20);
classificationEnsemble = fitcensemble(...
    predictors, ...

```

```

response, ...
'Method', 'AdaBoostM1', ...
'NumLearningCycles', 30, ...
'Learners', template, ...
'LearnRate', 0.1, ...
'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
ensemblePredictFcn = @(x) predict(classificationEnsemble, x);
trainedClassifier.predictFcn = @(x) ensemblePredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
trainedClassifier.ClassificationEnsemble = classificationEnsemble;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2018b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
"trainedModel". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationEnsemble, 'Kfold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## E. CODE TO TRAIN GENTLE BOOSTED TREES

```
%%%%%%%%%%
% Created on: 05/01/2019
% Input: Internally called from obj1a_matlab_2a
% Output: Trained Gentle Boosted model
% Author: MATLAB Auto Generation implemented by Ridhi Deo (R2018b [65]) )
%File name: obj1a_matlab_2f
%Description: Used to train the gentle boost model. Outputs the trained model back to
obj1a_matlab_2a
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = smote_gentle(trainingData)

% Auto-generated by MATLAB on 01-May-2019 11:19:41

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateTree(...
    'MaxNumSplits', 20);
classificationEnsemble = fitcensemble(...
    predictors, ...
    response, ...
    'Method', 'GentleBoost', ...
    'NumLearningCycles', 30, ...
    'Learners', template, ...
    'LearnRate', 0.1, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
ensemblePredictFcn = @(x) predict(classificationEnsemble, x);
trainedClassifier.predictFcn = @(x) ensemblePredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
```

```

trainedClassifier.ClassificationEnsemble = classificationEnsemble;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2018b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
"trainedModel". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Var1', 'Var2', 'Var3', 'Var4', 'Var5', 'Var6'};
predictors = inputTable(:, predictorNames);
response = inputTable.Var7;
isCategoricalPredictor = [false, false, false, false, false, false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationEnsemble, 'KFold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

*The flow of code is as follows:*

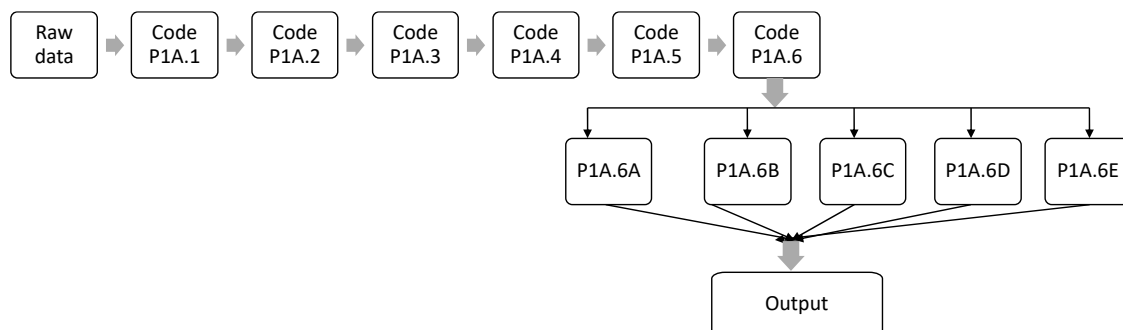


Figure P1.A.1: Figure outlining the flow of code used in this research objective (1A)

## APPENDIX B. - CODE FOR OBJECTIVE 1B

### 1. SAS CODE TO KEEP VARIABLES OF INTEREST AND DISCARD THE REST

```
%%%%%%%%%
% Created on: 03/21/21
% Input: Raw data from NHANES
% Output: Data with only variables of interest, specific to objective 1B
% Author: Ridhi Deo
% File name: obj1b_sas_1.sas
% Description: Used eliminate the variables that are not required and to only keep the variables
of interest from the raw datasets. This program was developed using SAS 2019 [64].
%%%%%%%%%

% set the data path and choose the variables to keep. Variable codes are as provided by
% NHANESIII

LIBNAME NH "Raw data path";

data adult;
set NH.adult;
keep SEQN HSAGEIR HSSEX DMARETHN HAD1 HAD6 HAD10;
proc sort; by seqn; run;

data lab;
set NH.lab;
keep SEQN AHP HBP SSP SAP HCP DHP NAPSI SKPSI CLPSI C3PSI SCPSI
PSPSI UAPSI G1P G2P BUPSI TBPSI CEPsi SFPSI CHPSI TRPSI ASPSI ATPSI
GGPSI LDPSI APPSI TPPSI AMPSI GBPSI OSPSI GHP GHPMETH G1PSI
G1PCODE G2PSI C1PSI C2PSI I1PSI I2PSI UDPSI URPSI UBP UIP PLPSI PVPSI
PBPSI FEPSI VBPSI VCPSI ICPSI CAPSI SEPSI VAPSI VEPSI ACPSI BCPSI
TCPSI TGPSI LCPSI HDPSI AAPSI ABPSI LPPSI;
proc sort; by seqn; run;

data exam;
set NH.exam;
keep SEQN PEP6DR BMPBMI BMPWAIST MAPA1 MAPA2A MAPA2B MAPA3
MAPE1 MAPE2 MAPE4;
proc sort; by seqn; run;

data HGUHS;
set NH.HGUHS;
keep SEQN GUPHSQC GUPHSLKC GUPHSPB GUPHSDBA GUPHSVW
GUPHSDGB GUPHSPF GUPHSPFR GUPHSC GUPHSREV;
proc sort; by seqn; run;
```

```

proc contents data = NH.adult;
run;

proc contents data = NH.lab;
run;

proc contents data = NH.exam;
run;

proc contents data = NH.HGUHS;
run;

```

## 2. SAS CODE TO MERGE DATASETS

```

% % % % % % %
% Created on: 03/21/21
% Input: Processed data with only variables of interest, specific to objective 1B
% Output: Multiple datasets of interest merged into one dataset
% Author: Ridhi Deo
% File name: obj1b_sas_2.sas
% Description: Used to combine different datasets of interest into one. This program was
developed using SAS 2019 [64].
% % % % % % %

% Sorting data by the sequential number
proc sort data=work.adult;
    by SEQN;
proc sort data=work.lab;
    by SEQN;
proc sort data=work.exam;
    by SEQN;
proc sort data=work.hguhs;
    by SEQN;
%Merging data using the sequential number
data NH.merged;
merge work.adult
      work.lab
      work.exam
      work.hguhs;
    by SEQN;

proc contents data = NH.merged varnum;
proc means data=NH.merged N Nmiss min max maxdec=2;
run;

```

### 3. MATLAB CODE FOR INITIAL DATA PROCESSING, SPLIT INTO MALE, FEMALE SUB DATASETS

```
%%%%%%%%%
% Created on: 03/21/21
% Input: Merged dataset from SAS
% Output: Processed data, split into male and female sub-datasets
% Author: Ridhi Deo
% File name: Obj1b_matlab_1.m (R2020b [65])
% Description: This code was written to process data and split it into male and female sub-
datasets.
%%%%%%%%%

clc
clear all;
%% Data import

data7 = data7(:,[1,3,4,24,47,48,51,58,61,72,78:80,88 ]);

data7.Properties.VariableNames{'HSAGEIR'} = 'Age';
data7.Properties.VariableNames{'HSSEX'} = 'Sex';
data7.Properties.VariableNames{'BMPBMI'} = 'BMI';
data7.Properties.VariableNames{'MAPE1'} = 'Alcohol_12_life';
data7.Properties.VariableNames{'MAPE2'} = 'Alcohol_12_last_year';
data7.Properties.VariableNames{'MAPE4'} = 'Drinks_per_day';
data7.Properties.VariableNames{'GUPHSPFR'} = 'HS';
data7.Properties.VariableNames{'ATPSI'} = 'ALT';
data7.Properties.VariableNames{'ASPSI'} = 'AST';
data7.Properties.VariableNames{'APPSI'} = 'ASP';
data7.Properties.VariableNames{'G1P'} = 'Plasma_glucose_1';
data7.Properties.VariableNames{'G2P'} = 'Plasma_glucose_2';
data7.Properties.VariableNames{'HDPST'} = 'HDL';

%% Alcohol data columns processing
    % Filling in missing data for the 12 drinks per year column with information from 12
    drinks in life column
% If a person has not had 12 drinks in their lifetime, the response on the
% variable 12 drinks in past year are missing
% To fix that, individuals who have not had 12 drinks in their life will
% have 0s on the column 12 drinks in past year
    % Same logic applies - making all those who have not had 12 drinks in their life 0 in the
    column drinks per day
for i = 1: size(data7,1)
    if (data7.Alcohol_12_life(i) == 2)
```



```

        data7.Alcohol_12_last_year(i) = 0;
        data7.Drinks_per_day(i) = 0;
    end
end

for i = 1: size(data7,1)
    if (data7.Alcohol_12_last_year(i) == 2)
        data7.Drinks_per_day(i) = 0;
    end
end

%% Cleaning up all the junk data (represented as 888 or 8888 or 999 etc.) withing
    variables of interest
% The information was referred from NHANES 3 documentation
% Since we have not used any youth data, all NaNs in the Age column could
% correspond to that
idx_age = find(isnan(data7.Age));
data7(idx_age,:) = []; %13,149 samples are eliminated in this step
Extra_Hb1AC(idx_age,:) = [];
clear idx_age;

% Sex
% No missing or junk data

% Plasma glucose
% G1P
% 88888 = blank but applicable
data7.Plasma_glucose_1(data7.Plasma_glucose_1 == 88888) = NaN;

% G2P
% 88888 = blank but applicable
data7.Plasma_glucose_2(data7.Plasma_glucose_2 == 88888) = NaN;

% AST
% 888 Blank but applicable
data7.AST(data7.AST == 888) = NaN;

% ALT
% 888 Blank but applicable
data7.ALT(data7.ALT == 888) = NaN;

% ASP
% 888 Blank but applicable
data7.ASP(data7.ASP == 888) = NaN;
data7.ASP(data7.ASP == 8888) = NaN;

```

```

% BMI
% 8888 was found as junk data. Although I didnt see this on the website for
% NHANES, it is removed because 8888 is not appropriate BMI
data7.BMI(data7.BMI == 8888) = NaN;

% HS
% 7    Image is present, but ungradable
% 8    No image
data7.HS(data7.HS == 7) = NaN;
data7.HS(data7.HS == 8) = NaN;

% HDL
data7.HDL(data7.HDL == 8888) = NaN;

% MAPE1 In your entire life, have you had at least 12 drinks of any kind of alcoholic beverage?
Do not count small tastes.
% 8 - Blank but applicable, 9 - dont know.
data7.Alcohol_12_life(data7.Alcohol_12_life == 8) = NaN;
data7.Alcohol_12_life(data7.Alcohol_12_life == 9) = NaN;

% MAPE2 In the past 12 months did you
%have at least 12 drinks of any kind of alcoholic beverage?
% 8 - Blank but applicable, 9 - dont know.
data7.Alcohol_12_last_year(data7.Alcohol_12_last_year == 8) = NaN;
data7.Alcohol_12_last_year(data7.Alcohol_12_last_year == 9) = NaN;

% MAPE4 On the average, on the days that you drank alcohol, how many drinks did you have a
day? (By a drink, I mean a 12-oz beer, a 4-oz glass of wine, or an ounce of liquor.)
% 888 - Blank but applicable, 999 - dont know.
data7.Drinks_per_day(data7.Drinks_per_day == 888) = NaN;
data7.Drinks_per_day(data7.Drinks_per_day == 999) = NaN;

%% After executing the code up to this point, I have visually examined all
% the data columns to ensure junk data is removed - 03/21/21
    %% Eliminating missing data from HS - we need to eliminate this data because this is our
    output variable and ground truth
HS_missing_idx = find(isnan(data7.HS)); %6,194 cases of pmissing HS data
data7(HS_missing_idx,:) = []; %This step eliminates the 6,194 cases of missing HS data
clear HS_missing_idx;

%% Delete missing ALT and AST information
ALT_missing_idx = find(isnan(data7.ALT)); %773 cases of missing ALT data
data7(ALT_missing_idx,:) = [];

    AST_missing_idx = find(isnan(data7.AST)); %0 cases of missing AST data after
    removing ALT missing samples

```

```

data7(AST_missing_idx,:) = [];

    ASP_missing_idx = find(isnan(data7.ASP)); %2 cases of missing ASP data after
    removing ALT missing samples
data7(ASP_missing_idx,:) = [];

% IF there are NaNs in G1P, fill them with G2P. If both G1P and G2P are
% NaNs, then delete the sample
for i = 1:size(data7,1)
    if(isnan(data7.Plasma_glucose_1(i)))
        if(isnan(data7.Plasma_glucose_2(i)))
            idx_pg(i) = i;
        else
            data7.Plasma_glucose_1(i) = data7.Plasma_glucose_2(i);
        end
    end
end
end

data7.Plasma_glucose_2 = [];
    Plasma_glucose_idx = find(isnan(data7.Plasma_glucose_1)); %25 cases of missing
    plasma glucose samples after combining G1P and G2P
data7(Plasma_glucose_idx,:) = [];

BMI_idx = find(isnan(data7.BMI)); %20 cases of missing BMI
data7(BMI_idx,:) = [];

HDL_idx = find(isnan(data7.HDL)); %121 cases of missing HDL
data7(HDL_idx,:) = [];

    clear ALT_missing_idx AST_missing_idx ASP_missing_idx Plasma_glucose_idx
    idx_pg BMI_idx HDL_idx;

%% Split datasets into HS and non-HS

    data7.HS(data7.HS == 1) = 0; % 1 is Normal - Mild as per NHANES. Changing it to 0 to
    indicate no risk
    data7.HS(data7.HS == 2) = 1; % 2 is Moderate - Severe as per NHANES. Changing it to
    1 to indicate risk

idx_disease = data7.HS == 1;
dataset_HS = data7(idx_disease,:); %2,956

idx_non_disease = data7.HS == 0;
dataset_non_HS = data7(idx_non_disease,:); % 9,959
clear idx_disease idx_non_disease;

```

```

%% Split further into Male HS, Non-HS and Female HS, non-HS
    dataset_HS_male = dataset_HS(dataset_HS.Sex == 1, :); % Sex = 1 is male and 2 is
    female per NHANES documentation
%1,517
dataset_HS_female = dataset_HS(dataset_HS.Sex == 2,:); %1,439

dataset_non_HS_male = dataset_non_HS(dataset_non_HS.Sex == 1,:); %4,533
dataset_non_HS_female = dataset_non_HS(dataset_non_HS.Sex == 2,:); %5,426

%% Apply exclusion criteria for alcohol
% HS and No-HS male exclusion criteria - > 21 drinks/week should be
% excluded
k = 1;
for i = 1: size(dataset_HS_male,1)
    if(dataset_HS_male.Sex(i) == 1 && dataset_HS_male.Drinks_per_day(i) > 3)
        idx_HS_men(k) = i;
        k = k + 1;
    end
end
dataset_HS_male(idx_HS_men,:) = []; %437 samples are eliminated
clear k idx_HS_men;

j = 1;
for i = 1: size(dataset_non_HS_male,1)
    if(dataset_non_HS_male.Sex(i) == 1 && dataset_non_HS_male.Drinks_per_day(i) > 3)
        idx_non_HS_men(j) = i;
        j = j + 1;
    end
end
dataset_non_HS_male(idx_non_HS_men,:) = []; %1,228 samples are elminiated
clear j idx_non_HS_men;

% HS and No-HS female exclusion criteria - > 14 drinks/week should be
% excluded
k = 1;
for i = 1: size(dataset_HS_female,1)
    if(dataset_HS_female.Sex(i) == 2 && dataset_HS_female.Drinks_per_day(i) > 2)
        idx_HS_women(k) = i;
        k = k + 1;
    end
end
dataset_HS_female(idx_HS_women,:) = []; %195 are eliminated
clear k idx_HS_women;

j = 1;

```

```

for i = 1: size(dataset_non_HS_female,1)
    if(dataset_non_HS_female.Sex(i) == 2 && dataset_non_HS_female.Drinks_per_day(i)
        > 2)
        idx_non_HS_women(j) = i;
        j = j + 1;
    end
end
dataset_non_HS_female(idx_non_HS_women,:) = []; %716 are eliminated
clear j idx_non_HS_women;

%% Delete data related to drinks per day
idx_male_HS_drinks = find(isnan(dataset_HS_male.Drinks_per_day));
dataset_HS_male(idx_male_HS_drinks,:) = []; % 1038 x 13

idx_male_non_HS_drinks = find(isnan(dataset_non_HS_male.Drinks_per_day));
dataset_non_HS_male(idx_male_non_HS_drinks,:) = []; %3,177 x 13

idx_female_HS_drinks = find(isnan(dataset_HS_female.Drinks_per_day));
dataset_HS_female(idx_female_HS_drinks,:) = []; %1,212 x 13

idx_female_non_HS_drinks = find(isnan(dataset_non_HS_female.Drinks_per_day));
dataset_non_HS_female(idx_female_non_HS_drinks,:) = []; %4,591 x 13

clear idx_female_HS_drinks idx_female_non_HS_drinks idx_male_HS_drinks
idx_male_non_HS_drinks;

%% Delete alcohol columns from 4 datasets
dataset_HS_male.Alcohol_12_last_year = [];
dataset_HS_male.Alcohol_12_life = [];
dataset_HS_male.Drinks_per_day = [];

dataset_HS_female.Alcohol_12_last_year = [];
dataset_HS_female.Alcohol_12_life = [];
dataset_HS_female.Drinks_per_day = [];

dataset_non_HS_male.Alcohol_12_last_year = [];
dataset_non_HS_male.Alcohol_12_life = [];
dataset_non_HS_male.Drinks_per_day = [];

dataset_non_HS_female.Alcohol_12_last_year = [];
dataset_non_HS_female.Alcohol_12_life = [];
dataset_non_HS_female.Drinks_per_day = [];

%% Delete sex column from all 4 datasets
dataset_HS_male.Sex = [];
dataset_HS_female.Sex = [];

```

```
dataset_non_HS_male.Sex = [];
dataset_non_HS_female.Sex = [];
```

#### 4. MATLAB CODE FOR SEX SPECIFIC PROCESSED – UNDER SAMPLING

##### A. MALE SPECIFIC CODE

```
%%%%%%%%%%
% Created on: 03/21/21
% Input: Male Sub-Dataset
% Output: Training and test datasets for male population
% Author: Ridhi Deo
% File name: Obj1b_matlab_2a.m (R2020b [65]) )
% Description: This code was written specifically for male population. Data were processed,
four derived features were created and populated with normalized data. Undersampling was
conducted and data were split into training and test in a class balanced way.
%%%%%%%%%%
```

```
% Creating derived variables to store normalized values
```

```
ULN_ALT = 33;
%% Liver foundation vicki shah: Female: AST ULN: 20 IU/L
ULN_AST = 30;
```

```
ALT_percent = zeros(size(dataset_HS_male, 1),1);
dataset_HS_male(:,end+1) = array2table(ALT_percent);
dataset_HS_male.Properties.VariableNames{'Var10'} = 'ALT_percent';
```

```
ALT_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(ALT_percent);
dataset_non_HS_male.Properties.VariableNames{'Var10'} = 'ALT_percent';
```

```
AST_percent = zeros(size(dataset_HS_male, 1),1);
dataset_HS_male(:,end+1) = array2table(AST_percent);
dataset_HS_male.Properties.VariableNames{'Var11'} = 'AST_percent';
```

```
AST_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(AST_percent);
dataset_non_HS_male.Properties.VariableNames{'Var11'} = 'AST_percent';
```

```
Plasma_glucose_percent = zeros(size(dataset_HS_male, 1),1);
dataset_HS_male(:,end+1) = array2table(Plasma_glucose_percent);
dataset_HS_male.Properties.VariableNames{'Var12'} = 'Plasma_glucose_percent';
```

```
Plasma_glucose_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(Plasma_glucose_percent);
```

```

dataset_non_HS_male.Properties.VariableNames{'Var12'} = 'Plasma_glucose_percent';

BMI_percent = zeros(size(dataset_HS_male, 1),1);
dataset_HS_male(:,end+1) = array2table(BMI_percent);
dataset_HS_male.Properties.VariableNames{'Var13'} = 'BMI_percent';

BMI_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(BMI_percent);
dataset_non_HS_male.Properties.VariableNames{'Var13'} = 'BMI_percent';
%% Computing % values
for i = 1: size(dataset_HS_male,1)
    dataset_HS_male.ALT_percent(i) = ((dataset_HS_male.ALT(i) -
    ULN__ALT)/ULN__ALT)*100;
    dataset_HS_male.AST_percent(i) = ((dataset_HS_male.AST(i) -
    ULN__AST)/ULN__AST)*100;
    dataset_HS_male.Plasma_glucose_percent(i) =
    ((dataset_HS_male.Plasma_glucose_1(i) - 120)/120)*100;
    dataset_HS_male.BMI_percent(i) = ((dataset_HS_male.BMI(i) - 25)/25)*100;

end

for i = 1: size(dataset_non_HS_male,1)
    dataset_non_HS_male.ALT_percent(i) = ((dataset_non_HS_male.ALT(i) -
    ULN__ALT)/ULN__ALT)*100;
    dataset_non_HS_male.AST_percent(i) = ((dataset_non_HS_male.AST(i) -
    ULN__AST)/ULN__AST)*100;
    dataset_non_HS_male.Plasma_glucose_percent(i) =
    ((dataset_non_HS_male.Plasma_glucose_1(i) - 120)/120)*100;
    dataset_non_HS_male.BMI_percent(i) = ((dataset_non_HS_male.BMI(i) - 25)/25)*100;
end

%% Converting % values to 0 if they are negative - see the top of this script for details

for i = 1: size(dataset_HS_male,1)
    if(dataset_HS_male.ALT_percent(i) <= 0)
        dataset_HS_male.ALT_percent(i) = 0;
    end
    if(dataset_HS_male.AST_percent(i) <= 0)
        dataset_HS_male.AST_percent(i) = 0;
    end
    if(dataset_HS_male.Plasma_glucose_percent(i) <= 0)
        dataset_HS_male.Plasma_glucose_percent(i) = 0;
    end
    if(dataset_HS_male.BMI_percent(i) <= 0)
        dataset_HS_male.BMI_percent(i) = 0;
    end
end

```

end

```
for i = 1: size(dataset_non_HS_male,1)
    if(dataset_non_HS_male.ALT_percent(i) <= 0)
        dataset_non_HS_male.ALT_percent(i) = 0;
    end
    if(dataset_non_HS_male.AST_percent(i) <= 0)
        dataset_non_HS_male.AST_percent(i) = 0;
    end
    if(dataset_non_HS_male.Plasma_glucose_percent(i) <= 0)
        dataset_non_HS_male.Plasma_glucose_percent(i) = 0;
    end
    if(dataset_non_HS_male.BMI_percent(i) <= 0)
        dataset_non_HS_male.BMI_percent(i) = 0;
    end
end
```

%% Randomly select samples without replacement

% Note that MATLAB's datasample function has replace = true as default

```
dataset_non_HS_male_reduced = datasample(dataset_non_HS_male,
    size(dataset_HS_male,1), 'Replace', false);
```

%% Split into training and test

```
Q = size(dataset_HS_male,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,~,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_disease_male = dataset_HS_male(trainInd,:);
test_disease_male = dataset_HS_male(testInd,:);
```

```
Q = size(dataset_non_HS_male_reduced,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,valInd,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_no_disease_male = dataset_non_HS_male_reduced(trainInd,:);
test_no_disease_male = dataset_non_HS_male_reduced(testInd,:);
```

```
training_male = [train_disease_male; train_no_disease_male];
test_male = [test_disease_male; test_no_disease_male];
test_male = test_male(randperm(size(test_male,1)),:);
training_male = training_male(randperm(size(training_male,1)),:);
```



```

%% Remove SEQN from training and test datasets
training_male.SEQN = [];
test_male.SEQN = [];
%% Reorder training and test datasets to have HS as the end variable
training_male = [training_male(:,1:7) training_male(:,9:12) training_male(:, 8)];
test_male = [test_male(:,1:7) test_male(:,9:12) test_male(:, 8)];

```

## B. FEMALE SPECIFIC CODE

```

%% % % % % % % %
% Created on: 03/21/21
% Input: Female Sub-Dataset
% Output: Training and test datasets for female population
% Author: Ridhi Deo
% File name: Obj1b_matlab_2b.m
% Description: This code was written specifically for female population. Data were processed,
four derived features were created and populated with normalized data. Undersampling was
conducted and data were split into training and test in a class balanced way.
% % % % % % %

```

```

% Creating derived variables to store normalized values

```

```

ULN_ALT = 25;
ULN_AST = 20;

```

```

ALT_percent = zeros(size(dataset_HS_female, 1),1);
dataset_HS_female(:,end+1) = array2table(ALT_percent);
dataset_HS_female.Properties.VariableNames{'Var10'} = 'ALT_percent';

```

```

ALT_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(ALT_percent);
dataset_non_HS_female.Properties.VariableNames{'Var10'} = 'ALT_percent';

```

```

AST_percent = zeros(size(dataset_HS_female, 1),1);
dataset_HS_female(:,end+1) = array2table(AST_percent);
dataset_HS_female.Properties.VariableNames{'Var11'} = 'AST_percent';

```

```

AST_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(AST_percent);
dataset_non_HS_female.Properties.VariableNames{'Var11'} = 'AST_percent';

```

```

Plasma_glucose_percent = zeros(size(dataset_HS_female, 1),1);
dataset_HS_female(:,end+1) = array2table(Plasma_glucose_percent);
dataset_HS_female.Properties.VariableNames{'Var12'} = 'Plasma_glucose_percent';

```

```

Plasma_glucose_percent = zeros(size(dataset_non_HS_female, 1),1);

```

```
dataset_non_HS_female(:,end+1) = array2table(Plasma_glucose_percent);
dataset_non_HS_female.Properties.VariableNames{'Var12'} = 'Plasma_glucose_percent';
```

```
BMI_percent = zeros(size(dataset_HS_female, 1),1);
dataset_HS_female(:,end+1) = array2table(BMI_percent);
dataset_HS_female.Properties.VariableNames{'Var13'} = 'BMI_percent';
```

```
BMI_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(BMI_percent);
dataset_non_HS_female.Properties.VariableNames{'Var13'} = 'BMI_percent';
for i = 1: size(dataset_HS_female,1)
    dataset_HS_female.ALT_percent(i) = ((dataset_HS_female.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;
    dataset_HS_female.AST_percent(i) = ((dataset_HS_female.AST(i) -
    ULN_AST)/ULN_AST)*100;
    dataset_HS_female.Plasma_glucose_percent(i) =
    ((dataset_HS_female.Plasma_glucose_1(i) - 120)/120)*100;
    dataset_HS_female.BMI_percent(i) = ((dataset_HS_female.BMI(i) - 25)/25)*100;
```

```
end
```

```
for i = 1: size(dataset_non_HS_female,1)
    dataset_non_HS_female.ALT_percent(i) = ((dataset_non_HS_female.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;
    dataset_non_HS_female.AST_percent(i) = ((dataset_non_HS_female.AST(i) -
    ULN_AST)/ULN_AST)*100;
    dataset_non_HS_female.Plasma_glucose_percent(i) =
    ((dataset_non_HS_female.Plasma_glucose_1(i) - 120)/120)*100;
    dataset_non_HS_female.BMI_percent(i) = ((dataset_non_HS_female.BMI(i) -
    25)/25)*100;
```

```
end
```

```
%% Converting % values to 0 if they are negative - see the top of this script for details
```

```
for i = 1: size(dataset_HS_female,1)
    if(dataset_HS_female.ALT_percent(i) <= 0)
        dataset_HS_female.ALT_percent(i) = 0;
    end
    if(dataset_HS_female.AST_percent(i) <= 0)
        dataset_HS_female.AST_percent(i) = 0;
    end
    if(dataset_HS_female.Plasma_glucose_percent(i) <= 0)
        dataset_HS_female.Plasma_glucose_percent(i) = 0;
    end
    if(dataset_HS_female.BMI_percent(i) <= 0)
        dataset_HS_female.BMI_percent(i) = 0;
```

```

    end
end

for i = 1: size(dataset_non_HS_female,1)
    if(dataset_non_HS_female.ALT_percent(i) <= 0)
        dataset_non_HS_female.ALT_percent(i) = 0;
    end
    if(dataset_non_HS_female.AST_percent(i) <= 0)
        dataset_non_HS_female.AST_percent(i) = 0;
    end
    if(dataset_non_HS_female.Plasma_glucose_percent(i) <= 0)
        dataset_non_HS_female.Plasma_glucose_percent(i) = 0;
    end
    if(dataset_non_HS_female.BMI_percent(i) <= 0)
        dataset_non_HS_female.BMI_percent(i) = 0;
    end
end

%% Randomly select samples without replacement
% Note that MATLAB's datasample function has replace = true as default

    dataset_non_HS_female_reduced = datasample(dataset_non_HS_female,
        size(dataset_HS_female,1), 'Replace', false);

%% Split into training and test
Q = size(dataset_HS_female,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,~,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_disease_female = dataset_HS_female(trainInd,:);
test_disease_female = dataset_HS_female(testInd,:);

Q = size(dataset_non_HS_female_reduced,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,valInd,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_no_disease_female = dataset_non_HS_female_reduced(trainInd,:);
test_no_disease_female = dataset_non_HS_female_reduced(testInd,:);

training_female = [train_disease_female; train_no_disease_female];
test_female = [test_disease_female; test_no_disease_female];
test_female = test_female(randperm(size(test_female,1)),:);
training_female = training_female(randperm(size(training_female,1)),:);

```

```

%% Remove SEQN from training and test datasets
training_female.SEQN = [];
test_female.SEQN = [];

%% Reorder training and test datasets to have HS as the end variable
training_female = [training_female(:,1:7) training_female(:,9:12) training_female(:, 8)];
test_female = [test_female(:,1:7) test_female(:,9:12) test_female(:, 8)];

```

## 5. MATLAB CODE FOR TRAINING AND TESTING ML MODELS

```

%% % % % % % % %
% Created on: 03/21/21
% Input: Training and test data
% Output: Model performances
% Author: Ridhi Deo
% File name: Obj1b_matlab_3.m (R2020b [65]) )
% Description: This code was written to train and test the models, then compute the model
performances and output them.
%% % % % % % % %

test = test_female; % Need to change this depending on male/female
training = training_female; % Need to change this depending on male/female

%% Model 1: fine tree
[mod_1, train_acc_1] = finetree2(training); % Training the model using training set
yfit_1 = mod_1.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_1(:,2) = table2array(test(:,end)); % Ground truth
g1 = yfit_1(:,2)'; % Transposed values of Known values - Ground Truth
g2 = yfit_1(:,1)'; % Transposed values of predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Fine Tree')
[tpr_1, fpr_1,~] = roc(g1, g2); % Extracting the true-positive and false-positive rates
sens_1 = tpr_1(1,2); % Calculating sensitivity
spec_1 = 1- fpr_1(1,2); % Calculating specificity
[X,Y,~,AUC_1] = perfcurve(g1,g2,'1'); % Extracting values to plot the AUC curve with
the AUC value
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Fine Tree')
txt = ['AUC for Fine Tree is ',num2str(AUC_1)];
text(0.5,0.9,txt)
clear X Y;
cp_1 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_1_accuracy = cp_1.CorrectRate;

```

```

%% Model 2: logistic regression
[mod_2, train_acc_2] = logisticregression2(training); % Training the model using
training set
yfit_2 = mod_2.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_2(:,2) = table2array(test(:,end));
g1 = yfit_2(:,2)'; %Known values - Ground Truth
g2 = yfit_2(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('logistic regression')
[tpr_2, fpr_2,~] = roc(g1, g2);
sens_2 = tpr_2(1,2);
spec_2 = 1- fpr_2(1,2);
[X,Y,~,AUC_2] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('logistic regression')
txt = ['AUC for logistic regression is ',num2str(AUC_2)];
text(0.5,0.9,txt)
clear X Y;
cp_2 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_2_accuracy = cp_2.CorrectRate;

%% Model 3: linear svm
[mod_3, train_acc_3] = linearsvm2(training); % Training the model using training set
yfit_3 = mod_3.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_3(:,2) = table2array(test(:,end));
g1 = yfit_3(:,2)'; %Known values - Ground Truth
g2 = yfit_3(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('linear svm')
[tpr_3, fpr_3,~] = roc(g1, g2);
sens_3 = tpr_3(1,2);
spec_3 = 1- fpr_3(1,2);
[X,Y,~,AUC_3] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('linear svm')
txt = ['AUC for linear svm is ',num2str(AUC_3)];
text(0.5,0.9,txt)
clear X Y;
cp_3 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_3_accuracy = cp_3.CorrectRate;

%% Model 4: quadratic svm
[mod_4, train_acc_4] = quadraticsvm2(training); % Training the model using training set

```

```

        yfit_4 = mod_4.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
        using the test dataset
    yfit_4(:,2) = table2array(test(:,end));
    g1 = yfit_4(:,2)'; %Known values - Ground Truth
    g2 = yfit_4(:,1)'; % predicted values
    figure %Plotting confusion matrix
    plotconfusion(g1,g2), title('quadratic svm')
    [tpr_4, fpr_4,~] = roc(g1, g2);
    sens_4 = tpr_4(1,2);
    spec_4 = 1- fpr_4(1,2);
    [X,Y,~,AUC_4] = perfcurve(g1,g2,'1');
    figure %Plotting ROC with AUC value printed on the graph
    plot(X,Y), title('quadratic svm')
    txt = ['AUC for quadratic svm is ',num2str(AUC_4)];
    text(0.5,0.9,txt)
    clear X Y;
    cp_4 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
    cp_4_accuracy = cp_4.CorrectRate;

%% Model 5: fine gaussian svm
    [mod_5, train_acc_5] = finegaussiansvm2(training); % Training the model using training
    set
    yfit_5 = mod_5.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
    using the test dataset
    yfit_5(:,2) = table2array(test(:,end));
    g1 = yfit_5(:,2)'; %Known values - Ground Truth
    g2 = yfit_5(:,1)'; % predicted values
    figure %Plotting confusion matrix
    plotconfusion(g1,g2), title('fine gaussian svm')
    [tpr_5, fpr_5,~] = roc(g1, g2);
    sens_5 = tpr_5(1,2);
    spec_5 = 1- fpr_5(1,2);
    [X,Y,~,AUC_5] = perfcurve(g1,g2,'1');
    figure %Plotting ROC with AUC value printed on the graph
    plot(X,Y), title('fine gaussian svm')
    txt = ['AUC for fine gaussian svm is ',num2str(AUC_5)];
    text(0.5,0.9,txt)
    clear X Y;
    cp_5 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
    cp_5_accuracy = cp_5.CorrectRate;

%% Model 6: medium gaussian svm
    [mod_6, train_acc_6] = mediumgaussiansvm2(training); % Training the model using
    training set
    yfit_6 = mod_6.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
    using the test dataset

```

```

yfit_6(:,2) = table2array(test(:,end));
g1 = yfit_6(:,2)'; %Known values - Ground Truth
g2 = yfit_6(:,1)'; % predicted values
figure %PLotting confusion matrix
plotconfusion(g1,g2), title('medium gaussian svm')
[tpr_6, fpr_6,~] = roc(g1, g2);
sens_6 = tpr_6(1,2);
spec_6 = 1- fpr_6(1,2);
[X,Y,~,AUC_6] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('medium gaussian svm')
txt = ['AUC for medium gaussian svm is ',num2str(AUC_6)];
text(0.5,0.9,txt)
clear X Y;
cp_6 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_6_accuracy = cp_6.CorrectRate;

%% Model 7: coarse gaussian svm
[mod_7, train_acc_7] = coarsegaussiansvm2(training); % Training the model using
training set
yfit_7 = mod_7.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_7(:,2) = table2array(test(:,end));
g1 = yfit_7(:,2)'; %Known values - Ground Truth
g2 = yfit_7(:,1)'; % predicted values
figure %PLotting confusion matrix
plotconfusion(g1,g2), title('coarse gaussian svm')
[tpr_7, fpr_7,~] = roc(g1, g2);
sens_7 = tpr_7(1,2);
spec_7 = 1- fpr_7(1,2);
[X,Y,~,AUC_7] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('coarse gaussian svm')
txt = ['AUC for coarse gaussian svm is ',num2str(AUC_7)];
text(0.5,0.9,txt)
clear X Y;
cp_7 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_7_accuracy = cp_7.CorrectRate;

%% Model 8: fine knn
[mod_8, train_acc_8] = fineknn2(training); % Training the model using training set
yfit_8 = mod_8.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_8(:,2) = table2array(test(:,end));
g1 = yfit_8(:,2)'; %Known values - Ground Truth

```

```

g2 = yfit_8(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('fine knn')
[tpr_8, fpr_8,~] = roc(g1, g2);
sens_8 = tpr_8(1,2);
spec_8 = 1- fpr_8(1,2);
[X,Y,~,AUC_8] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('fine knn')
txt = ['AUC for fine knn is ',num2str(AUC_8)];
text(0.5,0.9,txt)
clear X Y;
cp_8 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_8_accuracy = cp_8.CorrectRate;

%% Model 9: Medium knn
[mod_9, train_acc_9] = mediumknn2(training); % Training the model using training set
yfit_9 = mod_9.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_9(:,2) = table2array(test(:,end));
g1 = yfit_9(:,2)'; %Known values - Ground Truth
g2 = yfit_9(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Medium knn')
[tpr_9, fpr_9,~] = roc(g1, g2);
sens_9 = tpr_9(1,2);
spec_9 = 1- fpr_9(1,2);
[X,Y,~,AUC_9] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Medium knn')
txt = ['AUC for Medium knn is ',num2str(AUC_9)];
text(0.5,0.9,txt)
clear X Y;
cp_9 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_9_accuracy = cp_9.CorrectRate;

%% Model 10: Coarse knn
[mod_10, train_acc_10] = coarseknn2(training); % Training the model using training set
yfit_10 = mod_10.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_10(:,2) = table2array(test(:,end));
g1 = yfit_10(:,2)'; %Known values - Ground Truth
g2 = yfit_10(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Coarse knn')

```



```

[tpr_10, fpr_10,~] = roc(g1, g2);
sens_10 = tpr_10(1,2);
spec_10 = 1- fpr_10(1,2);
[X,Y,~,AUC_10] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Coarse knn')
txt = ['AUC for Coarse knn is ',num2str(AUC_10)];
text(0.5,0.9,txt)
clear X Y;
cp_10 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_10_accuracy = cp_10.CorrectRate;

%% Model 11: Cosine knn
[mod_11, train_acc_11] = cosineknn2(training); % Training the model using training set
yfit_11 = mod_11.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_11(:,2) = table2array(test(:,end));
g1 = yfit_11(:,2)'; %Known values - Ground Truth
g2 = yfit_11(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Cosine knn')
[tpr_11, fpr_11,~] = roc(g1, g2);
sens_11 = tpr_11(1,2);
spec_11 = 1- fpr_11(1,2);
[X,Y,~,AUC_11] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Cosine knn')
txt = ['AUC for Cosine knn is ',num2str(AUC_11)];
text(0.5,0.9,txt)
clear X Y;
cp_11 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_11_accuracy = cp_11.CorrectRate;

%% Model 12: Cubic knn
[mod_12, train_acc_12] = cubicknn2(training); % Training the model using training set
yfit_12 = mod_12.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_12(:,2) = table2array(test(:,end));
g1 = yfit_12(:,2)'; %Known values - Ground Truth
g2 = yfit_12(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Cubic knn')
[tpr_12, fpr_12,~] = roc(g1, g2);
sens_12 = tpr_12(1,2);
spec_12 = 1- fpr_12(1,2);

```

```

[X,Y,~,AUC_12] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Cubic knn')
txt = ['AUC for Cubic knn is ',num2str(AUC_12)];
text(0.5,0.9,txt)
clear X Y;
cp_12 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_12_accuracy = cp_12.CorrectRate;

%% Model 13: Weighted knn
[mod_13, train_acc_13] = weightedknn2(training); % Training the model using training
set
yfit_13 = mod_13.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_13(:,2) = table2array(test(:,end));
g1 = yfit_13(:,2)'; %Known values - Ground Truth
g2 = yfit_13(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Weighted knn')
[tpr_13, fpr_13,~] = roc(g1, g2);
sens_13 = tpr_13(1,2);
spec_13 = 1- fpr_13(1,2);
[X,Y,~,AUC_13] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Weighted knn')
txt = ['AUC for Weighted knn is ',num2str(AUC_13)];
text(0.5,0.9,txt)
clear X Y;
cp_13 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_13_accuracy = cp_13.CorrectRate;

%% Model 14: Boosted Trees
[mod_14, train_acc_14] = boostedtrees2(training); % Training the model using training
set
yfit_14 = mod_14.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_14(:,2) = table2array(test(:,end));
g1 = yfit_14(:,2)'; %Known values - Ground Truth
g2 = yfit_14(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Boosted Trees')
[tpr_14, fpr_14,~] = roc(g1, g2);
sens_14 = tpr_14(1,2);
spec_14 = 1- fpr_14(1,2);
[X,Y,~,AUC_14] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph

```

```

plot(X,Y), title('Boosted Trees')
txt = ['AUC for Boosted Trees is ',num2str(AUC_14)];
text(0.5,0.9,txt)
clear X Y;
cp_14 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_14_accuracy = cp_14.CorrectRate;

%% Model 15: Bagged Trees
[mod_15, train_acc_15] = baggedtrees2(training); % Training the model using training
set
yfit_15 = mod_15.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_15(:,2) = table2array(test(:,end));
g1 = yfit_15(:,2)'; % Known values - Ground Truth
g2 = yfit_15(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Bagged Trees')
[tpr_15, fpr_15,~] = roc(g1, g2);
sens_15 = tpr_15(1,2);
spec_15 = 1- fpr_15(1,2);
[X,Y,~,AUC_15] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Bagged Trees')
txt = ['AUC for Bagged Trees is ',num2str(AUC_15)];
text(0.5,0.9,txt)
clear X Y;
cp_15 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_15_accuracy = cp_15.CorrectRate;

%% Model 16: Subspace Discriminant
[mod_16, train_acc_16] = subspace2(training); % Training the model using training
set
yfit_16 = mod_16.predictFcn(test(:,1:end-1));
yfit_16(:,2) = table2array(test(:,end));
g1 = yfit_16(:,2)'; % Known values - Ground Truth
g2 = yfit_16(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Subspace Disc')
[tpr_16, fpr_16,~] = roc(g1, g2);
sens_16 = tpr_16(1,2);
spec_16 = 1- fpr_16(1,2);
[X,Y,~,AUC_16] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Subspace Disc')
txt = ['AUC for Subspace Disc is ',num2str(AUC_16)];
text(0.5,0.9,txt)

```

```

clear X Y;
cp_16 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_16_accuracy = cp_16.CorrectRate;

%% Model 17: RUS Boosted trees
[mod_17, train_acc_17] = rusboostedtrees2(training); % Training the model using
training set
yfit_17 = mod_17.predictFcn(test(:,1:end-1));
yfit_17(:,2) = table2array(test(:,end));
g1 = yfit_17(:,2); % Known values - Ground Truth
g2 = yfit_17(:,1); % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('RUS Boosted trees')
[tpr_17, fpr_17,~] = roc(g1, g2);
sens_17 = tpr_17(1,2);
spec_17 = 1- fpr_17(1,2);
[X,Y,~,AUC_17] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('RUS Boosted trees')
txt = ['AUC for RUS Boosted trees is ',num2str(AUC_17)];
text(0.5,0.9,txt)
clear X Y;
cp_17 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_17_accuracy = cp_17.CorrectRate;

%% Display results in a table
Model =
    {'Fine_Tree';'Logistic_Regression';'Linear_SVM';'Quadratic_SVM';'Fine_Gaussian_SV
M';...
    'Medium_Gaussian_SVM';'Coarse_Gaussian_SVM';'Fine_KNN';'Medium_KNN';'Coarse
_KNN';...
    'Cosine_KNN';'Cubic_KNN';'Weighted_KNN';'Ensemble_Boosted';'Ensemble_Bagged';...
    'Ensemble_Subspace_Disc';'Ensemble_RUS_Boosted_Trees'};

Training_Acc = [train_acc_1; train_acc_2; train_acc_3; train_acc_4; train_acc_5;...
    train_acc_6; train_acc_7; train_acc_8; train_acc_9; train_acc_10; train_acc_11;...
    train_acc_12; train_acc_13; train_acc_14; train_acc_15; train_acc_16; train_acc_17];
Test_Acc = [cp_1_accuracy; cp_2_accuracy; cp_3_accuracy; cp_4_accuracy;
    cp_5_accuracy;...
    cp_6_accuracy; cp_7_accuracy; cp_8_accuracy; cp_9_accuracy; cp_10_accuracy;
    cp_11_accuracy;...
    cp_12_accuracy; cp_13_accuracy; cp_14_accuracy; cp_15_accuracy; cp_16_accuracy;
    cp_17_accuracy];
AUC = [AUC_1; AUC_2; AUC_3; AUC_4; AUC_5;...
    AUC_6; AUC_7; AUC_8; AUC_9; AUC_10; AUC_11;...

```

```

AUC_12; AUC_13; AUC_14; AUC_15; AUC_16; AUC_17];
Sensitivity = [sens_1; sens_2; sens_3; sens_4; sens_5;...
sens_6; sens_7; sens_8; sens_9; sens_10; sens_11;...
sens_12; sens_13; sens_14; sens_15; sens_16; sens_17];
Specificity = [spec_1; spec_2; spec_3; spec_4; spec_5;...
spec_6; spec_7; spec_8; spec_9; spec_10; spec_11;...
spec_12; spec_13; spec_14; spec_15; spec_16; spec_17];
Results = table(Model, Training_Acc, Test_Acc, AUC, Sensitivity, Specificity);

```

## A. TRAINING LINEAR SVM

```

%%%%%%%%%%
% Created on: 03/21/21
% Input: Training data
% Output: Trained Linear SVM Model
% Author: Auto-generated by Matlab, implemented by Ridhi Deo
% File name: Obj1b_matlab_3a.m (R2020b [65]) )
% Description: This code was called internally from Obj1b_matlab_3.m to train the linear SVM
model. The trained model is returned as output to the Obj1b_matlab_3.m code and processed
further there.
%%%%%%%%%%

```

```

function [trainedClassifier, validationAccuracy] = linearsvm2(trainingData)

```

```

% Auto-generated by MATLAB on 21-Mar-2021 18:54:13

```

```

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',
    'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
    false];

```

```

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'linear', ...
    'PolynomialOrder', [], ...

```

```

'KernelScale', 'auto', ...
'BoxConstraint', 1, ...
'Standardize', true, ...
'ClassNames', [0; 1],...
'RemoveDuplicates',true);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'Plasma_glucose_1',
    'Plasma_glucose_percent'};
trainedClassifier.ClassificationSVM = classificationSVM;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2020b.';
trainedClassifier.HowToPredict = sprintf("To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing \"c\" with the name of the variable that is this struct, e.g.,
\"trainedModel\". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href='\"matlab:helpview(fullfile(docroot, \"stats\", \"stats.map\")),
\"appclassification_exportmodeltoworkspace\"}\">How to predict using an exported
model</a>.");

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI', 'ALT_percent',
    'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
    false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'Kfold', 5);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## B. TRAINING QUADRATIC SVM

```
%%%%%%%%%%  
% Created on: 03/21/21  
% Input: Training data  
% Output: Trained Quadratic SVM Model  
% Author: Auto-generated by Matlab, implemented by Ridhi Deo  
% File name: Obj1b_matlab_3b.m (R2020b [65]) )  
% Description: This code was called internally from Obj1b_matlab_3.m to train the  
QuadraticSVM model. The trained model is returned as output to the Obj1b_matlab_3.m code  
and processed further there.  
%%%%%%%%%%
```

```
function [trainedClassifier, validationAccuracy] = quadraticsvm2(trainingData)
```

```
% Auto-generated by MATLAB on 21-Mar-2021 18:58:33
```

```
% Extract predictors and response  
% This code processes the data into the right shape for training the  
% model.  
inputTable = trainingData;  
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',  
    'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};  
predictors = inputTable(:, predictorNames);  
response = inputTable.HS;  
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,  
    false];
```

```
% Train a classifier  
% This code specifies all the classifier options and trains the classifier.  
classificationSVM = fitsvm(...  
    predictors, ...  
    response, ...  
    'KernelFunction', 'polynomial', ...  
    'PolynomialOrder', 2, ...  
    'KernelScale', 'auto', ...  
    'BoxConstraint', 1, ...  
    'Standardize', true, ...  
    'ClassNames', [0; 1]);
```

```
% Create the result struct with predict function  
predictorExtractionFcn = @(t) t(:, predictorNames);  
svmPredictFcn = @(x) predict(classificationSVM, x);  
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));
```

```
% Add additional fields to the result struct
```

```

trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'Plasma_glucose_1',
'Plasma_glucose_percent'};
trainedClassifier.ClassificationSVM = classificationSVM;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
"trainedModel". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',
'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 5);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

### C. TRAINING GAUSSIAN SCALE 1 SVM

```

%% % % % % % % %
% Created on: 03/21/21
% Input: Training data
% Output: Trained Gaussian I SVM Model
% Author: Auto-generated by Matlab, implemented by Ridhi Deo
% File name: Obj1b_matlab_3c.m (R2020b [65]) )
% Description: This code was called internally from Obj1b_matlab_3.m to train the Gaussian I
SVM model. The trained model is returned as output to the Obj1b_matlab_3.m code and
processed further there.

```



```
%%%%%%%%%
```

```
function [trainedClassifier, validationAccuracy] = finegaussiansvm2(trainingData)
```

```
% Auto-generated by MATLAB on 21-Mar-2021 18:59:01
```

```
% Extract predictors and response
```

```
% This code processes the data into the right shape for training the  
% model.
```

```
inputTable = trainingData;
```

```
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',  
    'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
```

```
predictors = inputTable(:, predictorNames);
```

```
response = inputTable.HS;
```

```
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,  
    false];
```

```
% Train a classifier
```

```
% This code specifies all the classifier options and trains the classifier.
```

```
classificationSVM = fitsvm(...
```

```
    predictors, ...
```

```
    response, ...
```

```
    'KernelFunction', 'gaussian', ...
```

```
    'PolynomialOrder', [], ...
```

```
    'KernelScale', 0.83, ...
```

```
    'BoxConstraint', 1, ...
```

```
    'Standardize', true, ...
```

```
    'ClassNames', [0; 1]);
```

```
% Create the result struct with predict function
```

```
predictorExtractionFcn = @(t) t(:, predictorNames);
```

```
svmPredictFcn = @(x) predict(classificationSVM, x);
```

```
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));
```

```
% Add additional fields to the result struct
```

```
    trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',  
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'Plasma_glucose_1',  
    'Plasma_glucose_percent'};
```

```
trainedClassifier.ClassificationSVM = classificationSVM;
```

```
    trainedClassifier.About = 'This struct is a trained model exported from Classification  
    Learner R2020b.';
```

```
    trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n  
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,  
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
```

```

c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>.);

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',
        'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
        false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 5);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

#### **D. TRAINING GAUSSIAN SCALE 2 SVM**

```

%%%%%%%%%%
% Created on: 03/21/21
% Input: Training data
% Output: Trained Gaussian II SVM Model
% Author: Auto-generated by Matlab, implemented by Ridhi Deo
% File name: Obj1b_matlab_3d.m (R2020b [65]) )
% Description: This code was called internally from Obj1b_matlab_3.m to train the Gaussian II
SVM model. The trained model is returned as output to the Obj1b_matlab_3.m code and
processed further there.
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = mediumgaussiansvm2(trainingData)
% Auto-generated by MATLAB on 21-Mar-2021 18:59:35

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.

```

```

inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',
        'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 3.3, ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'Plasma_glucose_1',
    'Plasma_glucose_percent'};
trainedClassifier.ClassificationSVM = classificationSVM;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2020b.';
trainedClassifier.HowToPredict = sprintf("To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing \"c\" with the name of the variable that is this struct, e.g.,
\"trainedModel\". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href=\"matlab:helpview(fullfile(docroot, \"stats\", \"stats.map\"),
\"appclassification_exportmodeltoworkspace\")\">How to predict using an exported
model</a>.");

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;

```

```

    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',
        'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
    predictors = inputTable(:, predictorNames);
    response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,
        false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'Kfold', 5);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## E. TRAINING GAUSSIAN SCALE 3 SVM

```

%%%%%%%%%%
% Created on: 03/21/21
% Input: Training data
% Output: Trained Gaussian III SVM Model
% Author: Auto-generated by Matlab, implemented by Ridhi Deo
% File name: Obj1b_matlab_3e.m (R2020b [65]) )
% Description: This code was called internally from Obj1b_matlab_3.m to train the Gaussian III
SVM model. The trained model is returned as output to the Obj1b_matlab_3.m code and
processed further there.
%%%%%%%%%%

```

```

function [trainedClassifier, validationAccuracy] = coarsegaussiansvm2(trainingData)
% Auto-generated by MATLAB on 21-Mar-2021 19:00:16

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',
        'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
    predictors = inputTable(:, predictorNames);
    response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,
        false];

```

```

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 13, ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'Plasma_glucose_1',
    'Plasma_glucose_percent'};
trainedClassifier.ClassificationSVM = classificationSVM;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
"trainedModel". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'Plasma_glucose_1', 'BMI',
    'ALT_percent', 'AST_percent', 'Plasma_glucose_percent', 'BMI_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
    false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 5);

```

```
% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');
```

## 6. MATLAB CODE FOR SEX SPECIFIC PROCESSEDING – SMOTE

### A. MALE – SPECIFIC CODE

```
%%%%%%%%%%
% Created on: 03/18/2022
% Input: Male Sub-Dataset with synthetic data
% Output: Training and test datasets for male population
% Author: Ridhi Deo
% File name: obj1b_matlab_4a.m (R2020b [65]) )
% Description: This code was written specifically for male population. Data were processed,
SMOTE was applied, four derived features were created and populated with normalized data.
Data were split into training and test in a class balanced way.
%%%%%%%%%%

%% SMOTE
temp_dataset = dataset_HS_male;
temp_dataset(:,[1,9]) = []; % Removing the SEQN and HS columnns
size_disease = size(temp_dataset,1); %Measure of number of disease samples
N = 2; % Equivalent of N*100% synthetic sample generation
k = 2; % Setting number of nearest neighbours
num_attrs = size(temp_dataset,2); %Number of variables
new_index = 0; % Variable to keep a count of newly generated synthetic samples
    synthetic_sample_male.N{N} = zeros(size_disease,num_attrs); %Since we are
    generating N*100% synthetic, this value is N{2}
    nn_array = zeros(size_disease,k+1); %Tp keep a list of nearest neighbours for each
    sample
nn_values = zeros(size_disease,k+1);
R = zeros(num_attrs,1); %Range of continous variables is represented by the array R
    temp_range = table2array(temp_dataset); %Temporary conversions to array - for
    computational ease. This is essential dataset_HS_male
temp_dist = table2array(temp_dataset);
    distance = zeros(size_disease,size_disease); %Preallocating a matrix to store all the
    distances
for i = 1:num_attrs %Calculating ranges
```

```

    R(i,1) = (max(temp_range(:,i)) - min(temp_range(:,i)));
end
% Finding the k-nn
    [nn_array,nn_values] = knnsearch(temp_dist, temp_dist,'K',k+1); % the first nn will be
    itself so we will need to remove that

nn_array(:,1) = []; %Remove the first one because it is the same sample
nn_values(:,1) = []; %First nn is itself so distance is 0
while (N~=0) %To perform N*100% synthetic sampling
    for i = 1:size_disease
        for attr=1:num_attrs
            nn = randi([1 k],1); % Randomly choose the nearest neighbor
            dif = temp_dist(nn_array(i,nn),attr) - temp_dist(i,attr);
            gap = 0 + rand(1,1);
            synthetic_sample_male.N{N}(i,attr) = temp_dist(i,attr) + (gap*dif); %Synthetic
            continous attributes
        end
    end
    N = N-1; %To avoid infinite loops
end

    total_synthetic_sample_males =
    [synthetic_sample_male.N{1};synthetic_sample_male.N{2}];
total_synthetic_sample_males = array2table(total_synthetic_sample_males);
    total_synthetic_sample_males.Properties.VariableNames =
    temp_dataset.Properties.VariableNames;
    hybrid_disease_male = [total_synthetic_sample_males; temp_dataset]; %Hybrid =
    synthetic + disease
hybrid_disease_male.HS = ones(size(hybrid_disease_male,1),1);

%% removing seqn
dataset_non_HS_male.SEQN = [];

%% Based on American Liver Foundation video - Vicki Shah
%% Based on American Liver Foundation video - Vicki Shah
% Normal value for ALT: 10 - 55 U/L. Actual levels 30
% AST: 10-40 U/L, but prefer 30
% ASP: 45-115, also based on age

%% AASLD: Male: ALT: 29-33 IU/L
% Using 33 as the ALT ULN for men based on AASLD guidelines
ULN_ALT = 33;
%% Liver foundation vicki shah: Male: AST ULN: 30 IU/L
ULN_AST = 30;

```

```

%% Creating a new % variable per discussion with Dr. P on March 18th
ALT_percent = zeros(size(hybrid_disease_male, 1),1);
hybrid_disease_male(:,end+1) = array2table(ALT_percent);
hybrid_disease_male.Properties.VariableNames{'Var9'} = 'ALT_percent';

ALT_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(ALT_percent);
dataset_non_HS_male.Properties.VariableNames{'Var9'} = 'ALT_percent';

AST_percent = zeros(size(hybrid_disease_male, 1),1);
hybrid_disease_male(:,end+1) = array2table(AST_percent);
hybrid_disease_male.Properties.VariableNames{'Var10'} = 'AST_percent';

AST_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(AST_percent);
dataset_non_HS_male.Properties.VariableNames{'Var10'} = 'AST_percent';

Plasma_glucose_percent = zeros(size(hybrid_disease_male, 1),1);
hybrid_disease_male(:,end+1) = array2table(Plasma_glucose_percent);
hybrid_disease_male.Properties.VariableNames{'Var11'} = 'Plasma_glucose_percent';

Plasma_glucose_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(Plasma_glucose_percent);
dataset_non_HS_male.Properties.VariableNames{'Var11'} = 'Plasma_glucose_percent';

BMI_percent = zeros(size(hybrid_disease_male, 1),1);
hybrid_disease_male(:,end+1) = array2table(BMI_percent);
hybrid_disease_male.Properties.VariableNames{'Var12'} = 'BMI_percent';

BMI_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(BMI_percent);
dataset_non_HS_male.Properties.VariableNames{'Var12'} = 'BMI_percent';
for i = 1: size(hybrid_disease_male,1)
    hybrid_disease_male.ALT_percent(i) = ((hybrid_disease_male.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;
    hybrid_disease_male.AST_percent(i) = ((hybrid_disease_male.AST(i) -
    ULN_AST)/ULN_AST)*100;
    hybrid_disease_male.Plasma_glucose_percent(i) =
    ((hybrid_disease_male.Plasma_glucose_1(i) - 120)/120)*100;
    hybrid_disease_male.BMI_percent(i) = ((hybrid_disease_male.BMI(i) - 25)/25)*100;
end

for i = 1: size(dataset_non_HS_male,1)
    dataset_non_HS_male.ALT_percent(i) = ((dataset_non_HS_male.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;

```



```

        dataset_non_HS_male.AST_percent(i) = ((dataset_non_HS_male.AST(i) -
        ULN_AST)/ULN_AST)*100;
        dataset_non_HS_male.Plasma_glucose_percent(i) =
        ((dataset_non_HS_male.Plasma_glucose_1(i) - 120)/120)*100;
        dataset_non_HS_male.BMI_percent(i) = ((dataset_non_HS_male.BMI(i) -
        25)/25)*100;
    end

%% Converting % values to 0 if they are negative - see the top of this script for details

for i = 1: size(hybrid_disease_male,1)
    if(hybrid_disease_male.ALT_percent(i) <= 0)
        hybrid_disease_male.ALT_percent(i) = 0;
    end
    if(hybrid_disease_male.AST_percent(i) <= 0)
        hybrid_disease_male.AST_percent(i) = 0;
    end
    if(hybrid_disease_male.Plasma_glucose_percent(i) <= 0)
        hybrid_disease_male.Plasma_glucose_percent(i) = 0;
    end
    if(hybrid_disease_male.BMI_percent(i) <= 0)
        hybrid_disease_male.BMI_percent(i) = 0;
    end
end

for i = 1: size(dataset_non_HS_male,1)
    if(dataset_non_HS_male.ALT_percent(i) <= 0)
        dataset_non_HS_male.ALT_percent(i) = 0;
    end
    if(dataset_non_HS_male.AST_percent(i) <= 0)
        dataset_non_HS_male.AST_percent(i) = 0;
    end
    if(dataset_non_HS_male.Plasma_glucose_percent(i) <= 0)
        dataset_non_HS_male.Plasma_glucose_percent(i) = 0;
    end
    if(dataset_non_HS_male.BMI_percent(i) <= 0)
        dataset_non_HS_male.BMI_percent(i) = 0;
    end
end

    hybrid_disease_male = [hybrid_disease_male(:,1:7), hybrid_disease_male(:,9:end),
    hybrid_disease_male(:,8)];
    dataset_non_HS_male = [dataset_non_HS_male(:,1:7), dataset_non_HS_male(:,9:end),
    dataset_non_HS_male(:,8)];

```

```

%% Split into training and test
Q = size(hybrid_disease_male,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,~,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_disease_male = hybrid_disease_male(trainInd,:);
test_disease_male = hybrid_disease_male(testInd,:);

Q = size(dataset_non_HS_male,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,valInd,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_no_disease_male = dataset_non_HS_male(trainInd,:);
test_no_disease_male = dataset_non_HS_male(testInd,:);

training_male = [train_disease_male; train_no_disease_male];
test_male = [test_disease_male; test_no_disease_male];
test_male = test_male(randperm(size(test_male,1)),:);
training_male = training_male(randperm(size(training_male,1)),:);

```

## B. FEMALE – SPECIFIC CODE

```

%% % % % % % % %
% Created on: 03/18/2022
% Input: Female Sub-Dataset with synthetic data
% Output: Training and test datasets for female population
% Author: Ridhi Deo
% File name: obj1b_matlab_4b.m (R2020b [65]) )
% Description: This code was written specifically for female population. Data were processed,
SMOTE was applied, four derived features were created and populated with normalized data.
Data were split into training and test in a class balanced way.
%% % % % % % % %

```

```

%% SMOTE
temp_dataset = dataset_HS_female;
temp_dataset(:,[1,9]) = []; % Removing the SEQN and HS columns
size_disease = size(temp_dataset,1); % Measure of number of disease samples
N = 3; % Equivalent of N*100% synthetic sample generation
k = 2; % Setting number of nearest neighbours
num_attrs = size(temp_dataset,2); % Number of variables
new_index = 0; % Variable to keep a count of newly generated synthetic samples
    synthetic_sample_female.N{N} = zeros(size_disease,num_attrs); % Since we are
    generating N*100% synthetic, this value is N{2}

```

```

        nn_array = zeros(size_disease,k+1); %Tp keep a list of nearest neighbours for each
        sample
nn_values = zeros(size_disease,k+1);
R = zeros(num_attrs,1); %Range of continous variables is represented by the array R
        temp_range = table2array(temp_dataset); %Temporary conversions to array - for
        computational ease. This is essential dataset_HS_female
temp_dist = table2array(temp_dataset);
        distance = zeros(size_disease,size_disease); %Preallocating a matrix to store all the
        distances
for i = 1:num_attrs %Calculating ranges
    R(i,1) = (max(temp_range(:,i)) - min(temp_range(:,i)));
end
% Finding the k-nn
    [nn_array,nn_values] = knnsearch(temp_dist, temp_dist,'K',k+1); % the first nn will be
    itself so we will need to remove that

nn_array(:,1) = []; %Remove the first one because it is the same sample
nn_values(:,1) = []; %First nn is itself so distance is 0
while (N~=0) %To perform N*100% synthetic sampling
    for i = 1:size_disease
        for attr=1:num_attrs
            nn = randi([1 k],1); % Randomly choose the nearest neighbor
            dif = temp_dist(nn_array(i,nn),attr) - temp_dist(i,attr);
            gap = 0 + rand(1,1);
            synthetic_sample_female.N{N}(i,attr) = temp_dist(i,attr) + (gap*dif); %Synthetic
            continous attributes
        end
    end
    N = N-1; %To avoid infinite loops
end

total_synthetic_sample_females =
[synthetic_sample_female.N{1};synthetic_sample_female.N{2};synthetic_sample_femal
e.N{3}];
total_synthetic_sample_females = array2table(total_synthetic_sample_females);
total_synthetic_sample_females.Properties.VariableNames =
temp_dataset.Properties.VariableNames;
hybrid_disease_female = [total_synthetic_sample_females; temp_dataset]; %Hybrid =
synthetic + disease
hybrid_disease_female.HS = ones(size(hybrid_disease_female,1),1);

%% removing seqn
dataset_non_HS_female.SEQN = [];

```

```

%% Based on American Liver Foundation video - Vicki Shah
% Normal value for ALT: 10 - 55, but actually 20.
% AST: 9 - 32, but prefer 20
% ASP: 30 - 100, also based on age
% NAFLD: AST and ALT are up to less than 4 times the ULN
%% AASLD: Female: ALT: 19 - 25 IU/L
% Using 25IU/L as the ULN for female based on the AASLD guidelines

ULN_ALT = 25;
%% Liver foundation vicki shah: Female: AST ULN: 20 IU/L
ULN_AST = 20;
%% Creating a new % variable per discussion with Dr. P on March 18th
ALT_percent = zeros(size(hybrid_disease_female, 1),1);
hybrid_disease_female(:,end+1) = array2table(ALT_percent);
hybrid_disease_female.Properties.VariableNames{'Var9'} = 'ALT_percent';

ALT_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(ALT_percent);
dataset_non_HS_female.Properties.VariableNames{'Var9'} = 'ALT_percent';

AST_percent = zeros(size(hybrid_disease_female, 1),1);
hybrid_disease_female(:,end+1) = array2table(AST_percent);
hybrid_disease_female.Properties.VariableNames{'Var10'} = 'AST_percent';

AST_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(AST_percent);
dataset_non_HS_female.Properties.VariableNames{'Var10'} = 'AST_percent';

Plasma_glucose_percent = zeros(size(hybrid_disease_female, 1),1);
hybrid_disease_female(:,end+1) = array2table(Plasma_glucose_percent);
hybrid_disease_female.Properties.VariableNames{'Var11'} = 'Plasma_glucose_percent';

Plasma_glucose_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(Plasma_glucose_percent);
dataset_non_HS_female.Properties.VariableNames{'Var11'} = 'Plasma_glucose_percent';

BMI_percent = zeros(size(hybrid_disease_female, 1),1);
hybrid_disease_female(:,end+1) = array2table(BMI_percent);
hybrid_disease_female.Properties.VariableNames{'Var12'} = 'BMI_percent';

BMI_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(BMI_percent);
dataset_non_HS_female.Properties.VariableNames{'Var12'} = 'BMI_percent';
%% Computing % values

for i = 1: size(hybrid_disease_female,1)

```

```

        hybrid_disease_female.ALT_percent(i) = ((hybrid_disease_female.ALT(i) -
        ULN_ALT)/ULN_ALT)*100;
        hybrid_disease_female.AST_percent(i) = ((hybrid_disease_female.AST(i) -
        ULN_AST)/ULN_AST)*100;
        hybrid_disease_female.Plasma_glucose_percent(i) =
        ((hybrid_disease_female.Plasma_glucose_1(i) - 120)/120)*100;
        hybrid_disease_female.BMI_percent(i) = ((hybrid_disease_female.BMI(i) -
        25)/25)*100;

end

for i = 1: size(dataset_non_HS_female,1)
    dataset_non_HS_female.ALT_percent(i) = ((dataset_non_HS_female.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;
    dataset_non_HS_female.AST_percent(i) = ((dataset_non_HS_female.AST(i) -
    ULN_AST)/ULN_AST)*100;
    dataset_non_HS_female.Plasma_glucose_percent(i) =
    ((dataset_non_HS_female.Plasma_glucose_1(i) - 120)/120)*100;
    dataset_non_HS_female.BMI_percent(i) = ((dataset_non_HS_female.BMI(i) -
    25)/25)*100;
end

%% Converting % values to 0 if they are negative - see the top of this script for details

for i = 1: size(hybrid_disease_female,1)
    if(hybrid_disease_female.ALT_percent(i) <= 0)
        hybrid_disease_female.ALT_percent(i) = 0;
    end
    if(hybrid_disease_female.AST_percent(i) <= 0)
        hybrid_disease_female.AST_percent(i) = 0;
    end
    if(hybrid_disease_female.Plasma_glucose_percent(i) <= 0)
        hybrid_disease_female.Plasma_glucose_percent(i) = 0;
    end
    if(hybrid_disease_female.BMI_percent(i) <= 0)
        hybrid_disease_female.BMI_percent(i) = 0;
    end
end

for i = 1: size(dataset_non_HS_female,1)
    if(dataset_non_HS_female.ALT_percent(i) <= 0)
        dataset_non_HS_female.ALT_percent(i) = 0;
    end
    if(dataset_non_HS_female.AST_percent(i) <= 0)

```

```

    dataset_non_HS_female.AST_percent(i) = 0;
end
if(dataset_non_HS_female.Plasma_glucose_percent(i) <= 0)
    dataset_non_HS_female.Plasma_glucose_percent(i) = 0;
end
if(dataset_non_HS_female.BMI_percent(i) <= 0)
    dataset_non_HS_female.BMI_percent(i) = 0;
end
end

hybrid_disease_female = [hybrid_disease_female(:,1:7), hybrid_disease_female(:,9:end),
    hybrid_disease_female(:,8)];
dataset_non_HS_female = [dataset_non_HS_female(:,1:7),
    dataset_non_HS_female(:,9:end), dataset_non_HS_female(:,8)];

%% Split into training and test
Q = size(hybrid_disease_female,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,~,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_disease_female = hybrid_disease_female(trainInd,:);
test_disease_female = hybrid_disease_female(testInd,:);

Q = size(dataset_non_HS_female,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,valInd,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_no_disease_female = dataset_non_HS_female(trainInd,:);
test_no_disease_female = dataset_non_HS_female(testInd,:);

training_female = [train_disease_female; train_no_disease_female];
test_female = [test_disease_female; test_no_disease_female];
test_female = test_female(randperm(size(test_female,1)),:);
training_female = training_female(randperm(size(training_female,1)),:);

```

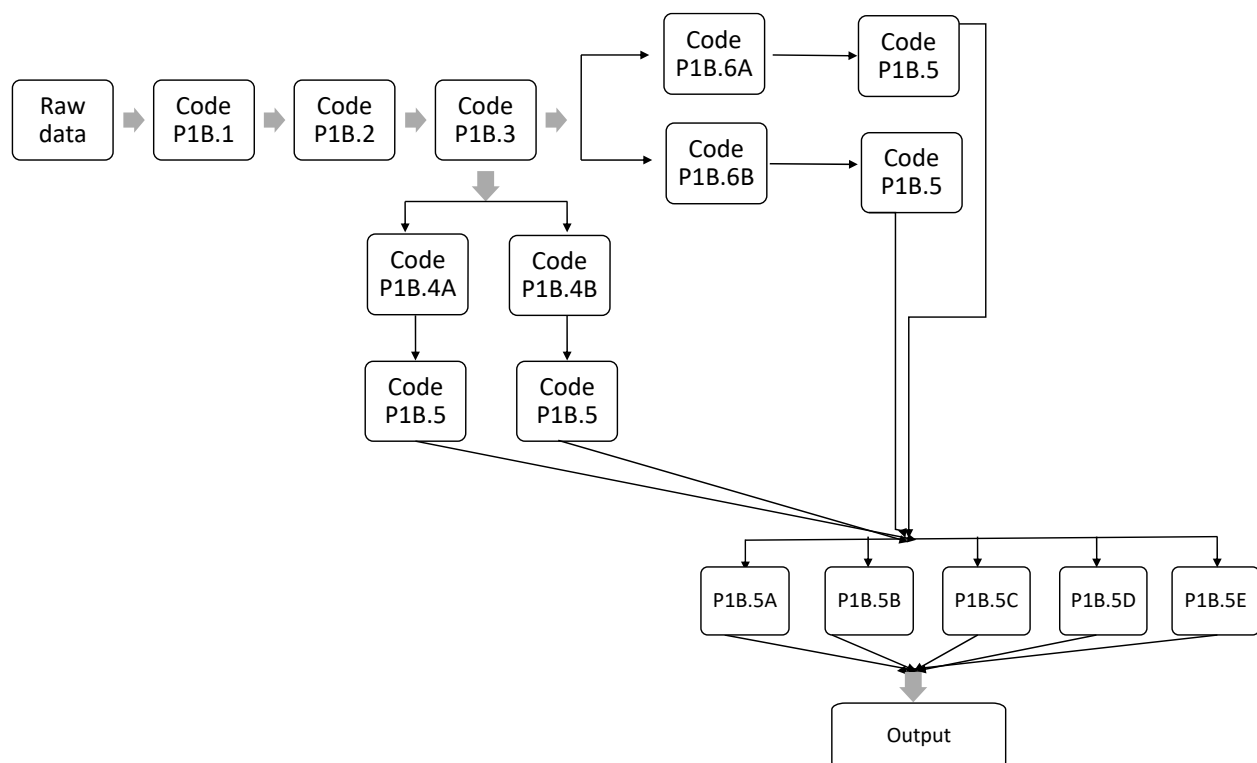


Figure P1.B.1: Figure outlining the flow of code used in this research objective (1B)

## **CHAPTER 3. EXPLAINABLE ARTIFICIAL INTELLIGENCE (XAI) APPLIED TO HS-SCREENING MODELS (PAPER 2)**

*A portion of the work in this paper was published in a peer-reviewed conference proceeding – Healthcare Innovations – Point of Care Technologies (HI-POCT 2022), Houston, Texas, March 10 – 11, 2022.*

### **3.1 Abstract**

Use of computational models for early screening of Hepatic Steatosis (HS) were discussed in paper 1. Selected models from Paper 1 were further evaluated in this paper to understand more about how the models interpret their input features. ML models are inherently black box in nature. Interpreting the models understanding is a critical step towards building transparency in machine learning models. In this work, a global explainability tool called Partial Dependency was implemented for the selected models (separately for male and female – specific models). Results were compared with the clinically defined normal values and the best performing features and models were identified.

### **3.2 Introduction**

The research objective in paper 1 was to develop screening tools for Hepatic Steatosis (HS) using machine learning and selected parameters (physiological and liver biochemical data – objective 1B). These models show potential as clinical decision support tools for HS screening. However, ML models are inherently black-box in nature, that is, they do not indicate the reasoning related to the prediction of HS or no-HS. Understanding how a model is learning (the relationship between the predictors and output variable) is useful. This understanding can improve the trust in the model's predictions and provide insights that can be used to improve future model performance.

Recently, an increased research thrust in the field of explainable AI has been observed [1], [2]. The goal of XAI is to provide “justification, transparency, and traceability” to the black-box model-based decisions [1]. Recent developments in the field of XAI and specific tools that can be used to understand the ML models are provided in the next section.



In this research, the models developed using physiological and liver biochemical data (objective 1B) were explored further using an XAI tool. The XAI tool ‘Partial Dependence’ was explored to gain insights on mapping the specific input parameters for predicting the output (HS/no-HS). Detailed background, definitions, and recent work in the XAI domain are outlined in the literature review section.

### **3.3 Literature review**

#### **3.3.1 Background and importance of XAI in healthcare research**

With an increase in data, data availability, and computational power there has been an increase in the use of machine learning (ML) and artificial neural network (ANN) tools to generate prediction/estimation models. ML and ANN models are being implemented using large datasets to predict (classify or estimate) different parameters across multiple application domains. Use of such models in the healthcare domain has also been an increasing area of research, including in the Objective 1 of this dissertation. While artificial intelligence-based models are highly useful, most of them are inherently black-box in nature.

Typically, as the models get more complex in nature, their interpretability decreases. Figure 3.1 shows an example of interpretability vs complexity for some of the commonly used ML models. Note that increasing size of the circle denotes increasing performance. Typically, complex models perform better than simple regression or rule-based models, but complex models lack interpretability. Linear models, rule-based models and simple tree models are highly interpretable intrinsically. Such models can be called intrinsically open/explainable. On the other hand, neural networks, support vector machines and K-nearest neighbor methods are more complex in nature and therefore require post-hoc explainability methods.

Broadly, there are multiple XAI approaches to gain an understanding of a specific model [3]–[5]. A simple categorization method for XAI tools is model-agnostic vs model-specific tools. Model-agnostic XAI can be applied to any black-box model, irrespective of which algorithm was used to train the model. On the other hand, model-specific tools are limited to only a specific category of AI/ML models. Another way to categorize XAI tools is using a global vs local interpretation. Global XAI methods are used to estimate the entire logic used by the model (for all the data points used in the model), whereas local XAI methods can be used to estimate the

reasoning for an individual data point or a small group of data points. The choice of the most relevant XAI method can be made using the specific model in question and its application domain.

Recent surveys have summarized the types of explainable AI tools [6], tools relevant to health data [1] and more specifically, explainable AI used for electronic health records (EHR) data [3]. The review of XAI using EHRs identified three major model trends identified in the survey were: 1) “if-then” rule-based models 2) Use of low complexity ML models first and then improving model performance using optimization 3) Dimensionality reduction techniques [3]. The recent tools and techniques developed for XAI are outlined in more detail in the next section.

### **3.3.2 Tools and techniques for XAI**

Broadly, explainable AI techniques for machine learning models can be categorized by the stage at which they were applied, the scope of the data being used, the type of ML/AI tool, the type of input and output features [6]. The categorization of the various tools is explained in greater detail in a recent systematic review article [6]. The definitions of each of these categories are briefly listed below:

1. Stage: The period when the tool/method is able to provide explanations in of two stages: “ante-hoc” or “post-hoc” [6]. Ante-hoc models are generated to be interpretable or transparent from the beginning, whereas post-hoc models use an external explainer during model testing [6].
2. Scope: The explanation provided by a particular method can either be “global”, “local” or “cohort” [6], [7]. Global explanations aim to interpret the model’s functionality as a whole. Local explanations are useful to interpret each observation used in a model. Cohort explanations provide insights into how the model is interpreting a group of observations (sub-set of the whole dataset) [6], [7].
3. Type of ML/AI: Classification type models are used for binary or categorical output whereas regression type models are used for continuous outputs. The type of explainability being used can depend on the type of the underlying model [6].
4. Input & output data types: Different explainability techniques are useful based on the type of the input and output data [6]. These data types can be one of the following:
  - a. Input: Categorical or continuous text data, images, longitudinal (time-series) .
  - b. Output: Numerical/text, rule-based, images, mixed data.

Choice of the tool or technique for explainability is therefore based on different parameters of the ML/AI model, the complexity of the model and the desired outputs. While several researchers have specified the importance of using XAI tools for improving model understanding [3], [4], [8]–[10], some other researchers also warn about the need to use XAI as a supplementary tool, particularly for healthcare applications [3], [11]. A collaboration between AI developers and healthcare practitioners is recommended [3], [11]. Further, in healthcare, human expertise is critical since health-related decisions can have large implications. As such, human-in-loop studies are recommended and require further research [3].

In this dissertation, the focus is on understanding models developed (in paper 1) as potential tools for clinical decision support. These models are numerical, black box, non-linear in nature and could use post-hoc explainability analysis. Therefore, the focus of this literature review is limited to identifying existing methods for post-hoc analysis of numerical data.

In terms of scope of XAI for model interpretability, there are two types [6]:

- i. Local explainability
  - a. Cohort
  - b. Individual
- ii. Global explainability

Briefly, local explainability is useful in understanding the model’s reasoning for how one observation (local) or a group of selected observations (cohort) are being classified. In both these cases, a smaller subset of the training dataset can be used to understand the model reasoning. Local explainability tools work by fitting smaller, linear models to understand a larger, non-linear model locally [12]. Local explainability is particularly useful to understand why a particular group of observations might be misclassified by the model. It can help in identifying any potential biases within a group of data or for a single observation. Local explainability also helps in understanding the influence of individual input features on a model’s decision making by bringing to attention the weight of a feature in a decision [3]. Drawing the user’s attention to insights about an individual, that may have been initially missed, is extremely impactful, especially for clinicians, as they work with patients in real-time [3], [8], [9].

Global explainability, on the other hand, is useful to understand the model’s reasoning as a whole. This is particularly useful in the initial stages of training a model. The overall reasoning

is obtained by averaging the model's decisions across all the predictions. It also allows the developer to understand to what extent each input feature is contributing to the overall decision making of the model. Based on the results from a global XAI tool, feature selection and feature engineering can be modified to align the model with its expectations.

In this research, three XAI methods were considered. They are LIME [12], SHAP [13] and Partial Dependence [14], [15]. Each of these methods has a different approach to explain a model's predictions and provide insights. These methods are explained below.

*a. LIME*

Local Interpretable Model-Agnostic Explanations or LIME is a local XAI technique that can approximate any complex machine learning model (linear or non-linear) using a simple interpretable model (linear model or a decision tree) [12]. The simpler model is used as a surrogate model to explain the original (complex ML) model. LIME can be used as a tool for interpreting the "local" behavior of a complex model. The advantage of this method is that it can be applied to any ML model, irrespective of the complexity of the model.

*b. SHAP*

Shapley Additive Explanations or SHAP is another XAI technique which ranks the importance of each input feature for any observation [13]. Like LIME, SHAP is also a local XAI technique and works with any specific observation of interest. However, SHAP can be modified to also provide global understanding [16]. The output from SHAP can quantify the deviation of each predictor from its average, for any given observation. SHAP is also a model-agnostic technique and can therefore be applied to any machine learning model.

*c. Partial dependence*

Partial dependence calculates the averaged relationship between any one or two input feature(s) (predictor variable) and the output feature of a trained classification or regression model by marginalizing on all other input features [14], [15]. It is a global XAI tool and can be used to identify the effect that one or two input features have on the overall model prediction. Partial dependence averages the output of the model over the entire range of input feature values

[17]. These partial dependencies can then be plotted to visualize the impact of any chosen input feature on the model's predicted output. Partial Dependency is a global XAI tool that provides insights about the features being used in the model as a whole. This is particularly useful to identify trends of the input features with respect to the outputs.

In this dissertation, partial dependence plots were implemented using MATLAB R2020b [17] and the results are elaborated in the sections to follow. The central idea of conducting explainability analysis is to better explore the individual relationships between each of the seven predictors with the outcome variable (HS) for selected machine learning models. The specific research tasks are:

1. To identify the differences between the model interpretability for male vs female data.
2. To identify the top predictors of Hepatic Steatosis in both male and female categories.
3. To identify the ML model(s) that interpret data with highest alignment to the clinically defined normal values.

### **3.4 Methods**

The overall procedure used for conducting the explainability analysis is described in the sections to follow.

#### **3.4.1 Data**

Of the five SVM models developed in paper 1, three best performing models in terms of sensitivity and specificity metrics were chosen for explainability analysis. The three selected models were: Quadratic, Gaussian 1 SVM, and Gaussian 2 SVM. The models for male and female population were analyzed separately to understand the differences between the male vs female model interpretability in this chapter.

#### **3.4.2 Model & explainable AI tool selection**

##### ***a. Model selection***

In objective 1B, five ML models were developed for each male and female populations. These models used the following predictor variables: Age, BMI, HDL, ALT, AST, ASP, and

glucose. The highest testing accuracy for male and female models were at 69% and 71%, respectively. Sensitivity and specificity ranges were between 64 – 72% and 61-74%, respectively for male population. Similarly for female population, the sensitivity range was from 67 – 71% and specificity ranged from 68 – 75%.

To assess the performance of healthcare-related ML models, considering both sensitivity and specificity performance metrics are important instead of only considering the test accuracy. However, when screening for a disease, sensitivity is important to identify those with the disease quickly, such that appropriate care can be provided to those with the disease condition.

Overall, the best performing male-specific models demonstrated approximately 72% sensitivity (Gaussian scale 1) and 74% specificity (Gaussian scale 2). Similarly, the best performing female-specific models provided 71% (Gaussian scale 1) and 75% (Quadratic SVM) sensitivity and specificity, respectively. Therefore, the top three models for XAI in this research were selected to be: Quadratic SVM, Gaussian scale 1 SVM and Gaussian scale 2 SVM.

### ***b. Partial dependency***

To gain insights on model's understanding of the data, a global XAI tool was selected in this research. Specifically, 'Partial dependency' was computed and plotted in this research using MATLAB [17]. The idea behind using a global XAI tool instead of any other local XAI tools was to gain an initial understanding of how the data (as a whole) was being interpreted by the model. This initial understanding is important to first identify any unexpected trends in the models understanding. This technique will also help in identifying the contribution of each individual parameter, which is useful in the context of complex datasets like the one used in Objective 1.

Finally, comparing the model's understanding of each input feature with the clinically defined normal for each input feature will provide significant insights and help in identifying the best model. For example, if increasing BMI is considered a risk factor by clinical literature for hepatic steatosis (HS), then understanding the model's interpretability of BMI is valuable to evaluate the model performance. If the model interprets BMI in the same manner that is defined by clinical literature, then the model can be considered as performing per expectations.

The models were analyzed using an explainable AI tool called 'Partial Dependency' in MATLAB R2020b [17]. Partial dependency is used to identify individual relationships between one input/predictor variable and the output variable used in any classification ML model. The

identified relationships are ‘partial’ in nature because they are obtained by marginalizing over all other predictor features. Partial dependency for support vector machines utilizes a classification score. The classification score of any observation can be computed using the ‘predict’ function in MATLAB R2020b [18]. Classification score for an observation in a SVM model can be defined as the signed boundary between that observation and the decision boundary in the hyperplane. The decision boundary is two-dimensional and centered around 0 with a range of  $(-\infty, +\infty)$ . Higher classification score implies that the observation is farther away from the decision boundary, whereas classification scores closer to 0 imply that the observations are closer. Mathematically, the classification score can be described using the equation P2.1.

$$Score(x) = \sum_{j=1}^{100} a_i y_j S(x_j, x) + c \quad (P2.1)$$

For each observation used for training, ‘j’, the input predictors ‘x’ and output feature ‘y’ are used. A dot product between the input ‘x’ and the corresponding support vector ‘x<sub>j</sub>’ is represented by  $S(x_j, x)$ . The variables ‘a’ and ‘c’ are estimated parameters specific to the SVM. In this research, the observations that lie on the positive side of zero are in the ‘Disease’/HS positive category. Observations that lie on the negative side of zero are in the ‘No-Disease’/No-HS category.

The partial relationships were identified between one input/predictor variable (at a time) and a subset of output responses (N=100). These values were identified while marginalizing over the other six predictor variables. Then, the process was repeated for each predictor, for each of the three selected models, for each sex. For robust interpretation of the partial relationships, each combination was run 10-times independently and the results were averaged. Therefore, the presented results are averaged partial relationships.

### *c. Ambiguity zone*

As explained earlier, partial dependencies greater than zero indicate that the observation lies in the disease category and those less than zero fall in the no-disease category. Higher partial dependency of a feature implies large contributions made by the feature in the model’s decision making. For example, a partial dependency score of 3.149 is more helpful (to the model) in making a prediction, than a score of 0. Therefore, scores closer to zero are less helpful and create

ambiguity. Average partial dependencies that lie close to 0 are therefore defined to be in the “ambiguous” zone, in this research. That is, partial dependencies that fall between (-0.15, 0.15) lie in the ambiguous zone. A visual of the partial prediction plot with the ambiguity zone centered around 0 is shown in Figure 3.2.

In this work, the ambiguity percentage (Amb %) of each partial dependency was computed. A ratio of number of partial dependencies in the (-0.15, 0.15) range divided by the total count of partial dependencies was used to calculate the Amb %. These values are presented alongside detailed observations in

Table 3.2.

Features with low Amb % can contribute highly to the model’s decision making. If a feature has high Amb %, it implies that the usefulness of the feature to the model in making predictions is low. Therefore, in an ideal case, the Amb % of any feature is expected to be either zero or also low as possible.

#### ***d. Clinical normal values***

In this work, the clinically defined normal ranges of the following input parameters were identified: BMI, HDL, ALT, AST, ASP, and Glucose. Although increasing age is found to increase the risk for NAFLD in general [19], [20], no specific “normal” for age was used in this study. Instead, the trend of increasing risk of NAFLD with increasing age was interpreted as “normal”.

It is important to note that a majority of the clinically defined normal values are different for male populations and for female populations. They were interpreted as such in this work. Further, the definition of clinical normal varied slightly between different literature sources. The data from some of the commonly cited literature was used in this research. These normal values are shown in the Table 3.1 below along with their references. Figure 3.3 shows the summary of the methods used in this work.

The results from the partial dependency plots were interpreted in a systematic manner for each model. These results are documented in

Table 3.2 - Table 3.4 for male-specific models and



Table 3.5 -

Table 3.7 for female-specific models. First, the observations from each graph were documented – these observations are in the column titled “Layer 1” in each table. Next, the observations were compared with the clinical normal values. The implications of these comparisons were documented in the column titled “Layer 2”. The results and their relevant discussion are presented in the sections to follow.

### **3.5 Results and discussion**

The partial dependencies for each predictor were plotted and were analyzed further by comparison with clinical normal values (defined in Table 3.1). The partial dependency plots (PDP) for Quadratic SVM, Gaussian Scale 1 SVM and Gaussian Scale 2 SVM are shown in Figure 3.4 - Figure 3.9. The X-axis of each plot is the feature value, for example: Age, BMI, HDL, etc. The Y-axis is the average partial dependence score (or partial dependency) of that specific feature in predicting HS. For example - partial dependency of Age for HS prediction. If the line on a plot shows an increasing or upward trend, then it indicates increasing risk for HS. If the line trends downwards or decreases, then it indicates a decreasing for HS.

Note that red asterisk(s) on each plot indicate the defined normal value or the normal range as per Table 3.1. The asterisk(s) are provided to highlight how the model interprets data in the normal range. The “normal” values are defined for six of the seven input features. There is not a clearly defined “normal” value for the age parameter in the context of risk for HS as explained earlier. Therefore, an increasing risk for HS with increasing age is the logical, expected trend in this work.

By examining each of the PDPs within Figure 3.4 to Figure 3.9, the comparisons with clinical normal values or ranges are documented for male and female – specific models in

Table 3.8 &

Table 3.9, respectively. In an ideal case, each model would have all the features following the logical trend as per the clinical literature. However, in the real-world scenario, it might not be possible for any one model to have all the features following the clinical trend perfectly. Therefore, in this work, the aim is to find the most optimal models that maximize the number of features following the logical trend.

Each cell within these tables is color coded to indicate whether a not a feature follows the logical trend from a clinical standpoint. The cells in green indicate model parameters that follow the logical trend. Cells in yellow indicate features that follow the logical trend partially whereas those in red indicate features that do not follow the logical trend and are in fact being interpreted by the model as the opposite of the defined clinical normal.

The results from male & female - specific population models are discussed in the sections below. Another section compares and discusses results of male and female – specific models.

### **3.5.1 Analysis of the models for male population**

Results from Figure 3.4 - Figure 3.6 were assessed and the interpretations of XAI analyses are shown in

Table 3.2 -Table 3.4. A summary of all the male-specific model results is in

Table 3.8. The analysis of each model is presented in the sub-sections below.

*a. Quadratic SVM*

The PDPs related to Quadratic SVM for male population are shown in Figure 3.4. Each figure within Figure 3.4 corresponds to an input feature. The first graph shows the partial dependency of Age with respect to HS risk. It is observed that the Quadratic SVM model is interpreting the feature 'Age' in the clinical range as expected. However, 45% of the Age data fall in the ambiguity zone. Overall, the input feature 'Age' follows the trend as expected from a general clinical normal perspective

The second feature, BMI, shows a linearly increasing range between normal BMI of 18.5 to 25 but from clinical perspective this should remain steady. That is, the risk between BMI ranges of 18.5 to 25 is considered to be low and similar, from a clinical standpoint. But in this case, it is postulated that the model might be indicating additional useful information, and that needs more investigation. However, the curve above 25 follows the logical, expected trend. Finally, none of the data from BMI falls in the ambiguity range – this implies that the BMI feature contributes strongly to the model's decision making.

HDL shows a decreasing risk trend after 1 mmol/L until approximately 2 mmol/L. While the HDL PDP makes logical sense in the 1 – 2 mmol/L range, the increase in slope after HDL of 2 mmol/L is not expected. From a clinical perspective, higher values of HDL indicate lower disease risk. However, other lipid profile information, for example: triglycerides, total cholesterol and LDL were not included in this study due to lack of sufficient data. It is postulated that inclusion of these parameters might improve the model performance. Therefore, the increasing trend after 2 mmol/L of HDL might need additional investigation. The HDL curve has 16% data in the ambiguity zone, which is not ideal but is a low amount.

ALT shows an increasingly linear trend with HS risk. The ALT curve in the normal range (10 – 55 U/L), is expected to be steady. In this case, the ALT curve in the normal range shows slightly increasing trend. After the normal range, the ALT PDP shows increasing risk which follows the logical trend. Therefore, the ALT curve is overall following the clinical trend as expected. The ALT curve also does not have any data in the ambiguity zone, which indicates the high contribution of ALT in Quadratic SVM's decision making.

AST shows a similar trend to that of ALT. The AST curve in the normal range of 10 – 40 U/L is close to steady and shows an increasing risk after the normal range. This behavior follows the clinical trend as expected. The ambiguity % (Amb %) is also very low at 6%, indicating that AST is also a high contributor in the model's decision making.

The ASP curve shows a decreasing slope within the normal range of 45 – 115 U/L. From a clinical perspective, the curve in this range is expected to be steady, not decreasing. However, the curve after the normal range (after 200 U/L) shows an increasing trend, which follows the logical trend. Overall, the AST parameter is following the clinical trend to some extent but not perfectly. 38% of ASP data falls in the ambiguity zone, which is not ideal. Future work might include processing the ASP parameter to help the model learn the correct trend.

The glucose curve has low and steady risk below 120 mg/dL and increasing/high risk after 120 mg/dL. This trend is perfectly following the clinical normal range. The Amb % related to glucose was 7%, which is a low amount. Therefore, in this case, glucose is found to be following the clinical trend and contributing highly to the model's decision making.

In summary, for the Quadratic SVM model, the following features follow the logical trend with no or low Amb %: BMI, ALT, AST, and Glucose. Age and ASP follow the clinical trend partially but have high Amb % values at 45% and 38%, respectively. While HDL has a lower Amb % of 16%, the increasing HDL trend after 2 mmol/L needs additional investigation. See

Table 3.2 for a summary of the results.

#### ***b. Gaussian scale I SVM***

The partial dependency plots (PDP) for Gaussian Scale I SVM for male specific models are shown in Figure 3.5.

The parameter Age shows an increasing trend with a steep increase between 30 – 40 years. While the Age curve in the PDP is generally following the logical trend, 55% of the values fall in the ambiguity zone. Therefore, the contribution of Age in Gaussian I SVM's predictions is low.

BMI shows a decreasing risk curve within the clinical normal range of 18.5 to 25. The curve within the normal range is expected to be steady and not decreasing but after 25, the risk of disease increases with increasing BMI until 40, and then plateaus out. Overall, this trend is following the logical pattern but 33% of the values lie in the ambiguous range.

HDL shows a decrease in risk after 1 mmol/L which follows the logical trend but after 2 mmol/L, the risk for HS shows an increase from the PDP. This increase in risk was also observed in Quadratic SVM's HDL curve and needs further investigation in the future. 16% of HDL data lies in the ambiguity zone – therefore, HDL plays a significant role in the model's decision making.

The ALT curve shows low risk between 6.25 to 25 U/L but increasing risk after 25 and until 55 U/L. This is not following the logical trend. Per the clinically defined normal, the ALT values in the 10 – 55 U/L should have low and steady risk. In this case, the ALT feature does not seem to follow that trend. Therefore, more work is needed to either process the ALT data such that the model can learn the correct trend or find other models that can read the ALT data as expected.

Like ALT, the AST curve also shows increasing risk in the normal range of 10 – 40 U/L. This feature also does not follow the logical trend and needs further processing and investigation. The results from other explainable AI tools can also be explored to understand the interpretation of this feature better.

The risk curve for ASP shows a fluctuating but low risk between 45 – 115 U/L (clinical normal). After 115 U/L the risk increases and eventually plateaus out. This feature overall follows the logical trend but 37% of the feature's partial dependencies lie in the ambiguity zone. Hence the feature is not able to contribute as highly to the model's decisions although its trend is as expected.

The glucose PDP shows a low risk until 100 mg/dL and increasing risk after that until 150 mg/dL. The risk after 150 mg/dL seems to be plateauing out. This feature does not follow the logical trend perfectly (with low risk until 100mg/dL instead of 120 mg/dL) but overall, it follows the trend. Further, only 12% of the feature's partial dependencies lie in the ambiguous zone.

Overall, the ASP and Glucose parameters in the Gaussian I SVM follow the logical trend from a clinical normal standpoint and contribute to the model's decision making. The features Age, BMI and ASP also follow the logical trend as expected but have high Amb % values of 55%, 33% and 37%, respectively. ALT and AST do not follow the logical trend and need further investigation. See Table 3.3 for a summary of the results

### *c. Gaussian scale II SVM*

The PDPs related to Gaussian II SVM for male population are shown in Figure 3.6. The first graph shows the partial dependency of Age with respect to HS risk. The model is interpreting

the feature 'Age' in the clinical range as expected. However, 38% of the Age data fall in the ambiguity zone. Overall, the input feature 'Age' follows the trend as expected from a general clinical normal perspective.

The BMI feature shows a gradual increase in risk between 18.5 to 25. While the curve in this range is expected to be steady, it is following the general logical trend. After 25, the partial dependency curve increases steeply and plateaus off after 45. Overall, the BMI feature is following the logical trend and has only 11% data in the ambiguity zone.

The HDL feature has a decreasing risk after the clinical normal value of 1 mmol/L which follows the normal trend as expected but after 2.75 mmol/L the risk starts to increase slightly. This slight increase in risk is not expected and needs additional investigation. Overall, the HDL PDP follows the logical trend until 2.75 mmol/L and has a low ambiguity percentage of 17%.

The ALT PDP shows a steeply increasing risk in the normal range of 10 – 55 U/L and thereafter shows a reduced, then plateaued out risk curve. This behavior does not follow the logical trend and requires additional investigation.

The AST PDP shows slowly increasing risk in the normal range of 10 – 40 U/L but increasing risk thereafter. Overall, this feature follows the logical trend and has only 12% ambiguity. Therefore, it contributes significantly to the model's decision making.

The ASP PDP shows low and steady risk in the normal range, followed by an increasing risk after the normal range. This behavior is expected and follows the logical trend from a clinical perspective. However, 85% of its values lie in the ambiguity range and therefore are not useful in the model's decision making. Additional data processing in the form of weighing the ASP parameter might benefit the overall model performance in the future.

The Glucose PDP has low and slowly increasing risk until the clinical normal of 120 mg/dL. After the normal range, the risk increases and plateaus out after 400 mg/dL. This feature follows the logical trend overall and has a low ambiguity percentage of 9%.

In summary for the Gaussian II SVM Model, the parameters: BMI, HDL, AST and Glucose follow the logical trend and have low ambiguity percentages. The features Age and ASP also follow the logical trend but have high percentage ambiguity at 38% and 85%, respectively. ALT does not follow the logical trend and needs additional investigation. See Table 3.4 for a summary of the results.



#### ***d. Comparison of model performances – male specific models***

Within the male-specific models, the Quadratic SVM model has maximum features (six of seven) that follow the logical trend. Except for the HDL feature, which needs additional investigation, all other features follow the logical trend in the Quadratic SVM model. The gaussian I and II SVM models have features that do not follow the logical trend. In Gaussian I SVM, the relationship of ALT and AST with HS (individually) is interpreted by the model against the clinical understanding. From a clinical standpoint, increasing levels of ALT and AST are associated with increasing HS risk. However, in this case, the increasing levels are being interpreted as decreasing risk by the model. Therefore, they are not considered to follow the logical trends. Similarly in Gaussian II SVM, the ALT parameter alone is being interpreted against the clinical understanding of ALT with HS risk.

Overall, in this work, the Quadratic SVM model for male data is found to be the best performing model as six of the seven input parameters in the model follow the logical trend (Age, BMI, ALT, AST, ASP, and Glucose), and one parameter follows the logical trend partially (HDL). Within the male – specific models, none of the models had all seven parameters following the logical trend. This finding can be considered to develop a future hybrid model that combines two models. It is envisioned that the six parameters following the logical trend are fed into the Quadratic SVM model and other feature (HDL) could be fed into a different model (possibly a simple logistic regression or tree based) that every feature is interpreted in the expected, logical manner. A hybrid model combining Quadratic SVM and tree-based/logistic regression can then be developed to predict HS.

### **3.5.2 Analysis of the models for female population**

Partial dependency plots related to the female population are shown in Figure 3.7 to Figure 3.9. The interpretations of these plots are shown in Table 3.5 – 3.7. A summary of the results for the female specific models is in table 3.9.

#### ***a. Quadratic SVM***

The PDPs related to Quadratic SVM for female population are shown in Figure 3.7. Each figure within the figure corresponds to an input feature. It is observed that the Gaussian II SVM

model is interpreting the feature 'Age' in the clinical range as expected. However, 27% of the Age data fall in the ambiguity zone. Overall, the input feature 'Age' follows the trend as expected from a general clinical normal perspective

The BMI feature shows a steadily increasing risk in the normal range with low risk between 50 – 60 kg/m<sup>2</sup>. Both these behaviors do not follow the clinical normal trend and need further investigation or feature processing.

The HDL feature shows a consistently decreasing risk with increasing HDL values, which follows the clinical normal trend logically. This pattern is expected and with only 13% of data in the ambiguity zone, the HDL feature contributes highly to the model's decision making.

The ALT feature shows a steady and low risk in the normal range of 10 – 55 U/L. After the normal range, the risk increases steeply. This trend also follows the clinical normal as expected and has 0% ambiguity. Therefore, it contributes highly to the model's decision making.

The AST feature shows highest and steady risk in the normal range with decreasing risk consistently thereafter. This does not follow the logical trend and is in fact being interpreted in the opposite of the expected, logical trend. Therefore, an inversion or similar other feature processing is necessary to potentially gain higher model performance. Additional investigation using other XAI tools can also be conducted in the future to interpret the feature's contribution to the Quadratic SVM model.

The ASP feature shows low but increasing risk in the normal range of 30 – 100 U/L. In an ideal case, the curve in the normal range should have been steady but is found to be linearly increasing in this case. However, after the normal value, the risk continues to increase, peaks at about 300 U/L and slightly decreases thereafter. Overall, the feature's behavior is interpreted to be following a logical trend with 22% of the data in the ambiguity zone.

The glucose curve has low and steady risk below 120 mg/dL and increasing/high risk after 120 mg/dL. This trend is perfectly following the clinical normal range. The Amb % related to glucose was 4%, which is a low amount. Therefore, in this case, glucose is found to be following the clinical trend and contributing highly to the model's decision making.

In summary for the Quadratic SVM model (female), the features: HDL, ALT, and Glucose are found to be following the logical trend from a clinical normal standpoint with zero or low ambiguity percentages. The features Age and ASP also follow the logical trend but have slightly higher ambiguity percentages at 27% and 22%, respectively. Finally, the BMI and AST features

need additional investigation and/or feature processing as they are not following their respective trends as expected. See Table 3.5 for a summary of the results.

#### ***b. Gaussian scale I SVM***

The PDPs related to Gaussian I SVM are in Figure 3.8. The first graph shows the partial dependence of Age on HS prediction. It is observed that the Age feature is being interpreted in a logical manner. However, 40% of the Age partial dependence data falls in the ambiguity zone and therefore, this feature does not contribute well to the model's decision making.

The BMI feature's partial dependence curve shows a steady and low risk curve in the normal range of 18.5 to 25. After 25, the risk increases linearly, peaks at about 35 kg/m<sup>2</sup> and plateaus thereafter. 18% of the BMI partial dependence data falls in the ambiguity zone. Overall, the BMI feature follows the logical trend and contributes to the Gaussian Scale I SVM's decision making.

The PDP of HDL reduces after the normal value of 1 mmol/L until 1.8 mmol/L. The increase in risk after 1.8 mmol/L does not follow the logical trend and needs further investigation. Further, the ambiguity percentage of HDL data was 21%. However, like that in the Male – specific models, the data regarding LDL and triglycerides was not used in this research due to a lack of data availability and therefore, the behavior of the HDL PDP is not clear.

The ALT PDP indicates linearly rising risk in the clinical normal range of 10 – 55 U/L. After the normal range, the plot slightly decreases and plateaus out. Both these behaviors do not follow the logical trend as expected, from a clinical normal standpoint. Although only 2% of the data is in the ambiguity zone, more investigation is needed in the future to understand and potentially correct the interpretation of ALT by the model.

The AST plot indicates low but slowly increasing risk in the normal range of 9 – 32 U/L. After the normal range of 32 U/L, the risk continues to increase until 50 U/L and then plateaus out. This curve largely follows the logical trend and has only 6% data in the ambiguity zone.

ASP partial dependency plot shows a decreasing and then slowly increasing risk in the normal range of 30 – 100 U/L. In an ideal case, the curve in the normal range would be steady. After the normal range, the ASP curve continues to increase and plateaus out eventually. Overall, this curve follows the logical trend and has only 14% data in the ambiguity range.

Glucose PDP has a low but slowly increasing risk in the normal range below 120 mg/dL. After the normal range, the risk continues to increase until 170 mg/dL and plateaus out thereafter. This PDP overall follows the logical trend with only 14% of the data in the ambiguity zone.

Overall, for the Gaussian I SVM model, the following features follow the logical trend and have low ambiguity percentages: BMI, AST, ASP, and Glucose. Age also follows the logical trend but has a high ambiguity percentage of 40%. ALT does not follow the logical trend and HDL needs additional investigation. See table 3.6 for a summary of the results.

### *c. Gaussian scale II SVM*

The PDPs related to Gaussian II SVM are in Figure 3.9. The first graph shows the partial dependence of Age with respect to HS risk. It is observed that the Gaussian II SVM model is interpreting the feature 'Age' in the clinical range as expected. However, 34% of the Age data fall in the ambiguity zone. Overall, the input feature 'Age' follows the trend as expected from a general clinical normal perspective

The PDP of the BMI feature shows a slowly increasing risk between the clinical normal range of 18.5 to 25. After 25, the risk increases steeply, plateauing out after 50. The BMI PDP has only 11% of its values in the ambiguity zone. Therefore, BMI is considered to be following the logical and expected trend, while contributing strongly to the model's decision making.

The plot for HDL indicates that the disease risk decreases after the clinical normal value of 1.3 mmol/L. However, the risk increases again after 2.5mmol/L. This increase in risk is not expected from a clinical standpoint. More investigation is required in the future to understand this trend. Further, in this case, 47% of the values from the PDP lie in the ambiguity zone and are therefore not very useful to the model.

The ALT PDP indicates steeply rising risk in the clinical normal range of 10 – 55 U/L. After the normal range, the plot decreases in slope, indicating low risk. Both these behaviors do not follow the logical trend as expected, when compared to the clinical normal. Although only 3% of the data is in the ambiguity zone, more investigation is needed to understand and potentially correct the interpretation of ALT by the model.

The AST curve shows a steady and low risk in the clinical normal range of 9 – 32 U/L. After the normal range, the risk appears to increase with increase in ALT values. These behaviors

are following the clinical normal trend alongside only 17% data in the ambiguity zone. Overall, AST is following the logical trend as expected and is contributing to the model's decision making.

The ASP curve also shows steady and low risk in the normal range of 30 – 100 U/L. The risk after the normal value increases but there is another low-risk period between 300 – 400 U/L. This low-risk period is does not follow the logical trend but overall, ASP follows the logical trend between 0 – 300 U/L and 400 to 700 U/L. 36% of the ASP data however lies in the ambiguity zone. Therefore, this feature might require additional processing to allow the model to capture the correct/logical trend.

A low and slowly increasing curve under 120 mg/dL is observed in the Glucose PDP. After 120, the curve increases steeply, peaks at around 250 and plateaus out thereafter. Overall, this pattern is mostly following the logical trend and has only 8% data in the ambiguity zone. Therefore, this parameter contributes significantly to decision making of the Gaussian II SVM model.

Overall, the parameters: BMI, AST and Glucose are found to be following the logical trend and have low ambiguity percentages. While Age and HDL also follow the logical trend, they have high ambiguity percentages of 34% and 47%, respectively. Finally, the parameter ALT is not following the logical trend and need additional investigation as part of future work in the Gaussian II SVM – female specific model. See

Table 3.7 for a summary of the results.

***d. Comparison of model performances – Female specific models***

Within the female-specific models, the Quadratic SVM model had two features that did not follow the logical trend. BMI and AST were being interpreted against the clinical understanding of their individual relationships with HS risk in this model. Further the HDL feature followed the logical trend only up to a certain threshold and needs additional investigation. The gaussian I and II SVM models had similar model interpretabilities. They both had one feature that does not follow the logical trend - ALT. The relationship of ALT with HS is interpreted by the models against the clinical understanding of ALT with HS. From a clinical standpoint, increasing levels of ALT are associated with increasing risk. However, in this case, the increasing levels are being interpreted as decreasing risk. Both the Gaussian I and II SVM models need additional investigation of their HDL parameters.

Overall, in this work, the Gaussian I and II models for female data had similar model interpretabilities. Each had five features that follow the logical trend (Age, BMI, AST, ASP, and Glucose), one feature that followed the logical trend partially (HDL), and one feature that did not follow the logical trend (ALT). However, the Gaussian I SVM model had a higher sensitivity of approximately 70% (compared to 67% sensitivity of Gaussian II SVM), and lower ambiguity for the HDL parameter. Therefore, the Gaussian I SVM is found to be the best model for female population in this work.

Like in male-specific models, none of the female-specific models had all seven input parameters following the expected logical trend. Therefore, using the insights from partial dependency plots, future development of a hybrid model can also be applied to female – population. In this case, a combination of Gaussian scale I and Quadratic SVM models can be implemented to create a hybrid model. Five features (Age, BMI, AST, ASP, and glucose) could be fed into the Gaussian I model, and two features (HDL and ALT) could be fed into the Quadratic SVM model. In this way, each model interprets the features in the expected, logical way. The outputs of both these models could then be combined into a hybrid model for HS prediction. It is postulated that such a hybrid model would improve the performance of HS prediction models.

### **3.5.3 Top predictors of HS**

Model performance in the context of XAI was assessed based on how many parameters are being interpreted in alignment with the clinical normal definition. To identify the best predictors, i.e., the features that provide most contribution to the model in decision making, the mean classification scores of each feature were compared. These results are available in Table 3.11 & Table 3.12 for male and female – specific models, respectively.

Using data from Table 3.11 & Table 3.12, bar charts were generated, as shown in Figure 3.10 & Figure 3.11. The top three predictors were found to be: ALT, AST, and Glucose, across all three models for male and female specific models. These results imply that the three parameters are making the most individual contributions in the model's decision making. More investigation of the combined effects of these three parameters will be useful in screening for HS, however it is out of scope for this research.

### **3.5.4 Comparison of results in male vs female populations**

The top three highly contributing predictors were found to be the same for all the male and female- specific models: ALT, AST, and Glucose. However, the best performing models were different for each sex.

In male-specific models, the quadratic SVM model was found to be the best in terms of following the logical trends from clinical perspective. In female-specific models, the Gaussian I was found to be the best. However, in both the sexes, none of the models had all seven input parameters following the logical trend. Therefore, in both cases, a theoretical framework for developing a hybrid model was provided in sections 3.5.1 d and 3.5.2 d.

Features that follow the logical trend for the best performing male vs female – specific models are shown in Table 3.10. While six features follow the logical trend for the Quadratic SVM (male-specific) model, only five features follow the logical trend in the Gaussian I SVM (female-specific) model. The one parameter that is different between these sexes is ALT. ALT in Quadratic SVM for male follows the logical trend but ALT in Gaussian I female does not. However, the ALT feature in Quadratic SVM female specific model also follows the logical trend. While the Quadratic SVM female is not the best performing model, it could be used within the theoretical framework described earlier to develop a hybrid model and improve the overall performance.

Another difference is in the best model for male vs female. The best model for male-data uses the quadratic/ polynomial kernel, whereas the best model for female-data uses the gaussian/ radial-basis kernel to create decision boundaries in higher-dimensional spaces. The polynomial and radial-basis functions are both non-linear kernels, but they utilize different orders of non-linear transformation to transform the training data. The best kernel choice depends on the input data used [21], and in this case, the kernel that best fits the male data was found to be Quadratic and that for female-data was found to be Gaussian Scale I. In general, polynomial kernels are found to perform better when the input-output relationship is simpler, whereas gaussian/radial-basis kernels are found to fit the complex relationships better [22]. Therefore, it is postulated that the relationships of input parameters with HS in female data are more complex (than in male-data) and fit the gaussian I SVM model better than the Quadratic SVM model.

Future implementation of these models with additional data, like lipid panel information and other relevant parameters, might improve the fit of the model leading to improved prediction performances. Comparison of results from other XAI tools might lead to additional insights regarding the best fitting model for male vs female datasets. Implementation of additional XAI tools and additional features are out of scope for this work and are provided as recommendations for future work.

### **3.6 Summary and conclusions**

In this chapter, three selected models (each for male and female specific populations) from objective 1 were evaluated from an interpretability perspective to include model transparency. The models for male and female populations were evaluated and interpreted separately for interpretability. Averaged partial dependency plots for each of seven predictors were plotted using MATLAB R2020b [17]. The obtained plots were compared with their respective clinically defined normal ranges. The results from the comparison were discussed and directions for potential future research were identified.

1. The best performing models were identified as Quadratic SVM in male population and Gaussian SVM scale 1 in female population.
2. The top three independent predictors for male and female data were identified using the mean of the partial dependencies. In both sexes, ALT, AST, and Glucose were



- found to be the most individually contributing features. These three parameters are found to be individually contributing highly to HS prediction.
3. Results for male and female populations were found to vary slightly with male models outperforming the female models in terms of alignment with clinical normal values. These findings need to be investigated further, with larger, richer, and more robust datasets.
  4. Impacts of HDL on NAFLD also need more investigation, particularly the increase of HDL after a certain threshold and its relationship with NAFLD risk
  5. A theoretical framework for developing hybrid models is provided in sections 3.5.1 d and 3.5.2 d.

### **3.7 Recommendations for future work**

Based on the above summary and conclusions, the following are recommended for future work.

1. Developing hybrid models using the theoretical framework provided in sections 3.5.1 d and 3.5.2 d. are recommended.
2. Additional testing the combined effect of ALT, AST, and Glucose on model performances are recommended.
3. Use of other XAI approaches including additional features (hormonal data, other lipid panel data like triglycerides, total cholesterol and LDL) might provide additional insights and improve prediction performances.

### 3.8 Figures

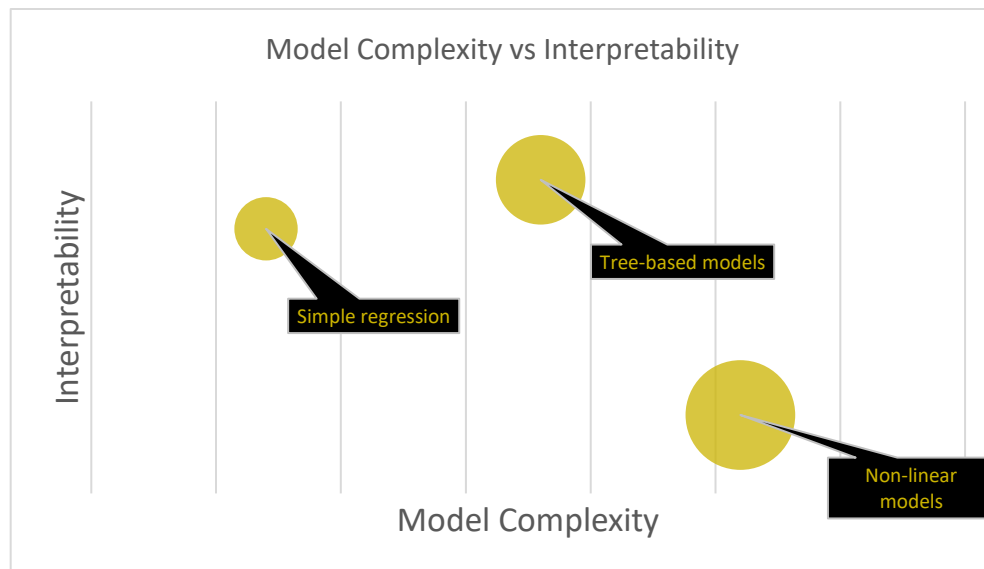


Figure 3.1: Typical machine learning models complexity vs interpretability

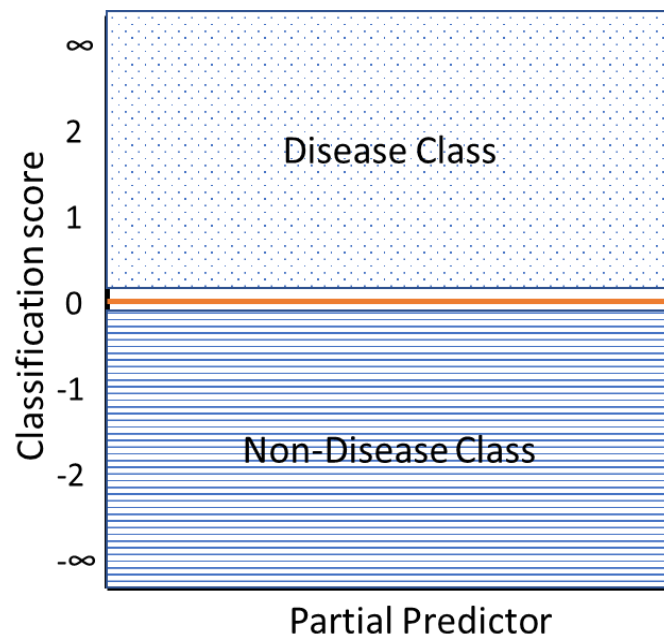


Figure 3.2: Partial prediction plot with ambiguity zone ( $0 \pm 0.15$ )

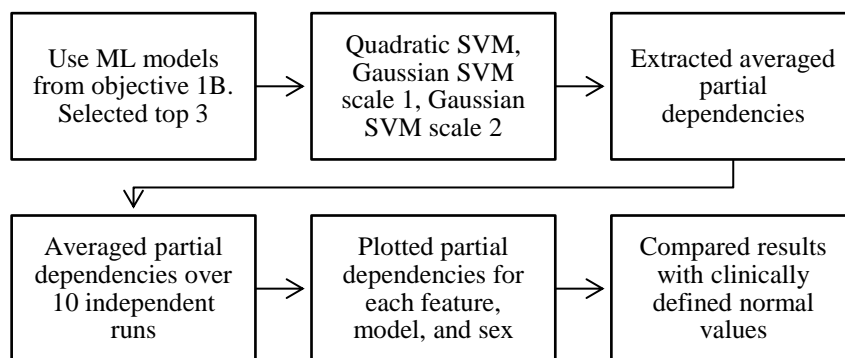


Figure 3.3: Methods used for explainability analysis

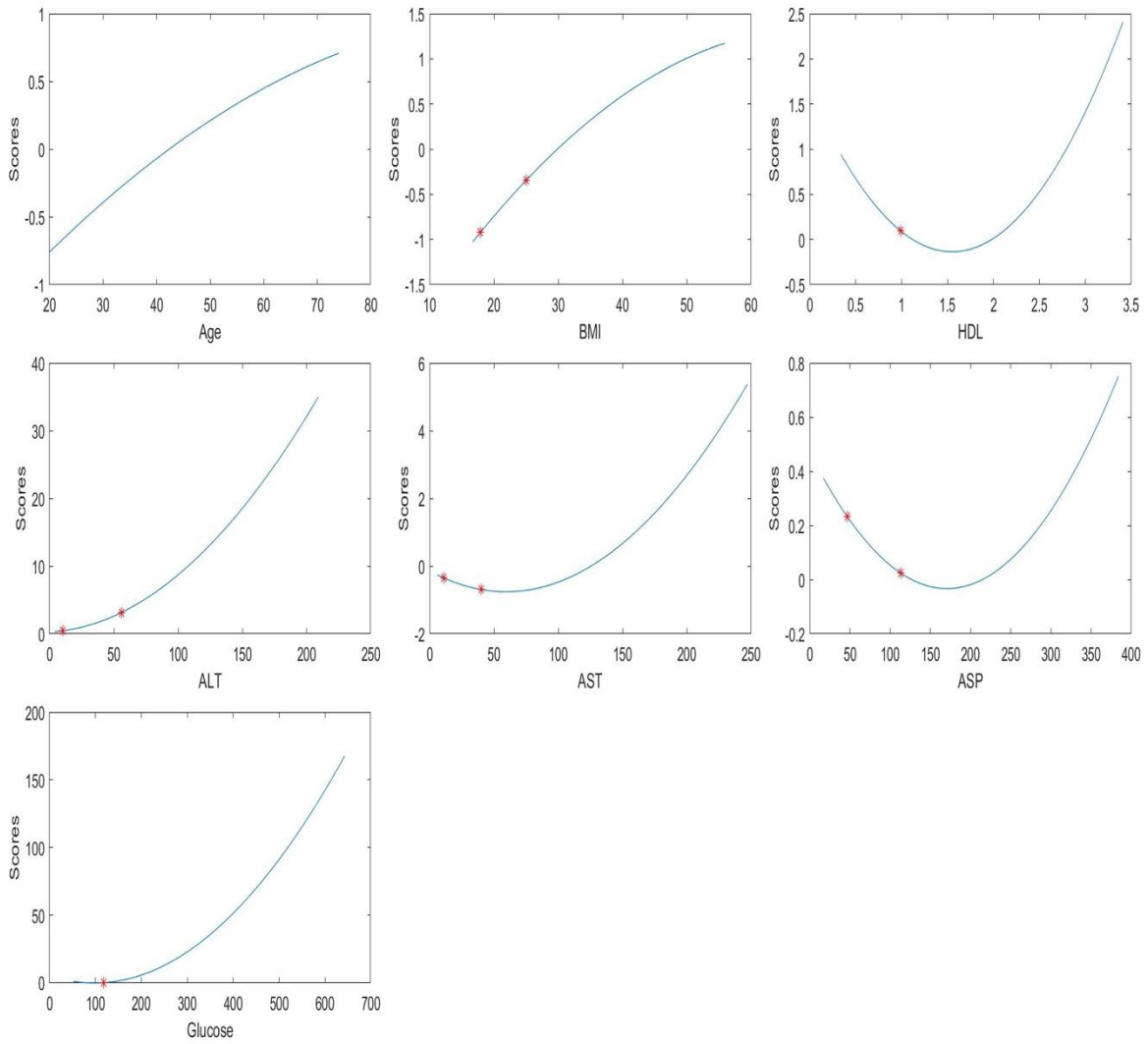


Figure 3.4: Quadratic SVM - Partial dependency plots for each predictor - male population

*Note: The asterisk(s) on each plot indicates the clinically defined normal value or the normal range as per Table 3.1*

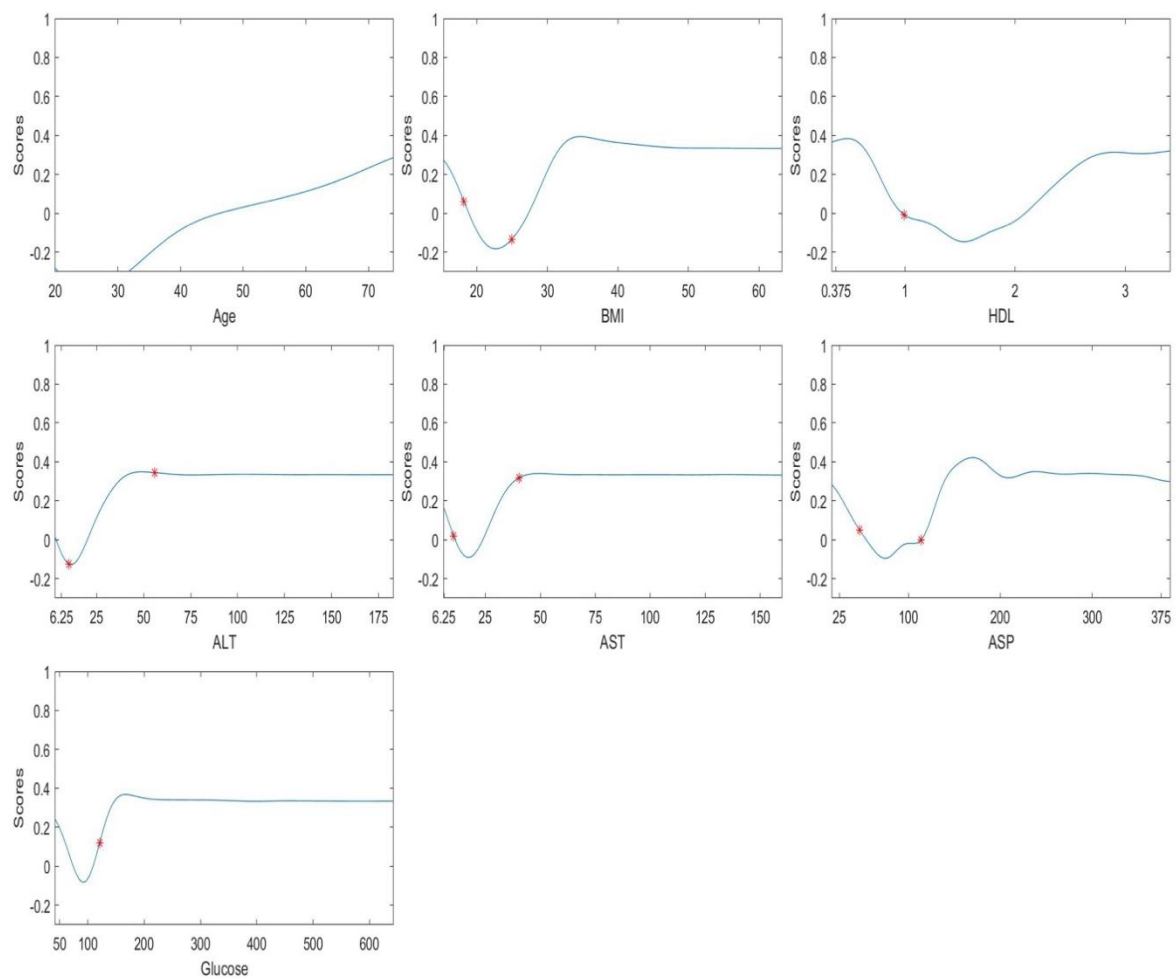


Figure 3.5: Gaussian SVM – Scale 1 - Partial dependency plots for each predictor - male population

*Note: The asterisk(s) on each plot indicates the clinically defined normal value or the normal range as per Table 3.1*

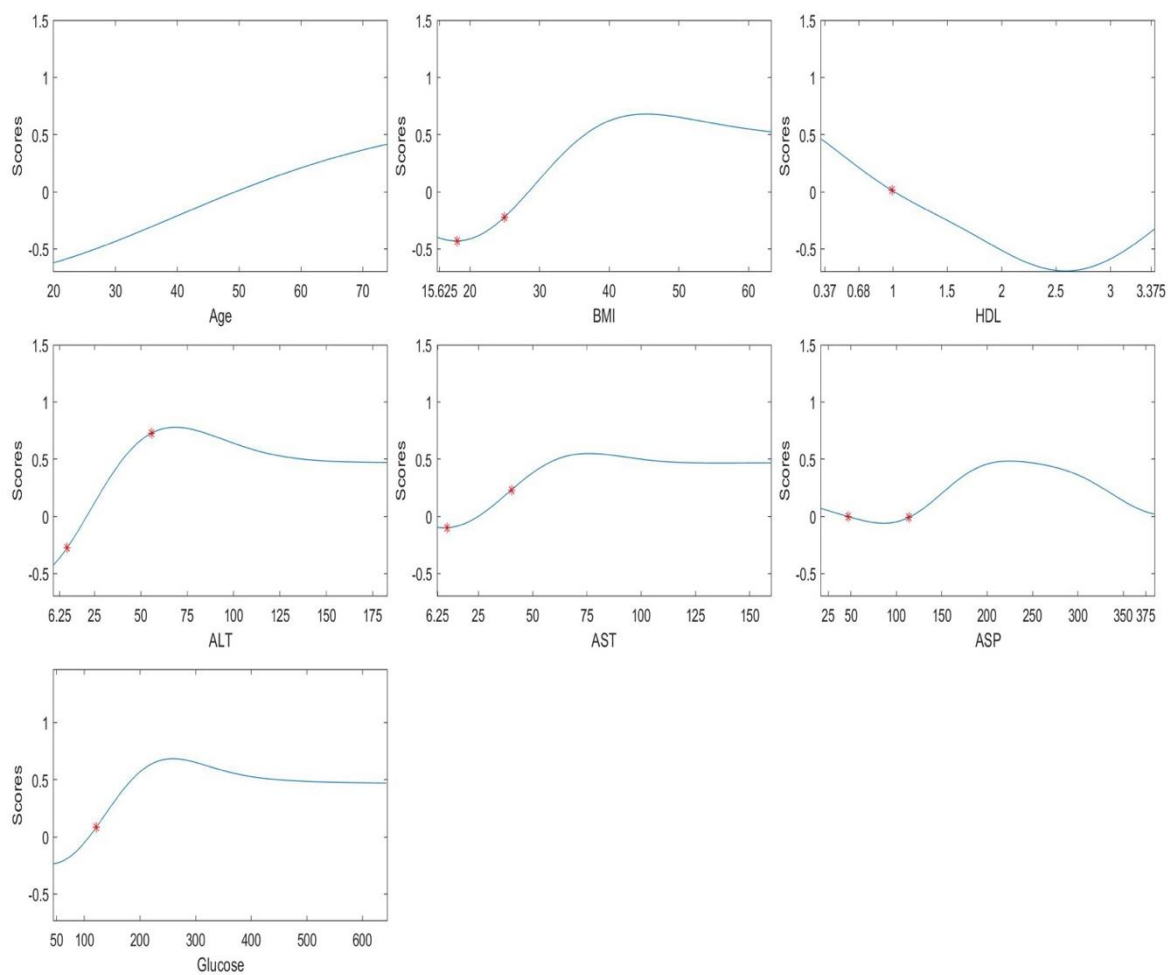


Figure 3.6: Gaussian SVM – Scale 2 - Partial dependency plots for each predictor - male population

*Note: The asterisk(s) on each plot indicates the clinically defined normal value or the normal range as per Table 3.1*

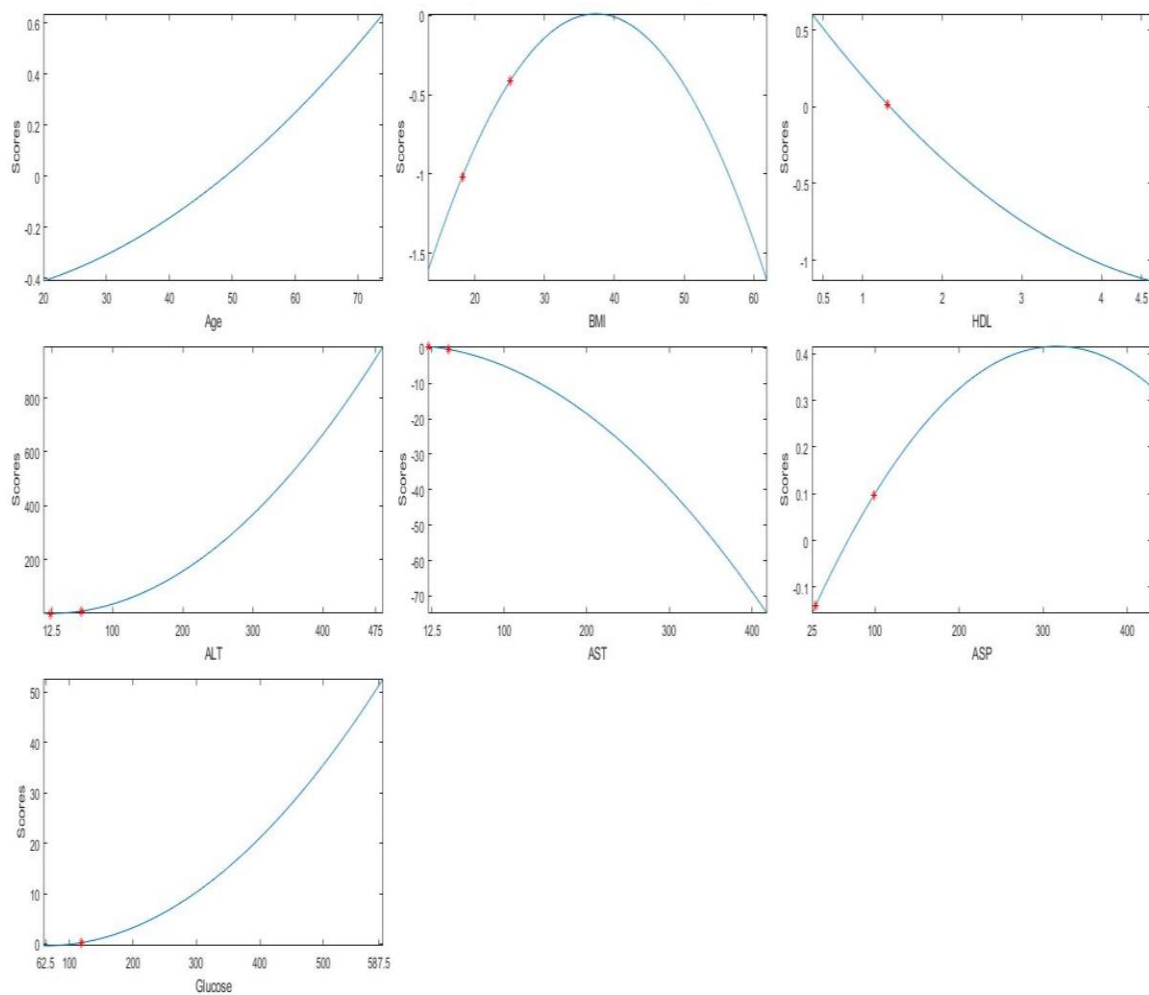


Figure 3.7: Quadratic SVM - Partial dependency plots for each predictor - female population

*Note: The asterisk(s) on each plot indicates the clinically defined normal value or the normal range as per Table 3.1*

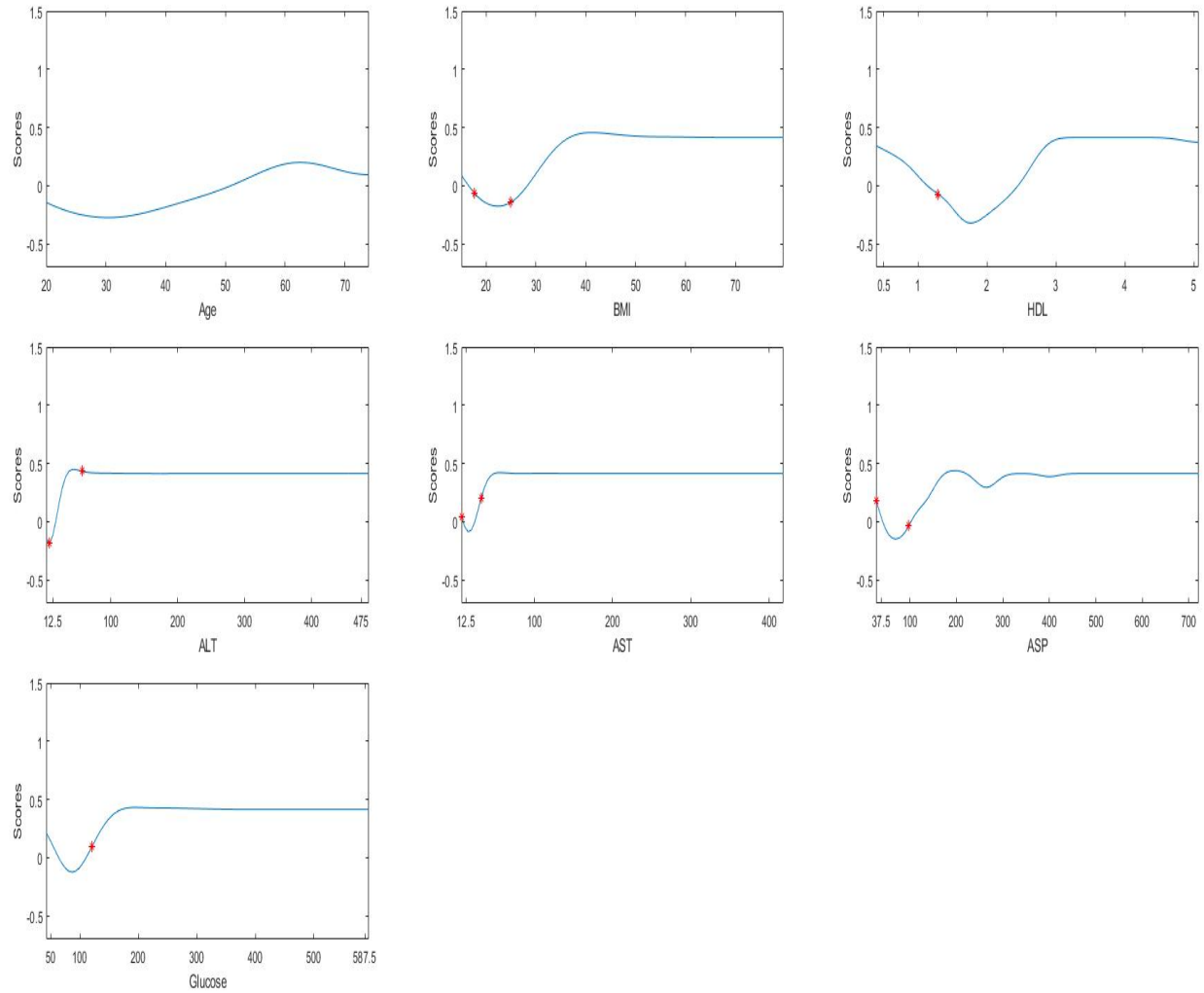


Figure 3.8: Gaussian SVM – Scale 1 - Partial dependency plots for each predictor - female population

*Note: The asterisk(s) on each plot indicates the clinically defined normal value or the normal range as per Table 3.1*



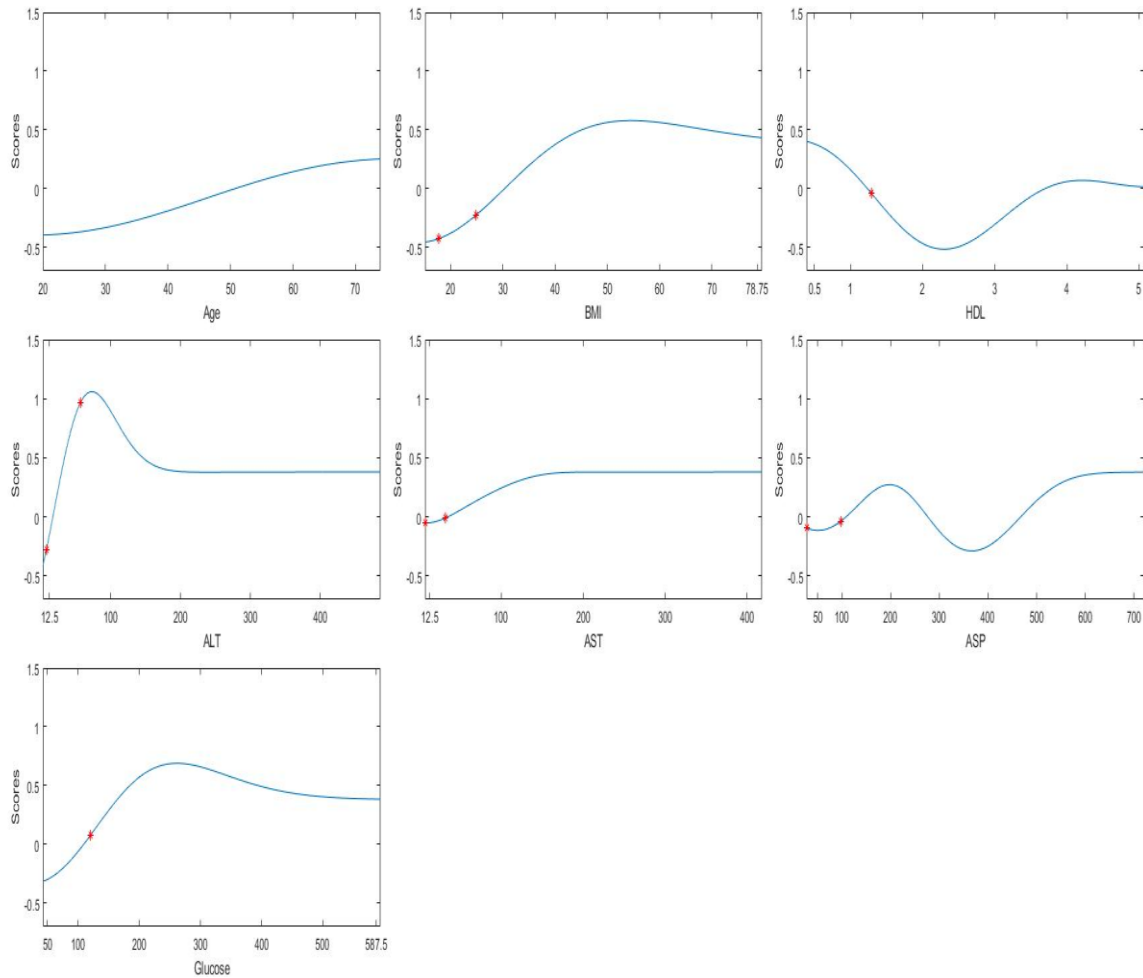


Figure 3.9: Gaussian SVM – Scale 2 - Partial dependency plots for each predictor - female population

*Note: The asterisk(s) on each plot indicates the clinically defined normal value or the normal range as per Table 3.1*

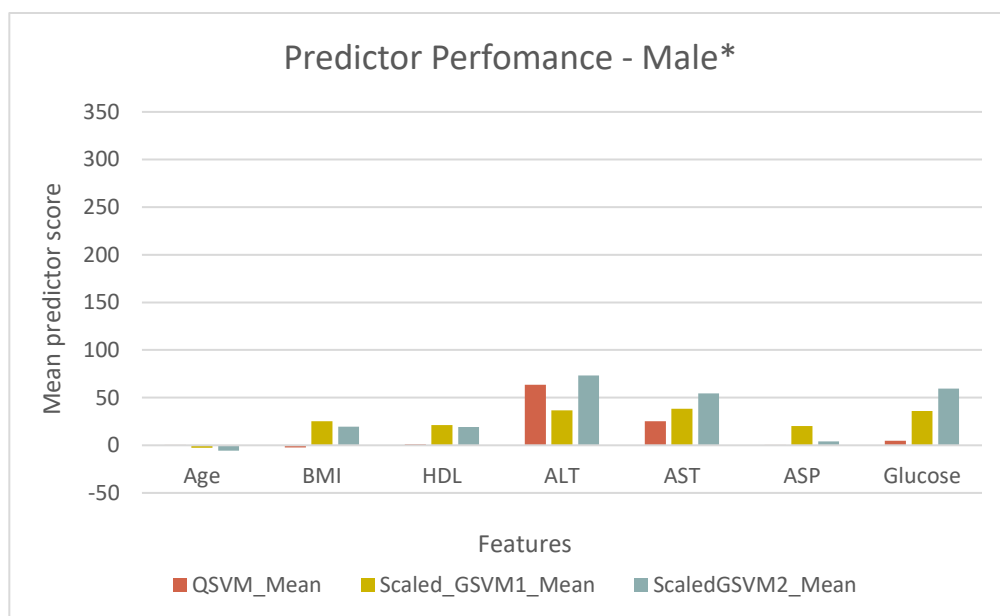


Figure 3.10: Individual predictor performance - male population

*\*Note: Data in Gaussian scales 1 & 2 are scaled by 100 for representation purposes*

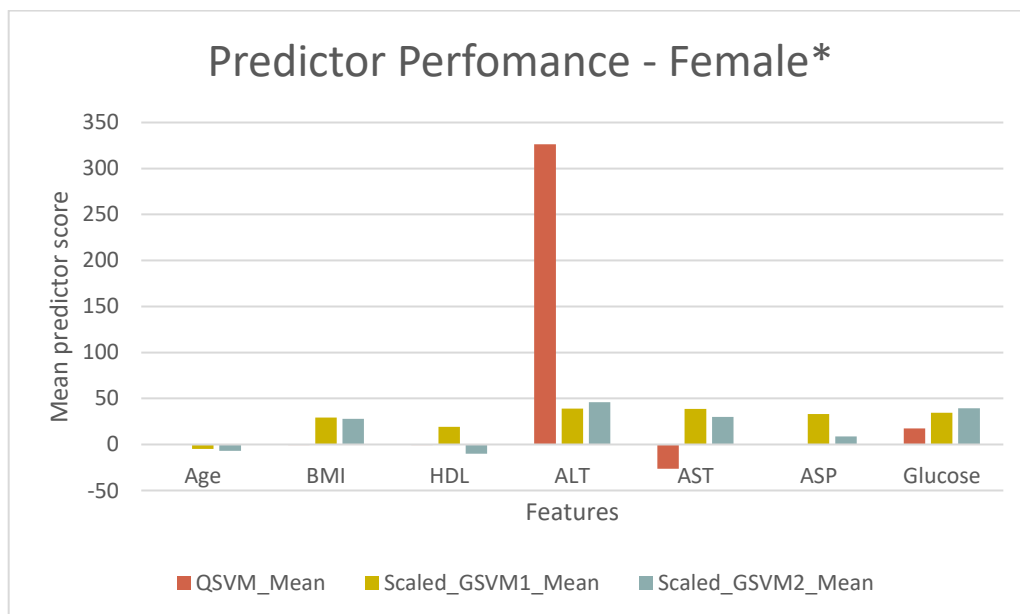


Figure 3.11: Individual predictor performance - female population

*\*Note: Data in Gaussian scales 1 & 2 are scaled by 100 for representation purposes*

### 3.9 Tables

Table 3.1: Clinically defined normal values for male and female populations

Feature	Clinical normal values		Reference
	Male	Female	
BMI	18.5 to 25 kg/m <sup>2</sup>	18.5 to 25 kg/m <sup>2</sup>	[23]
HDL	$\geq 1$ mmol/L	$\geq 1.3$ mmol/L	[24]
ALT	10 – 55 U/L	10 – 55 U/L	[25]
AST	10 – 40 U/L	9 – 32 U/L	[26]
ASP	45 – 115 U/L	30 – 100 U/L	[26]
Glucose	$< 120$ mg/dL	$< 120$ mg/dL	[27]

Table 3.2: Male quadratic SVM – partial dependency result analysis

Feature	Layer 1 (Observations from XAI results)	Clinical Normal	Layer 2 (Implication as defined by clinical practices)	Ambiguity %
Age	Linearly/Consistently increasing with age	NA	Follows the logical trend	45
BMI	Steadily increasing with BMI, even within the normal range	18.5 to 25 kg/m <sup>2</sup>	Follows the logical trend	0
HDL	Decreasing then increasing curve. High risk between 0.5 to 1. Decreasing risk after 1 but increasing again after 2.	$\geq 1$ mmol/L	Follows the logical trend until 2 mmol/L but needs additional investigation after 2 mmol/L	16
ALT	Increasing risk with increasing ALT, slowly increasing until 50, then increases steeply	10 – 55 U/L	Follows the logical trend	0
AST	Stable, lower risk between 10 to 90, but increasing risk after 90	10 – 40 U/L	Follows the logical trend	6
ASP	Decreasing risk curve from 0 to 200, then increases after 200 steadily	45 – 115 U/L	Follows the logical trend	38
Glucose	Low risk until 150. Slow increase in risk from 150 to 200, then steep increase after 200	$< 120$ mg/dL	Follows the logical trend	7

Table 3.3: Male gaussian I SVM – partial dependency result analysis

Feature	Layer 1 (Observations from XAI results)	Clinical Normal	Layer 2 (Implication as defined by clinical practices)	Ambiguity %
Age	Linear increase in risk from 30 to 40, then slow increase in risk after 40	NA	Follows the logical trend after 30	55
BMI	Decreasing risk until 25. Especially low risk between 20 to 25	18.5 to 25 kg/m <sup>2</sup>	Follows the logical trend	33
HDL	Starts at high risk with low HDL but decreasing risk as HDL increases, until 1.5. Increasing risk after 1.5	$\geq 1$ mmol/L	Follows the logical trend until 1.5 mmol/L and needs additional investigation after 1.5 mmol/L	16
ALT	Low risk between 6.25 to 25, high risk between 25 to 50. Reduced but overall stable, high risk after 50	10 – 55 U/L	Increasing risk curve in the normal range does not follow the logical trend	7
AST	Decreased risk from 0 to 25, steady, high risk after 25, stable high risk after 55	10 – 40 U/L	Increasing risk curve in the normal range does not follow the logical trend	7
ASP	Decreasing risk between 0 to 115. Steady increase in risk after 115, with highest risk between 115 to 150	45 – 115 U/L	Follows the logical trend	37
Glucose	Decreasing risk between 0 to 120. Lowest risk between 75 to 120. Increasing risk between 120 to 200. Stable high risk after 200	$< 120$ mg/dL	Follows the logical trend	12

Table 3.4: Male gaussian II SVM – partial dependency result analysis

Feature	Layer 1 (Observations from XAI results)	Clinical Normal	Layer 2 (Implication as defined by clinical practices)	Ambiguity %
Age	Linearly/Consistently increasing with age	NA	Follows the logical trend	38
BMI	Low but increasing risk between 15 to 25. Highest risk at 40. Increasing risk overall with increasing BMI	18.5 to 25 kg/m <sup>2</sup>	Follows the logical trend	11
HDL	Decreasing risk with increasing HDL until 2.75. Lowest risk at 2.75. Slow increase in risk after 2.75	$\geq 1$ mmol/L	Follows the logical trend until 2.75 mmol/L and needs additional investigation after 2.75 mmol/L	17
ALT	Low but increasing risk from 0 to 60. Highest risk between 60 to 70. Stable, high risk after 75	10 – 55 U/L	Increasing risk in the normal range does not follow the logical trend	4
AST	Low increasing risk between 0 to 40. Highest risk between 50 to 75. Stable high risk after 75	10 – 40 U/L	Follows the logical trend	12
ASP	Low, decreasing risk between 0 to 120. Increasing risk after 120 with highest risk between 200 to 250	45 – 115 U/L	Follows the logical trend	85
Glucose	Low increasing risk until 120. Highest risk between 200 to 300. Stable high risk after 300.	$< 120$ mg/dL	Follows the logical trend	9

Table 3.5: Female quadratic SVM – partial dependency result analysis

Feature	Layer 1 (Observations from XAI results)	Clinical Normal	Layer 2 (Implication as defined by clinical practices)	Ambiguity %
Age	Linearly/Consistently increasing with age	NA	Follows the logical trend	27
BMI	Increasing with increase in BMI, but decreasing risk after 40	18.5 to 25 kg/m <sup>2</sup>	Increasing risk in the normal range and decreasing thereafter, does not follow the logical trend	31
HDL	Decreasing risk with increase HDL until 2.5, then increases	$\geq 1.3$ mmol/L	Follows the logical trend	13
ALT	Low risk from 0 to 50, then steady increase	10 – 55 U/L	Follows the logical trend	0
AST	Decreasing risk as AST increases, almost linearly	9 – 32 U/L	Decreasing risk with increasing AST does not follow the logical trend	2
ASP	Increase until 300, then decreasing risk with increasing value	30-100 U/L	Follows the logical trend	22
Glucose	Linearly/Consistently increasing with increase in glucose	$< 120$ mg/dL	Follows the logical trend	4

Table 3.6: Female gaussian I SVM – partial dependency result analysis

Feature	Layer 1 (Observations from XAI results)	Clinical Normal	Layer 2 (Implication as defined by clinical practices)	Ambiguity %
Age	Low risk between 20 to 40 with lowest risk at 30. Increasing risk from 40 to end	NA	Follows the logical trend	40
BMI	Decreasing risk from 12.5 to 25. Lowest risk between 18 to 25. Increase after 30	18.5 to 25 kg/m <sup>2</sup>	Follows the logical trend	18
HDL	Decreasing risk from 0.5 to 1.8. Lowest risk at 1.5 to 1.7. Increasing risk after 2.	$\geq 1.3$ mmol/L	Follows the logical trend until 1.8 mmol/L, needs additional investigation after 1.8 mmol/L	21
ALT	Increasing risk from 10 - 50. Highest risk at 50. Slightly reduced but stable risk after 50.	10 – 55 U/L	Increasing risk in the normal range does not follow the logical trend	2
AST	Lowest risk between 0 to 20. Increasing between 20 to 40 and steady high risk after 40.	9 – 32 U/L	Follows the logical trend	6
ASP	Decreasing risk from 0 to 90. Increasing risk from 90 to 275	30 - 100U/L	Follows the logical trend	14
Glucose	Low risk from 0 to 110. Slowly increasing risk after 110. Highest risk after 180	$< 120$ mg/dL	Follows the logical trend	14

Table 3.7: Female gaussian II SVM – partial dependency result analysis

Feature	Layer 1 (Observations from XAI results)	Clinical Normal	Layer 2 (Implication as defined by clinical practices)	Ambiguity %
Age	Increasing with age	NA	Follows the logical trend	34
BMI	Increasing with BMI. Highest risk between 40 - 60	18.5 to 25 kg/m <sup>2</sup>	Follows the logical trend	11
HDL	Decreasing with increasing HDL, lowest at 2.2, increasing after 2.5	$\geq 1.3$ mmol/L	Follows the logical trend until 2.5 mmol/L, needs additional investigation after 2.5 mmol/L	47
ALT	Increasing until 60, then decreasing and steady after 100	10 – 55 U/L	Steeply increasing and high risk in the normal range does not follow the logical trend	2
AST	Low but increasing until 90. Steady high after 90	9 – 32 U/L	Follows the logical trend	17
ASP	Steady low from 25 to 100, increases slightly after 100. Lowest risk between 300 - 400	30-100U/L	Follows the logical trend except for the low risk between 300 - 400 U/L	36
Glucose	Increasing but lowest from 0 to 120. Then increasing after that, highest at 250	$< 120$ mg/dL	Follows the logical trend	8



Table 3.8: Male-specific model observations

Models (→)	Quadratic		Gaussian I		Gaussian II	
Features (↓)	Comment on the trend of the partial dependency plot, compared to the clinical normal	Amb %	Comment on the trend of the partial dependency plot, compared to the clinical normal	Amb %	Comment on the trend of the partial dependency plot, compared to the clinical normal	Amb %
Age	Follows the logical trend	45	Follows the logical trend after 30	55	Follows the logical trend	38
BMI	Follows the logical trend	0	Follows the logical trend	33	Follows the logical trend	11
HDL	Follows the logical trend until 2 mmol/L but needs additional investigation after 2 mmol/L	16	Follows the logical trend until 2 mmol/L and needs additional investigation after 2 mmol/L	16	Follows the logical trend until 2.75 mmol/L and needs additional investigation after 2.75 mmol/L	17
ALT	Follows the logical trend	0	Increasing risk curve in the normal range does not follow the logical trend	7	Increasing risk in the normal range does not follow the logical trend	4
AST	Follows the logical trend	6	Increasing risk curve in the normal range does not follow the logical trend	7	Follows the logical trend	12
ASP	Follows the logical trend	38	Follows the logical trend	37	Follows the logical trend	85
Glucose	Follows the logical trend	7	Follows the logical trend	12	Follows the logical trend	9

Table 3.9: Female-specific model observations

Models (→)	Quadratic		Gaussian I		Gaussian II	
Features (↓)	Comment on the trend of the partial dependency plot, compared to the clinical normal	Amb %	Comment on the trend of the partial dependency plot, compared to the clinical normal	Amb %	Comment on the trend of the partial dependency plot, compared to the clinical normal	Amb %
Age	Follows the logical trend	27	Follows the logical trend	40	Follows the logical trend	34
BMI	Increasing risk in the normal range and decreasing thereafter, does not follow the logical trend	31	Follows the logical trend	18	Follows the logical trend	11
HDL	Follows the logical trend	13	Follows the logical trend until 1.8 mmol/L, needs additional investigation after 1.8 mmol/L	21	Follows the logical trend until 2.5 mmol/L, needs additional investigation after 2.5 mmol/L	47
ALT	Follows the logical trend	0	Increasing risk in the normal range does not follow the logical trend	2	Steeply increasing and high risk in the normal range does not follow the logical trend	2
AST	Decreasing risk with increasing AST does not follow the logical trend	2	Follows the logical trend	6	Follows the logical trend	17
ASP	Follows the logical trend	22	Follows the logical trend	14	Follows the logical trend	36
Glucose	Follows the logical trend	4	Follows the logical trend	14	Follows the logical trend	8

Table 3.10: Comparison of best performing models in male vs in female populations

Models	Features that follow the logical trend					
Male – Quadratic SVM	Age	BMI	ALT	AST	ASP	Glucose
Female – Gaussian I SVM	Age	BMI	AST	ASP	Glucose	

Table 3.11: Mean of predictor performances – male-specific models

Param	QSVM Mean	GSVM1 Mean	GSVM2 Mean
Age	0.123	-0.026	-0.058
BMI	-2.317	0.252	0.196
HDL	0.969	0.214	0.192
ALT	63.554	0.368	0.732
AST	25.097	0.384	0.545
ASP	0.136	0.203	0.041
Glucose	4.727	0.360	0.595

Table 3.12: Mean of predictor performances – female-specific models

Param	QSVM Mean	GSVM1 Mean	GSVM2 Mean
Age	0.014	-0.048	-0.069
BMI	-0.551	0.293	0.278
HDL	-0.457	0.191	-0.100
ALT	326.341	0.391	0.459
AST	-26.476	0.385	0.299
ASP	0.269	0.332	0.087
Glucose	17.335	0.346	0.395

### 3.10 References

- [1] A. Shaban-Nejad, M. Michalowski, and D. L. Buckeridge, “Explainability and Interpretability: Keys to Deep Medicine,” in *Explainable AI in Healthcare and Medicine*, vol. 914, A. Shaban-Nejad, M. Michalowski, and D. L. Buckeridge, Eds. Cham: Springer International Publishing, 2021, pp. 1–10. doi: 10.1007/978-3-030-53352-6\_1.
- [2] Y. Li, X. Wang, J. Zhang, S. Zhang, and J. Jiao, “Applications of artificial intelligence (AI) in researches on non-alcoholic fatty liver disease (NAFLD) : A systematic review,” *Rev. Endocr. Metab. Disord.*, Aug. 2021, doi: 10.1007/s11154-021-09681-x.
- [3] S. N. Payrovnaziri *et al.*, “Explainable artificial intelligence models using real-world electronic health record data: a systematic scoping review,” *J. Am. Med. Inform. Assoc.*, vol. 27, no. 7, pp. 1173–1185, Jul. 2020, doi: 10.1093/jamia/ocaa053.
- [4] F. K. Došilović, M. Brčić, and N. Hlupić, “Explainable artificial intelligence: A survey,” in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2018, pp. 0210–0215. doi: 10.23919/MIPRO.2018.8400040.
- [5] J. Duell, X. Fan, B. Burnett, G. Aarts, and S.-M. Zhou, “A Comparison of Explanations Given by Explainable Artificial Intelligence Methods on Analysing Electronic Health Records,” in *2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI)*, Jul. 2021, pp. 1–4. doi: 10.1109/BHI50953.2021.9508618.
- [6] G. Vilone and L. Longo, “Explainable Artificial Intelligence: a Systematic Review,” *ArXiv200600093 Cs*, Oct. 2020, Accessed: Feb. 24, 2022. [Online]. Available: <http://arxiv.org/abs/2006.00093>
- [7] “Interpretability.” <https://www.mathworks.com/discovery/interpretability.html> (accessed Feb. 24, 2022).
- [8] O. Asan, A. E. Bayrak, and A. Choudhury, “Artificial Intelligence and Human Trust in Healthcare: Focus on Clinicians,” *J. Med. Internet Res.*, vol. 22, no. 6, p. e15154, Jun. 2020, doi: 10.2196/15154.
- [9] L. Gordon, T. Grantcharov, and F. Rudzicz, “Explainable Artificial Intelligence for Safe Intraoperative Decision Support,” *JAMA Surg.*, vol. 154, no. 11, pp. 1064–1065, Nov. 2019, doi: 10.1001/jamasurg.2019.2821.
- [10] S. M. Lauritsen *et al.*, “Explainable artificial intelligence model to predict acute critical illness from electronic health records,” *Nat. Commun.*, vol. 11, no. 1, p. 3852, Jul. 2020, doi: 10.1038/s41467-020-17431-x.
- [11] M. Ghassemi, L. Oakden-Rayner, and A. L. Beam, “The false hope of current approaches to explainable artificial intelligence in health care,” *Lancet Digit. Health*, vol. 3, no. 11, pp. e745–e750, Nov. 2021, doi: 10.1016/S2589-7500(21)00208-9.

- [12] M. T. Ribeiro, S. Singh, and C. Guestrin, ““Why Should I Trust You?”: Explaining the Predictions of Any Classifier,” *ArXiv160204938 Cs Stat*, Aug. 2016, Accessed: Sep. 29, 2021. [Online]. Available: <http://arxiv.org/abs/1602.04938>
- [13] S. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” *ArXiv170507874 Cs Stat*, Nov. 2017, Accessed: Feb. 24, 2022. [Online]. Available: <http://arxiv.org/abs/1705.07874>
- [14] J. H. Friedman, “Greedy function approximation: A gradient boosting machine.,” *Ann. Stat.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001, doi: 10.1214/aos/1013203451.
- [15] T. Hastie and R. Tibshirani, *J. Friedman The Elements of Statistical Learning. Chapter 6*. Springer Verlag, New York, 2001.
- [16] H. Chen, S. Lundberg, and S.-I. Lee, “Explaining Models by Propagating Shapley Values of Local Components,” *ArXiv191111888 Cs Stat*, Nov. 2019, Accessed: Oct. 07, 2021. [Online]. Available: <http://arxiv.org/abs/1911.11888>
- [17] “Compute partial dependence - MATLAB partialDependence.” <https://www.mathworks.com/help/stats/regressiontree.partialdependence.html> (accessed Nov. 01, 2021).
- [18] “Classify observations using support vector machine (SVM) classifier - MATLAB predict.” <https://www.mathworks.com/help/stats/classreg.learning.classif.compactclassificationsvm.predict.html> (accessed Nov. 01, 2021).
- [19] P. Golabi, J. Paik, R. Reddy, E. Bugianesi, G. Trimble, and Z. M. Younossi, “Prevalence and long-term outcomes of non-alcoholic fatty liver disease among elderly individuals from the United States,” *BMC Gastroenterol.*, vol. 19, no. 1, p. 56, Dec. 2019, doi: 10.1186/s12876-019-0972-6.
- [20] J. Frith, C. P. Day, E. Henderson, A. D. Burt, and J. L. Newton, “Non-Alcoholic Fatty Liver Disease in Older People,” *Gerontology*, vol. 55, no. 6, pp. 607–613, 2009, doi: 10.1159/000235677.
- [21] A. Ben-Hur, C. S. Ong, S. Sonnenburg, B. Schölkopf, and G. Rätsch, “Support Vector Machines and Kernels for Computational Biology,” *PLoS Comput. Biol.*, vol. 4, no. 10, p. e1000173, Oct. 2008, doi: 10.1371/journal.pcbi.1000173.
- [22] M. F. Hussain, R. R. Barton, and S. B. Joshi, “Metamodeling: Radial basis functions, versus polynomials,” *Eur. J. Oper. Res.*, vol. 138, no. 1, pp. 142–154, Apr. 2002, doi: 10.1016/S0377-2217(01)00076-5.
- [23] “Hyperglycemia in diabetes - Diagnosis and treatment - Mayo Clinic.” <https://www.mayoclinic.org/diseases-conditions/hyperglycemia/diagnosis-treatment/drc-20373635> (accessed Mar. 30, 2021).

- [24] N. C. for B. Information, U. S. N. L. of M. 8600 R. Pike, B. MD, and 20894 Usa, *High cholesterol: Overview*. Institute for Quality and Efficiency in Health Care (IQWiG), 2017. Accessed: Mar. 15, 2022. [Online]. Available: <https://www.ncbi.nlm.nih.gov/books/NBK279318/>
- [25] P. Y. Kwo, S. M. Cohen, and J. K. Lim, “ACG Clinical Guideline: Evaluation of Abnormal Liver Chemistries,” *Am. J. Gastroenterol.*, vol. 112, no. 1, pp. 18–35, Jan. 2017, doi: 10.1038/ajg.2016.517.
- [26] “Understand Liver Enzyme Test Results — American Liver Foundation.” <https://liverfoundation.org/understand-liver-enzyme-test-results-2/> (accessed Mar. 30, 2021).
- [27] “Risk Factors for Diabetes | NIDDK,” *National Institute of Diabetes and Digestive and Kidney Diseases*. <https://www.niddk.nih.gov/health-information/professionals/clinical-tools-patient-management/diabetes/game-plan-preventing-type-2-diabetes/prediabetes-screening-how-why/risk-factors-diabetes> (accessed Mar. 30, 2021).

## APPENDIX C. P2 - CODE

### 1. MATLAB CODE TO EXTRACT PLOT PARTIAL DEPENDENCIES

```
%%%%%%%%%
% Created on: 08/09/2021
% Input: Selected trained models from paper 1
% Output: Partial dependence plots
% Author: Ridhi Deo
% File name: obj2_matlab_3 (R2020b [17]) )
% Description: This code was testing the ML models from paper 1, and plot their partial
dependencies
%%%%%%%%%

%% For marking the clinically normal values for Male or female, make sure to
% change the values based on male/female runs

%% Male clinically defined normals:
% BMI: 18.5 to 25
% HDL: 1mmol/L or higher
% ALT: 10 to 55 U/L
% AST: 10 - 40 U/L
% ASP: 45 - 115 U/L
% Glucose: 120mg/dL or lower

%% Female clinically defined normals:
% BMI: 18.5 to 25
% HDL: 1.3 mmol/L or higher
% ALT: 10 to 55 U/L
% AST: 9 - 32 U/L
% ASP: 30 - 100 U/L
% Glucose: 120mg/dL or lower

test = test_male; % Need to change this depending on female/male
training = training_male; % Need to change this depending on female/male

%% Model 4: quadratic svm
QSVM_Age_pd = zeros(100,10);
QSVM_Age_x = zeros(100,10);
QSVM_BMI_pd = zeros(100,10);
QSVM_BMI_x = zeros(100,10);
QSVM_HDL_pd = zeros(100,10);
QSVM_HDL_x = zeros(100,10);
QSVM_ALT_pd = zeros(100,10);
```

```

QSVM_ALT_x = zeros(100,10);
QSVM_AST_pd = zeros(100,10);
QSVM_AST_x = zeros(100,10);
QSVM_ASP_pd = zeros(100,10);
QSVM_ASP_x = zeros(100,10);
QSVM_Glucose_pd = zeros(100,10);
QSVM_Glucose_x = zeros(100,10);
for i = 1:10
    [mod_4, train_acc_4] = quadraticsvm2(training); % Training the model using training set
    [Age_pd, Age_x] = partialDependence(mod_4.ClassificationSVM, 'Age', ...
        mod_4.ClassificationSVM.ClassNames(2));
    QSVM_Age_pd(:,i) = Age_pd';
    QSVM_Age_x(:,i) = Age_x';
    [BMI_pd, BMI_x] = partialDependence(mod_4.ClassificationSVM, 'BMI', ...
        mod_4.ClassificationSVM.ClassNames(2));
    QSVM_BMI_pd(:,i) = BMI_pd';
    QSVM_BMI_x(:,i) = BMI_x';
    [HDL_pd, HDL_x] = partialDependence(mod_4.ClassificationSVM, 'HDL', ...
        mod_4.ClassificationSVM.ClassNames(2));
    QSVM_HDL_pd(:,i) = HDL_pd';
    QSVM_HDL_x(:,i) = HDL_x';
    [ALT_pd, ALT_x] = partialDependence(mod_4.ClassificationSVM, 'ALT', ...
        mod_4.ClassificationSVM.ClassNames(2));
    QSVM_ALT_pd(:,i) = ALT_pd';
    QSVM_ALT_x(:,i) = ALT_x';
    [AST_pd, AST_x] = partialDependence(mod_4.ClassificationSVM, 'AST', ...
        mod_4.ClassificationSVM.ClassNames(2));
    QSVM_AST_pd(:,i) = AST_pd';
    QSVM_AST_x(:,i) = AST_x';
    [ASP_pd, ASP_x] = partialDependence(mod_4.ClassificationSVM, 'ASP', ...
        mod_4.ClassificationSVM.ClassNames(2));
    QSVM_ASP_pd(:,i) = ASP_pd';
    QSVM_ASP_x(:,i) = ASP_x';
    [Glucose_pd, Glucose_x] =
partialDependence(mod_4.ClassificationSVM, 'Plasma_glucose_1', ...
        mod_4.ClassificationSVM.ClassNames(2));
    QSVM_Glucose_pd(:,i) = Glucose_pd';
    QSVM_Glucose_x(:,i) = Glucose_x';
end

% Average the pd and x values and plot using nexttile
QSVM_Age_pd_avg = mean(QSVM_Age_pd, 2);
QSVM_BMI_pd_avg = mean(QSVM_BMI_pd, 2);
QSVM_HDL_pd_avg = mean(QSVM_HDL_pd, 2);
QSVM_ALT_pd_avg = mean(QSVM_ALT_pd, 2);
QSVM_AST_pd_avg = mean(QSVM_AST_pd, 2);

```



```

QSVM_ASP_pd_avg = mean(QSVM_ASP_pd, 2);
QSVM_Glucose_pd_avg = mean(QSVM_Glucose_pd, 2);

QSVM_Age_x_avg = mean(QSVM_Age_x, 2);
QSVM_BMI_x_avg = mean(QSVM_BMI_x, 2);
QSVM_HDL_x_avg = mean(QSVM_HDL_x, 2);
QSVM_ALT_x_avg = mean(QSVM_ALT_x, 2);
QSVM_AST_x_avg = mean(QSVM_AST_x, 2);
QSVM_ASP_x_avg = mean(QSVM_ASP_x, 2);
QSVM_Glucose_x_avg = mean(QSVM_Glucose_x, 2);

%% Writing the excel file to save and use the data for plotting later

QSVM_Male_data = table(Age_x, QSVM_Age_pd_avg, BMI_x, QSVM_BMI_pd_avg , ...
    HDL_x, QSVM_HDL_pd_avg , ALT_x, QSVM_ALT_pd_avg , AST_x,
    QSVM_AST_pd_avg ,...
    ASP_x, QSVM_ASP_pd_avg , Glucose_x, QSVM_Glucose_pd_avg);
writetable(QSVM_Male_data,'QSVM_Male_code2_plotting_data.xlsx');

%% plotting

figure
t = tiledlayout(3,3,'TileSpacing','compact');
title(t,'Quadratic SVM-Disease - Averaged Partial Dependency Plot')
nexttile; plot(Age_x,QSVM_Age_pd_avg); title("");
xlabel("Age"), ylabel("Scores"), xticks(20:20:80)

nexttile; plot(BMI_x,QSVM_BMI_pd_avg); title("");
xlabel("BMI"), ylabel("Scores"), xticks(20:10:80)
idx_BMI_1 = interp1(BMI_x,1:length(BMI_x),18,'nearest'); % Approx BMI = 18
hold on; plot(BMI_x(idx_BMI_1),QSVM_BMI_pd_avg(idx_BMI_1),'*r'); hold off;
idx_BMI_2 = interp1(BMI_x,1:length(BMI_x),25,'nearest'); % Approx BMI = 25
hold on; plot(BMI_x(idx_BMI_2),QSVM_BMI_pd_avg(idx_BMI_2),'*r'); hold off;

nexttile; plot(HDL_x,QSVM_HDL_pd_avg); title("");
xlabel("HDL"), ylabel("Scores"), xticks(0:1:4)
idx_HDL = interp1(HDL_x,1:length(HDL_x),1,'nearest');
hold on; plot(HDL_x(idx_HDL),QSVM_HDL_pd_avg(idx_HDL),'*r'); hold off;

nexttile; plot(ALT_x,QSVM_ALT_pd_avg); title("");
xlabel("ALT"), ylabel("Scores"), xticks(0:100:400)
idx_ALT_1 = interp1(ALT_x,1:length(ALT_x),10,'nearest'); % Approx ALT = 10
hold on; plot(ALT_x(idx_ALT_1),QSVM_ALT_pd_avg(idx_ALT_1),'*r'); hold off;

```

```

idx_ALT_2 = interp1(ALT_x,1:length(ALT_x),55,'nearest'); % Approx ALT = 55
hold on; plot(ALT_x(idx_ALT_2),QSVM_ALT_pd_avg(idx_ALT_2),'*r'); hold off;

nexttile; plot(AST_x,QSVM_AST_pd_avg); title("");
xlabel("AST"), ylabel("Scores"), xticks(0:100:400)
idx_AST_1 = interp1(AST_x,1:length(AST_x),10,'nearest');
hold on; plot(AST_x(idx_AST_1),QSVM_AST_pd_avg(idx_AST_1),'*r'); hold off;
idx_AST_2 = interp1(AST_x,1:length(AST_x),40,'nearest');
hold on; plot(AST_x(idx_AST_2),QSVM_AST_pd_avg(idx_AST_2),'*r'); hold off;

nexttile; plot(ASP_x,QSVM_ASP_pd_avg); title("");
xlabel("ASP"), ylabel("Scores"), xticks(0:100:400)
idx_ASP_1 = interp1(ASP_x,1:length(ASP_x),45,'nearest');
hold on; plot(ASP_x(idx_ASP_1),QSVM_ASP_pd_avg(idx_ASP_1),'*r'); hold off;
idx_ASP_2 = interp1(ASP_x,1:length(ASP_x),115,'nearest');
hold on; plot(ASP_x(idx_ASP_2),QSVM_ASP_pd_avg(idx_ASP_2),'*r'); hold off;

nexttile; plot(Glucose_x,QSVM_Glucose_pd_avg); title("");
xlabel("Glucose"), ylabel("Scores"), xticks(0:100:600)
idx_Glucose = interp1(Glucose_x,1:length(Glucose_x),120,'nearest');
hold on; plot(Glucose_x(idx_Glucose),QSVM_Glucose_pd_avg(idx_Glucose),'*r'); hold off;

%% Model 5: Gaussian Scale 1 svm
GSVM1_Age_pd = zeros(100,10);
GSVM1_Age_x = zeros(100,10);
GSVM1_BMI_pd = zeros(100,10);
GSVM1_BMI_x = zeros(100,10);
GSVM1_HDL_pd = zeros(100,10);
GSVM1_HDL_x = zeros(100,10);
GSVM1_ALT_pd = zeros(100,10);
GSVM1_ALT_x = zeros(100,10);
GSVM1_AST_pd = zeros(100,10);
GSVM1_AST_x = zeros(100,10);
GSVM1_ASP_pd = zeros(100,10);
GSVM1_ASP_x = zeros(100,10);
GSVM1_Glucose_pd = zeros(100,10);
GSVM1_Glucose_x = zeros(100,10);
for i = 1:10
    [mod_5, train_acc_5] = finegaussiansvm2(training); % Training the model using training set
    [Age_pd,Age_x] = partialDependence(mod_5.ClassificationSVM,'Age',...
        mod_5.ClassificationSVM.ClassNames(2));
    GSVM1_Age_pd(:,i) = Age_pd';
    GSVM1_Age_x(:,i) = Age_x';
    [BMI_pd,BMI_x] = partialDependence(mod_5.ClassificationSVM,'BMI',...
        mod_5.ClassificationSVM.ClassNames(2));
    GSVM1_BMI_pd(:,i) = BMI_pd';

```

```

GSVM1_BMI_x(:,i) = BMI_x';
[HDL_pd,HDL_x] = partialDependence(mod_5.ClassificationSVM,'HDL',...
    mod_5.ClassificationSVM.ClassNames(2));
GSVM1_HDL_pd(:,i) = HDL_pd';
GSVM1_HDL_x(:,i) = HDL_x';
[ALT_pd,ALT_x] = partialDependence(mod_5.ClassificationSVM,'ALT',...
    mod_5.ClassificationSVM.ClassNames(2));
GSVM1_ALT_pd(:,i) = ALT_pd';
GSVM1_ALT_x(:,i) = ALT_x';
[AST_pd,AST_x] = partialDependence(mod_5.ClassificationSVM,'AST',...
    mod_5.ClassificationSVM.ClassNames(2));
GSVM1_AST_pd(:,i) = AST_pd';
GSVM1_AST_x(:,i) = AST_x';
[ASP_pd,ASP_x] = partialDependence(mod_5.ClassificationSVM,'ASP',...
    mod_5.ClassificationSVM.ClassNames(2));
GSVM1_ASP_pd(:,i) = ASP_pd';
GSVM1_ASP_x(:,i) = ASP_x';
[Glucose_pd,Glucose_x] =
partialDependence(mod_5.ClassificationSVM,'Plasma_glucose_1',...
    mod_5.ClassificationSVM.ClassNames(2));
GSVM1_Glucose_pd(:,i) = Glucose_pd';
GSVM1_Glucose_x(:,i) = Glucose_x';
end

```

```

% Average the pd and x values and plot using nexttile
GSVM1_Age_pd_avg = mean(GSVM1_Age_pd, 2);
GSVM1_BMI_pd_avg = mean(GSVM1_BMI_pd, 2);
GSVM1_HDL_pd_avg = mean(GSVM1_HDL_pd, 2);
GSVM1_ALT_pd_avg = mean(GSVM1_ALT_pd, 2);
GSVM1_AST_pd_avg = mean(GSVM1_AST_pd, 2);
GSVM1_ASP_pd_avg = mean(GSVM1_ASP_pd, 2);
GSVM1_Glucose_pd_avg = mean(GSVM1_Glucose_pd, 2);

```

```

GSVM1_Age_x_avg = mean(GSVM1_Age_x, 2);
GSVM1_BMI_x_avg = mean(GSVM1_BMI_x, 2);
GSVM1_HDL_x_avg = mean(GSVM1_HDL_x, 2);
GSVM1_ALT_x_avg = mean(GSVM1_ALT_x, 2);
GSVM1_AST_x_avg = mean(GSVM1_AST_x, 2);
GSVM1_ASP_x_avg = mean(GSVM1_ASP_x, 2);
GSVM1_Glucose_x_avg = mean(GSVM1_Glucose_x, 2);

```

```

%% Writing the excel file to save and use the data for plotting later

```

```

GSVM1_Male_data = table(Age_x, GSVM1_Age_pd_avg, BMI_x, GSVM1_BMI_pd_avg , ...

```

```

HDL_x, GSVM1_HDL_pd_avg , ALT_x, GSVM1_ALT_pd_avg , AST_x,
GSVM1_AST_pd_avg ,...
ASP_x, GSVM1_ASP_pd_avg , Glucose_x, GSVM1_Glucose_pd_avg);
writetable(GSVM1_Male_data,'GSVM1_Male_code2_plotting_data.xlsx');

%% Plotting
figure
t = tiledlayout(3,3,'TileSpacing','compact');
title(t,'Gaussian 1 SVM-Disease - Averaged Partial Dependency Plot')
nexttile; plot(Age_x,GSVM1_Age_pd_avg); title("");
xlabel("Age"), ylabel("Scores"), axis([-inf inf -0.3 1]), xticks(20:20:80)

nexttile; plot(BMI_x,GSVM1_BMI_pd_avg); title("");
xlabel("BMI"), ylabel("Scores"), axis([-inf inf -0.3 1]), xticks(20:10:50)
idx_BMI_1 = interp1(BMI_x,1:length(BMI_x),18,'nearest'); % Approx BMI = 18
hold on; plot(BMI_x(idx_BMI_1),GSVM1_BMI_pd_avg(idx_BMI_1),'*r'); hold off;
idx_BMI_2 = interp1(BMI_x,1:length(BMI_x),25,'nearest'); % Approx BMI = 25
hold on; plot(BMI_x(idx_BMI_2),GSVM1_BMI_pd_avg(idx_BMI_2),'*r'); hold off;

nexttile; plot(HDL_x,GSVM1_HDL_pd_avg); title("");
xlabel("HDL"), ylabel("Scores"), axis([-inf inf -0.3 1]), xticks(0:1:4)
idx_HDL = interp1(HDL_x,1:length(HDL_x),1,'nearest');
hold on; plot(HDL_x(idx_HDL),GSVM1_HDL_pd_avg(idx_HDL),'*r'); hold off;

nexttile; plot(ALT_x,GSVM1_ALT_pd_avg); title("");
xlabel("ALT"), ylabel("Scores"), axis([-inf inf -0.3 1]), xticks(0:50:300)
idx_ALT_1 = interp1(ALT_x,1:length(ALT_x),10,'nearest'); % Approx ALT = 10
hold on; plot(ALT_x(idx_ALT_1),GSVM1_ALT_pd_avg(idx_ALT_1),'*r'); hold off;
idx_ALT_2 = interp1(ALT_x,1:length(ALT_x),55,'nearest'); % Approx ALT = 55
hold on; plot(ALT_x(idx_ALT_2),GSVM1_ALT_pd_avg(idx_ALT_2),'*r'); hold off;

nexttile; plot(AST_x,GSVM1_AST_pd_avg); title("");
xlabel("AST"), ylabel("Scores"), axis([-inf inf -0.3 1]), xticks(0:50:300)
idx_AST_1 = interp1(AST_x,1:length(AST_x),10,'nearest');
hold on; plot(AST_x(idx_AST_1),GSVM1_AST_pd_avg(idx_AST_1),'*r'); hold off;
idx_AST_2 = interp1(AST_x,1:length(AST_x),40,'nearest');
hold on; plot(AST_x(idx_AST_2),GSVM1_AST_pd_avg(idx_AST_2),'*r'); hold off;

nexttile; plot(ASP_x,GSVM1_ASP_pd_avg); title("");
xlabel("ASP"), ylabel("Scores"), axis([-inf inf -0.3 1]), xticks(0:100:400)
idx_ASP_1 = interp1(ASP_x,1:length(ASP_x),45,'nearest');
hold on; plot(ASP_x(idx_ASP_1),GSVM1_ASP_pd_avg(idx_ASP_1),'*r'); hold off;
idx_ASP_2 = interp1(ASP_x,1:length(ASP_x),115,'nearest');

```

```

hold on; plot(ASP_x(idx_ASP_2),GSVM1_ASP_pd_avg(idx_ASP_2),'*r'); hold off;

nexttile; plot(Glucose_x,GSVM1_Glucose_pd_avg); title("");
xlabel("Glucose"), ylabel("Scores"), axis([-inf inf -0.3 1]), xticks(0:200:600)
idx_Glucose = interp1(Glucose_x,1:length(Glucose_x),120,'nearest');
hold on; plot(Glucose_x(idx_Glucose),GSVM1_Glucose_pd_avg(idx_Glucose),'*r'); hold off;

%% Model 6: Gaussian Scale 2 svm
GSVM2_Age_pd = zeros(100,10);
GSVM2_Age_x = zeros(100,10);
GSVM2_BMI_pd = zeros(100,10);
GSVM2_BMI_x = zeros(100,10);
GSVM2_HDL_pd = zeros(100,10);
GSVM2_HDL_x = zeros(100,10);
GSVM2_ALT_pd = zeros(100,10);
GSVM2_ALT_x = zeros(100,10);
GSVM2_AST_pd = zeros(100,10);
GSVM2_AST_x = zeros(100,10);
GSVM2_ASP_pd = zeros(100,10);
GSVM2_ASP_x = zeros(100,10);
GSVM2_Glucose_pd = zeros(100,10);
GSVM2_Glucose_x = zeros(100,10);
for i = 1:10
    [mod_6, train_acc_6] = mediumgaussiansvm2(training); % Training the model using training
    set
    [Age_pd,Age_x] = partialDependence(mod_6.ClassificationSVM,'Age',...
        mod_6.ClassificationSVM.ClassNames(2));
    GSVM2_Age_pd(:,i) = Age_pd';
    GSVM2_Age_x(:,i) = Age_x';
    [BMI_pd,BMI_x] = partialDependence(mod_6.ClassificationSVM,'BMI',...
        mod_6.ClassificationSVM.ClassNames(2));
    GSVM2_BMI_pd(:,i) = BMI_pd';
    GSVM2_BMI_x(:,i) = BMI_x';
    [HDL_pd,HDL_x] = partialDependence(mod_6.ClassificationSVM,'HDL',...
        mod_6.ClassificationSVM.ClassNames(2));
    GSVM2_HDL_pd(:,i) = HDL_pd';
    GSVM2_HDL_x(:,i) = HDL_x';
    [ALT_pd,ALT_x] = partialDependence(mod_6.ClassificationSVM,'ALT',...
        mod_6.ClassificationSVM.ClassNames(2));
    GSVM2_ALT_pd(:,i) = ALT_pd';
    GSVM2_ALT_x(:,i) = ALT_x';
    [AST_pd,AST_x] = partialDependence(mod_6.ClassificationSVM,'AST',...
        mod_6.ClassificationSVM.ClassNames(2));
    GSVM2_AST_pd(:,i) = AST_pd';
    GSVM2_AST_x(:,i) = AST_x';

```

```

[ASP_pd,ASP_x] = partialDependence(mod_6.ClassificationSVM,'ASP',...
    mod_6.ClassificationSVM.ClassNames(2));
GSVM2_ASP_pd(:,i) = ASP_pd';
GSVM2_ASP_x(:,i) = ASP_x';
[Glucose_pd,Glucose_x] =
partialDependence(mod_6.ClassificationSVM,'Plasma_glucose_1',...
    mod_6.ClassificationSVM.ClassNames(2));
GSVM2_Glucose_pd(:,i) = Glucose_pd';
GSVM2_Glucose_x(:,i) = Glucose_x';
end

% Average the pd and x values and plot using nexttile
GSVM2_Age_pd_avg = mean(GSVM2_Age_pd, 2);
GSVM2_BMI_pd_avg = mean(GSVM2_BMI_pd, 2);
GSVM2_HDL_pd_avg = mean(GSVM2_HDL_pd, 2);
GSVM2_ALT_pd_avg = mean(GSVM2_ALT_pd, 2);
GSVM2_AST_pd_avg = mean(GSVM2_AST_pd, 2);
GSVM2_ASP_pd_avg = mean(GSVM2_ASP_pd, 2);
GSVM2_Glucose_pd_avg = mean(GSVM2_Glucose_pd, 2);

GSVM2_Age_x_avg = mean(GSVM2_Age_x, 2);
GSVM2_BMI_x_avg = mean(GSVM2_BMI_x, 2);
GSVM2_HDL_x_avg = mean(GSVM2_HDL_x, 2);
GSVM2_ALT_x_avg = mean(GSVM2_ALT_x, 2);
GSVM2_AST_x_avg = mean(GSVM2_AST_x, 2);
GSVM2_ASP_x_avg = mean(GSVM2_ASP_x, 2);
GSVM2_Glucose_x_avg = mean(GSVM2_Glucose_x, 2);

%% Writing the excel file to save and use the data for plotting later

GSVM2_Male_data = table(Age_x, GSVM2_Age_pd_avg, BMI_x, GSVM2_BMI_pd_avg , ...
    HDL_x, GSVM2_HDL_pd_avg , ALT_x, GSVM2_ALT_pd_avg , AST_x,
    GSVM2_AST_pd_avg ,...
    ASP_x, GSVM2_ASP_pd_avg , Glucose_x, GSVM2_Glucose_pd_avg);
writetable(GSVM2_Male_data,'GSVM2_Male_code2_plotting_data.xlsx');

%% Plotting
figure
t = tiledlayout(3,3,'TileSpacing','compact');
title(t,'Gaussian 2 SVM-Disease - Averaged Partial Dependency Plot')
nexttile; plot(Age_x,GSVM2_Age_pd_avg); title("");
xlabel("Age"), ylabel("Scores"), axis([-inf inf -0.7 1.5]), xticks(20:20:80)

nexttile; plot(BMI_x,GSVM2_BMI_pd_avg); title("");

```

```

xlabel("BMI"), ylabel("Scores"), axis([-inf inf -0.7 1.5]), xticks(20:10:50)
idx_BMI_1 = interp1(BMI_x,1:length(BMI_x),18,'nearest'); % Approx BMI = 18
hold on; plot(BMI_x(idx_BMI_1),GSVM2_BMI_pd_avg(idx_BMI_1),'*r'); hold off;
idx_BMI_2 = interp1(BMI_x,1:length(BMI_x),25,'nearest'); % Approx BMI = 25
hold on; plot(BMI_x(idx_BMI_2),GSVM2_BMI_pd_avg(idx_BMI_2),'*r'); hold off;

nexttile; plot(HDL_x,GSVM2_HDL_pd_avg); title("");
xlabel("HDL"), ylabel("Scores"), axis([-inf inf -0.7 1.5]), xticks(0:1:4)
idx_HDL = interp1(HDL_x,1:length(HDL_x),1,'nearest');
hold on; plot(HDL_x(idx_HDL),GSVM2_HDL_pd_avg(idx_HDL),'*r'); hold off;

nexttile; plot(ALT_x,GSVM2_ALT_pd_avg); title("");
xlabel("ALT"), ylabel("Scores"), axis([-inf inf -0.7 1.5]), xticks(0:100:400)
idx_ALT_1 = interp1(ALT_x,1:length(ALT_x),10,'nearest'); % Approx ALT = 10
hold on; plot(ALT_x(idx_ALT_1),GSVM2_ALT_pd_avg(idx_ALT_1),'*r'); hold off;
idx_ALT_2 = interp1(ALT_x,1:length(ALT_x),55,'nearest'); % Approx ALT = 55
hold on; plot(ALT_x(idx_ALT_2),GSVM2_ALT_pd_avg(idx_ALT_2),'*r'); hold off;

nexttile; plot(AST_x,GSVM2_AST_pd_avg); title("");
xlabel("AST"), ylabel("Scores"), axis([-inf inf -0.7 1.5]), xticks(0:100:400)
idx_AST_1 = interp1(AST_x,1:length(AST_x),10,'nearest');
hold on; plot(AST_x(idx_AST_1),GSVM2_AST_pd_avg(idx_AST_1),'*r'); hold off;
idx_AST_2 = interp1(AST_x,1:length(AST_x),40,'nearest');
hold on; plot(AST_x(idx_AST_2),GSVM2_AST_pd_avg(idx_AST_2),'*r'); hold off;

nexttile; plot(ASP_x,GSVM2_ASP_pd_avg); title("");
xlabel("ASP"), ylabel("Scores"), axis([-inf inf -0.7 1.5]), xticks(0:100:400)
idx_ASP_1 = interp1(ASP_x,1:length(ASP_x),45,'nearest');
hold on; plot(ASP_x(idx_ASP_1),GSVM2_ASP_pd_avg(idx_ASP_1),'*r'); hold off;
idx_ASP_2 = interp1(ASP_x,1:length(ASP_x),115,'nearest');
hold on; plot(ASP_x(idx_ASP_2),GSVM2_ASP_pd_avg(idx_ASP_2),'*r'); hold off;

nexttile; plot(Glucose_x,GSVM2_Glucose_pd_avg); title("");
xlabel("Glucose"), ylabel("Scores"), axis([-inf inf -0.7 1.5]), xticks(0:100:500)
idx_Glucose = interp1(Glucose_x,1:length(Glucose_x),120,'nearest');
hold on; plot(Glucose_x(idx_Glucose),GSVM2_Glucose_pd_avg(idx_Glucose),'*r'); hold off;

```

## 2. MATLAB CODE TO EXTRACT STATISTICS FROM PARTIAL DEPENDENCY DATA

```

%% % % % % % % %
% Created on: 08/09/2021
% Input: Partial dependency data

```

```

% Output: Computed statistics
% Author: Ridhi Deo
% File name: obj2_matlab_4 (R2020b [17]) )
% Description: This code extracts the means and standard deviations from partial dependency
data
%%%%%%%%

%% Quadratic
QSV_M_Age_pd = QSV_M_Female_data.QSV_M_Age_pd_avg;
QSV_M_BMI_pd = QSV_M_Female_data.QSV_M_BMI_pd_avg;
QSV_M_HDL_pd = QSV_M_Female_data.QSV_M_HDL_pd_avg;
QSV_M_ALT_pd = QSV_M_Female_data.QSV_M_ALT_pd_avg;
QSV_M_AST_pd = QSV_M_Female_data.QSV_M_AST_pd_avg;
QSV_M_ASP_pd = QSV_M_Female_data.QSV_M_ASP_pd_avg;
QSV_M_Glucose_pd = QSV_M_Female_data.QSV_M_Glucose_pd_avg;

QSV_M_Age_pd_avg = mean(QSV_M_Age_pd, 2);
QSV_M_BMI_pd_avg = mean(QSV_M_BMI_pd, 2);
QSV_M_HDL_pd_avg = mean(QSV_M_HDL_pd, 2);
QSV_M_ALT_pd_avg = mean(QSV_M_ALT_pd, 2);
QSV_M_AST_pd_avg = mean(QSV_M_AST_pd, 2);
QSV_M_ASP_pd_avg = mean(QSV_M_ASP_pd, 2);
QSV_M_Glucose_pd_avg = mean(QSV_M_Glucose_pd, 2);

QSV_M_pds_only = QSV_M_Female_data(:,[2,4,6,8,10,12,14]);

[QSV_M_age_amb, ~] = find(QSV_M_pds_only.QSV_M_Age_pd_avg > -0.15 & ...
    QSV_M_pds_only.QSV_M_Age_pd_avg < 0.15);
QSV_M_Age_amb_percentage =
(size(QSV_M_age_amb,1)/size(QSV_M_pds_only.QSV_M_Age_pd_avg,1))*100;
[QSV_M_bmi_amb, ~] = find(QSV_M_pds_only.QSV_M_BMI_pd_avg > -0.15 & ...
    QSV_M_pds_only.QSV_M_BMI_pd_avg < 0.15);
QSV_M_BMI_amb_percentage =
(size(QSV_M_bmi_amb,1)/size(QSV_M_pds_only.QSV_M_BMI_pd_avg,1))*100;

[QSV_M_HDL_amb, ~] = find(QSV_M_pds_only.QSV_M_HDL_pd_avg > -0.15 & ...
    QSV_M_pds_only.QSV_M_HDL_pd_avg < 0.15);
QSV_M_HDL_amb_percentage =
(size(QSV_M_HDL_amb,1)/size(QSV_M_pds_only.QSV_M_HDL_pd_avg,1))*100;

[QSV_M_ALT_amb, ~] = find(QSV_M_pds_only.QSV_M_ALT_pd_avg > -0.15 & ...
    QSV_M_pds_only.QSV_M_ALT_pd_avg < 0.15);
QSV_M_ALT_amb_percentage =
(size(QSV_M_ALT_amb,1)/size(QSV_M_pds_only.QSV_M_ALT_pd_avg,1))*100;

```



```

[QSV_M_AST_amb, ~] = find(QSV_M_pds_only.QSV_M_AST_pd_avg > -0.15 & ...
    QSV_M_pds_only.QSV_M_AST_pd_avg < 0.15);
QSV_M_AST_amb_percentage =
(size(QSV_M_AST_amb,1)/size(QSV_M_pds_only.QSV_M_AST_pd_avg,1))*100;

[QSV_M_ASP_amb, ~] = find(QSV_M_pds_only.QSV_M_ASP_pd_avg > -0.15 & ...
    QSV_M_pds_only.QSV_M_ASP_pd_avg < 0.15);
QSV_M_ASP_amb_percentage =
(size(QSV_M_ASP_amb,1)/size(QSV_M_pds_only.QSV_M_ASP_pd_avg,1))*100;

[QSV_M_Glucose_amb, ~] = find(QSV_M_pds_only.QSV_M_Glucose_pd_avg > -0.15 & ...
    QSV_M_pds_only.QSV_M_Glucose_pd_avg < 0.15);
QSV_M_Glucose_amb_percentage =
(size(QSV_M_Glucose_amb,1)/size(QSV_M_pds_only.QSV_M_Glucose_pd_avg,1))*100;

Ambiguity_QSV_M = [QSV_M_Age_amb_percentage, QSV_M_BMI_amb_percentage,
    QSV_M_HDL_amb_percentage,...
    QSV_M_ALT_amb_percentage, QSV_M_AST_amb_percentage,
    QSV_M_ASP_amb_percentage,...
    QSV_M_Glucose_amb_percentage];

%% Quadratic SVM Score extraction for pd
figure;histfit(QSV_M_Age_pd_avg),title('QSV_M-Age');
[QSV_M_Age_pd_mean,QSV_M_Age_pd_SD,QSV_M_Age_pd_var,QSV_M_Age_pd_min,QSV_M
_Age_pd_max,QSV_M_Age_pd_range] ...
    = grpstats(QSV_M_Age_pd_avg,[],{'mean','std','var','min','max','range'});
QSV_M_Age_pd_Quantiles = quantile(QSV_M_Age_pd_avg,[0.25,0.5,0.75,1]);

figure;histfit(QSV_M_BMI_pd_avg),title('QSV_M-BMI');
[QSV_M_BMI_pd_mean,QSV_M_BMI_pd_SD,QSV_M_BMI_pd_var,QSV_M_BMI_pd_min,QSV
M_BMI_pd_max,QSV_M_BMI_pd_range] ...
    = grpstats(QSV_M_BMI_pd_avg,[],{'mean','std','var','min','max','range'});
QSV_M_BMI_pd_Quantiles = quantile(QSV_M_BMI_pd_avg,[0.25,0.5,0.75,1]);

figure;histfit(QSV_M_HDL_pd_avg),title('QSV_M-HDL');
[QSV_M_HDL_pd_mean,QSV_M_HDL_pd_SD,QSV_M_HDL_pd_var,QSV_M_HDL_pd_min,QSV
VM_HDL_pd_max,QSV_M_HDL_pd_range] ...
    = grpstats(QSV_M_HDL_pd_avg,[],{'mean','std','var','min','max','range'});
QSV_M_HDL_pd_Quantiles = quantile(QSV_M_HDL_pd_avg,[0.25,0.5,0.75,1]);

figure;histfit(QSV_M_ALT_pd_avg),title('QSV_M-ALT');
[QSV_M_ALT_pd_mean,QSV_M_ALT_pd_SD,QSV_M_ALT_pd_var,QSV_M_ALT_pd_min,QSV
M_ALT_pd_max,QSV_M_ALT_pd_range] ...
    = grpstats(QSV_M_ALT_pd_avg,[],{'mean','std','var','min','max','range'});

```

```
QSVM_ALT_pd_Quantiles = quantile(QSVM_ALT_pd_avg,[0.25,0.5,0.75,1]);
```

```
figure;histfit(QSVM_AST_pd_avg),title('QSVM-AST');
[QSVM_AST_pd_mean,QSVM_AST_pd_SD,QSVM_AST_pd_var,QSVM_AST_pd_min,QSV
M_AST_pd_max,QSVM_AST_pd_range] ...
    = grpstats(QSVM_AST_pd_avg,[],{'mean','std','var','min','max','range'});
QSVM_AST_pd_Quantiles = quantile(QSVM_AST_pd_avg,[0.25,0.5,0.75,1]);
```

```
figure;histfit(QSVM_ASP_pd_avg),title('QSVM-ASP');
[QSVM_ASP_pd_mean,QSVM_ASP_pd_SD,QSVM_ASP_pd_var,QSVM_ASP_pd_min,QSV
M_ASP_pd_max,QSVM_ASP_pd_range] ...
    = grpstats(QSVM_ASP_pd_avg,[],{'mean','std','var','min','max','range'});
QSVM_ASP_pd_Quantiles = quantile(QSVM_ASP_pd_avg,[0.25,0.5,0.75,1]);
```

```
figure;histfit(QSVM_Glucose_pd_avg),title('QSVM-Glucose');
[QSVM_Glucose_pd_mean,QSVM_Glucose_pd_SD,QSVM_Glucose_pd_var,QSVM_Glucose_
pd_min,QSVM_Glucose_pd_max,QSVM_Glucose_pd_range] ...
    = grpstats(QSVM_Glucose_pd_avg,[],{'mean','std','var','min','max','range'});
QSVM_Glucose_pd_Quantiles = quantile(QSVM_Glucose_pd_avg,[0.25,0.5,0.75,1]);
```

```
%% Display results in a table for QSVM
```

```
Param = {'Age';'BMI';'HDL';'ALT';'AST';'ASP';'Glucose'};
```

```
QSVM_Mean =
```

```
{ QSVM_Age_pd_mean;QSVM_BMI_pd_mean;QSVM_HDL_pd_mean;QSVM_ALT_pd_mea
n;...
```

```
    QSVM_AST_pd_mean;QSVM_ASP_pd_mean;QSVM_Glucose_pd_mean};
```

```
SD={ QSVM_Age_pd_SD;QSVM_BMI_pd_SD;QSVM_HDL_pd_SD;QSVM_ALT_pd_SD;...
```

```
    QSVM_AST_pd_SD;QSVM_ASP_pd_SD;QSVM_Glucose_pd_SD};
```

```
Min =
```

```
{ QSVM_Age_pd_min;QSVM_BMI_pd_min;QSVM_HDL_pd_min;QSVM_ALT_pd_min;...
```

```
    QSVM_AST_pd_min;QSVM_ASP_pd_min;QSVM_Glucose_pd_min};
```

```
Max =
```

```
{ QSVM_Age_pd_max;QSVM_BMI_pd_max;QSVM_HDL_pd_max;QSVM_ALT_pd_max;...
```

```
    QSVM_AST_pd_max;QSVM_ASP_pd_max;QSVM_Glucose_pd_max};
```

```
Range =
```

```
{ QSVM_Age_pd_range;QSVM_BMI_pd_range;QSVM_HDL_pd_range;QSVM_ALT_pd_rang
e;...
```

```
    QSVM_AST_pd_range;QSVM_ASP_pd_range;QSVM_Glucose_pd_range};
```

```
Q25 =
```

```
{ QSVM_Age_pd_Quantiles(1,1);QSVM_BMI_pd_Quantiles(1,1);QSVM_HDL_pd_Quantiles(1
,1);QSVM_ALT_pd_Quantiles(1,1);...
```

```

QSV_M_AST_pd_Quantiles(1,1);QSV_M_ASP_pd_Quantiles(1,1);QSV_M_Glucose_pd_Quantile
s(1,1));
Q50 =
{QSV_M_Age_pd_Quantiles(1,2);QSV_M_BMI_pd_Quantiles(1,2);QSV_M_HDL_pd_Quantiles(1
,2);QSV_M_ALT_pd_Quantiles(1,2);...

QSV_M_AST_pd_Quantiles(1,2);QSV_M_ASP_pd_Quantiles(1,2);QSV_M_Glucose_pd_Quantile
s(1,2));
Q75 =
{QSV_M_Age_pd_Quantiles(1,3);QSV_M_BMI_pd_Quantiles(1,3);QSV_M_HDL_pd_Quantiles(1
,3);QSV_M_ALT_pd_Quantiles(1,3);...

QSV_M_AST_pd_Quantiles(1,3);QSV_M_ASP_pd_Quantiles(1,3);QSV_M_Glucose_pd_Quantile
s(1,3));
Q100 =
{QSV_M_Age_pd_Quantiles(1,4);QSV_M_BMI_pd_Quantiles(1,4);QSV_M_HDL_pd_Quantiles(1
,4);QSV_M_ALT_pd_Quantiles(1,4);...

QSV_M_AST_pd_Quantiles(1,4);QSV_M_ASP_pd_Quantiles(1,4);QSV_M_Glucose_pd_Quantile
s(1,4));

QSV_M_pd_Stats = table(Param, QSV_M_Mean, SD, Min, Max, Range, Q25, Q50, Q75, Q100);
%writetable(QSV_M_pd_Stats,'Female_QSV_M_pd_code2_stats_pd_03_30.xlsx');

%% Gaussian 1
GSVM1_Age_pd = GSVM1_Female_data.GSVM1_Age_pd_avg;
GSVM1_BMI_pd = GSVM1_Female_data.GSVM1_BMI_pd_avg;
GSVM1_HDL_pd = GSVM1_Female_data.GSVM1_HDL_pd_avg;
GSVM1_ALT_pd = GSVM1_Female_data.GSVM1_ALT_pd_avg;
GSVM1_AST_pd = GSVM1_Female_data.GSVM1_AST_pd_avg;
GSVM1_ASP_pd = GSVM1_Female_data.GSVM1_ASP_pd_avg;
GSVM1_Glucose_pd = GSVM1_Female_data.GSVM1_Glucose_pd_avg;

GSVM1_Age_pd_avg = mean(GSVM1_Age_pd, 2);
GSVM1_BMI_pd_avg = mean(GSVM1_BMI_pd, 2);
GSVM1_HDL_pd_avg = mean(GSVM1_HDL_pd, 2);
GSVM1_ALT_pd_avg = mean(GSVM1_ALT_pd, 2);
GSVM1_AST_pd_avg = mean(GSVM1_AST_pd, 2);
GSVM1_ASP_pd_avg = mean(GSVM1_ASP_pd, 2);
GSVM1_Glucose_pd_avg = mean(GSVM1_Glucose_pd, 2);

GSVM1_pds_only = GSVM1_Female_data(:,[2,4,6,8,10,12,14]);

[GSVM1_age_amb, ~] = find(GSVM1_pds_only.GSVM1_Age_pd_avg > -0.15 & ...

```

```

GSVM1_pds_only.GSVM1_Age_pd_avg < 0.15);
GSVM1_Age_amb_percentage =
(size(GSVM1_age_amb,1)/size(GSVM1_pds_only.GSVM1_Age_pd_avg,1))*100;
[GSVM1_bmi_amb, ~] = find(GSVM1_pds_only.GSVM1_BMI_pd_avg > -0.15 & ...
    GSVM1_pds_only.GSVM1_BMI_pd_avg < 0.15);
GSVM1_BMI_amb_percentage =
(size(GSVM1_bmi_amb,1)/size(GSVM1_pds_only.GSVM1_BMI_pd_avg,1))*100;

[GSVM1_HDL_amb, ~] = find(GSVM1_pds_only.GSVM1_HDL_pd_avg > -0.15 & ...
    GSVM1_pds_only.GSVM1_HDL_pd_avg < 0.15);
GSVM1_HDL_amb_percentage =
(size(GSVM1_HDL_amb,1)/size(GSVM1_pds_only.GSVM1_HDL_pd_avg,1))*100;

[GSVM1_ALT_amb, ~] = find(GSVM1_pds_only.GSVM1_ALT_pd_avg > -0.15 & ...
    GSVM1_pds_only.GSVM1_ALT_pd_avg < 0.15);
GSVM1_ALT_amb_percentage =
(size(GSVM1_ALT_amb,1)/size(GSVM1_pds_only.GSVM1_ALT_pd_avg,1))*100;

[GSVM1_AST_amb, ~] = find(GSVM1_pds_only.GSVM1_AST_pd_avg > -0.15 & ...
    GSVM1_pds_only.GSVM1_AST_pd_avg < 0.15);
GSVM1_AST_amb_percentage =
(size(GSVM1_AST_amb,1)/size(GSVM1_pds_only.GSVM1_AST_pd_avg,1))*100;

[GSVM1_ASP_amb, ~] = find(GSVM1_pds_only.GSVM1_ASP_pd_avg > -0.15 & ...
    GSVM1_pds_only.GSVM1_ASP_pd_avg < 0.15);
GSVM1_ASP_amb_percentage =
(size(GSVM1_ASP_amb,1)/size(GSVM1_pds_only.GSVM1_ASP_pd_avg,1))*100;

[GSVM1_Glucose_amb, ~] = find(GSVM1_pds_only.GSVM1_Glucose_pd_avg > -0.15 & ...
    GSVM1_pds_only.GSVM1_Glucose_pd_avg < 0.15);
GSVM1_Glucose_amb_percentage =
(size(GSVM1_Glucose_amb,1)/size(GSVM1_pds_only.GSVM1_Glucose_pd_avg,1))*100;

Ambiguity_GSVM1 = [GSVM1_Age_amb_percentage, GSVM1_BMI_amb_percentage,
GSVM1_HDL_amb_percentage,...
    GSVM1_ALT_amb_percentage, GSVM1_AST_amb_percentage,
GSVM1_ASP_amb_percentage,...
    GSVM1_Glucose_amb_percentage];

%% Gaussian Scale 1 Score extraction
figure;histfit(GSVM1_Age_pd_avg),title('GSVM1-Age');
[GSVM1_Age_pd_mean,GSVM1_Age_pd_SD,GSVM1_Age_pd_var,GSVM1_Age_pd_min,G
SVM1_Age_pd_max,GSVM1_Age_pd_range] ...
    = grpstats(GSVM1_Age_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM1_Age_pd_Quantiles = quantile(GSVM1_Age_pd_avg,[0.25,0.5,0.75,1]);

```

```
figure;histfit(GSVM1_BMI_pd_avg),title('GSVM1-BMI');
[GSVM1_BMI_pd_mean,GSVM1_BMI_pd_SD,GSVM1_BMI_pd_var,GSVM1_BMI_pd_min,
GSVM1_BMI_pd_max,GSVM1_BMI_pd_range] ...
= grpstats(GSVM1_BMI_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM1_BMI_pd_Quantiles = quantile(GSVM1_BMI_pd_avg,[0.25,0.5,0.75,1]);
```

```
figure;histfit(GSVM1_HDL_pd_avg),title('GSVM1-HDL');
[GSVM1_HDL_pd_mean,GSVM1_HDL_pd_SD,GSVM1_HDL_pd_var,GSVM1_HDL_pd_min,
GSVM1_HDL_pd_max,GSVM1_HDL_pd_range] ...
= grpstats(GSVM1_HDL_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM1_HDL_pd_Quantiles = quantile(GSVM1_HDL_pd_avg,[0.25,0.5,0.75,1]);
```

```
figure;histfit(GSVM1_ALT_pd_avg),title('GSVM1-ALT');
[GSVM1_ALT_pd_mean,GSVM1_ALT_pd_SD,GSVM1_ALT_pd_var,GSVM1_ALT_pd_min,
GSVM1_ALT_pd_max,GSVM1_ALT_pd_range] ...
= grpstats(GSVM1_ALT_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM1_ALT_pd_Quantiles = quantile(GSVM1_ALT_pd_avg,[0.25,0.5,0.75,1]);
```

```
figure;histfit(GSVM1_AST_pd_avg),title('GSVM1-AST');
[GSVM1_AST_pd_mean,GSVM1_AST_pd_SD,GSVM1_AST_pd_var,GSVM1_AST_pd_min,
GSVM1_AST_pd_max,GSVM1_AST_pd_range] ...
= grpstats(GSVM1_AST_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM1_AST_pd_Quantiles = quantile(GSVM1_AST_pd_avg,[0.25,0.5,0.75,1]);
```

```
figure;histfit(GSVM1_ASP_pd_avg),title('GSVM1-ASP');
[GSVM1_ASP_pd_mean,GSVM1_ASP_pd_SD,GSVM1_ASP_pd_var,GSVM1_ASP_pd_min,
GSVM1_ASP_pd_max,GSVM1_ASP_pd_range] ...
= grpstats(GSVM1_ASP_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM1_ASP_pd_Quantiles = quantile(GSVM1_ASP_pd_avg,[0.25,0.5,0.75,1]);
```

```
figure;histfit(GSVM1_Glucose_pd_avg),title('GSVM1-Glucose');
[GSVM1_Glucose_pd_mean,GSVM1_Glucose_pd_SD,GSVM1_Glucose_pd_var,GSVM1_Glucose_pd_min,
GSVM1_Glucose_pd_max,GSVM1_Glucose_pd_range] ...
= grpstats(GSVM1_Glucose_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM1_Glucose_pd_Quantiles = quantile(GSVM1_Glucose_pd_avg,[0.25,0.5,0.75,1]);
```

```
%% Display results in a table for GSVM1
Param = {'Age','BMI','HDL','ALT','AST','ASP','Glucose'};
```

```

GSVM1_Mean =
{ GSVM1_Age_pd_mean;GSVM1_BMI_pd_mean;GSVM1_HDL_pd_mean;GSVM1_ALT_pd_
mean;...
  GSVM1_AST_pd_mean;GSVM1_ASP_pd_mean;GSVM1_Glucose_pd_mean};
SD
={ GSVM1_Age_pd_SD;GSVM1_BMI_pd_SD;GSVM1_HDL_pd_SD;GSVM1_ALT_pd_SD;..
.
  GSVM1_AST_pd_SD;GSVM1_ASP_pd_SD;GSVM1_Glucose_pd_SD};
Min =
{ GSVM1_Age_pd_min;GSVM1_BMI_pd_min;GSVM1_HDL_pd_min;GSVM1_ALT_pd_min;
...
  GSVM1_AST_pd_min;GSVM1_ASP_pd_min;GSVM1_Glucose_pd_min};
Max =
{ GSVM1_Age_pd_max;GSVM1_BMI_pd_max;GSVM1_HDL_pd_max;GSVM1_ALT_pd_ma
x;...
  GSVM1_AST_pd_max;GSVM1_ASP_pd_max;GSVM1_Glucose_pd_max};
Range =
{ GSVM1_Age_pd_range;GSVM1_BMI_pd_range;GSVM1_HDL_pd_range;GSVM1_ALT_pd
_range;...
  GSVM1_AST_pd_range;GSVM1_ASP_pd_range;GSVM1_Glucose_pd_range};
Q25 =
{ GSVM1_Age_pd_Quantiles(1,1);GSVM1_BMI_pd_Quantiles(1,1);GSVM1_HDL_pd_Quantil
es(1,1);GSVM1_ALT_pd_Quantiles(1,1);...

GSVM1_AST_pd_Quantiles(1,1);GSVM1_ASP_pd_Quantiles(1,1);GSVM1_Glucose_pd_Quan
tiles(1,1)};
Q50 =
{ GSVM1_Age_pd_Quantiles(1,2);GSVM1_BMI_pd_Quantiles(1,2);GSVM1_HDL_pd_Quantil
es(1,2);GSVM1_ALT_pd_Quantiles(1,2);...

GSVM1_AST_pd_Quantiles(1,2);GSVM1_ASP_pd_Quantiles(1,2);GSVM1_Glucose_pd_Quan
tiles(1,2)};
Q75 =
{ GSVM1_Age_pd_Quantiles(1,3);GSVM1_BMI_pd_Quantiles(1,3);GSVM1_HDL_pd_Quantil
es(1,3);GSVM1_ALT_pd_Quantiles(1,3);...

GSVM1_AST_pd_Quantiles(1,3);GSVM1_ASP_pd_Quantiles(1,3);GSVM1_Glucose_pd_Quan
tiles(1,3)};
Q100 =
{ GSVM1_Age_pd_Quantiles(1,4);GSVM1_BMI_pd_Quantiles(1,4);GSVM1_HDL_pd_Quantil
es(1,4);GSVM1_ALT_pd_Quantiles(1,4);...

GSVM1_AST_pd_Quantiles(1,4);GSVM1_ASP_pd_Quantiles(1,4);GSVM1_Glucose_pd_Quan
tiles(1,4)};

```

```

GSVM1_pd_Stats = table(Param, GSVM1_Mean, SD, Min, Max, Range, Q25, Q50, Q75,
Q100);
% writetable(GSVM1_pd_Stats,'Female_GSVM1_pd_code2_stats_pd_03_30.xlsx');

%% Gaussian 2
GSVM2_Age_pd = GSVM2_Female_data.GSVM2_Age_pd_avg;
GSVM2_BMI_pd = GSVM2_Female_data.GSVM2_BMI_pd_avg;
GSVM2_HDL_pd = GSVM2_Female_data.GSVM2_HDL_pd_avg;
GSVM2_ALT_pd = GSVM2_Female_data.GSVM2_ALT_pd_avg;
GSVM2_AST_pd = GSVM2_Female_data.GSVM2_AST_pd_avg;
GSVM2_ASP_pd = GSVM2_Female_data.GSVM2_ASP_pd_avg;
GSVM2_Glucose_pd = GSVM2_Female_data.GSVM2_Glucose_pd_avg;

GSVM2_Age_pd_avg = mean(GSVM2_Age_pd, 2);
GSVM2_BMI_pd_avg = mean(GSVM2_BMI_pd, 2);
GSVM2_HDL_pd_avg = mean(GSVM2_HDL_pd, 2);
GSVM2_ALT_pd_avg = mean(GSVM2_ALT_pd, 2);
GSVM2_AST_pd_avg = mean(GSVM2_AST_pd, 2);
GSVM2_ASP_pd_avg = mean(GSVM2_ASP_pd, 2);
GSVM2_Glucose_pd_avg = mean(GSVM2_Glucose_pd, 2);

GSVM2_pds_only = GSVM2_Female_data(:,[2,4,6,8,10,12,14]);

[GSVM2_age_amb, ~] = find(GSVM2_pds_only.GSVM2_Age_pd_avg > -0.15 & ...
    GSVM2_pds_only.GSVM2_Age_pd_avg < 0.15);
GSVM2_Age_amb_percentage =
(size(GSVM2_age_amb,1)/size(GSVM2_pds_only.GSVM2_Age_pd_avg,1))*100;
[GSVM2_bmi_amb, ~] = find(GSVM2_pds_only.GSVM2_BMI_pd_avg > -0.15 & ...
    GSVM2_pds_only.GSVM2_BMI_pd_avg < 0.15);
GSVM2_BMI_amb_percentage =
(size(GSVM2_bmi_amb,1)/size(GSVM2_pds_only.GSVM2_BMI_pd_avg,1))*100;

[GSVM2_HDL_amb, ~] = find(GSVM2_pds_only.GSVM2_HDL_pd_avg > -0.15 & ...
    GSVM2_pds_only.GSVM2_HDL_pd_avg < 0.15);
GSVM2_HDL_amb_percentage =
(size(GSVM2_HDL_amb,1)/size(GSVM2_pds_only.GSVM2_HDL_pd_avg,1))*100;

[GSVM2_ALT_amb, ~] = find(GSVM2_pds_only.GSVM2_ALT_pd_avg > -0.15 & ...
    GSVM2_pds_only.GSVM2_ALT_pd_avg < 0.15);
GSVM2_ALT_amb_percentage =
(size(GSVM2_ALT_amb,1)/size(GSVM2_pds_only.GSVM2_ALT_pd_avg,1))*100;

[GSVM2_AST_amb, ~] = find(GSVM2_pds_only.GSVM2_AST_pd_avg > -0.15 & ...
    GSVM2_pds_only.GSVM2_AST_pd_avg < 0.15);

```

```

GSVM2_AST_amb_percentage =
(size(GSVM2_AST_amb,1)/size(GSVM2_pds_only.GSVM2_AST_pd_avg,1))*100;

[GSVM2_ASP_amb, ~] = find(GSVM2_pds_only.GSVM2_ASP_pd_avg > -0.15 & ...
    GSVM2_pds_only.GSVM2_ASP_pd_avg < 0.15);
GSVM2_ASP_amb_percentage =
(size(GSVM2_ASP_amb,1)/size(GSVM2_pds_only.GSVM2_ASP_pd_avg,1))*100;

[GSVM2_Glucose_amb, ~] = find(GSVM2_pds_only.GSVM2_Glucose_pd_avg > -0.15 & ...
    GSVM2_pds_only.GSVM2_Glucose_pd_avg < 0.15);
GSVM2_Glucose_amb_percentage =
(size(GSVM2_Glucose_amb,1)/size(GSVM2_pds_only.GSVM2_Glucose_pd_avg,1))*100;

Ambiguity_GSVM2 = [GSVM2_Age_amb_percentage, GSVM2_BMI_amb_percentage,
    GSVM2_HDL_amb_percentage,...
    GSVM2_ALT_amb_percentage, GSVM2_AST_amb_percentage,
    GSVM2_ASP_amb_percentage,...
    GSVM2_Glucose_amb_percentage];

%% Gaussian_Scale_2 SVM Score extraction

figure;histfit(GSVM2_Age_pd_avg),title('GSVM2-Age');
[GSVM2_Age_pd_mean,GSVM2_Age_pd_SD,GSVM2_Age_pd_var,GSVM2_Age_pd_min,G
SVM2_Age_pd_max,GSVM2_Age_pd_range] ...
    = grpstats(GSVM2_Age_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM2_Age_pd_Quantiles = quantile(GSVM2_Age_pd_avg,[0.25,0.5,0.75,1]);

figure;histfit(GSVM2_BMI_pd_avg),title('GSVM2-BMI');
[GSVM2_BMI_pd_mean,GSVM2_BMI_pd_SD,GSVM2_BMI_pd_var,GSVM2_BMI_pd_min,
GSVM2_BMI_pd_max,GSVM2_BMI_pd_range] ...
    = grpstats(GSVM2_BMI_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM2_BMI_pd_Quantiles = quantile(GSVM2_BMI_pd_avg,[0.25,0.5,0.75,1]);

figure;histfit(GSVM2_HDL_pd_avg),title('GSVM2-HDL');
[GSVM2_HDL_pd_mean,GSVM2_HDL_pd_SD,GSVM2_HDL_pd_var,GSVM2_HDL_pd_mi
n,GSVM2_HDL_pd_max,GSVM2_HDL_pd_range] ...
    = grpstats(GSVM2_HDL_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM2_HDL_pd_Quantiles = quantile(GSVM2_HDL_pd_avg,[0.25,0.5,0.75,1]);

figure;histfit(GSVM2_ALT_pd_avg),title('GSVM2-ALT');
[GSVM2_ALT_pd_mean,GSVM2_ALT_pd_SD,GSVM2_ALT_pd_var,GSVM2_ALT_pd_min,
GSVM2_ALT_pd_max,GSVM2_ALT_pd_range] ...

```



```

    = grpstats(GSVM2_ALT_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM2_ALT_pd_Quantiles = quantile(GSVM2_ALT_pd_avg,[0.25,0.5,0.75,1]);

```

```

figure;histfit(GSVM2_AST_pd_avg),title('GSVM2-AST');
[GSVM2_AST_pd_mean,GSVM2_AST_pd_SD,GSVM2_AST_pd_var,GSVM2_AST_pd_min,
GSVM2_AST_pd_max,GSVM2_AST_pd_range] ...
    = grpstats(GSVM2_AST_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM2_AST_pd_Quantiles = quantile(GSVM2_AST_pd_avg,[0.25,0.5,0.75,1]);

```

```

figure;histfit(GSVM2_ASP_pd_avg),title('GSVM2-ASP');
[GSVM2_ASP_pd_mean,GSVM2_ASP_pd_SD,GSVM2_ASP_pd_var,GSVM2_ASP_pd_min,
GSVM2_ASP_pd_max,GSVM2_ASP_pd_range] ...
    = grpstats(GSVM2_ASP_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM2_ASP_pd_Quantiles = quantile(GSVM2_ASP_pd_avg,[0.25,0.5,0.75,1]);

```

```

figure;histfit(GSVM2_Glucose_pd_avg),title('GSVM2-Glucose');
[GSVM2_Glucose_pd_mean,GSVM2_Glucose_pd_SD,GSVM2_Glucose_pd_var,GSVM2_Glu
cose_pd_min,GSVM2_Glucose_pd_max,GSVM2_Glucose_pd_range] ...
    = grpstats(GSVM2_Glucose_pd_avg,[],{'mean','std','var','min','max','range'});
GSVM2_Glucose_pd_Quantiles = quantile(GSVM2_Glucose_pd_avg,[0.25,0.5,0.75,1]);

```

```

%% Display results in a table for GSVM2
Param = {'Age';'BMI';'HDL';'ALT';'AST';'ASP';'Glucose'};
GSVM2_Mean =
{GSVM2_Age_pd_mean;GSVM2_BMI_pd_mean;GSVM2_HDL_pd_mean;GSVM2_ALT_pd_
mean;...
    GSVM2_AST_pd_mean;GSVM2_ASP_pd_mean;GSVM2_Glucose_pd_mean};
SD
={GSVM2_Age_pd_SD;GSVM2_BMI_pd_SD;GSVM2_HDL_pd_SD;GSVM2_ALT_pd_SD;..
.
    GSVM2_AST_pd_SD;GSVM2_ASP_pd_SD;GSVM2_Glucose_pd_SD};
Min =
{GSVM2_Age_pd_min;GSVM2_BMI_pd_min;GSVM2_HDL_pd_min;GSVM2_ALT_pd_min;
...
    GSVM2_AST_pd_min;GSVM2_ASP_pd_min;GSVM2_Glucose_pd_min};
Max =
{GSVM2_Age_pd_max;GSVM2_BMI_pd_max;GSVM2_HDL_pd_max;GSVM2_ALT_pd_ma
x;...
    GSVM2_AST_pd_max;GSVM2_ASP_pd_max;GSVM2_Glucose_pd_max};

```

```

Range =
{GSVM2_Age_pd_range;GSVM2_BMI_pd_range;GSVM2_HDL_pd_range;GSVM2_ALT_pd
_range;...
GSVM2_AST_pd_range;GSVM2_ASP_pd_range;GSVM2_Glucose_pd_range};
Q25 =
{GSVM2_Age_pd_Quantiles(1,1);GSVM2_BMI_pd_Quantiles(1,1);GSVM2_HDL_pd_Quantil
es(1,1);GSVM2_ALT_pd_Quantiles(1,1);...

GSVM2_AST_pd_Quantiles(1,1);GSVM2_ASP_pd_Quantiles(1,1);GSVM2_Glucose_pd_Quan
tiles(1,1)};
Q50 =
{GSVM2_Age_pd_Quantiles(1,2);GSVM2_BMI_pd_Quantiles(1,2);GSVM2_HDL_pd_Quantil
es(1,2);GSVM2_ALT_pd_Quantiles(1,2);...

GSVM2_AST_pd_Quantiles(1,2);GSVM2_ASP_pd_Quantiles(1,2);GSVM2_Glucose_pd_Quan
tiles(1,2)};
Q75 =
{GSVM2_Age_pd_Quantiles(1,3);GSVM2_BMI_pd_Quantiles(1,3);GSVM2_HDL_pd_Quantil
es(1,3);GSVM2_ALT_pd_Quantiles(1,3);...

GSVM2_AST_pd_Quantiles(1,3);GSVM2_ASP_pd_Quantiles(1,3);GSVM2_Glucose_pd_Quan
tiles(1,3)};
Q100 =
{GSVM2_Age_pd_Quantiles(1,4);GSVM2_BMI_pd_Quantiles(1,4);GSVM2_HDL_pd_Quantil
es(1,4);GSVM2_ALT_pd_Quantiles(1,4);...

GSVM2_AST_pd_Quantiles(1,4);GSVM2_ASP_pd_Quantiles(1,4);GSVM2_Glucose_pd_Quan
tiles(1,4)};

GSVM2_pd_Stats = table(Param, GSVM2_Mean, SD, Min, Max, Range, Q25, Q50, Q75,
Q100);
% writetable(GSVM2_pd_Stats,'Female_GSVM2_pd_code2_stats_pd_03_30.xlsx');

%% Combined Female Results

Combined_pd_Stats = table(Param, QSVM_Mean, GSVM1_Mean, GSVM2_Mean);
writetable(Combined_pd_Stats, 'Female_pd_combined_means_03_30.xlsx');

Combined_ambiguity = table(Ambiguity_QSVM', Ambiguity_GSVM1', Ambiguity_GSVM2');
Combined_ambiguity.Properties.RowNames = {'Age','BMI','HDL','ALT','AST','ASP','Glucose'};
Combined_ambiguity.Properties.VariableNames = {'Ambiguity%_QSVM',...
'Ambiguity%_GSVM1','Ambiguity%_GSVM2'};
writetable(Combined_ambiguity, 'Female_ambiguity_03_30.xlsx');

```

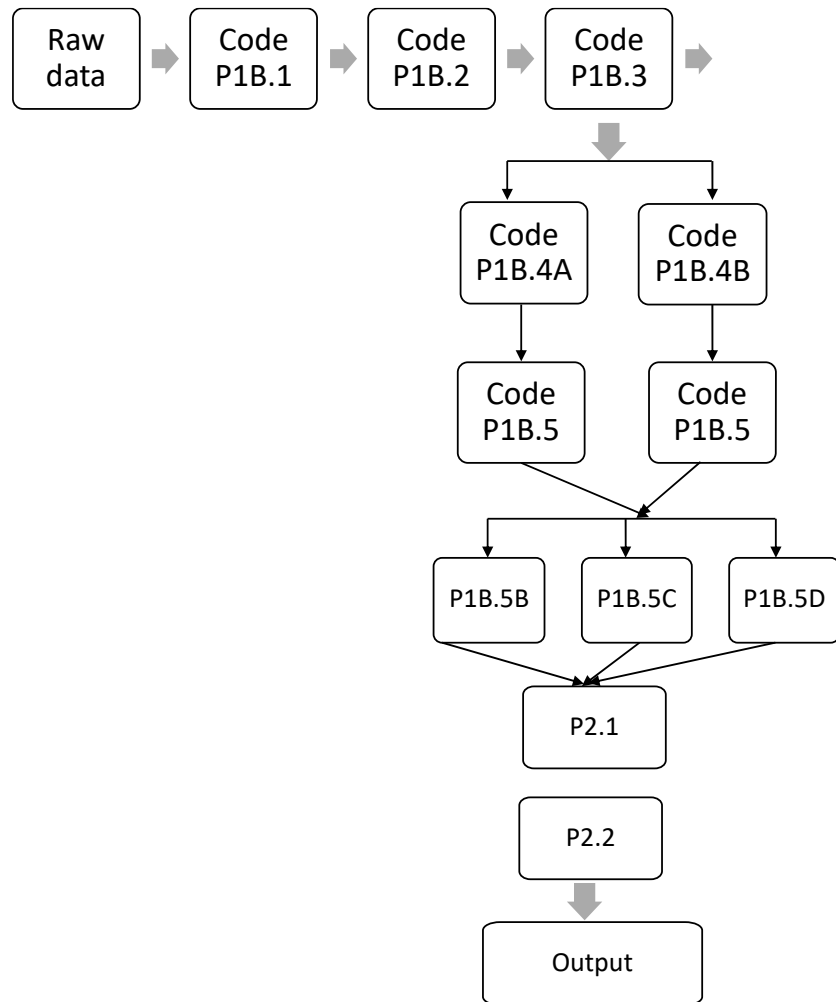


Figure P2.1: Figure outlining the flow of code used in this research objective

## **CHAPTER 4. ASSESSMENT OF HS PREDICTION MODELS USING HEAVY METAL EXPOSURE DATA (PAPER 3)**

### **4.1 Introduction**

Liver is the major detoxification organ in the human system and is therefore susceptible to toxicity [1]. All endogenous as well as exogenous toxicants pass through the liver and the liver is responsible for their metabolism [1]. The toxic mechanisms of most toxicants involve passing through the liver [1]–[3]. This process can cause harm to the liver, especially if the human body is chronically exposed to toxicants. Exposure to Cadmium [4], [5], Arsenic [2], Lead [5], [6], and Mercury [7]–[10] are all found to be linked with liver dysfunction. Interestingly, high levels of Iron are also related to liver disease [11], [12].

Researchers have found a significant impact of toxicants on the liver, and the terms “Toxicant” Associated Fatty Liver Disease (TAFLD) and “Toxicant” Associated Steatohepatitis (TASH) were coined to label liver injury caused specifically due to toxicants [13], [14]. It is important to note that TAFLD and TASH are similar in pathology to NAFLD and NASH, respectively [13], [14].

### **4.2 Literature review**

A detailed literature review of heavy metals and their relationship with NAFLD was covered in the general literature review section of this work (section 2.3). Several studies have found associations between liver dysfunction and heavy metal exposure.

One study used NHANES data to identify the relationship between NAFLD and Arsenic (urinary Arsenic levels) [15]. A “positive association” between Arsenic exposure and risk of NAFLD was reported [15]. However, it is important to note that they used elevated levels of the ALT enzyme as a proxy marker for NAFLD. Specifically, the used ALT levels greater than 25 U/L and 22 U/L for boys and girls (under 17 years), respectively, as a marker for liver dysfunction [15]. Levels over 30 U/L and 19 U/L were used to define liver dysfunction for men and women, respectively, in their work [15]. It is important to note here that while elevated ALT indicates liver dysfunction, it is not a biomarker for NAFLD [16].

Another research group used NHANES data to measure the associations between Lead, Mercury and liver disease [6]. They also reported a dose-dependent association between heavy metal exposure and ALT elevation [6]. Increasing levels of Lead and Mercury in the blood were individually found to be associated with ALT elevation [6].

In this work, a preliminary study was intended to understand the relationship, if any, between HS and heavy metal exposure, within the specific NHANES III dataset. However, all the data of interest were not available within the NHANES III dataset. For example, Arsenic is a known hepatotoxin, but Arsenic exposure data were not available within NHANES III. While a different NHANES dataset – ‘Continuous NHANES’, includes urinary Arsenic information, it does not include the Hepatic Steatosis data (using Ultrasound tests).

The relationship of heavy metal exposure and the biological mechanism leading to HS is complex. Although other researchers have used continuous NHANES data with Arsenic exposure to understand its association with NAFLD, they did not use the ultrasound-based HS parameter to assess NAFLD presence [15]. Instead, they used a proxy parameter, ALT, with an assumption that it could represent the presence or absence of NAFLD [15]. In this work, the abnormal ALT levels are not considered as a determinant of NAFLD or HS to build ML models. In this research, the presence or absence of HS is determined using ultrasound-based HS detection. The use of heavy metal exposure and HS (ultrasound-based) was not found in any other research, based on the literature review conducted in this work.

Therefore, the hypothesis used in this work is as follows:

Heavy metal exposure is related to NAFLD and could be useful in predicting HS (using a ML-based model).

Based on the above hypothesis, the research objective of this work is to:

Assess the effect of using specific heavy metal exposure information (along with other physiological and liver biochemistry data) in human system on the predictability of HS model.

### **4.3 Methods**

The effect of heavy metal exposure on the liver is explained in the general literature review portion of this dissertation. Toxicant associated fatty liver disease (TAFLD) and Toxicant associated steatohepatitis (TASH) were also discussed. The association between NAFLD and chronic heavy metal exposure was also briefly reviewed.

In this research objective, the link between heavy metal exposure and HS was evaluated by including exposure-related parameters to predict HS. The effect of heavy metal exposure data was tested by subsequent inclusion and exclusion of the relevant parameters and comparing their results.

#### **a. Data processing**

Data from NHANES III were used in this research objective [17], [18]. Compared with previous objectives, additional parameters related to heavy metal exposure were incorporated in this objective. The following parameters were chosen as input features in this objective: Age, BMI, HDL, ALT, AST, ASP, Lead, Iron, Cadmium, and Insulin resistance. The parameter Insulin resistance was derived using two other parameters: Fasting insulin and fasting glucose. Insulin resistance was found to be one of the largest risk factors associated with HS [19]–[22]. The output parameter was HS. The same dataset as that in objective 1B was imported from SAS into MATLAB.

All observations with any missing information were deleted. Any observations with fasting time of under eight hours were excluded from this research [23]. The derived parameter of insulin resistance was computed using the formula in equation (21). The equation for computing insulin resistance was referred to from previous literature [24].

$$\text{Insulin resistance} = \frac{\text{Fasting insulin} \times \text{fasting glucose}}{22.5} \quad (21)$$

Data were then split into male and female sub-datasets. Alcohol related exclusions were applied. Again, class imbalance was an inherent challenge and SMOTE was applied to augment the original data with generate synthetic HS data. After the processing, the dataset sizes were as shown in

Table 4.1 &

Table 4.2.

## **b. Model selection**

To identify the best ML model, the data were used to train 17 different models. These models belonged to the five model families: tree-based, ensemble-based (random forest, boosted trees), K-nearest neighbors, support vector machines (SVM), and logistic regression. Each of these model families is explained in detail in chapter 2 & 3. The next step of selecting relevant features is explained in the section below.

## **c. Feature selection**

A comparison of model performance was designed in this objective. First, a cluster of parameters called heavy metal exposure data was included in training the ML models. The heavy metal exposure data used in this research contains three parameters: Lead (Pb), Iron (Fe) and Cadmium (Cd). Although the initial research plan included the use of Arsenic data as well, it was not available in NHANES III. While Arsenic data is available in the larger continuous NHANES datasets, the ultrasound-based HS data was only available in NHANES III. Therefore, the research methods in this objective do not include Arsenic, although it is a critical hepatotoxin [2].

The models trained using the heavy metal exposure data had a total 10 input features: Age, BMI, HDL, ALT, AST, ASP, Lead, Iron, Cadmium, and Insulin resistance. These trained models were then tested, and the best performing models were identified.

In the second approach, the features in the heavy metal exposure data were removed from the dataset and the ML models were trained again. The models trained using this approach had only seven input parameters: Age, BMI, HDL, ALT, AST, ASP, and Insulin resistance. The models were tested again, and the best performing models were identified.

The data were divided into training and test in a 70:30 ratio, respectively. Training data was fed to the models first and then they were tested on the separate test dataset. 10-fold cross validation was used to ensure the models do not overfit the data. Each training and test session was repeated 10-times for every model and the performance results were averaged. The average results of the best performing models are shown in

A comparison of the performances of models trained and tested with and without the heavy metal exposure data is presented in the results and discussion section.





#### 4.4 Results and discussion

A comparison of the models with and without heavy metal exposure data is presented in this section. The main research interest is to identify the impact of using Lead, Cadmium, and Iron data on model performance. For a quick reference, the models without the heavy metal exposure data have the following input parameters: Age, BMI, HDL, ALT, AST, ASP, and Insulin resistance. The models including the heavy metal exposure data have the following input parameters: Age, BMI, HDL, ALT, AST, ASP, Lead, Iron, Cadmium, and Insulin resistance. In both the approaches (with and without heavy metal exposure data), 17 models from five model families were trained separately for male and female populations.

The detailed dataset sizes are shown in

Table 4.1 &

Table 4.2. The performance results for male- specific models with heavy metal exposure data are in Table 4.3, without heavy metal exposure data are in . For female specific models with heavy metal exposure data are in

Table 4.5 and without the heavy metal exposure data are in Table 4.6.

##### *Male – specific models*

The results of the models with heavy metal exposure data had the following performance ranges: test accuracy: 66 – 74%, sensitivity: 66 – 81% and specificity: 51 – 83%. The model with highest test accuracy was Coarse KNN. The model with highest sensitivity was Gaussian SVM I at 83%. However, it has a very poor specificity of only 51%. These results imply that although the Gaussian SVM I model is 83% accurate at predicting those with HS, it does not perform well when predicting those without HS.

The results of the models without heavy metal exposure data had the following performance ranges: test accuracy: 70 - 72%, sensitivity: 66 - 78% and specificity: 62 - 79%. The model with highest test accuracy was Gaussian SVM II at 72%. The model with highest sensitivity was Gaussian SVM I at 78%. Similar to the models with heavy metal exposure data included, the

Gaussian SVM I has the best sensitivity performance. However, it continues to result in poor specificity (62% for the models without heavy metal exposure data).

Interestingly, the use of heavy metal exposure data only improved the performance of the models by 2% (test accuracy), 3% (sensitivity), and 4% (specificity). In both the cases, the model with highest sensitivity was Gaussian SVM I. But due to the significantly low specificity performance of the Gaussian SVM I, this model would require more training, and validation before use. Additional analysis to these models is out of scope for this dissertation but can be explored in the future.

It is postulated that including time-series heavy metal exposure data might be more beneficial in predicting HS than stationary data (which is used in this research). Further, additional research regarding the impacts of toxins on liver health might lead to intermediate parameters/biomarkers. In the future, if such research becomes available, those parameters can be included in the model to improve the HS prediction.

#### *Female – specific models*

The results of the models with the heavy metal exposure data had the following performance ranges: test accuracy: 66 – 72 %, sensitivity: 67 – 78 % and specificity: 54 – 76 %. The models with highest test accuracies were Logistic Regression and Ensemble Subspace Discriminant, both with a test accuracy of 72 %. The model with highest sensitivity was Gaussian SVM 1 at 78%. However, the specificity of the Gaussian SVM I model was very poor at 54%.

The results of the models without heavy metal exposure data had the following performance ranges: test accuracy: 70 – 73%, sensitivity: 69 – 74% and specificity: 66 – 77%. The model with highest test accuracies were Logistic Regression and Linear SVM with a test accuracy of 73%. The model with highest sensitivity was Gaussian Scale I SVM at 74%. Although the specificity of the Gaussian SVM I was still low at 66%, it was higher than the model with heavy metal exposure data.

Overall, the difference in performances was still minimal at 1% test accuracy, 4% sensitivity and 1% specificity. The models without heavy metal exposure data resulted in better test accuracy and better specificity than those with heavy metal exposure data. However, the sensitivity of the model with heavy metal exposure data included as higher than that of the model without.

While the impact of using toxicological data (Lead, Cadmium, and Iron) levels did not show a significant improvement with ML model performance, these parameters need additional investigation. Increasing research related to heavy metal exposure and the liver might benefit the prediction of HS using ML in the future.

#### **4.5 Summary and conclusions**

Models were developed to test the impact of including vs excluding a cluster of heavy metal exposure data (Lead, Iron, and Cadmium). The best performing models (with heavy metal exposure) had 74% and 72% test accuracies for male and female, respectively. The best performing models without heavy metal exposure had test accuracies of 72% and 73% for male and female, respectively. Interestingly, while the use of heavy metal exposure data improved the Male -specific model performance by 2%, it decreased the performance of the female-specific model by 1%. Overall, the use of heavy metal exposure parameters did not impact the model performance (less than 3%) in this work.

The conclusion of this research was:

Inclusion of heavy metal exposure (Lead, Iron, and Cadmium) did not have a numerically significant impact on the model performance in predicting HS.

#### **4.6 Recommendations for future work**

Future research regarding heavy metal exposure and its impact on the liver might lead to discovery of biomarkers for liver related heavy metal exposure. Any such potential biomarkers would be a good feature to include in screening of NAFLD. To understand the impact of heavy metal exposure on HS prediction, use of longitudinal/time-series data is recommended. Additional data related to heavy metal exposure like: Arsenic, mercury and other heavy metals is also recommended to be used along with Lead, Cadmium, and Iron data. Utilizing the sample weights provided by NHANES and developing models using weighted observations are also recommended.

## 4.7 Tables

Table 4.1: Dataset sizes after applying SMOTE – with heavy metal exposure parameters

Sex	HS (before SMOTE)	HS (after SMOTE)	No-HS	Total
Male	581	2,324	1,962	4,286
Female	684	2,736	2,803	5,539

Table 4.2: Dataset sizes after applying SMOTE – without heavy metal exposure parameters

Sex	HS (before SMOTE)	HS (after SMOTE)	No-HS	Total
Male	588	2,352	1,978	4,330
Female	690	2,760	2,830	5,590

Table 4.3: Best performing models using heavy metal exposure data - male populations

Models	Training (%)	Testing (%)	Sensitivity (%)	Specificity (%)
Gaussian SVM I	64.5	66.1	81.0	51.1
Coarse KNN	67.2	74.4	65.5	83.3
Ensemble Boosted	67.6	71.8	71.8	71.8
Ensemble RUS Boosted Trees	68.2	73.5	74.7	72.4

Table 4.4: Best performing models excluding heavy metal exposure data - male populations

Models	Training (%)	Testing (%)	Sensitivity (%)	Specificity (%)
Logistic Regression	71.3	71.9	69.4	74.6
Linear SVM	70.8	71.9	66.6	77.3
Gaussian SVM I	67.2	69.6	77.5	61.7
Gaussian SVM II	70.5	72.4	70.1	74.7
Gaussian SVM III	70.2	71.5	64.2	78.7

Table 4.5: Best performing models using heavy metal exposure data - female populations

Models	Training	Testing	Sensitivity	Specificity
Logistic Regression	71.9	71.9	68.5	75.4
Gaussian SVM I	66.5	65.9	77.7	54.1
Gaussian SVM II	71.5	71.4	67.2	75.6
Gaussian SVM III	71.5	71.4	67.1	75.8
Ensemble Subspace Discriminant	72.1	71.9	68.0	75.9

Table 4.6: Best performing models excluding heavy metal exposure data - female populations

Models	Training (%)	Testing (%)	Sensitivity (%)	Specificity (%)
Logistic Regression	72.2	73.0	68.9	77.1
Linear SVM	72.2	73.0	68.8	77.2
Gaussian SVM I	69.6	70.2	74.3	66.1
Ensemble Subspace Discriminant	72.7	72.8	69.6	76.1

## 4.8 References

- [1] J. Ozougwu, "Physiology of the liver," vol. 4, pp. 13–24, Jan. 2017.
- [2] G. E. Arteel, "Hepatotoxicity," in *Arsenic*, John Wiley & Sons, Ltd, 2015, pp. 249–265. doi: 10.1002/9781118876992.ch11.
- [3] H. Jaeschke, "Mechanisms of Hepatotoxicity," *Toxicol. Sci.*, vol. 65, no. 2, pp. 166–176, Feb. 2002, doi: 10.1093/toxsci/65.2.166.
- [4] T. Bhattacharjee, S. Bhattacharjee, and D. Choudhuri, "HEPATOTOXIC AND NEPHROTOXIC EFFECTS OF CHRONIC LOW DOSE EXPOSURE TO A MIXTURE OF HEAVY METALS – LEAD, CADMIUM AND ARSENIC," p. 10, 2016.
- [5] Y. Chen, X. Xu, Z. Zeng, X. Lin, Q. Qin, and X. Huo, "Blood lead and Cadmium levels associated with hematological and hepatic functions in patients from an e-waste-polluted area," *Chemosphere*, vol. 220, pp. 531–538, Apr. 2019, doi: 10.1016/j.chemosphere.2018.12.129.
- [6] Cave Matt, Appana Savitri, Patel Mihir, Falkner Keith Cameron, McClain Craig J., and Brock Guy, "Polychlorinated Biphenyls, Lead, and Mercury Are Associated with Liver Disease in American Adults: NHANES 2003–2004," *EnvIron. Health Perspect.*, vol. 118, no. 12, pp. 1735–1742, Dec. 2010, doi: 10.1289/ehp.1002720.
- [7] J. Choi *et al.*, "Mercury Exposure in Association With Decrease of Liver Function in Adults: A Longitudinal Study," *J. Prev. Med. Pub. Health*, vol. 50, no. 6, pp. 377–385, Nov. 2017, doi: 10.3961/jpmp.17.099.
- [8] H. Lee, Y. Kim, C.-S. Sim, J.-O. Ham, N.-S. Kim, and B.-K. Lee, "Associations between blood mercury levels and subclinical changes in liver enzymes among South Korean general adults: Analysis of 2008–2012 Korean national health and nutrition examination survey data," *EnvIron. Res.*, vol. 130, pp. 14–19, Apr. 2014, doi: 10.1016/j.envres.2014.01.005.
- [9] M.-R. Lee, Y.-H. Lim, B.-E. Lee, and Y.-C. Hong, "Blood mercury concentrations are associated with decline in liver function in an elderly population: a panel study," *EnvIron. Health*, vol. 16, no. 1, p. 17, Mar. 2017, doi: 10.1186/s12940-017-0228-2.
- [10] Y.-S. Lin *et al.*, "Association of body burden of mercury with liver function test status in the U.S. population," *EnvIron. Int.*, vol. 70, pp. 88–94, Sep. 2014, doi: 10.1016/j.envint.2014.05.010.
- [11] A. Pietrangelo, "Iron and the liver," *Liver Int.*, vol. 36, pp. 116–123, Jan. 2016, doi: 10.1111/liv.13020.
- [12] A. Pietrangelo, "Iron in NASH, chronic liver diseases and HCC: How much Iron is too much?," *J. Hepatol.*, vol. 50, no. 2, pp. 249–251, Feb. 2009, doi: 10.1016/j.jhep.2008.11.011.

- [13] B. Wahlang *et al.*, “Toxicant-associated Steatohepatitis,” *Toxicol. Pathol.*, vol. 41, no. 2, pp. 343–360, Feb. 2013, doi: 10.1177/0192623312468517.
- [14] M. Cave *et al.*, “Toxicant-associated steatohepatitis in vinyl chloride workers,” *Hepatology*, vol. 51, no. 2, pp. 474–481, Feb. 2010, doi: 10.1002/hep.23321.
- [15] J. K. Frediani, E. A. Naioti, M. B. Vos, J. Figueroa, C. J. Marsit, and J. A. Welsh, “Arsenic exposure and risk of nonalcoholic fatty liver disease (NAFLD) among U.S. adolescents and adults: an association modified by race/ethnicity, NHANES 2005–2014,” *Environ. Health*, vol. 17, no. 1, Dec. 2018, doi: 10.1186/s12940-017-0350-1.
- [16] “Understand Liver Enzyme Test Results — American Liver Foundation.” <https://liverfoundation.org/understand-liver-enzyme-test-results-2/> (accessed Mar. 30, 2021).
- [17] Centers for Disease Control and Prevention (CDC). National Center for Health Statistics (NCHS). National Health and Nutrition Examination Survey Data, “NHANES III (1988–1994) - Data Files.” <https://wwwn.cdc.gov/nchs/nhanes/nhanes3/datafiles.aspx#core> (accessed Apr. 23, 2021).
- [18] “NHANES 1988-1994: Hepatic/Gallbladder Ultrasound and Hepatic Steatosis Data Documentation, Codebook, and Frequencies.” [https://wwwn.cdc.gov/nchs/data/nhanes3/34a/HGUHS.htm#Data\\_Processing\\_and\\_Editing](https://wwwn.cdc.gov/nchs/data/nhanes3/34a/HGUHS.htm#Data_Processing_and_Editing) (accessed Jan. 30, 2019).
- [19] Z. M. Younossi *et al.*, “The global epidemiology of NAFLD and NASH in patients with type 2 diabetes: A systematic review and meta-analysis,” *J. Hepatol.*, vol. 71, no. 4, pp. 793–801, Oct. 2019, doi: 10.1016/j.jhep.2019.06.021.
- [20] G. Bhat and C. S. Baba, “Insulin resistance and metabolic syndrome in nonobese Indian patients with non-alcoholic fatty liver disease,” *Trop. Gastrology*, vol. 34, no. 1, pp. 18–24, Mar. 2013, doi: 10.7869/tg.2012.86.
- [21] B. Sears and M. Perry, “The role of fatty acids in insulin resistance,” *Lipids Health Dis.*, vol. 14, no. 1, p. 121, Dec. 2015, doi: 10.1186/s12944-015-0123-1.
- [22] H. Kitade, G. Chen, Y. Ni, and T. Ota, “Nonalcoholic fatty liver disease and insulin resistance: new insights and potential new treatments,” *Nutrients*, vol. 9, no. 4, p. 387, 2017.
- [23] “The A1C Test & Diabetes | NIDDK,” *National Institute of Diabetes and Digestive and Kidney Diseases*. <https://www.niddk.nih.gov/health-information/diagnostic-tests/a1c-test> (accessed Jun. 23, 2021).
- [24] “Association Between Serum Concentrations of Persistent Organic Pollutants and Insulin Resistance Among Nondiabetic Adults | Diabetes Care | American Diabetes Association.” <https://diabetesjournals.org/care/article/30/3/622/25699/Association-Between-Serum-Concentrations-of> (accessed Mar. 24, 2022).

## APPENDIX D. P3 - CODE FOR OBJECTIVE 3

### 1. SAS CODE TO KEEP VARIABLES OF INTEREST AND DISCARD THE REST

```
%%%%%%%%%%
% Created on: 02/17/2022
% Input: Raw data from NHANES
% Output: Data with only variables of interest, specific to objective 3
% Author: Ridhi Deo
% File name: obj1c_sas_1.sas
% Description: Used eliminate the variables that are not required and to only keep the variables
of interest from the raw datasets. This program was developed using SAS 2019 [64].
%%%%%%%%%%

% set the data path and choose the variables to keep. Variable codes are as provided by
% NHANESIII

LIBNAME NH "Raw data path";

data adult;
set NH.adult;
keep SEQN HSAGEIR HSSEX DMARETHN HAD1 HAD6 HAD10;
proc sort; by seqn; run;

data lab;
set NH.lab;
keep SEQN AHP HBP SSP SAP HCP DHP NAPSI SKPSI CLPSI C3PSI SCPSI PPSI
UAPSI G1P G2P BUPSI TBPSI CEPsi SFPSI CHPSI TRPSI ASPSI ATPSI GGPSI
LDPSI APPSI TPPSI AMPsi GBPSI OSPsi GHP GHPMETH G1PSI G1PCODE G2PSI
C1PSI C2PSI I1PSI I2PSI UDP UDPSI URPSI UBP UIP PLPSI PVPSI PBP PBPSI FEP
FEPSI VBPSI VCPSI ICPSI CAPSI SEPSI VAPSI VEPSI ACPSI BCPSI TCPSI TGPSI
LCPSI HDPSI AAPSI ABPSI LPPSI PHPFAST;
proc sort; by seqn; run;

data exam;
set NH.exam;
keep SEQN PEP6DR BMPBMI BMPWAIST MAPA1 MAPA2A MAPA2B MAPA3
MAPE1 MAPE2 MAPE4;
proc sort; by seqn; run;

data HGUHS;
set NH.HGUHS;
keep SEQN GUPHSQC GUPHSLKC GUPHSPB GUPHSDBA GUPHSVW
GUPHSDGB GUPHSPF GUPHSPFR GUPHSC GUPHSREV;
proc sort; by seqn; run;
```



```
proc contents data = NH.adult;  
run;
```

```
proc contents data = NH.lab;  
run;
```

```
proc contents data = NH.exam;  
run;
```

```
proc contents data = NH.HGUHS;  
run;
```

## 2. SAS CODE TO MERGE DATASETS

```
%%%%%%%%%%  
% Created on: 02/17/2022  
% Input: Processed data with only variables of interest, specific to objective 3  
% Output: Multiple datasets of interest merged into one dataset  
% Author: Ridhi Deo  
% File name: obj1c_sas_2.sas  
% Description: Used to combine different datasets of interest into one. This program was  
developed using SAS 2019 [64].  
%%%%%%%%%%
```

```
% Sorting data using sequential numbers  
proc sort data=work.adult;  
    by SEQN;  
proc sort data=work.lab;  
    by SEQN;  
proc sort data=work.exam;  
    by SEQN;  
proc sort data=work.hguhs;  
    by SEQN;  
%Merging data using the sequential number  
data NH.merged;  
    merge work.adult  
          work.lab  
          work.exam  
          work.hguhs;  
    by SEQN;
```

```
proc contents data = NH.merged varnum;  
proc means data=NH.merged N Nmiss min max maxdec=2;  
run;
```

### 3. MATLAB CODE TO PROCESS AND CREATE DISEASE AND NO-DISEASE DATASETS (WITH HEAVY METAL EXPOSURE DATA)

```
%%%%%%%%%%
% Created on: 02/17/22
% Input: Merged dataset from SAS
% Output: Processed data, split into male and female sub-datasets
% Author: Ridhi Deo
% File name: Obj1c_matlab_1a.m
% Description: This code was written to process data (with heavy metal exposure data) and split
it into male and female sub-datasets. This program was developed using SAS 2019 [64].
%%%%%%%%%%

clc
close all;
%% Data import
data = readtable(data directory);
%% Extracting the following features
% SEQN, Age, Sex, Lead, Iron, Cadmium
% Will also need to extract alcohol data so that exclusions can be applied
data = data(:,[1,3,4,8,12,14,27,50,51,54,62,65,68,69,71,76,82:84,92]);

data.Properties.VariableNames{'HSAGEIR'} = 'Age';
data.Properties.VariableNames{'HSSEX'} = 'Sex';
data.Properties.VariableNames{'BMPBMI'} = 'BMI';
data.Properties.VariableNames{'MAPE1'} = 'Alcohol_12_life';
data.Properties.VariableNames{'MAPE2'} = 'Alcohol_12_last_year';
data.Properties.VariableNames{'MAPE4'} = 'Drinks_per_day';
data.Properties.VariableNames{'GUPHSPFR'} = 'HS';
data.Properties.VariableNames{'ATPSI'} = 'ALT';
data.Properties.VariableNames{'ASPSI'} = 'AST';
data.Properties.VariableNames{'APPSI'} = 'ASP';
data.Properties.VariableNames{'G1PSI'} = 'Plasma_glucose_1';
data.Properties.VariableNames{'G2PSI'} = 'Plasma_glucose_2';
data.Properties.VariableNames{'I1PSI'} = 'Insulin_1';
data.Properties.VariableNames{'I2PSI'} = 'Insulin_2';
data.Properties.VariableNames{'HDPSI'} = 'HDL';
data.Properties.VariableNames{'PBPSI'} = 'Lead';
data.Properties.VariableNames{'FEPsi'} = 'Iron';
data.Properties.VariableNames{'UDPSI'} = 'Cadmium';
data.Properties.VariableNames{'PHPFAST'} = 'Fasting_time_hours';

%% Alcohol data columns processing
    % Filling in missing data for the 12 drinks per year column with information from 12
    drinks in life column
% If a person has not had 12 drinks in their lifetime, the response on the
% variable 12 drinks in past year are missing
% To fix that, individuals who have not had 12 drinks in their life will
```

```

% have 0s on the column 12 drinks in past year
    % Same logic applies - making all those who have not had 12 drinks in their life 0 in the
    column drinks per day
for i = 1: size(data,1)
    if (data.Alcohol_12_life(i) == 2)
        data.Alcohol_12_last_year(i) = 0;
        data.Drinks_per_day(i) = 0;
    end
end

for i = 1: size(data,1)
    if (data.Alcohol_12_last_year(i) == 2)
        data.Drinks_per_day(i) = 0;
    end
end

%% Cleaning up all the junk data (represented as 888 or 8888 or 999 etc.) withing
variables of interest
% The information was referred from NHANES 3 documentation
% Since we have not used any youth data, all NaNs in the Age column could
% correspond to that
idx_age = find(isnan(data.Age)); % Eliminated 13,149 samples
data(idx_age,:) = [];
clear idx_age; % Sample size: 20,050 x 17

% Sex
% No missing or junk data

% Fasting time
data.Fasting_time_hours(data.Fasting_time_hours == 88888) = NaN;

% Lead,Cadmium, Iron
data.Lead(data.Lead == 88888) = NaN;
data.Cadmium(data.Cadmium == 888888) = NaN;
data.Iron(data.Iron == 88888) = NaN;

% HS
% 7    Image is present, but ungradable
% 8    No image
data.HS(data.HS == 7) = NaN;
data.HS(data.HS == 8) = NaN;

% MAPE1 In your entire life, have you had at least 12 drinks of any kind of alcoholic
beverage? Do not count small tastes.
% 8 - Blank but applicable, 9 - dont know.
data.Alcohol_12_life(data.Alcohol_12_life == 8) = NaN;

```

```

data.Alcohol_12_life(data.Alcohol_12_life == 9) = NaN;

% MAPE2 In the past 12 months did you
% have at least 12 drinks of any kind of alcoholic beverage?
% 8 - Blank but applicable, 9 - dont know.
data.Alcohol_12_last_year(data.Alcohol_12_last_year == 8) = NaN;
data.Alcohol_12_last_year(data.Alcohol_12_last_year == 9) = NaN;

% MAPE4 On the average, on the days that you drank alcohol, how many drinks did you
% have a day? (By a drink, I mean a 12-oz beer, a 4-oz glass of wine, or an ounce of liquor.)
% 888 - Blank but applicable, 999 - dont know.
data.Drinks_per_day(data.Drinks_per_day == 888) = NaN;
data.Drinks_per_day(data.Drinks_per_day == 999) = NaN;

% Glucose and insulin related junk data
data.Plasma_glucose_1(data.Plasma_glucose_1 == 888888) = NaN;
data.Plasma_glucose_2(data.Plasma_glucose_2 == 888888) = NaN;
data.Insulin_1(data.Insulin_1 == 8888888) = NaN;
data.Insulin_2(data.Insulin_2 == 8888888) = NaN;
%% Eliminating missing data from HS - we need to eliminate this data because this is our
% output variable and ground truth
HS_missing_idx = find(isnan(data.HS)); %6,194 samples with missing HS
data(HS_missing_idx,:) = [];
clear HS_missing_idx;

% Lead, Cadmium, Iron
Lead_missing_idx = find(isnan(data.Lead)); %457 with missing lead data
data(Lead_missing_idx, :) = [];

Cd_missing_idx = find(isnan(data.Cadmium)); %149 with missing Cd data
data(Cd_missing_idx, :) = [];

Iron_missing_idx = find(isnan(data.Iron)); %0 missing Iron data
data(Iron_missing_idx, :) = [];

Fasting_missing_idx = find(isnan(data.Fasting_time_hours));
data(Fasting_missing_idx,:) = []; % 12 missing Fasting information

clear Iron_missing_idx Cd_missing_idx Lead_missing_idx Fasting_missing_idx;

%% Junk and missing data - ALT, AST, ASP, HDL, BMI
% AST
% 888 Blank but applicable
data.AST(data.AST == 888) = NaN;
% ALT

```

```

% 888 Blank but applicable
data.ALT(data.ALT == 888) = NaN;
% ASP
% 8888 Blank but applicable
data.ASP(data.ASP == 8888) = NaN;
% BMI
    % 8888 was found as junk data via visual examination of data. Although I didnt see this
    on the website for
% NHANES, it is removed because 8888 is not appropriate BMI
data.BMI(data.BMI == 8888) = NaN;
% HDL
data.HDL(data.HDL == 8888) = NaN;

ALT_missing_idx = find(isnan(data.ALT));
data(ALT_missing_idx,:) = [];

AST_missing_idx = find(isnan(data.AST));
data(AST_missing_idx,:) = [];

ASP_missing_idx = find(isnan(data.ASP));
data(ASP_missing_idx,:) = [];

BMI_idx = find(isnan(data.BMI));
data(BMI_idx,:) = [];

HDL_idx = find(isnan(data.HDL));
data(HDL_idx,:) = [];

clear ALT_missing_idx AST_missing_idx ASP_missing_idx BMI_idx HDL_idx;

%% IF there are NaNs in G1PSI, fill them with G2PSI. If both G1PSI and G2PSI are
% NaNs, then delete the sample
for i = 1:size(data,1)
    if(isnan(data.Plasma_glucose_1(i)))
        if(isnan(data.Plasma_glucose_2(i)))
            idx_pg(i) = i;
        else
            data.Plasma_glucose_1(i) = data.Plasma_glucose_2(i);
        end
    end
end
end

data.Plasma_glucose_2 = [];
    Plasma_glucose_idx = find(isnan(data.Plasma_glucose_1)); %25 cases of missing plasma
    glucose samples after combining G1PSI and G2PSI
data(Plasma_glucose_idx,:) = [];

```

```

%% IF there are NaNs in I1PSI, fill them with I2PSI. If both I1PSI and I2PSI are
% NaNs, then delete the sample
for i = 1:size(data,1)
    if(isnan(data.Insulin_1(i)))
        if(isnan(data.Insulin_2(i)))
            idx_in(i) = i;
        else
            data.Insulin_1(i) = data.Insulin_2(i);
        end
    end
end
end

data.Insulin_2 = [];
    Insulin_idx = find(isnan(data.Insulin_1)); %62 cases of missing insulin samples after
    combining I1PSI and I2PSI
data(Insulin_idx,:) = [];

clear idx_pg idx_in Plasma_glucose_idx Insulin_idx

%% Convert Insulin_1 from pmol/L to mU/L by multiplying with 0.144
data.Insulin_1 = data.Insulin_1*0.144;
% Glucose unit is correct for G1PSI so no need to convert

%% Exclude data related to less than permissible fasting time (X hours)
    idx_fasting = find(data.Fasting_time_hours<8); %Identify data that has fasting hours < 8
    hours
data(idx_fasting,:)=[]; %Eliminate data
%% Delete fasting hours variable
data.Fasting_time_hours = [];
%% Calculating Insulin resistance
data(:,end+1) = array2table(zeros(size(data,1),1));
data.Properties.VariableNames{'Var18'} = 'Insulin_resistance';
% Fasting insulin [mU/L] x fasting glucose [mmol/L] / 22.5
data.Insulin_resistance = data.Insulin_1.*data.Plasma_glucose_1/22.5;

%% Delete plasma glucose and insulin resistance
data.Plasma_glucose_1 = [];
data.Insulin_1 = [];

%% Split datasets into HS and non-HS
    data.HS(data.HS == 1) = 0; % 1 is Normal - Mild as per NHANES. Changing it to 0 to
    indicate no risk
    data.HS(data.HS == 2) = 1; % 2 is Moderate - Severe as per NHANES. Changing it to 1
    to indicate risk

idx_disease = data.HS == 1;

```

```

dataset_HS = data(idx_disease,:);

idx_non_disease = data.HS == 0;
dataset_non_HS = data(idx_non_disease,:);
clear idx_disease idx_non_disease;

%% Split further into Male HS, Non-HS and Female HS, non-HS
dataset_HS_male = dataset_HS(dataset_HS.Sex == 1, :);
dataset_HS_female = dataset_HS(dataset_HS.Sex == 2,:);

dataset_non_HS_male = dataset_non_HS(dataset_non_HS.Sex == 1,:);
dataset_non_HS_female = dataset_non_HS(dataset_non_HS.Sex == 2,:);

%% Apply exclusion criteria for alcohol
% HS and No-HS male exclusion criteria - > 21 drinks/week should be
% excluded
k = 1;
for i = 1: size(dataset_HS_male,1)
    if(dataset_HS_male.Sex(i) == 1 && dataset_HS_male.Drinks_per_day(i) > 3)
        idx_HS_men(k) = i;
        k = k + 1;
    end
end
dataset_HS_male(idx_HS_men,:) = [];
clear k idx_HS_men;

j = 1;
for i = 1: size(dataset_non_HS_male,1)
    if(dataset_non_HS_male.Sex(i) == 1 && dataset_non_HS_male.Drinks_per_day(i) > 3)
        idx_non_HS_men(j) = i;
        j = j + 1;
    end
end
dataset_non_HS_male(idx_non_HS_men,:) = [];
clear j idx_non_HS_men;

% HS and No-HS female exclusion criteria - > 14 drinks/week should be
% excluded
k = 1;
for i = 1: size(dataset_HS_female,1)
    if(dataset_HS_female.Sex(i) == 2 && dataset_HS_female.Drinks_per_day(i) > 2)
        idx_HS_women(k) = i;
        k = k + 1;
    end
end

```

```

dataset_HS_female(idx_HS_women,:) = [];
clear k idx_HS_women;

j = 1;
for i = 1: size(dataset_non_HS_female,1)
    if(dataset_non_HS_female.Sex(i) == 2 && dataset_non_HS_female.Drinks_per_day(i)
        > 2)
        idx_non_HS_women(j) = i;
        j = j + 1;
    end
end
dataset_non_HS_female(idx_non_HS_women,:) = [];
clear j idx_non_HS_women;

%% Delete missing data related to drinks per day
idx_male_HS_drinks = find(isnan(dataset_HS_male.Drinks_per_day));
dataset_HS_male(idx_male_HS_drinks,:) = [];
idx_male_non_HS_drinks = find(isnan(dataset_non_HS_male.Drinks_per_day));
dataset_non_HS_male(idx_male_non_HS_drinks,:) = [];
idx_female_HS_drinks = find(isnan(dataset_HS_female.Drinks_per_day));
dataset_HS_female(idx_female_HS_drinks,:) = [];
idx_female_non_HS_drinks = find(isnan(dataset_non_HS_female.Drinks_per_day));
dataset_non_HS_female(idx_female_non_HS_drinks,:) = [];

    clear idx_female_HS_drinks idx_female_non_HS_drinks idx_male_HS_drinks
    idx_male_non_HS_drinks;

%% Delete alcohol columns from 4 datasets
dataset_HS_male.Alcohol_12_last_year = [];
dataset_HS_male.Alcohol_12_life = [];
dataset_HS_male.Drinks_per_day = [];

dataset_HS_female.Alcohol_12_last_year = [];
dataset_HS_female.Alcohol_12_life = [];
dataset_HS_female.Drinks_per_day = [];

dataset_non_HS_male.Alcohol_12_last_year = [];
dataset_non_HS_male.Alcohol_12_life = [];
dataset_non_HS_male.Drinks_per_day = [];

dataset_non_HS_female.Alcohol_12_last_year = [];
dataset_non_HS_female.Alcohol_12_life = [];
dataset_non_HS_female.Drinks_per_day = [];

%% Delete sex column from all 4 datasets
dataset_HS_male.Sex = [];

```



```

dataset_HS_female.Sex = [];
dataset_non_HS_male.Sex = [];
dataset_non_HS_female.Sex = [];
clear i;

```

#### 4. MATLAB CODE TO PROCESS AND CREATE DISEASE AND NO-DISEASE DATASETS (WITHOUT HEAVY METAL EXPOSURE DATA)

```

%%%%%%%%%%
% Created on: 02/17/22
% Input: Merged dataset from SAS
% Output: Processed data, split into male and female sub-datasets
% Author: Ridhi Deo
% File name: Obj1c_matlab_1b.m
% Description: This code was written to process data (without heavy metal exposure data) and
split it into male and female sub-datasets
%%%%%%%%%%

clc
close all;
%% Data import
data = readtable(Raw_data_path);
%% Extracting the following features based on discussion with Dr. P
% SEQN, Age, Sex, Lead, Iron, Cadmium
% Will also need to extract alcohol data so that exclusions can be applied
data = data(:,[1,3,4,8,12,14,27,50,51,54,62,65,68,69,71,76,82:84,92]);

data.Properties.VariableNames{'HSAGEIR'} = 'Age';
data.Properties.VariableNames{'HSSEX'} = 'Sex';
data.Properties.VariableNames{'BMPBMI'} = 'BMI';
data.Properties.VariableNames{'MAPE1'} = 'Alcohol_12_life';
data.Properties.VariableNames{'MAPE2'} = 'Alcohol_12_last_year';
data.Properties.VariableNames{'MAPE4'} = 'Drinks_per_day';
data.Properties.VariableNames{'GUPHSPFR'} = 'HS';
data.Properties.VariableNames{'ATPSI'} = 'ALT';
data.Properties.VariableNames{'ASPSI'} = 'AST';
data.Properties.VariableNames{'APPSI'} = 'ASP';
data.Properties.VariableNames{'G1PSI'} = 'Plasma_glucose_1';
data.Properties.VariableNames{'G2PSI'} = 'Plasma_glucose_2';
data.Properties.VariableNames{'I1PSI'} = 'Insulin_1';
data.Properties.VariableNames{'I2PSI'} = 'Insulin_2';
data.Properties.VariableNames{'HDPSI'} = 'HDL';
data.Properties.VariableNames{'PBPSI'} = 'Lead';
data.Properties.VariableNames{'FEPSI'} = 'Iron';
data.Properties.VariableNames{'UDPSI'} = 'Cadmium';

```

```

data.Properties.VariableNames{'PHPFAST'} = 'Fasting_time_hours';

%% Alcohol data columns processing
    % Filling in missing data for the 12 drinks per year column with information from 12
    drinks in life column
% If a person has not had 12 drinks in their lifetime, the response on the
% variable 12 drinks in past year are missing
% To fix that, individuals who have not had 12 drinks in their life will
% have 0s on the column 12 drinks in past year
    % Same logic applies - making all those who have not had 12 drinks in their life 0 in the
    column drinks per day
for i = 1: size(data,1)
    if (data.Alcohol_12_life(i) == 2)
        data.Alcohol_12_last_year(i) = 0;
        data.Drinks_per_day(i) = 0;
    end
end

for i = 1: size(data,1)
    if (data.Alcohol_12_last_year(i) == 2)
        data.Drinks_per_day(i) = 0;
    end
end

    %% Cleaning up all the junk data (represented as 888 or 8888 or 999 etc.) withing
    variables of interest
% The information was referred from NHANES 3 documentation
% Since we have not used any youth data, all NaNs in the Age column could
% correspond to that
idx_age = find(isnan(data.Age)); % Eliminated 13,149 samples
data(idx_age,:) = [];
clear idx_age; % Sample size: 20,050 x 17

% Sex
% No missing or junk data

% Fasting time
data.Fasting_time_hours(data.Fasting_time_hours == 88888) = NaN;

% HS
% 7    Image is present, but ungradable
% 8    No image
data.HS(data.HS == 7) = NaN;
data.HS(data.HS == 8) = NaN;

```

```

    % MAPE1 In your entire life, have you had at least 12 drinks of any kind of alcoholic
    beverage? Do not count small tastes.
    % 8 - Blank but applicable, 9 - dont know.
    data.Alcohol_12_life(data.Alcohol_12_life == 8) = NaN;
    data.Alcohol_12_life(data.Alcohol_12_life == 9) = NaN;

    % MAPE2 In the past 12 months did you
    % have at least 12 drinks of any kind of alcoholic beverage?
    % 8 - Blank but applicable, 9 - dont know.
    data.Alcohol_12_last_year(data.Alcohol_12_last_year == 8) = NaN;
    data.Alcohol_12_last_year(data.Alcohol_12_last_year == 9) = NaN;

    % MAPE4 On the average, on the days that you drank alcohol, how many drinks did you
    have a day? (By a drink, I mean a 12-oz beer, a 4-oz glass of wine, or an ounce of liquor.)
    % 888 - Blank but applicable, 999 - dont know.
    data.Drinks_per_day(data.Drinks_per_day == 888) = NaN;
    data.Drinks_per_day(data.Drinks_per_day == 999) = NaN;

    % Glucose and insulin related junk data
    data.Plasma_glucose_1(data.Plasma_glucose_1 == 888888) = NaN;
    data.Plasma_glucose_2(data.Plasma_glucose_2 == 888888) = NaN;
    data.Insulin_1(data.Insulin_1 == 8888888) = NaN;
    data.Insulin_2(data.Insulin_2 == 8888888) = NaN;
    %% Eliminating missing data from HS - we need to eliminate this data because this is our
    output variable and ground truth
    HS_missing_idx = find(isnan(data.HS)); % 6,194 samples with missing HS
    data(HS_missing_idx,:) = [];
    clear HS_missing_idx;

    Fasting_missing_idx = find(isnan(data.Fasting_time_hours));
    data(Fasting_missing_idx,:) = []; % 12 missing Fasting information

    clear Fasting_missing_idx;

    %% Junk and missing data - ALT, AST, ASP, HDL, BMI
    % AST
    % 888 Blank but applicable
    data.AST(data.AST == 888) = NaN;
    % ALT
    % 888 Blank but applicable
    data.ALT(data.ALT == 888) = NaN;
    % ASP
    % 8888 Blank but applicable
    data.ASP(data.ASP == 8888) = NaN;
    % BMI

```

```

        % 8888 was found as junk data via visual examination of data. Although I didnt see this
        on the website for
% NHANES, it is removed because 8888 is not appropriate BMI
data.BMI(data.BMI == 8888) = NaN;
% HDL
data.HDL(data.HDL == 8888) = NaN;

ALT_missing_idx = find(isnan(data.ALT));
data(ALT_missing_idx,:) = [];

AST_missing_idx = find(isnan(data.AST));
data(AST_missing_idx,:) = [];

ASP_missing_idx = find(isnan(data.ASP));
data(ASP_missing_idx,:) = [];

BMI_idx = find(isnan(data.BMI));
data(BMI_idx,:) = [];

HDL_idx = find(isnan(data.HDL));
data(HDL_idx,:) = [];

clear ALT_missing_idx AST_missing_idx ASP_missing_idx BMI_idx HDL_idx;

%% IF there are NaNs in G1PSI, fill them with G2PSI. If both G1PSI and G2PSI are
% NaNs, then delete the sample
for i = 1:size(data,1)
    if(isnan(data.Plasma_glucose_1(i)))
        if(isnan(data.Plasma_glucose_2(i)))
            idx_pg(i) = i;
        else
            data.Plasma_glucose_1(i) = data.Plasma_glucose_2(i);
        end
    end
end
end

data.Plasma_glucose_2 = [];
    Plasma_glucose_idx = find(isnan(data.Plasma_glucose_1)); %25 cases of missing plasma
    glucose samples after combining G1PSI and G2PSI
data(Plasma_glucose_idx,:) = [];

%% IF there are NaNs in I1PSI, fill them with I2PSI. If both I1PSI and I2PSI are
% NaNs, then delete the sample
for i = 1:size(data,1)
    if(isnan(data.Insulin_1(i)))
        if(isnan(data.Insulin_2(i)))

```

```

        idx_in(i) = i;
    else
        data.Insulin_1(i) = data.Insulin_2(i);
    end
end
end

data.Insulin_2 = [];
    Insulin_idx = find(isnan(data.Insulin_1)); %62 cases of missing insulin samples after
    combining I1PSI and I2PSI
data(Insulin_idx,:) = [];

clear idx_pg idx_in Plasma_glucose_idx Insulin_idx

%% Convert Insulin_1 from pmol/L to mU/L by multiplying with 0.144
data.Insulin_1 = data.Insulin_1*0.144;
% Glucose unit is correct for G1PSI so no need to convert

%% Exclude data related to less than permissible fasting time (X hours)
    idx_fasting = find(data.Fasting_time_hours<8); %Identify data that has fasting hours < 8
    hours
data(idx_fasting,:)=[]; %Eliminate data
%% Delete fasting hours variable
data.Fasting_time_hours = [];

%% Calculating Insulin resistance
data(:,end+1) = array2table(zeros(size(data,1),1));
data.Properties.VariableNames{'Var18'} = 'Insulin_resistance';
% Fasting insulin [mU/L] x fasting glucose [mmol/L] / 22.5
data.Insulin_resistance = data.Insulin_1.*data.Plasma_glucose_1/22.5;

%% Delete plasma glucose and insulin resistance
data.Plasma_glucose_1 = [];
data.Insulin_1 = [];

%% Split datasets into HS and non-HS
    data.HS(data.HS == 1) = 0; % 1 is Normal - Mild as per NHANES. Changing it to 0 to
    indicate no risk
    data.HS(data.HS == 2) = 1; % 2 is Moderate - Severe as per NHANES. Changing it to 1
    to indiccate risk

idx_disease = data.HS == 1;
dataset_HS = data(idx_disease,:);

idx_non_disease = data.HS == 0;

```

```

dataset_non_HS = data(idx_non_disease,:);
clear idx_disease idx_non_disease;

%% Split further into Male HS, Non-HS and Female HS, non-HS
dataset_HS_male = dataset_HS(dataset_HS.Sex == 1, :);
dataset_HS_female = dataset_HS(dataset_HS.Sex == 2,:);

dataset_non_HS_male = dataset_non_HS(dataset_non_HS.Sex == 1,:);
dataset_non_HS_female = dataset_non_HS(dataset_non_HS.Sex == 2,:);

%% Apply exclusion criteria for alcohol
% HS and No-HS male exclusion criteria - > 21 drinks/week should be
% excluded
k = 1;
for i = 1: size(dataset_HS_male,1)
    if(dataset_HS_male.Sex(i) == 1 && dataset_HS_male.Drinks_per_day(i) > 3)
        idx_HS_men(k) = i;
        k = k + 1;
    end
end
dataset_HS_male(idx_HS_men,:) = [];
clear k idx_HS_men;

j = 1;
for i = 1: size(dataset_non_HS_male,1)
    if(dataset_non_HS_male.Sex(i) == 1 && dataset_non_HS_male.Drinks_per_day(i) > 3)
        idx_non_HS_men(j) = i;
        j = j + 1;
    end
end
dataset_non_HS_male(idx_non_HS_men,:) = [];
clear j idx_non_HS_men;

% HS and No-HS female exclusion criteria - > 14 drinks/week should be
% excluded
k = 1;
for i = 1: size(dataset_HS_female,1)
    if(dataset_HS_female.Sex(i) == 2 && dataset_HS_female.Drinks_per_day(i) > 2)
        idx_HS_women(k) = i;
        k = k + 1;
    end
end
dataset_HS_female(idx_HS_women,:) = [];
clear k idx_HS_women;

```

```

j = 1;
for i = 1: size(dataset_non_HS_female,1)
    if(dataset_non_HS_female.Sex(i) == 2 && dataset_non_HS_female.Drinks_per_day(i)
        > 2)
        idx_non_HS_women(j) = i;
        j = j + 1;
    end
end
dataset_non_HS_female(idx_non_HS_women,:) = [];
clear j idx_non_HS_women;

%% Delete missing data related to drinks per day
idx_male_HS_drinks = find(isnan(dataset_HS_male.Drinks_per_day));
dataset_HS_male(idx_male_HS_drinks,:) = [];
idx_male_non_HS_drinks = find(isnan(dataset_non_HS_male.Drinks_per_day));
dataset_non_HS_male(idx_male_non_HS_drinks,:) = [];
idx_female_HS_drinks = find(isnan(dataset_HS_female.Drinks_per_day));
dataset_HS_female(idx_female_HS_drinks,:) = [];
idx_female_non_HS_drinks = find(isnan(dataset_non_HS_female.Drinks_per_day));
dataset_non_HS_female(idx_female_non_HS_drinks,:) = [];

    clear idx_female_HS_drinks idx_female_non_HS_drinks idx_male_HS_drinks
    idx_male_non_HS_drinks;

%% Delete alcohol columns from 4 datasets
dataset_HS_male.Alcohol_12_last_year = [];
dataset_HS_male.Alcohol_12_life = [];
dataset_HS_male.Drinks_per_day = [];

dataset_HS_female.Alcohol_12_last_year = [];
dataset_HS_female.Alcohol_12_life = [];
dataset_HS_female.Drinks_per_day = [];

dataset_non_HS_male.Alcohol_12_last_year = [];
dataset_non_HS_male.Alcohol_12_life = [];
dataset_non_HS_male.Drinks_per_day = [];

dataset_non_HS_female.Alcohol_12_last_year = [];
dataset_non_HS_female.Alcohol_12_life = [];
dataset_non_HS_female.Drinks_per_day = [];

%% Delete sex column from all 4 datasets
dataset_HS_male.Sex = [];
dataset_HS_female.Sex = [];
dataset_non_HS_male.Sex = [];
dataset_non_HS_female.Sex = [];

```

```

clear i;

%% Delete heavy metal exposure parameters from all datasets
dataset_HS_male.Lead = [];
dataset_HS_female.Lead = [];
dataset_non_HS_male.Lead = [];
dataset_non_HS_female.Lead = [];

dataset_HS_male.Iron = [];
dataset_HS_female.Iron = [];
dataset_non_HS_male.Iron = [];
dataset_non_HS_female.Iron = [];

dataset_HS_male.Cadmium = [];
dataset_HS_female.Cadmium = [];
dataset_non_HS_male.Cadmium = [];
dataset_non_HS_female.Cadmium = [];

MATLAB Code specific to process MALE datasets
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Created on: 02/17/22
% Input: Datasets from obj_1c_matlab_1a.m (for heavy metal exposure data) or
obj_1c_matlab_1b.m (without heavy metal exposure data)
% Output: Processed data, split into training and test sub datasets
% Author: Ridhi Deo
% File name: Obj1c_matlab_2a.m
% Description: This code was written to process male population data, apply SMOTE, additional
processing and create training and test datasets in a 70:30 ratio, respectively.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% SMOTE
temp_dataset = dataset_HS_male;
temp_dataset(:,[1,8]) = []; % Removing the SEQN and HS columns
size_disease = size(temp_dataset,1); % Measure of number of disease samples
N = 3; % Equivalent of N*100% synthetic sample generation
k = 2; % Setting number of nearest neighbours
num_attrs = size(temp_dataset,2); % Number of variables
new_index = 0; % Variable to keep a count of newly generated synthetic samples
    synthetic_sample_male.N{N} = zeros(size_disease,num_attrs); % Since we are
    generating N*100% synthetic, this value is N{2}
    nn_array = zeros(size_disease,k+1); % To keep a list of nearest neighbours for each
    sample
nn_values = zeros(size_disease,k+1);
R = zeros(num_attrs,1); % Range of continuous variables is represented by the array R
    temp_range = table2array(temp_dataset); % Temporary conversions to array - for
    computational ease. This is essential dataset_HS_male

```



```

temp_dist = table2array(temp_dataset);
    distance = zeros(size_disease,size_disease); %Preallocating a matrix to store all the
    distances
for i = 1:num_attrs %Calculating ranges
    R(i,1) = (max(temp_range(:,i)) - min(temp_range(:,i)));
end
% Finding the k-nn
    [nn_array,nn_values] = knnsearch(temp_dist, temp_dist,'K',k+1); % the first nn will be
    itself so we will need to remove that

nn_array(:,1) = []; %Remove the first one because it is the same sample
nn_values(:,1) = []; %First nn is itself so distance is 0
while (N~=0) %To perform N*100% synthetic sampling
    for i = 1:size_disease
        for attr=1:num_attrs
            nn = randi([1 k],1); % Randomly choose the nearest neighbor
            dif = temp_dist(nn_array(i,nn),attr) - temp_dist(i,attr);
            gap = 0 + rand(1,1);
            synthetic_sample_male.N{N}(i,attr) = temp_dist(i,attr) + (gap*dif); %Synthetic
            continous attributes
        end
    end
    N = N-1; %To avoid infinite loops
end

    total_synthetic_sample_males =
    [synthetic_sample_male.N{1};synthetic_sample_male.N{2};synthetic_sample_male.N{3
    }];
total_synthetic_sample_males = array2table(total_synthetic_sample_males);
    total_synthetic_sample_males.Properties.VariableNames =
    temp_dataset.Properties.VariableNames;
    hybrid_disease_male = [total_synthetic_sample_males; temp_dataset]; %Hybrid =
    synthetic + disease
hybrid_disease_male.HS = ones(size(hybrid_disease_male,1),1);

%% removing seqn
dataset_non_HS_male.SEQN = [];
dataset_HS_male.SEQN = [];

%% Based on American Liver Foundation video - Vicki Shah
% Normal value for ALT: 10 - 55, but actually 20.
% AST: 9 - 32, but prefer 20
% ASP: 30 - 100, also based on age
% NAFLD: AST and ALT are up to less than 4 times the ULN
%% Data normalization

```

```

%% AASLD: male: ALT: 19 - 25 IU/L
% Using 25IU/L as the ULN for male based on the AASLD guidelines
% Pre-sets
% Pre-sets
ULN_ALT = 33;
ULN_AST = 30;
ULN_BMI = 25;
ULN_HDL = 1;
%% Creating a new % variables
ALT_percent = zeros(size(dataset_HS_male, 1),1);
dataset_HS_male(:,end+1) = array2table(ALT_percent);
dataset_HS_male.Properties.VariableNames{'Var9'} = 'ALT_percent';

ALT_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(ALT_percent);
dataset_non_HS_male.Properties.VariableNames{'Var9'} = 'ALT_percent';

AST_percent = zeros(size(dataset_HS_male, 1),1);
dataset_HS_male(:,end+1) = array2table(AST_percent);
dataset_HS_male.Properties.VariableNames{'Var10'} = 'AST_percent';

AST_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(AST_percent);
dataset_non_HS_male.Properties.VariableNames{'Var10'} = 'AST_percent';

BMI_percent = zeros(size(dataset_HS_male, 1),1);
dataset_HS_male(:,end+1) = array2table(BMI_percent);
dataset_HS_male.Properties.VariableNames{'Var11'} = 'BMI_percent';

BMI_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(BMI_percent);
dataset_non_HS_male.Properties.VariableNames{'Var11'} = 'BMI_percent';

HDL_percent = zeros(size(dataset_HS_male, 1),1);
dataset_HS_male(:,end+1) = array2table(HDL_percent);
dataset_HS_male.Properties.VariableNames{'Var12'} = 'HDL_percent';

HDL_percent = zeros(size(dataset_non_HS_male, 1),1);
dataset_non_HS_male(:,end+1) = array2table(HDL_percent);
dataset_non_HS_male.Properties.VariableNames{'Var12'} = 'HDL_percent';
%% Normalization equations
for i = 1: size(dataset_HS_male,1)
    dataset_HS_male.ALT_percent(i) = ((dataset_HS_male.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;

```

```

        dataset_HS_male.AST_percent(i) = ((dataset_HS_male.AST(i) -
        ULN_AST)/ULN_AST)*100;
        dataset_HS_male.BMI_percent(i) = ((dataset_HS_male.BMI(i) -
        ULN_BMI)/ULN_BMI)*100;
        dataset_HS_male.HDL_percent(i) = ((dataset_HS_male.HDL(i) -
        ULN_HDL)/ULN_HDL)*100;

end
clear i;
for i = 1: size(dataset_non_HS_male,1)
    dataset_non_HS_male.ALT_percent(i) = ((dataset_non_HS_male.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;
    dataset_non_HS_male.AST_percent(i) = ((dataset_non_HS_male.AST(i) -
    ULN_AST)/ULN_AST)*100;
    dataset_non_HS_male.BMI_percent(i) = ((dataset_non_HS_male.BMI(i) -
    ULN_BMI)/ULN_BMI)*100;
    dataset_non_HS_male.HDL_percent(i) = ((dataset_non_HS_male.HDL(i) -
    ULN_HDL)/ULN_HDL)*100;

end

% Converting negative to 0

for i = 1: size(dataset_HS_male,1)
    if(dataset_HS_male.ALT_percent(i) <= 0)
        dataset_HS_male.ALT_percent(i) = 0;
    end
    if(dataset_HS_male.AST_percent(i) <= 0)
        dataset_HS_male.AST_percent(i) = 0;
    end
    if(dataset_HS_male.BMI_percent(i) <= 0)
        dataset_HS_male.BMI_percent(i) = 0;
    end
    if(dataset_HS_male.HDL_percent(i) >= 0)
        dataset_HS_male.HDL_percent(i) = 0;
    end
end

for i = 1: size(dataset_non_HS_male,1)
    if(dataset_non_HS_male.ALT_percent(i) <= 0)
        dataset_non_HS_male.ALT_percent(i) = 0;
    end
    if(dataset_non_HS_male.AST_percent(i) <= 0)
        dataset_non_HS_male.AST_percent(i) = 0;
    end
end

```

```

    if(dataset_non_HS_male.BMI_percent(i) <= 0)
        dataset_non_HS_male.BMI_percent(i) = 0;
    end
    if(dataset_non_HS_male.HDL_percent(i) >= 0)
        dataset_non_HS_male.HDL_percent(i) = 0;
    end
end
% %% Remove SEQN

%% Randomly select samples without replacement
% Note that MATLAB's datasample function has replace = true as default

    dataset_non_HS_male_reduced = datasample(dataset_non_HS_male,
        size(dataset_HS_male,1), 'Replace', false);

%% Split into training and test
Q = size(dataset_HS_male,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,~,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_disease_male = dataset_HS_male(trainInd,:);
test_disease_male = dataset_HS_male(testInd,:);

Q = size(dataset_non_HS_male_reduced,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,valInd,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_no_disease_male = dataset_non_HS_male_reduced(trainInd,:);
test_no_disease_male = dataset_non_HS_male_reduced(testInd,:);

training_male = [train_disease_male; train_no_disease_male];
test_male = [test_disease_male; test_no_disease_male];
test_male = test_male(randperm(size(test_male,1)),:);
training_male = training_male(randperm(size(training_male,1)),:);

%% Reorder training and test datasets to have HS as the end variable
training_male = [training_male(:,1:6) training_male(:,8:12) training_male(:, 7)];
test_male = [test_male(:,1:6) test_male(:,8:12) test_male(:,7)];
MATLAB Code specific to process feMALE datasets
% % % % % % %
% Created on: 02/17/22
    % Input: Datasets from obj_1c_matlab_1a.m (for heavy metal exposure data) or
    obj_1c_matlab_1b.m (without heavy metal exposure data)
% Output: Processed data, split into training and test sub datasets

```

```

% Author: Ridhi Deo
% File name: Obj1c_matlab_2b.m
% Description: This code was written to process female population data, apply SMOTE,
additional processing and create training and test datasets in a 70:30 ratio, respectively.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% SMOTE
temp_dataset = dataset_HS_female;
temp_dataset(:,[1,8]) = []; % Removing the SEQN and HS columns
size_disease = size(temp_dataset,1); % Measure of number of disease samples
N = 3; % Equivalent of N*100% synthetic sample generation
k = 2; % Setting number of nearest neighbours
num_attrs = size(temp_dataset,2); % Number of variables
new_index = 0; % Variable to keep a count of newly generated synthetic samples
    synthetic_sample_female.N{N} = zeros(size_disease,num_attrs); % Since we are
    generating N*100% synthetic, this value is N{2}
    nn_array = zeros(size_disease,k+1); % To keep a list of nearest neighbours for each
    sample
nn_values = zeros(size_disease,k+1);
R = zeros(num_attrs,1); % Range of continuous variables is represented by the array R
    temp_range = table2array(temp_dataset); % Temporary conversions to array - for
    computational ease. This is essential dataset_HS_female
temp_dist = table2array(temp_dataset);
    distance = zeros(size_disease,size_disease); % Preallocating a matrix to store all the
    distances
for i = 1:num_attrs % Calculating ranges
    R(i,1) = (max(temp_range(:,i)) - min(temp_range(:,i)));
end
% Finding the k-nn
    [nn_array,nn_values] = knnsearch(temp_dist, temp_dist,'K',k+1); % the first nn will be
    itself so we will need to remove that

nn_array(:,1) = []; % Remove the first one because it is the same sample
nn_values(:,1) = []; % First nn is itself so distance is 0
while (N~=0) % To perform N*100% synthetic sampling
    for i = 1:size_disease
        for attr=1:num_attrs
            nn = randi([1 k],1); % Randomly choose the nearest neighbor
            dif = temp_dist(nn_array(i,nn),attr) - temp_dist(i,attr);
            gap = 0 + rand(1,1);
            synthetic_sample_female.N{N}(i,attr) = temp_dist(i,attr) + (gap*dif); % Synthetic
            continuous attributes
        end
    end
end

```

```

N = N-1; %To avoid infinite loops
end

total_synthetic_sample_females =
[synthetic_sample_female.N{1};synthetic_sample_female.N{2};synthetic_sample_femal
e.N{3}];
total_synthetic_sample_females = array2table(total_synthetic_sample_females);
total_synthetic_sample_females.Properties.VariableNames =
temp_dataset.Properties.VariableNames;
hybrid_disease_female = [total_synthetic_sample_females; temp_dataset]; %Hybrid =
synthetic + disease
hybrid_disease_female.HS = ones(size(hybrid_disease_female,1),1);

%% removing seqn
dataset_non_HS_female.SEQN = [];
dataset_HS_female.SEQN = [];

%% Based on American Liver Foundation video - Vicki Shah
% Normal value for ALT: 10 - 55, but actually 20.
% AST: 9 - 32, but prefer 20
% ASP: 30 - 100, also based on age
% NAFLD: AST and ALT are up to less than 4 times the ULN
%% Data normalization
%% AASLD: Female: ALT: 19 - 25 IU/L
% Using 25IU/L as the ULN for female based on the AASLD guidelines
% Pre-sets
ULN_ALT = 25;
ULN_AST = 20;

ULN_BMI = 25;

%ULN_Cadmium = ;
ULN_HDL = 1.3; %https://www.mayoclinic.org/diseases-conditions/high-blood-cholesterol/in-depth/hdl-cholesterol/art-20046388

%% Creating a new % variables
ALT_percent = zeros(size(dataset_HS_female, 1),1);
dataset_HS_female(:,end+1) = array2table(ALT_percent);
dataset_HS_female.Properties.VariableNames{'Var9'} = 'ALT_percent';

ALT_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(ALT_percent);
dataset_non_HS_female.Properties.VariableNames{'Var9'} = 'ALT_percent';

AST_percent = zeros(size(dataset_HS_female, 1),1);

```

```

dataset_HS_female(:,end+1) = array2table(AST_percent);
dataset_HS_female.Properties.VariableNames{'Var10'} = 'AST_percent';

AST_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(AST_percent);
dataset_non_HS_female.Properties.VariableNames{'Var10'} = 'AST_percent';

BMI_percent = zeros(size(dataset_HS_female, 1),1);
dataset_HS_female(:,end+1) = array2table(BMI_percent);
dataset_HS_female.Properties.VariableNames{'Var11'} = 'BMI_percent';

BMI_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(BMI_percent);
dataset_non_HS_female.Properties.VariableNames{'Var11'} = 'BMI_percent';

HDL_percent = zeros(size(dataset_HS_female, 1),1);
dataset_HS_female(:,end+1) = array2table(HDL_percent);
dataset_HS_female.Properties.VariableNames{'Var12'} = 'HDL_percent';

HDL_percent = zeros(size(dataset_non_HS_female, 1),1);
dataset_non_HS_female(:,end+1) = array2table(HDL_percent);
dataset_non_HS_female.Properties.VariableNames{'Var12'} = 'HDL_percent';
%% Normalization equations
for i = 1: size(dataset_HS_female,1)
    dataset_HS_female.ALT_percent(i) = ((dataset_HS_female.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;
    dataset_HS_female.AST_percent(i) = ((dataset_HS_female.AST(i) -
    ULN_AST)/ULN_AST)*100;
    dataset_HS_female.BMI_percent(i) = ((dataset_HS_female.BMI(i) -
    ULN_BMI)/ULN_BMI)*100;
    dataset_HS_female.HDL_percent(i) = ((dataset_HS_female.HDL(i) -
    ULN_HDL)/ULN_HDL)*100;

end
clear i;
for i = 1: size(dataset_non_HS_female,1)
    dataset_non_HS_female.ALT_percent(i) = ((dataset_non_HS_female.ALT(i) -
    ULN_ALT)/ULN_ALT)*100;
    dataset_non_HS_female.AST_percent(i) = ((dataset_non_HS_female.AST(i) -
    ULN_AST)/ULN_AST)*100;
    dataset_non_HS_female.BMI_percent(i) = ((dataset_non_HS_female.BMI(i) -
    ULN_BMI)/ULN_BMI)*100;
    dataset_non_HS_female.HDL_percent(i) = ((dataset_non_HS_female.HDL(i) -
    ULN_HDL)/ULN_HDL)*100;

```

```

end

% Converting negative to 0

for i = 1: size(dataset_HS_female,1)
    if(dataset_HS_female.ALT_percent(i) <= 0)
        dataset_HS_female.ALT_percent(i) = 0;
    end
    if(dataset_HS_female.AST_percent(i) <= 0)
        dataset_HS_female.AST_percent(i) = 0;
    end
    if(dataset_HS_female.BMI_percent(i) <= 0)
        dataset_HS_female.BMI_percent(i) = 0;
    end
    if(dataset_HS_female.HDL_percent(i) >= 0)
        dataset_HS_female.HDL_percent(i) = 0;
    end
end

for i = 1: size(dataset_non_HS_female,1)
    if(dataset_non_HS_female.ALT_percent(i) <= 0)
        dataset_non_HS_female.ALT_percent(i) = 0;
    end
    if(dataset_non_HS_female.AST_percent(i) <= 0)
        dataset_non_HS_female.AST_percent(i) = 0;
    end
    if(dataset_non_HS_female.BMI_percent(i) <= 0)
        dataset_non_HS_female.BMI_percent(i) = 0;
    end
    if(dataset_non_HS_female.HDL_percent(i) >= 0)
        dataset_non_HS_female.HDL_percent(i) = 0;
    end
end

% %% Remove SEQN

%% Randomly select samples without replacement
% Note that MATLAB's datasample function has replace = true as default

    dataset_non_HS_female_reduced = datasample(dataset_non_HS_female,
        size(dataset_HS_female,1), 'Replace', false);

%% Split into training and test
Q = size(dataset_HS_female,1);
valRatio = 0;
trainRatio = 0.70;

```



```

testRatio = 0.30;
[trainInd,~,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_disease_female = dataset_HS_female(trainInd,:);
test_disease_female = dataset_HS_female(testInd,:);

Q = size(dataset_non_HS_female_reduced,1);
valRatio = 0;
trainRatio = 0.70;
testRatio = 0.30;
[trainInd,valInd,testInd] = dividerand(Q,trainRatio,valRatio,testRatio);
train_no_disease_female = dataset_non_HS_female_reduced(trainInd,:);
test_no_disease_female = dataset_non_HS_female_reduced(testInd,:);

training_female = [train_disease_female; train_no_disease_female];
test_female = [test_disease_female; test_no_disease_female];
test_female = test_female(randperm(size(test_female,1)),:);
training_female = training_female(randperm(size(training_female,1)),:);

%% Reorder training and test datasets to have HS as the end variable
training_female = [training_female(:,1:6) training_female(:,8:12) training_female(:, 7)];
test_female = [test_female(:,1:6) test_female(:,8:12) test_female(:,7)];

```

## 5. MATLAB CODE TO TRAIN THE MODELS

```

%%%%%%%%%%
% Created on: 02/17/22
% Input: Datasets from obj_1c_matlab_2a.m or obj_1c_matlab_2b.m
% Output: Trained models
% Author: Ridhi Deo
% File name: Obj1c_matlab_3.m
% Description: This code was written to train ML models. This code internally calls several
other functions to train specific models.
%%%%%%%%%%

close all;
test = test_male; % Need to change this depending on male/female
training = training_male; % Need to change this depending on male/female

%% Model 1: fine tree
[mod_1, train_acc_1] = finetree2(training); % Training the model using training set
yfit_1 = mod_1.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_1(:,2) = table2array(test(:,end)); % Ground truth
g1 = yfit_1(:,2)'; % Transposed values of Known values - Ground Truth

```

```

g2 = yfit_1(:,1)'; % Transposed values of predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Fine Tree')
[tpr_1, fpr_1,~] = roc(g1, g2); % Extracting the true-positive and false-positive rates
sens_1 = tpr_1(1,2); % Calculating sensitivity
spec_1 = 1- fpr_1(1,2); % Calculating specificity
[X,Y,~,AUC_1] = perfcurve(g1,g2,'1'); % Extracting values to plot the AUC curve with
the AUC value
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Fine Tree')
txt = ['AUC for Fine Tree is ',num2str(AUC_1)];
text(0.5,0.9,txt)
clear X Y;
cp_1 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_1_accuracy = cp_1.CorrectRate;

%% Model 2: logistic regression
[mod_2, train_acc_2] = logisticregression2(training); % Training the model using
training set
yfit_2 = mod_2.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_2(:,2) = table2array(test(:,end));
g1 = yfit_2(:,2)'; % Known values - Ground Truth
g2 = yfit_2(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('logistic regression')
[tpr_2, fpr_2,~] = roc(g1, g2);
sens_2 = tpr_2(1,2);
spec_2 = 1- fpr_2(1,2);
[X,Y,~,AUC_2] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('logistic regression')
txt = ['AUC for logistic regression is ',num2str(AUC_2)];
text(0.5,0.9,txt)
clear X Y;
cp_2 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_2_accuracy = cp_2.CorrectRate;

%% Model 3: linear svm
[mod_3, train_acc_3] = linearsvm2(training); % Training the model using training set
yfit_3 = mod_3.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_3(:,2) = table2array(test(:,end));
g1 = yfit_3(:,2)'; % Known values - Ground Truth
g2 = yfit_3(:,1)'; % predicted values
figure %Plotting confusion matrix

```

```

plotconfusion(g1,g2), title('linear svm')
[tpr_3, fpr_3,~] = roc(g1, g2);
sens_3 = tpr_3(1,2);
spec_3 = 1- fpr_3(1,2);
[X,Y,~,AUC_3] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('linear svm')
txt = ['AUC for linear svm is ',num2str(AUC_3)];
text(0.5,0.9,txt)
clear X Y;
cp_3 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_3_accuracy = cp_3.CorrectRate;

%% Model 4: quadratic svm
[mod_4, train_acc_4] = quadraticsvm2(training); % Training the model using training set
yfit_4 = mod_4.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_4(:,2) = table2array(test(:,end));
g1 = yfit_4(:,2)'; %Known values - Ground Truth
g2 = yfit_4(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('quadratic svm')
[tpr_4, fpr_4,~] = roc(g1, g2);
sens_4 = tpr_4(1,2);
spec_4 = 1- fpr_4(1,2);
[X,Y,~,AUC_4] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('quadratic svm')
txt = ['AUC for quadratic svm is ',num2str(AUC_4)];
text(0.5,0.9,txt)
clear X Y;
cp_4 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_4_accuracy = cp_4.CorrectRate;

%% Model 5: fine gaussian svm
[mod_5, train_acc_5] = finegaussiansvm2(training); % Training the model using training
set
yfit_5 = mod_5.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_5(:,2) = table2array(test(:,end));
g1 = yfit_5(:,2)'; %Known values - Ground Truth
g2 = yfit_5(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('fine gaussian svm')
[tpr_5, fpr_5,~] = roc(g1, g2);

```

```

sens_5 = tpr_5(1,2);
spec_5 = 1- fpr_5(1,2);
[X,Y,~,AUC_5] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('fine gaussian svm')
txt = ['AUC for fine gaussian svm is ',num2str(AUC_5)];
text(0.5,0.9,txt)
clear X Y;
cp_5 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_5_accuracy = cp_5.CorrectRate;

%% Model 6: medium gaussian svm
[mod_6, train_acc_6] = mediumgaussiansvm2(training); % Training the model using
training set
yfit_6 = mod_6.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_6(:,2) = table2array(test(:,end));
g1 = yfit_6(:,2)'; %Known values - Ground Truth
g2 = yfit_6(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('medium gaussian svm')
[tpr_6, fpr_6,~] = roc(g1, g2);
sens_6 = tpr_6(1,2);
spec_6 = 1- fpr_6(1,2);
[X,Y,~,AUC_6] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('medium gaussian svm')
txt = ['AUC for medium gaussian svm is ',num2str(AUC_6)];
text(0.5,0.9,txt)
clear X Y;
cp_6 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_6_accuracy = cp_6.CorrectRate;

%% Model 7: coarse gaussian svm
[mod_7, train_acc_7] = coarsegaussiansvm2(training); % Training the model using
training set
yfit_7 = mod_7.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_7(:,2) = table2array(test(:,end));
g1 = yfit_7(:,2)'; %Known values - Ground Truth
g2 = yfit_7(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('coarse gaussian svm')
[tpr_7, fpr_7,~] = roc(g1, g2);
sens_7 = tpr_7(1,2);
spec_7 = 1- fpr_7(1,2);

```

```

[X,Y,~,AUC_7] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('coarse gaussian svm')
txt = ['AUC for coarse gaussian svm is ',num2str(AUC_7)];
text(0.5,0.9,txt)
clear X Y;
cp_7 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_7_accuracy = cp_7.CorrectRate;

%% Model 8: fine knn
[mod_8, train_acc_8] = fineknn2(training); % Training the model using training set
yfit_8 = mod_8.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_8(:,2) = table2array(test(:,end));
g1 = yfit_8(:,2)'; %Known values - Ground Truth
g2 = yfit_8(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('fine knn')
[tpr_8, fpr_8,~] = roc(g1, g2);
sens_8 = tpr_8(1,2);
spec_8 = 1- fpr_8(1,2);
[X,Y,~,AUC_8] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('fine knn')
txt = ['AUC for fine knn is ',num2str(AUC_8)];
text(0.5,0.9,txt)
clear X Y;
cp_8 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_8_accuracy = cp_8.CorrectRate;

%% Model 9: Medium knn
[mod_9, train_acc_9] = mediumknn2(training); % Training the model using training set
yfit_9 = mod_9.predictFcn(test(:,1:end-1)); % Predicting values from the trained model
using the test dataset
yfit_9(:,2) = table2array(test(:,end));
g1 = yfit_9(:,2)'; %Known values - Ground Truth
g2 = yfit_9(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Medium knn')
[tpr_9, fpr_9,~] = roc(g1, g2);
sens_9 = tpr_9(1,2);
spec_9 = 1- fpr_9(1,2);
[X,Y,~,AUC_9] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph

```

```

plot(X,Y), title('Medium knn')
txt = ['AUC for Medium knn is ',num2str(AUC_9)];
text(0.5,0.9,txt)
clear X Y;
cp_9 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_9_accuracy = cp_9.CorrectRate;

%% Model 10: Coarse knn
[mod_10, train_acc_10] = coarseknn2(training); % Training the model using training set
yfit_10 = mod_10.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_10(:,2) = table2array(test(:,end));
g1 = yfit_10(:,2)'; % Known values - Ground Truth
g2 = yfit_10(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Coarse knn')
[tpr_10, fpr_10,~] = roc(g1, g2);
sens_10 = tpr_10(1,2);
spec_10 = 1- fpr_10(1,2);
[X,Y,~,AUC_10] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Coarse knn')
txt = ['AUC for Coarse knn is ',num2str(AUC_10)];
text(0.5,0.9,txt)
clear X Y;
cp_10 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_10_accuracy = cp_10.CorrectRate;

%% Model 11: Cosine knn
[mod_11, train_acc_11] = cosineknn2(training); % Training the model using training set
yfit_11 = mod_11.predictFcn(test(:,1:end-1));% Predicting values from the trained model
using the test dataset
yfit_11(:,2) = table2array(test(:,end));
g1 = yfit_11(:,2)'; % Known values - Ground Truth
g2 = yfit_11(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Cosine knn')
[tpr_11, fpr_11,~] = roc(g1, g2);
sens_11 = tpr_11(1,2);
spec_11 = 1- fpr_11(1,2);
[X,Y,~,AUC_11] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Cosine knn')
txt = ['AUC for Cosine knn is ',num2str(AUC_11)];
text(0.5,0.9,txt)
clear X Y;

```

```

cp_11 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_11_accuracy = cp_11.CorrectRate;

%% Model 12: Cubic knn
[mod_12, train_acc_12] = cubicknn2(training); % Training the model using training set
yfit_12 = mod_12.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_12(:,2) = table2array(test(:,end));
g1 = yfit_12(:,2)'; % Known values - Ground Truth
g2 = yfit_12(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Cubic knn')
[tpr_12, fpr_12,~] = roc(g1, g2);
sens_12 = tpr_12(1,2);
spec_12 = 1- fpr_12(1,2);
[X,Y,~,AUC_12] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Cubic knn')
txt = ['AUC for Cubic knn is ',num2str(AUC_12)];
text(0.5,0.9,txt)
clear X Y;
cp_12 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_12_accuracy = cp_12.CorrectRate;

%% Model 13: Weighted knn
[mod_13, train_acc_13] = weightedknn2(training); % Training the model using training
set
yfit_13 = mod_13.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_13(:,2) = table2array(test(:,end));
g1 = yfit_13(:,2)'; % Known values - Ground Truth
g2 = yfit_13(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Weighted knn')
[tpr_13, fpr_13,~] = roc(g1, g2);
sens_13 = tpr_13(1,2);
spec_13 = 1- fpr_13(1,2);
[X,Y,~,AUC_13] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Weighted knn')
txt = ['AUC for Weighted knn is ',num2str(AUC_13)];
text(0.5,0.9,txt)
clear X Y;
cp_13 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_13_accuracy = cp_13.CorrectRate;

```

```

%% Model 14: Boosted Trees
[mod_14, train_acc_14] = boostedtrees2(training); % Training the model using training
set
yfit_14 = mod_14.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_14(:,2) = table2array(test(:,end));
g1 = yfit_14(:,2)'; %Known values - Ground Truth
g2 = yfit_14(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Boosted Trees')
[tpr_14, fpr_14,~] = roc(g1, g2);
sens_14 = tpr_14(1,2);
spec_14 = 1- fpr_14(1,2);
[X,Y,~,AUC_14] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Boosted Trees')
txt = ['AUC for Boosted Trees is ',num2str(AUC_14)];
text(0.5,0.9,txt)
clear X Y;
cp_14 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_14_accuracy = cp_14.CorrectRate;

%% Model 15: Bagged Trees
[mod_15, train_acc_15] = baggedtrees2(training); % Training the model using training
set
yfit_15 = mod_15.predictFcn(test(:,1:end-1)); % Predicting values from the trained
model using the test dataset
yfit_15(:,2) = table2array(test(:,end));
g1 = yfit_15(:,2)'; %Known values - Ground Truth
g2 = yfit_15(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Bagged Trees')
[tpr_15, fpr_15,~] = roc(g1, g2);
sens_15 = tpr_15(1,2);
spec_15 = 1- fpr_15(1,2);
[X,Y,~,AUC_15] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Bagged Trees')
txt = ['AUC for Bagged Trees is ',num2str(AUC_15)];
text(0.5,0.9,txt)
clear X Y;
cp_15 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_15_accuracy = cp_15.CorrectRate;

%% Model 16: Subspace Discriminant

```



```

[mod_16, train_acc_16] = subspacedisc2(training); % Training the model using training
set
yfit_16 = mod_16.predictFcn(test(:,1:end-1));
yfit_16(:,2) = table2array(test(:,end));
g1 = yfit_16(:,2)'; %Known values - Ground Truth
g2 = yfit_16(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('Subspace Disc')
[tpr_16, fpr_16,~] = roc(g1, g2);
sens_16 = tpr_16(1,2);
spec_16 = 1- fpr_16(1,2);
[X,Y,~,AUC_16] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('Subspace Disc')
txt = ['AUC for Subspace Disc is ',num2str(AUC_16)];
text(0.5,0.9,txt)
clear X Y;
cp_16 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_16_accuracy = cp_16.CorrectRate;

%% Model 17: RUS Boosted trees
[mod_17, train_acc_17] = rusboostedtrees2(training); % Training the model using
training set
yfit_17 = mod_17.predictFcn(test(:,1:end-1));
yfit_17(:,2) = table2array(test(:,end));
g1 = yfit_17(:,2)'; %Known values - Ground Truth
g2 = yfit_17(:,1)'; % predicted values
figure %Plotting confusion matrix
plotconfusion(g1,g2), title('RUS Boosted trees')
[tpr_17, fpr_17,~] = roc(g1, g2);
sens_17 = tpr_17(1,2);
spec_17 = 1- fpr_17(1,2);
[X,Y,~,AUC_17] = perfcurve(g1,g2,'1');
figure %Plotting ROC with AUC value printed on the graph
plot(X,Y), title('RUS Boosted trees')
txt = ['AUC for RUS Boosted trees is ',num2str(AUC_17)];
text(0.5,0.9,txt)
clear X Y;
cp_17 = classperf(g1,g2); % Extracting the accuracy of the testing dataset
cp_17_accuracy = cp_17.CorrectRate;

%% Display results in a table
Model =
{'Fine_Tree';'Logistic_Regression';'Linear_SVM';'Quadratic_SVM';'Fine_Gaussian_SV
M';...

```

```

'Medium_Gaussian_SVM'; 'Coarse_Gaussian_SVM'; 'Fine_KNN'; 'Medium_KNN'; 'Coarse_KNN'; ...
'Cosine_KNN'; 'Cubic_KNN'; 'Weighted_KNN'; 'Ensemble_Boosted'; 'Ensemble_Bagged'; ..
'Ensemble_Subspace_Disc'; 'Ensemble_RUS_Boosted_Trees'});

Training_Acc = [train_acc_1; train_acc_2; train_acc_3; train_acc_4; train_acc_5; ...
train_acc_6; train_acc_7; train_acc_8; train_acc_9; train_acc_10; train_acc_11; ...
train_acc_12; train_acc_13; train_acc_14; train_acc_15; train_acc_16; train_acc_17];
Test_Acc = [cp_1_accuracy; cp_2_accuracy; cp_3_accuracy; cp_4_accuracy;
cp_5_accuracy; ...
cp_6_accuracy; cp_7_accuracy; cp_8_accuracy; cp_9_accuracy; cp_10_accuracy;
cp_11_accuracy; ...
cp_12_accuracy; cp_13_accuracy; cp_14_accuracy; cp_15_accuracy; cp_16_accuracy;
cp_17_accuracy];
AUC = [AUC_1; AUC_2; AUC_3; AUC_4; AUC_5; ...
AUC_6; AUC_7; AUC_8; AUC_9; AUC_10; AUC_11; ...
AUC_12; AUC_13; AUC_14; AUC_15; AUC_16; AUC_17];
Sensitivity = [sens_1; sens_2; sens_3; sens_4; sens_5; ...
sens_6; sens_7; sens_8; sens_9; sens_10; sens_11; ...
sens_12; sens_13; sens_14; sens_15; sens_16; sens_17];
Specificity = [spec_1; spec_2; spec_3; spec_4; spec_5; ...
spec_6; spec_7; spec_8; spec_9; spec_10; spec_11; ...
spec_12; spec_13; spec_14; spec_15; spec_16; spec_17];
Results = table(Model, Training_Acc, Test_Acc, AUC, Sensitivity, Specificity);

```

## A. CODE TO TRAIN LOGISTIC REGRESSION

```

%%%%%%%%%%
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained Logistic Regression Model
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3a.m
% Description: This code was written to train the logistic regression model.
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = logisticregression2(trainingData)

% Auto-generated by MATLAB on 17-Feb-2022 13:02:57

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;

```

```

    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,
        false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
% For logistic regression, the response values must be converted to zeros
% and ones because the responses are assumed to follow a binomial
% distribution.
% 1 or true = 'successful' class
% 0 or false = 'failure' class
% NaN - missing response.
successClass = double(1);
failureClass = double(0);
% Compute the majority response class. If there is a NaN-prediction from
% fitglm, convert NaN to this majority class label.
numSuccess = sum(response == successClass);
numFailure = sum(response == failureClass);
if numSuccess > numFailure
    missingClass = successClass;
else
    missingClass = failureClass;
end
successFailureAndMissingClasses = [successClass; failureClass; missingClass];
isMissing = isnan(response);
zeroOneResponse = double(ismember(response, successClass));
zeroOneResponse(isMissing) = NaN;
% Prepare input arguments to fitglm.
concatenatedPredictorsAndResponse = [predictors, table(zeroOneResponse)];
% Train using fitglm.
GeneralizedLinearModel = fitglm(...
    concatenatedPredictorsAndResponse, ...
    'Distribution', 'binomial', ...
    'link', 'logit');

% Convert predicted probabilities to predicted class labels and scores.
convertSuccessProbsToPredictions = @(p) successFailureAndMissingClasses(
    ~isnan(p).*( (p<0.5) + 1 ) + isnan(p)*3 );
returnMultipleValuesFcn = @(varargin) varargin{1:max(1,nargout)};
scoresFcn = @(p) [1-p, p];
predictionsAndScoresFcn = @(p) returnMultipleValuesFcn( con
    vertSuccessProbsToPredictions(p), scoresFcn(p) );

```

```

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
logisticRegressionPredictFcn = @(x) predictionsAndScoresFcn( predict
    ict(GeneralizedLinearModel, x) );
trainedClassifier.predictFcn = @(x) log
    isticRegressionPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST', 'AST_percent',
    'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
trainedClassifier.GeneralizedLinearModel = GeneralizedLinearModel;
trainedClassifier.SuccessClass = successClass;
trainedClassifier.FailureClass = failureClass;
trainedClassifier.MissingClass = missingClass;
trainedClassifier.ClassNames = {successClass; failureClass};
trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
    original training data. \nAdditional variables are ignored. \n \nFor more information, see
    <a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
    "appclassification_exportmodeltoworkspace")">How to predict using an exported
    model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
        false];

% Perform cross-validation
KFolds = 10;
cvp = cvpartition(response, 'KFold', KFolds);
% Initialize the predictions to the proper sizes
validationPredictions = response;
numObservations = size(predictors, 1);
numClasses = 2;
validationScores = NaN(numObservations, numClasses);
for fold = 1:KFolds

```

```

trainingPredictors = predictors(cvp.training(fold), :);
trainingResponse = response(cvp.training(fold), :);
foldIsCategoricalPredictor = isCategoricalPredictor;

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
% For logistic regression, the response values must be converted to zeros
% and ones because the responses are assumed to follow a binomial
% distribution.
% 1 or true = 'successful' class
% 0 or false = 'failure' class
% NaN - missing response.
successClass = double(1);
failureClass = double(0);
% Compute the majority response class. If there is a NaN-prediction from
% fitglm, convert NaN to this majority class label.
numSuccess = sum(trainingResponse == successClass);
numFailure = sum(trainingResponse == failureClass);
if numSuccess > numFailure
    missingClass = successClass;
else
    missingClass = failureClass;
end
successFailureAndMissingClasses = [successClass; failureClass; missingClass];
isMissing = isnan(trainingResponse);
zeroOneResponse = double(ismember(trainingResponse, successClass));
zeroOneResponse(isMissing) = NaN;
% Prepare input arguments to fitglm.
concatenatedPredictorsAndResponse = [trainingPredictors, table(zeroOneResponse)];
% Train using fitglm.
GeneralizedLinearModel = fitglm(...
    concatenatedPredictorsAndResponse, ...
    'Distribution', 'binomial', ...
    'link', 'logit');

% Convert predicted probabilities to predicted class labels and scores.
convertSuccessProbsToPredictions = @(p) successFailureAndMissingClasses( ~isnan(p).*(
(p<0.5) + 1 ) + isnan(p)*3 );
returnMultipleValuesFcn = @(varargin) varargin{ 1:max(1,nargout) };
scoresFcn = @(p) [1-p, p];
predictionsAndScoresFcn = @(p) returnMultipleValuesFcn(
convertSuccessProbsToPredictions(p), scoresFcn(p) );

% Create the result struct with predict function
logisticRegressionPredictFcn = @(x) predictionsAndScoresFcn(
predict(GeneralizedLinearModel, x) );

```

```

validationPredictFcn = @(x) logisticRegressionPredictFcn(x);

% Add additional fields to the result struct

% Compute validation predictions
validationPredictors = predictors(cvp.test(fold), :);
[foldPredictions, foldScores] = validationPredictFcn(validationPredictors);

% Store predictions in the original order
validationPredictions(cvp.test(fold), :) = foldPredictions;
validationScores(cvp.test(fold), :) = foldScores;
end

% Compute validation accuracy
correctPredictions = (validationPredictions == response);
isMissing = isnan(response);
correctPredictions = correctPredictions(~isMissing);
validationAccuracy = sum(correctPredictions)/length(correctPredictions);

```

## B. CODE TO TRAIN LOGISTIC REGRESSION

```

%% % % % % %
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained Linear SVM model
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3b.m
% Description: This code was written to train Linear SVM model.
%% % % % % %

function [trainedClassifier, validationAccuracy] = linearsvm2(trainingData)

% Auto-generated by MATLAB on 17-Feb-2022 13:03:18

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
        false];

```

```

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'linear', ...
    'PolynomialOrder', [], ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
trainedClassifier.ClassificationSVM = classificationSVM;
trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
    original training data. \nAdditional variables are ignored. \n \nFor more information, see
    <a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
    "appclassification_exportmodeltoworkspace")">How to predict using an exported
    model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
        false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 10);

```

```
% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');
```

### C. CODE TO TRAIN GAUSSIAN SVM I

```
%%%%%%%%%%
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained Gaussian Scale I
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3c.m
% Description: This code was written to train Gaussian Scale I.
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = finegaussiansvm2(trainingData)
% Auto-generated by MATLAB on 17-Feb-2022 13:04:26

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
        false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 0.83, ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
```



```

svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
trainedClassifier.ClassificationSVM = classificationSVM;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
"trainedModel". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
        false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'Kfold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

#### **D. CODE TO TRAIN GAUSSIAN SVM II**

```

%%%%%%%%%%
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained Gaussian Scale II
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3d.m

```

```
% Description: This code was written to train Gaussian Scale II.
%%%%%%%%%
```

```
function [trainedClassifier, validationAccuracy] = mediumgaussiansvm2(trainingData)
```

```
% Auto-generated by MATLAB on 17-Feb-2022 13:04:48
```

```
% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false,
        false, false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 3.3, ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
    trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
        'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
trainedClassifier.ClassificationSVM = classificationSVM;
    trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2020b.';
    trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
```

```

original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>.);

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,
        false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## E. CODE TO TRAIN GAUSSIAN SVM III

```

%%%%%%%%%%
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained Gaussian Scale III
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3e.m
% Description: This code was written to train Gaussian Scale III.
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = coarsegaussiansvm2(trainingData)
% Auto-generated by MATLAB on 17-Feb-2022 13:05:07

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};

```

```

predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,
    false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationSVM = fitsvm(...
    predictors, ...
    response, ...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 13, ...
    'BoxConstraint', 1, ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
    trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
trainedClassifier.ClassificationSVM = classificationSVM;
    trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2020b.';
    trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
    original training data. \nAdditional variables are ignored. \n \nFor more information, see
    <a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
    "appclassification_exportmodeltoworkspace")">How to predict using an exported
    model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
    'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;

```

```

        isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,
        false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## F. CODE TO TRAIN COARSE KNN

```

%%%%%%%%%%
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained Coarse KNN
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3f.m
% Description: This code was written to train coarse KNN.
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = coarseknn2(trainingData)
% Auto-generated by MATLAB on 17-Feb-2022 13:06:19

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
    'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,
    false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationKNN = fitcknn(...
    predictors, ...
    response, ...
    'Distance', 'Euclidean', ...
    'Exponent', [], ...

```

```

    'NumNeighbors', 100, ...
    'DistanceWeight', 'Equal', ...
    'Standardize', true, ...
    'ClassNames', [0; 1]);
% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
knnPredictFcn = @(x) predict(classificationKNN, x);
trainedClassifier.predictFcn = @(x) knnPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
trainedClassifier.ClassificationKNN = classificationKNN;
trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
    original training data. \nAdditional variables are ignored. \n \nFor more information, see
    <a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
    "appclassification_exportmodeltoworkspace")">How to predict using an exported
    model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance', 'ALT_percent',
    'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
    false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationKNN, 'KFold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## G. CODE TO TRAIN BOOSTED TREES

```
%%%%%%%%%
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained Boosted Trees
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3g.m
% Description: This code was written to train Boosted Trees.
%%%%%%%%%

function [trainedClassifier, validationAccuracy] = boostedtrees2(trainingData)
% Auto-generated by MATLAB on 17-Feb-2022 13:08:09

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
        false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateTree(...
    'MaxNumSplits', 20);
classificationEnsemble = fitcensemble(...
    predictors, ...
    response, ...
    'Method', 'AdaBoostM1', ...
    'NumLearningCycles', 30, ...
    'Learners', template, ...
    'LearnRate', 0.1, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
ensemblePredictFcn = @(x) predict(classificationEnsemble, x);
trainedClassifier.predictFcn = @(x) ensemblePredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
```

```

trainedClassifier.ClassificationEnsemble = classificationEnsemble;
trainedClassifier.About = 'This struct is a trained model exported from Classification
Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
"trainedModel". \n \nThe table, T, must contain the variables returned by: \n
c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
original training data. \nAdditional variables are ignored. \n \nFor more information, see
<a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
"appclassification_exportmodeltoworkspace")">How to predict using an exported
model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationEnsemble, 'Kfold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## H. CODE TO TRAIN SUBSPACE DISCRIMINANT

```

%%%%%%%%%%
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained Subspace Discriminant Model
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3h.m
% Description: This code was written to train Subspace Discriminant.
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = subspacedisc2(trainingData)
% Auto-generated by MATLAB on 17-Feb-2022 13:08:51

```



```

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false,
        false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
subspaceDimension = max(1, min(6, width(predictors) - 1));
classificationEnsemble = fitcensemble(...
    predictors, ...
    response, ...
    'Method', 'Subspace', ...
    'NumLearningCycles', 30, ...
    'Learners', 'discriminant', ...
    'NPredToSample', subspaceDimension, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
ensemblePredictFcn = @(x) predict(classificationEnsemble, x);
trainedClassifier.predictFcn = @(x) ensemblePredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
trainedClassifier.ClassificationEnsemble = classificationEnsemble;
trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
    original training data. \nAdditional variables are ignored. \n \nFor more information, see
    <a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
    "appclassification_exportmodeltoworkspace")">How to predict using an exported
    model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.

```

```

inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false, false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationEnsemble, 'Kfold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');

```

## I. CODE TO TRAIN RUS BOOSTED MODEL

```

%%%%%%%%%%
% Created on: 02/17/22
% Input: Internally called from obj1c_matlab_3.m
% Output: Trained RUS Boosted Model
% Author: Auto-generated by MATLAB, initiated by Ridhi Deo
% File name: Obj1c_matlab_3i.m
% Description: This code was written to train RUS Boosted Model.
%%%%%%%%%%

function [trainedClassifier, validationAccuracy] = rusboostedtrees2(trainingData)
% Auto-generated by MATLAB on 17-Feb-2022 13:09:13

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance', 'ALT_percent',
    'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateTree(...

```

```

'MaxNumSplits', 20);
classificationEnsemble = fitcensemble(...
    predictors, ...
    response, ...
    'Method', 'RUSBoost', ...
    'NumLearningCycles', 30, ...
    'Learners', template, ...
    'LearnRate', 0.1, ...
    'ClassNames', [0; 1]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
ensemblePredictFcn = @(x) predict(classificationEnsemble, x);
trainedClassifier.predictFcn = @(x) ensemblePredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'ALT', 'ALT_percent', 'ASP', 'AST',
    'AST_percent', 'Age', 'BMI', 'BMI_percent', 'HDL', 'HDL_percent', 'Insulin_resistance'};
trainedClassifier.ClassificationEnsemble = classificationEnsemble;
trainedClassifier.About = 'This struct is a trained model exported from Classification
    Learner R2020b.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n
    yfit = c.predictFcn(T) \nreplacing "c" with the name of the variable that is this struct, e.g.,
    "trainedModel". \n \nThe table, T, must contain the variables returned by: \n
    c.RequiredVariables \nVariable formats (e.g., matrix/vector, datatype) must match the
    original training data. \nAdditional variables are ignored. \n \nFor more information, see
    <a href="matlab:helpview(fullfile(docroot, "stats", "stats.map"),
    "appclassification_exportmodeltoworkspace")">How to predict using an exported
    model</a>');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
    predictorNames = {'Age', 'HDL', 'AST', 'ALT', 'ASP', 'BMI', 'Insulin_resistance',
        'ALT_percent', 'AST_percent', 'BMI_percent', 'HDL_percent'};
predictors = inputTable(:, predictorNames);
response = inputTable.HS;
    isCategoricalPredictor = [false, false, false, false, false, false, false, false, false,
        false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationEnsemble, 'KFold', 10);

% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

```

```
% Compute validation accuracy  
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');
```

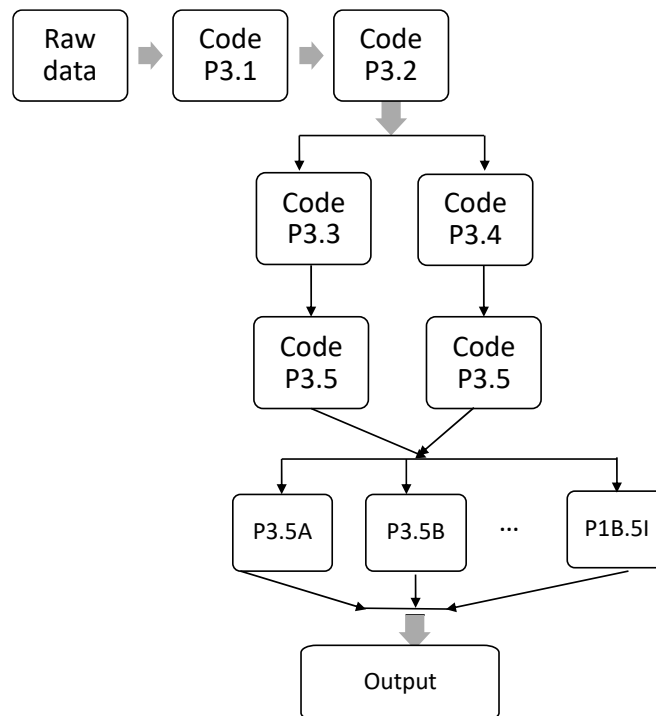


Figure P3.1: Figure outlining the flow of code used in this research objective

## **CHAPTER 5. ASSESSMENT OF A COMMERCIALLY AVAILABLE SENSOR FOR ARSENIC DETECTION IN WATER (PAPER 4)**

### **5.1 Introduction**

Chronic heavy metal exposure (As, Pb, Hg and Cd) is found to correlate with abnormal liver biochemistry, with NAFLD and, with liver damage in general [1]–[3]. The inability of the liver to metabolize heavy metals warrants the screening, diagnosis, and early intervention for individuals living with chronic heavy metal exposure. The impact chronic of heavy metal exposure and its relationship with NAFLD is reviewed in detail in an under-review paper, attached in Appendix H.

While any amount of heavy metal exposure is not ideal for the human system, there are certain permissible levels of heavy metal (in drinking water), determined by agencies like US – Environmental Protection Agency (EPA) and the World Health Organization (WHO) [4], [5]. In the USA, the safe water drinking act requires the US – EPA to ‘establish and enforce’ standards for public water systems [6]. While these standards are enforced by local agencies for public water sources, there are no specific regulations for private sources of water like wells. Further, any leaks or damages in the plumbing could lead to an unexpected heavy metal exposure at a household level.

Monitoring of drinking water quality at a home/field-level can be used in a preventative way to avoid the risk of liver disease. Use of a sensor that does not involve elaborate or sophisticated equipment can be useful for an at home test. Recently, a few commercially available sensor kits were found in the marketplace. One such kit used a color-based Arsenic detection concept. Although color-based output is easy to use, human beings are subjective in color assessment and there many other associated challenges associated with this approach. A need for a proof the concept assessment for such a kit in laboratory condition was therefore justified.

Thus, the objective of this study was to assess a commercially available Arsenic sensor kit under lab conditions. The associated tasks are:

1. To evaluate the performance of the selected kit in laboratory conditions.
2. To develop and test a computer imaging algorithm for quantitative assessment of the color information on the sensor testing strip.

## 5.2 Methods

### *Background*

A commercial test kit – “Industrial Test Systems Quick 481396 Arsenic for Water Quality Testing, 100 Tests, 12 Minutes Test Time” was purchased from Amazon’s website and was tested under lab conditions. The detailed instructions can be found on the manufacturer’s site [7]. The kit detects Arsenic based on color change and the manufacturer provides a scale for visual color comparison and result determination. Figure 5.2 has the color scale provided by the manufacturer [7].

### *Experiment design*

In this study, a total of five concentrations of Arsenic and four replicates per concentration were used. The concentrations were: blank (0 ppb (micrograms/liter)), 10 ppb, 50 ppb, 100 ppb and 200 ppb Arsenic in water. The concentrations are henceforward labelled as C0 (blank), C1, C2, C3 and C4, for 10, 50, 100, and 200 ppb, respectively. The four replicates are labelled as R1, R2, R3, and R4.

All samples were prepared using Nalgene grade plastic labware. All labware was cleaned five to six times with distilled water. Distilled water was also used to prepare the five different Arsenic concentrations. The Arsenic concentrations were prepared in the Mass Spectrometry lab (Chemistry building), Purdue University. Arsenic stock solution from Exaxol, Florida, USA (1000 ppm Arsenic in 2% nitric acid) was purchased and used to prepare the samples. Refrigerated stock solution was allowed to thaw and reach room temperature two hours before the sample preparation.

Serial dilution was conducted to obtain the five Arsenic concentrations. The samples used in the commercial kit were diluted using DI water and the appropriate Arsenic stock solution. The commercial kit requires that the samples do not have nitric acid in them. Each concentration was prepared four times to obtain four replicates. All samples and replicates were prepared on the same day. All the experiments with the test kit using the prepared samples were also conducted on the same day.

Samples were analyzed on an ELEMENT2 High Resolution Inductively Coupled Plasma Mass Spectrometer (Thermo Fisher Scientific, Bremen, Germany). Samples were introduced into the ICP via an Aridus II Desolvating Sample Introduction system with a 100ul/min PFA low flow

concentric nebulizer (Teledyne Cetac Technologies, 14306 Industrial Rd, Omaha, NE 68144). A Teledyne Cetac Autosampler ASX-112FR was used. After tuning and calibration using the 1ppb Thermo Fisher Tune-Up solution, the samples were analyzed for Arsenic.

Note that one replicate from each concentration was sent to the chemistry lab for analysis as a part of the ICP-MS calibration. All the twenty samples were not analyzed by ICP-MS. Results from the ICP-MS are shown in Figure 5.3. A  $R^2$  value of 0.9998 was obtained and thus it validated the appropriateness of the sample preparation method. The ICP-MS analysis was conducted by Dr. Anusha Hettiyadura, who is affiliated with the Department of Chemistry at Purdue University.

The experiments to test for Arsenic in water using the selected kit were conducted in the laboratory at Purdue University. The reagents and test strips were part of the Arsenic kit and were used for evaluating the kit [7]. The instructions provided with the kit were followed. Briefly, Arsenic samples or blank were added to a reaction bottle (provided with the kit) and the reagents were added as per the kit instructions [7]. The provided test strip was added in the last step to the cap of the reaction bottle and 10 minutes were allowed to pass for the color change to occur on the test strip [7]. The timing and measurements were followed as instructed in the kit manual [7]. After 10 minutes, the test strip was extracted and transferred into a photography box for digital image capture.

The photography box was purchased and used to capture images consistently with fixed lighting and camera position (Figure 5.1). Note that the photography box is not a part of the Arsenic kit and was separately purchased to allow consistent image capture conditions. Images of five concentrations, each with four replicates were captured (total 20 images), one at a time, using an iPhone XR. Each image was then exported into a Windows PC for further processing. Every image was individually examined and if needed, the image was straightened using the window photo viewer tool. In this step, any background from the images was also removed manually.

Next, all the images were re-sized using the “`imresize()`” function which part of the image processing toolbox in MATLAB R2020b [8]. The `resize` function is used to convert all the images into standard pixel dimensions. In this case, the pixel dimensions of the smallest image were identified (“`final_dim`”). The `final_dim` was used as a reference to convert all images to the same size as `final_dim`. The output of this step was 20 images of the same pixel dimensions. The resized images were then used as part of the visual snapshot analysis. The summary of the steps used, are



shown in a flowchart (Figure 5.4). The code written to resize the images is in Appendix E.1. All the digital images were then processed using two methods:

1. Visual snapshot analysis.
2. Customized image analysis and pattern recognition algorithm.

### ***Step 1: Visual snapshot analysis***

The concept of visual snapshot analysis, a new analysis method [9] was applied in this analysis. The method uses more than one replication to identify Arsenic concentration instead of just one reference image provided by the test kit [9]. As the kit uses a paper-based sensor, the color development on the strip might not be uniform across it.

In the visual snapshot analysis, there are ' $N_{sc}$ ' replicates. In this case,  $N_{sc}=4$ . In the future, it is recommended to increase  $N_{sc}$  as at least 10 for better analysis. Note that this is a preliminary study and will not take into consideration the reference images provided by the test kit [9].

#### *Process used for visual snapshot analysis [9]:*

The digital images of all concentrations were collected, and their replicates were placed in a tabular manner, on a single screen (Figure 5.6) [9]. It was ensured that the computer screen on which these images are displayed was color calibrated and the background was a uniform color (black in this case). The surrounding environment of the display screen was also well lit with daylight spectrum. The visual snapshot analysis was conducted in two stages:

1. Within concentration analysis
2. Pairwise concentration analysis

#### *Parameters for within concentration analysis [9]*

1. Minimum concentration for discrimination:

The concentrations  $C_i$  ( $i = 1, 2, 3$ , or  $4$ ) were compared with the blank ( $C_0$ ) to identify which concentration had all its replicates different from  $C_0$ . This process was still based on visual color perception. Based on this visual analysis, a concentration  $C_{min}$  was identified as the minimum concentration that the kit could differentiate from

the blank. *This is an important finding alternate to the process provided by the manufacturer of the commercial kit.*

## 2. Similarity within a concentration and repeatability [9]:

A parameter  $V_{sc}$  is defined here.  $V_{sc}$  describes the similarity among the replicates within a given concentration. It was defined as the number of images that are visually perceived to be similar, divided by the total replicates. It was expressed in percentages (%). For example, if the  $N_{sc} = 4$ , the minimum resolution is  $\frac{1}{4} = 25\%$ . That is, testing with  $N_{sc} = 4$  leads to a 25%, 50%, 75% or 100% similarity. Ideally, a  $V_{sc}$  of 100% would be desired but in this case, a  $V_{sc}$  of 90% would be acceptable [9]. Again, for this study, as  $N_{sc} = 4$ , the resolution does not allow to capture a 90%  $V_{sc}$ . Therefore, a  $V_{sc}$  of 100% is required in this study [9].

The  $V_{sc}$  reflects how different variables (variations among the test kits, the experimental variations due to any human error or any unforeseen situation) affect the color output on the test kit for a given concentration [9]. In this study, one of the goals was to identify how many concentrations met the defined  $V_{sc}$ . Let this parameter be  $V_{cm}$ . Say maximum number of concentrations being tested is  $C_t$ . Then, a repeatability factor can be defined as [9]:

$$RF = \frac{V_{cm}}{C_t} \times 100 \quad (4.1)$$

This was calculated and  $RF$  [9] was determined. The maximum  $RF$  is 100%.

## *Parameters for pairwise concentration analysis & identification of seed images [9]*

Two concentrations  $C_i$  and  $C_j$  were chosen from  $C_t$ . The concentration pair was labelled  $C_{ij}$ . For each pair, the sample images (Figure 5.6) were observed and identified how many images were visually different from each other. Then the percentage differentiability was defined as [9]:

$$PDIF \% = \frac{V_{dc}}{N_{sc}} \times 100 \quad (4.2)$$

Aside from calculating the  $PDIF$  value, another benefit of this analysis was the identification of sample images that can be used as seed images for subsequent development of image analysis and pattern recognition [9].

### ***Step 2: Customized image analysis and pattern recognition algorithm***

The resized images from the above step (Figure 5.4) were further processed using MATLAB R2020b image processing toolbox [8] as part of the image analysis. The aim of this analysis was to quantify the image concentrations using a preliminary algorithm such that the subjectivity in visually examining the images is minimized.

The resized images were used and smoothed to reduce any image noises and improve the overall image appearance. The function `imgaussfilt()` was used to smooth the images [10]. The code used to smooth the resized images is in Appendix E.2. The original image scale was Red, Green, Blue (RGB), by default. However, a different color model was used in this work to interpret the color of the test strips.

Hue, saturation, value or HSV color scale is a transformation of the RGB color scale. The HSV scale was developed in 1978 [11]. HSV scale is used commonly when human perception of color is required for various applications. Some examples are in agriculture (fruit or crop color identification), in the medical domain (bio image color identification) or in other industries like manufacturing etc. [12]–[16]. While the RGB scale defines color in terms of the dominant red, green or blue colors, the HSV scale represents the image as a mixture of hue, saturation and intensity/brightness [11]. Hue is the part of the image that represents different colors in a wheel like model. The hue wheel can for example range as: blue, magenta, red, yellow, green, cyan, and back to blue again. Hue of an image is measured in angles, and it ranges from 0 – 360 degrees [11], [16]. Saturation on the other hand, refers to the intensity of a particular Hue [16]. Saturation is measured in pixel intensities and ranges from 0 – 255 with 0 being the lowest intensity (black) and 255 being the highest intensity (white). Value refers to image brightness and is also measured from 0 – 255. However, in this work, the focus is on identifying the hue and saturation of the images of interest.

Since the visual color perception is required in this case, the RGB images were converted to HSV scale using the `rgb2hsv()` function in MATLAB. The code used to convert the RGB images into HSV is in Appendix E.3. The hue and saturation data of each image were then processed to determine the mean hue and mean saturation values for each of the 20 images (5 concentrations, 4 replicates/concentration). The MATLAB code used to extract these descriptive statistics is in Appendix E.4. The hue and saturation data for each replicate are in Table 5.3 -

Table 5.5, respectively.

To determine the hue and saturation data for every concentration, the data from four replicates were further processed to identify mean and standard deviations (SD). These data for hue and saturation are in Table 5.4 & Table 5.6, respectively. Boxplots were plotted using the mean and SD data. These plots are in Figure 5.16 & Figure 5.17. The results pertaining to this section are discussed below.

The hue and saturation data were transformed to a 2-dimensional plane by plotting a scatter plot of Hue vs Saturation using the mean hue and saturation data points for each concentration (e.g.:  $H_i$  and  $S_i$  where  $i, j = 0, 10, 50, 100$ , or  $200$  ppb) (Figure 5.18) [9]. Next, the distance between each geometric coordinate ( $H_i, S_i$ ), and ( $H_j, S_j$ ) were computed for all combinations of  $i$  and  $j$ . In this work, the Euclidean distance metric was used [9]. The distances between each coordinate are shown as a confusion matrix in Table 5.7.

### 5.3 Results & discussion

#### *Visual snapshot analysis*

Each of the processed images was observed visually and their observations were recorded. Upon visually observing the Figure 5.6, the  $C_{min}$  concentration was found to be 100 ppb of Arsenic in water. That is each replicate of the 100-ppb concentration (Arsenic in water) was found to be visually different from all replicates of the blank. It is important to note here that the WHO and EPA limits for permissible amounts of Arsenic in drinking water are 1 ppb and 10 ppb, respectively [4], [5]. Based on the limited testing conducted in this work, the test kit was not able to detect Arsenic in water below 100 ppb reliably. Additional details from the visual inspection are outlined in the sections below.

#### *Within concentration observations from visual snapshot analysis:*

In this section, the focus was to identify how much similarity can be visually identified within four replicates (R1, R2, R3, and R4) for each concentration. Brief comments regarding the comparison of sample strips with manufacturer provided results are also mentioned for every concentration. Note that images of both the sample strips and the manufacturer provided results were captured under the same conditions in our lab.

1. 0 ppb/Blank: Of the four replicates, three appear whitish (R2, R3, R4), but one appears a very light shade of yellow (R1) (ref Figure 5.7A). Three of the four replicates appear consistent (R2, R3, R4) but one does not (R1). When compared to the manufacture-provided result, the sample images (R2, R3 and R4) appear to be brighter shades of white in our testing, than in the manufacturer's provided reference chart [7] (Figure 5.7B).
2. 10 ppb Arsenic in water: Among the four replicates, R1 and R3 appear whitish but R2 & R4 are pale yellow in appearance (in Figure 5.8A). R2, R4 appear inconsistently yellow and white in patches over the strip area. Overall, there is no consistency between the replicates in the 10 ppb (in Figure 5.8A) test results. Visually, none of the replicates from the testing in this work were found to match the manufacturer's provided reference chart [7].
3. 50 ppb Arsenic in water: The replicates R1, R2, and R4 appear alike with similar shades of yellow (Figure 5.9). However, R3 looks completely whitish. Although R1, R2 and R4 are yellowish in color, they vary in the intensity with R2 being the darkest and R1, R4 being lighter in comparison to R4. None of the replicates from our testing look like the manufacturer's provided image, although three of the four replicates appear consistent. The results from our testing of the commercial kit for 50 ppb look a lighter intensity of yellow than the provided result. In this case, the provided image and the obtained results can be very misleading and there could be a high chance on misinterpreting the 50 ppb result as a lower concentration – which can have significant impact of the health and safety of the user.
4. 100 ppb Arsenic in water: The R2, R3 and R4 images appear similar in this case. However, R3 and R4 have a darker outer border around them, compared to R2 (Figure 5.10). R1 appears to be different from the rest of the replicates and has an unexpected whitish spot on it. The test images (all replicates in Figure 5.10A) and the manufacturer's result (Figure 5.10B) are all captured under the same light conditions using a photo box. R2, R3, and R4 obtained using the test kit look similar in shade to the manufacturer's provided reference chart [7]. But R1 appears much darker in intensity compared to the other three replicates.
5. 200 ppb Arsenic in water: The R2 and R3 images appear similar to each other but have a darker outer border and a lighter shade in the center. R4 has no outer border but a consistent yellow shade all through the image. Overall, there is no uniformity in color across the test

strip surface in any of the replicates. The intensity of the yellow shade varies within a replicate and across replicates. The test images (all replicates in Figure 5.11A) and the manufacturer's provided reference chart [7] (in Figure 5.11B) are all captured under the same light conditions using a photo box. When compared with the manufacturer provided result, the replicates R1, R2, and R3 have an outer border effect. Overall, the consistent shade of yellow seen in the central part of the manufacturer's provided reference chart [7] was not observed in the replicates obtained in our testing.

*Between concentration observations from visual snapshot analysis:*

In this section, observations are made by comparing neighboring concentrations in a pairwise manner. Concentrations are compared in the following pairs: 0 vs 10 ppb, 10 ppb vs 50 ppb, 50 ppb vs 100 ppb, and 100 ppb vs 200 ppb. The differentiability between these concentrations is quantified based on visual observations. In each comparison below, eight images, four each from a concentration in the pair are visually compared with each other and a differentiability percentage (PDIF %) is noted.

1. Blank vs 10 ppb Arsenic in water: The eight images (in Figure 5.12) look mostly whitish to the eye when compared side by side. The 10 ppb R1, R3 and 0 ppb R2, R3 look whitish in color but overall, they look very similar to each other. The other images, i.e., 0 ppb R1, R4 and 10 ppb R2, R4 also look similar to each other. Visually, it would be very difficult to distinguish between 0 ppb and 10 ppb based on our testing. The *PDIF* % in this case was found to be 0%.
2. 10 vs 50 ppb Arsenic in water: The comparison of 10 vs 50 ppb is a bit more distinguishable (see Figure 5.13) than that of 0 vs 10 ppb. However, 50 ppb R3 looks very similar to most of the 10 ppb replicates and could be misleading to a user. Based on our testing, we see that three of the four 50 ppb replicates are yellow in shade and are visually different from the 10 ppb replicates. The 10 ppb can be differentiated using two of the four replicates. In total five test strips (10 ppb R1, R3 and 50 ppb R1, R2, R4) are clearly distinguishable. Therefore, the *PDIF* % was found to be 62.5.
3. 50 vs 100 ppb Arsenic in water: Refer to Figure 5.14 for comparing 50 vs 100 ppb detection using four replicates each. There is a clear difference in color between the 50 and 100 ppb sensors except for one of the 50 ppb (R3) which looks entirely whitish. The R1 100 ppb

also has a whitish spot on the sensor surface and is not consistent with the other replicates. However, the difference in yellow intensity between 50 to 100 ppb is a good indicator of the increasing Arsenic concentration and the *PDIF* % was 75%.

4. 100 vs 200 ppb Arsenic in water: Figure 5.15 shows the comparison of 100 vs 200 ppb replicates. Overall, all the eight replicates look similar visually and are very hard to distinguish from each other. Particularly, replicates R2, R3, R4 100 ppb and R1 200 ppb represent a similar shade of yellow. R1 100 ppb is darker (and has an unexpected white patch) and looks like R3, R4 200 ppb in terms of the intensity of yellow. Further most of the images seem to have a border effect and uneven intensity of yellow across the sensor surface. The differentiability percentage is therefore 0 in this case.

In summary, based on the pairwise comparison of neighboring concentrations, it was found that visual difference in color is not differentiable in 0 vs 10 ppb and 100 vs 200 ppb (Arsenic in water). The differentiability of 10 vs 50 ppb is better at 62.5% but it still does not meet the high differentiability criteria (of 100%). The 50 vs 100 ppb (Arsenic in water) had the best performance among all other concentrations at a *PDIF* % of 75 but it still does not meet the required *PDIF* of 100%.

Based on the limited testing conducted under lab conditions, the commercial kit was found to be able to detect the presence or absence of Arsenic in water, but it was unable to differentiate between Arsenic concentrations with high *PDIF*. While the test strips showed difference in results between 50 ppb and 100 ppb Arsenic, these concentrations might be too high in terms of permissible Arsenic concentrations in drinking water. The WHO and EPA standards are 1ppb and 10 ppb, respectively [4], [5].

### ***Image analysis & pattern recognition algorithm***

The test strip images were analyzed using a preliminary image processing algorithm and the obtained results are elaborated in this section. For reference, the flowchart of the methods used is in Figure 5.5.

The hue and saturation data were captured, and their means and standard deviations were computed as described in the methods section. A boxplot for Hue and another for Saturation data were plotted (Figure 5.16 & Figure 5.17).

The results from hue box plot indicate that the mean hue for 0 ppb and 10 ppb (Arsenic in water) varied significantly within the replicates (as shown by the large ranges on the boxplot). The Hue range for 0 ppb replicates (Arsenic in water) was 16 to 49 and the hue range for 10 ppb replicates (Arsenic in water) was 36 to 53. The hues of 0 and 10 ppb (Arsenic in water) have an overlap region between 36 to 49, which indicates that the 0 and 10 ppb cannot be differentiated based on the hue data. It is important to note that the hue parameter is generally used to measure how humans view color. Therefore, the overlap of hue between 0 and 10 ppb indicates that the test strips for 0 and 10 ppb (Arsenic in water) cannot be differentiated both visually and using the hue data. Similar result was found based on the visual snapshot analysis of 0 and 10 ppb test strips. 10 ppb and 50 ppb (Arsenic in water) showed an overlap from 42-29 and therefore were not differentiable.

The 50 ppb (Arsenic in water) mean Hue range was very low (42-43) and therefore indicated consistent Hue within the replicates. The mean hue for 100 ppb (Arsenic in water) was between 33-38 and that of 200 ppb was between 27 – 35. The clear difference in Hue between 50 and 100 ppb indicated that the test strips for these concentrations were clearly differentiable. While the 50 ppb also had low variability within the hues for its replicates, the 100 ppb had a slightly higher hue range. However, the hue values for 100 and 200 ppb (Arsenic in water) had an overlap between 33-35 and this overlap again indicated that these two concentrations could not be differentiated using the hue data. The visual snapshot analysis also provided the same conclusions as seen with the hue data.

The results from the Saturation box plot can be used to understand the intensity of the color on each test strip. The saturations of 0 ppb test replicates ranged from 1 – 17 and that of 10 ppb test replicates range from 4 – 23. There was a large overlap of 4 – 17 between the 0 and 10 ppb (Arsenic in water) test replicates. This also indicated that the color saturation of the 0 and 10 ppb images are similar and could not be differentiated from each other.

The 50-ppb (Arsenic in water) saturation ranged from 6 to 95, whereas the 100-ppb (Arsenic in water) saturation ranged from 200 – 235. There was no overlap in the 50 and 100 ppb test strip saturations, and these could be separated from one another. Interestingly, the 200-ppb (Arsenic in water) saturation ranged from 174 – 222. This not only had a significant overlap with 100 ppb saturation (overlap from 200 to 222) but also the 200 ppb had lower saturation than that in 100 ppb. With increasing Arsenic concentrations, an increasing color saturation was expected



(See Figure 5.2). However, based on these results, the saturation was found to be decreasing slightly from 100 to 200 ppb, with a large overlap between these two concentrations. Therefore, based on saturation data, it was concluded that the 100 and 200 ppb (Arsenic in water) were not differentiable using the kit while testing in lab conditions.

The results from the visual snapshot and the image analysis were similar. Both methods indicated that the test kit was not 100% accurate in differentiating any of the concentrations. However, the differentiability was highest with the 50 ppb and 100 ppb (Arsenic in water). 0 ppb and 10 ppb (Arsenic in water) were not differentiable at all. Similarly, 100 ppb and 200 ppb (Arsenic in water) were also not differentiable using the test kit under lab conditions.

The results from the Euclidean distance confusion matrix (Table 5.7) indicated that the increase in distances between concentrations were not linear. For example, distance between 0 and 10 ppb, say  $\Delta_{0,10}$  is 13.2, which was similar to the increase in 10 to 50 ppb at  $\Delta_{10,50} = 51.5$ . But after that, the distance between 50 to 100 ppb was very high ( $\Delta_{50,100} = 151.2$ ). And finally, the distance between 100 and 200 ppb was very low at  $\Delta_{100,200} = 36.2$ . These results indicated that the color saturation as the concentration increased was not linear.

It is important to note that only four replicates per concentration were tested in this work due to a constraint of time and resources. Additional testing is recommended in the future, to further assess the test kit performance in detecting Arsenic in water.

## **5.4 Summary & conclusions**

A commercially available Arsenic detection kit was tested for determining Arsenic in water, and the results were examined. From the visual inspection, differentiating some concentrations were not clear (example 0 and 10, 10 and 50, 100 and 200 ppb Arsenic in water). The instructions provided in the commercial kit were found not to be sufficient for the user to quantify the color of the test strips. Visual inspection of the images was recommended by matching it with the reference color to determine the Arsenic concentration.

Thus, an alternate method [9] without relying on the use of the test kit reference images was adapted and evaluated in this study. The associated testing parameters were systematically analyzed. The following conclusions are outlined below:

1. The following concentrations were not differentiable based on our testing: 0 vs 10 ppb (Arsenic in water), 10 vs 50 ppb (Arsenic in water) and 100 vs 200 ppb (Arsenic in water).
2. The 50-ppb vs 100-ppb (Arsenic in water) were the most differentiable at 75% PDIF.
3. The increase in Euclidean distance measure (computed using hue and saturation data - 8 bit with 0 to 255 scale) was not linear with increase in concentration of Arsenic in water.

*A limitation of the study: All the samples for the Arsenic experiment were prepared using the same protocol and on the same day. Only one sub-sample of the four samples for each concentration was sent for ICP-MS analysis. It was assumed that all the samples have the same concentration as that of the sub-sample. Therefore, it might be possible that a specific replicate might not have the exact expected concentration.*

## **5.5 Recommendations for future work**

In the future, additional experiments using the same test kit or similar other kits for detecting Arsenic in water using multiple concentrations and/or multiple replicates are recommended.

### **Disclaimer:**

Mention of a company's name, product(s) or trademark(s) does not constitute any endorsement or recommendation or any lack of endorsement or any lack of recommendation by the author, and/or the faculty members associated with the study, and their affiliated employers, their affiliated organizations and/or entities.

## 5.6 Figures



Figure 5.1: Image of the photography box (purchased from a commercial source) used to capture images with consistent lighting

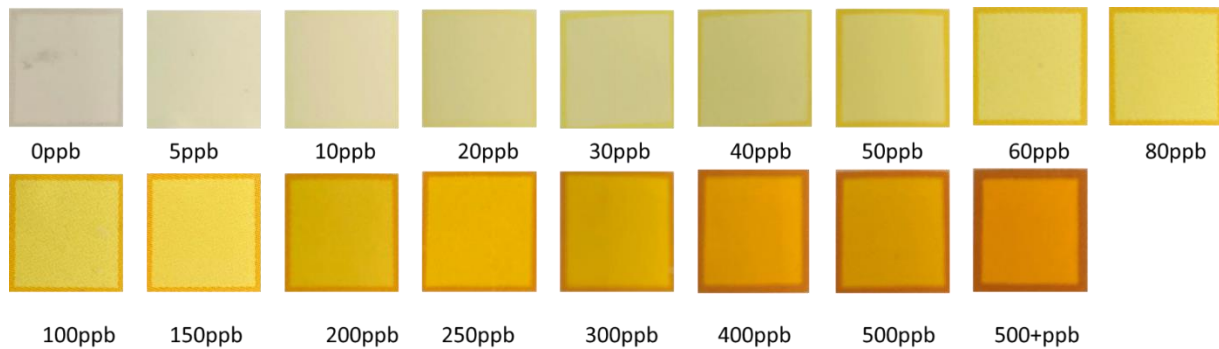


Figure 5.2: Calibration scale provided by the manufacturer\*

*\*Note: Images of the scale provided by the manufacturer were captured in our lab using the same photography box used to capture sample images in this work*

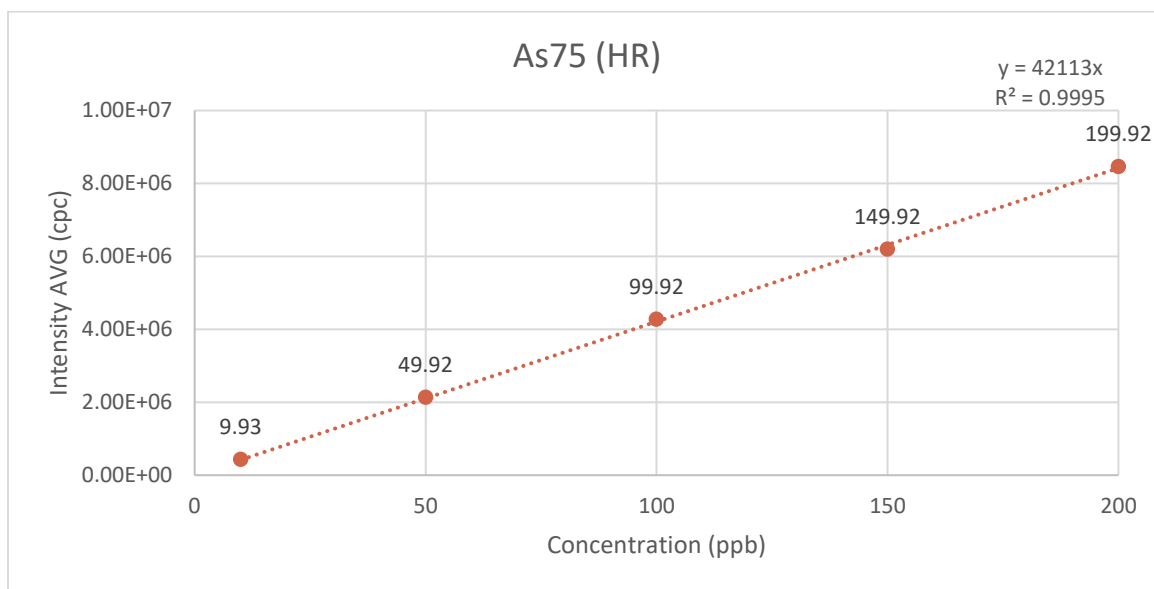


Figure 5.3: Results of the Arsenic calibration curve using ICP-Mass spec  
*Note: Only 1 replicate per concentration was analyzed using ICP-MS*

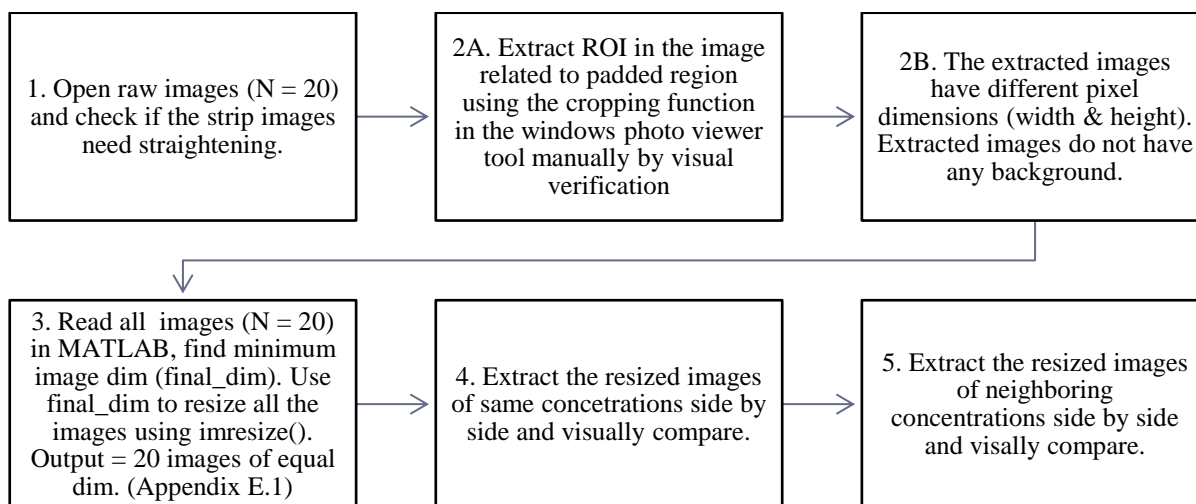


Figure 5.4: Flowchart indicating the process used for visual analysis

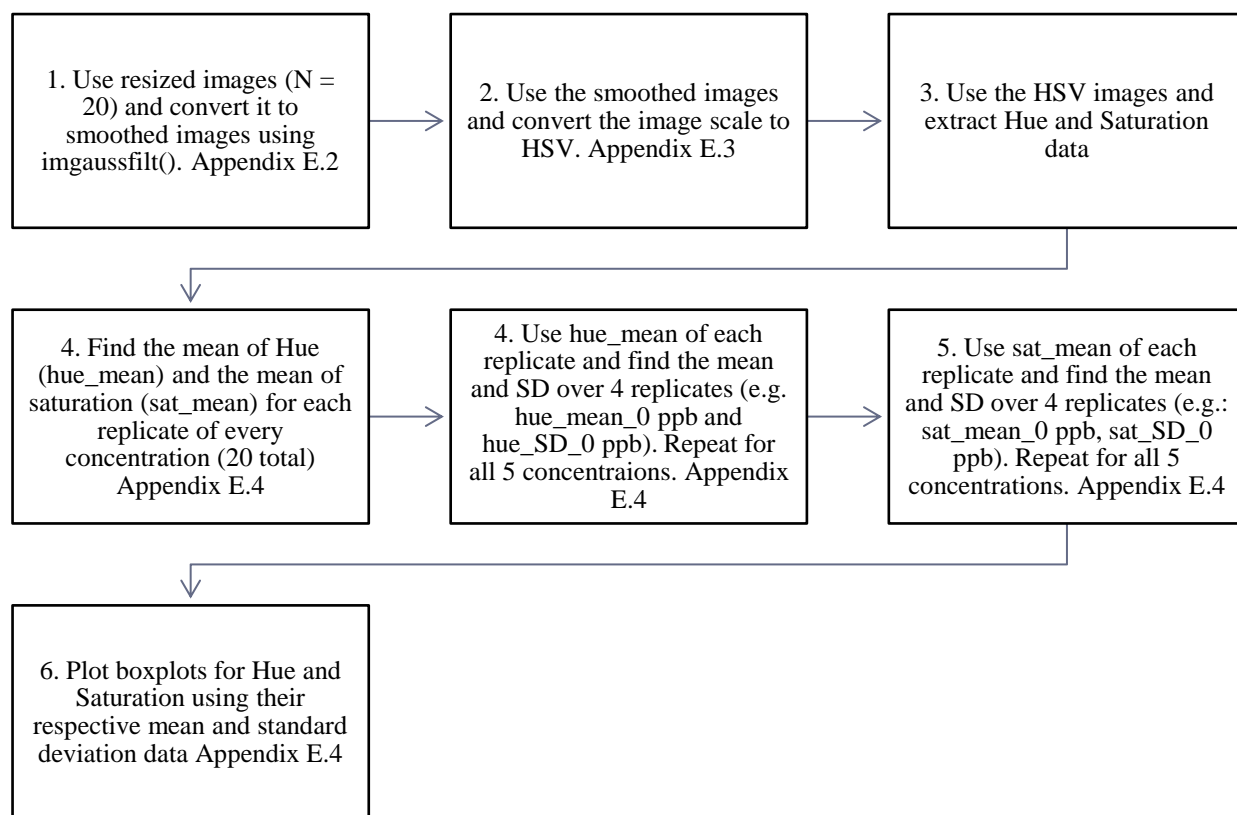


Figure 5.5: Flowchart indicating the process used for image analysis and pattern recognition

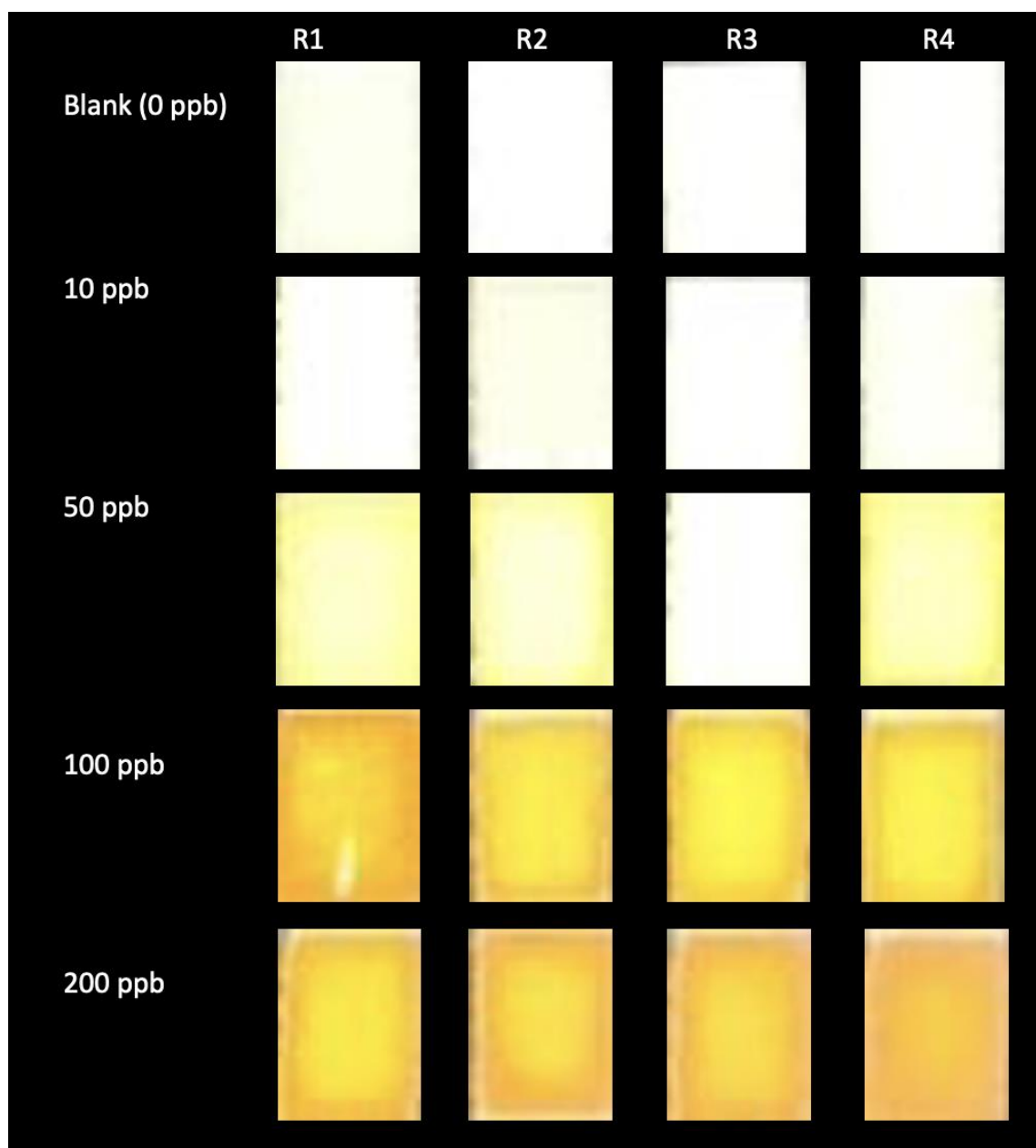


Figure 5.6: Graphical lay out for the digital images of test kit samples arranged in row for each concentration (rows) and each replicate (columns) [9]  
*Note: Only one replicate was confirmed by ICP-MS analysis*

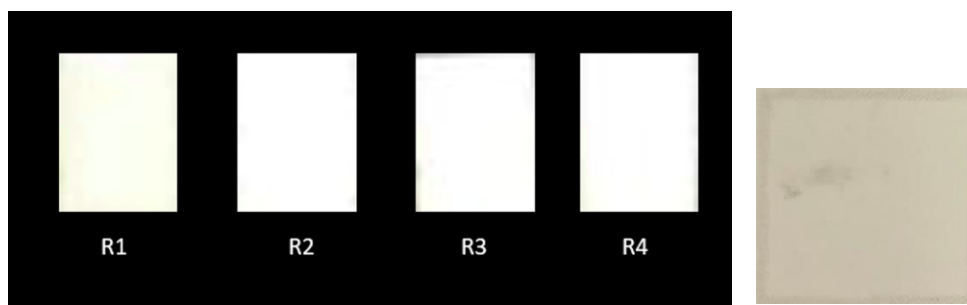


Figure 5.7: A: 0 ppb (Arsenic in water) test results for four replicates (R1,R2,R3,R4)  
B: Manufacturer's result

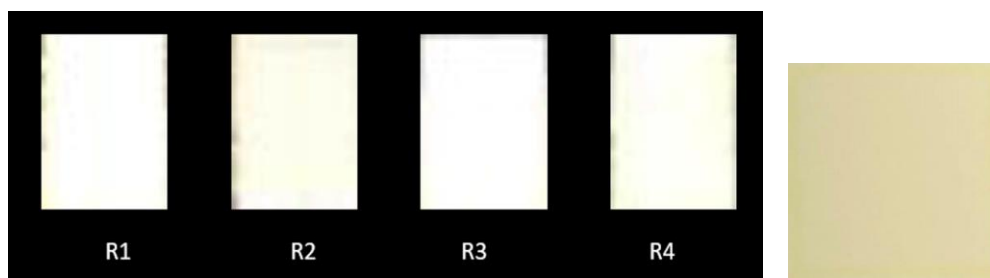


Figure 5.8: A: 10 ppb (Arsenic in water) test results for 4 replicates (R1,R2,R3, R4) B:  
Manufacturer's result

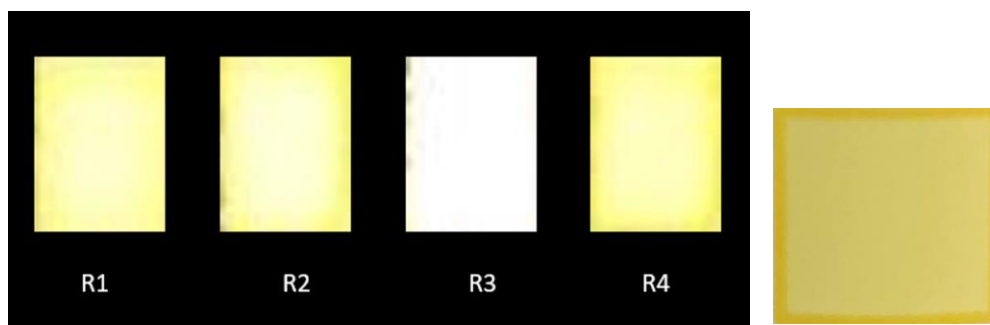


Figure 5.9: A: 50 ppb (Arsenic in water) test results for four replicates (R1,R2,R3, R4) B:  
Manufacturer's result

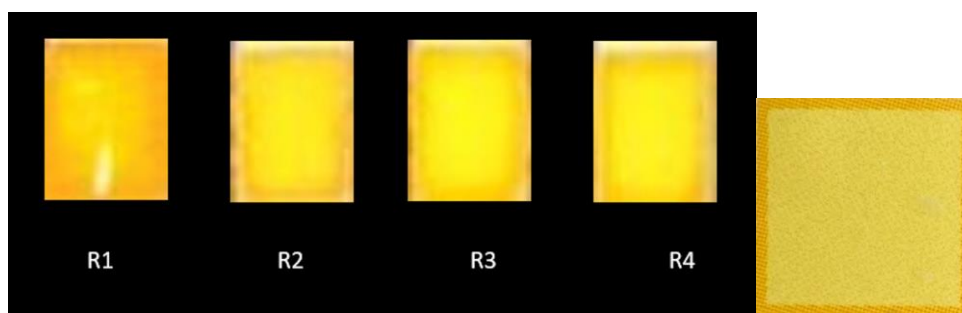


Figure 5.10: A. 100 ppb (Arsenic in water) test results for four replicates (R1, R2, R3, R4) B: Manufacturer's result

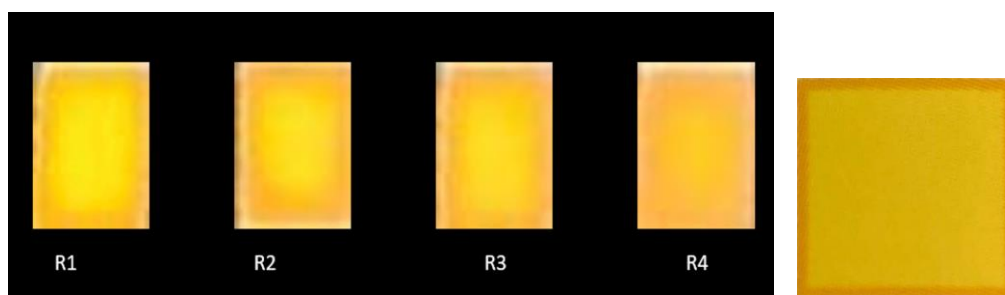


Figure 5.11: A. 200 ppb (Arsenic in water) test results for four replicates (R1,R2,R3, R4) B: Manufacturer's result

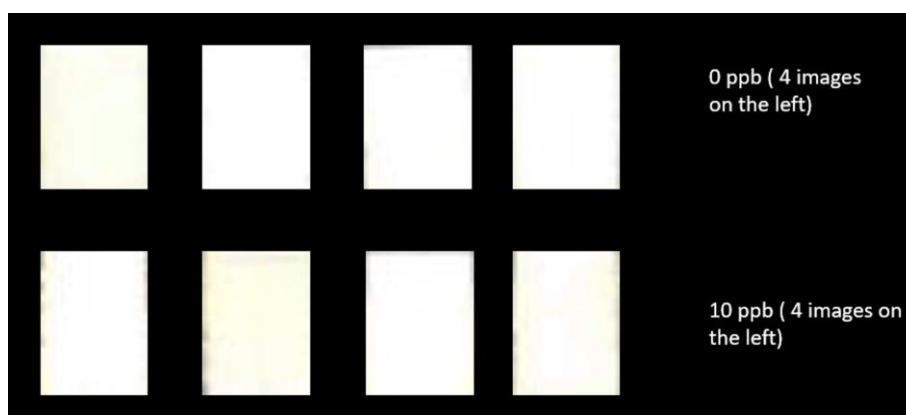


Figure 5.12: 0 ppb vs 10 ppb (Arsenic in water) side by side comparison



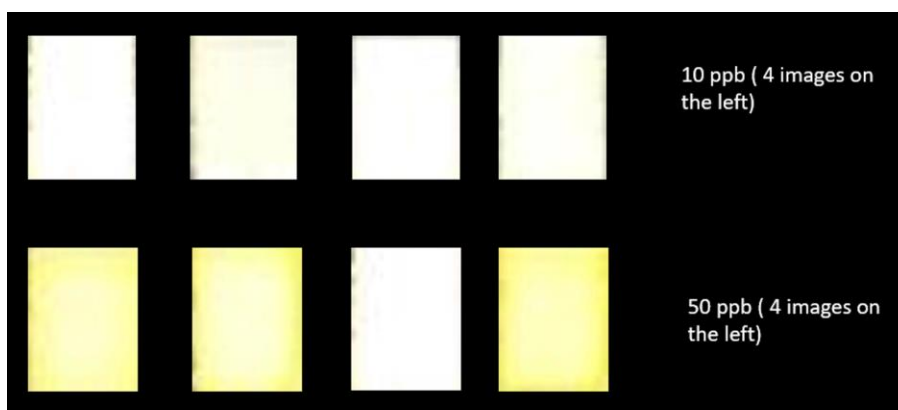


Figure 5.13: 10 ppb vs 50 ppb (Arsenic in water) side by side comparison

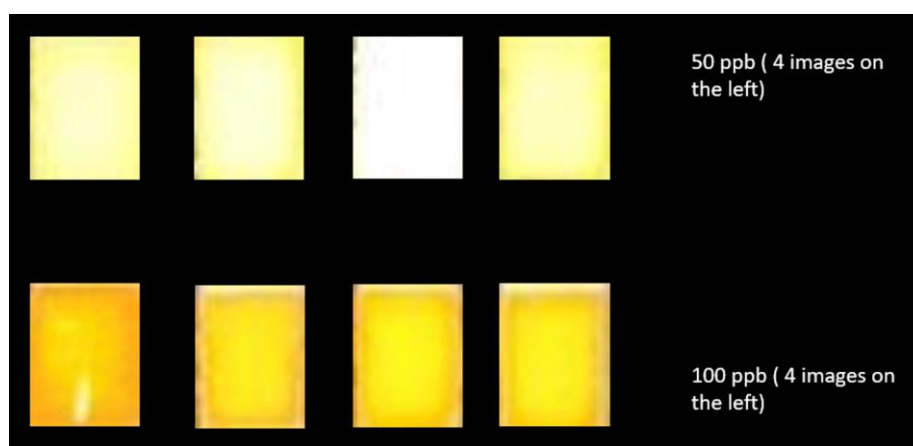


Figure 5.14: 50 ppb vs 100 ppb (Arsenic in water) side by side comparison

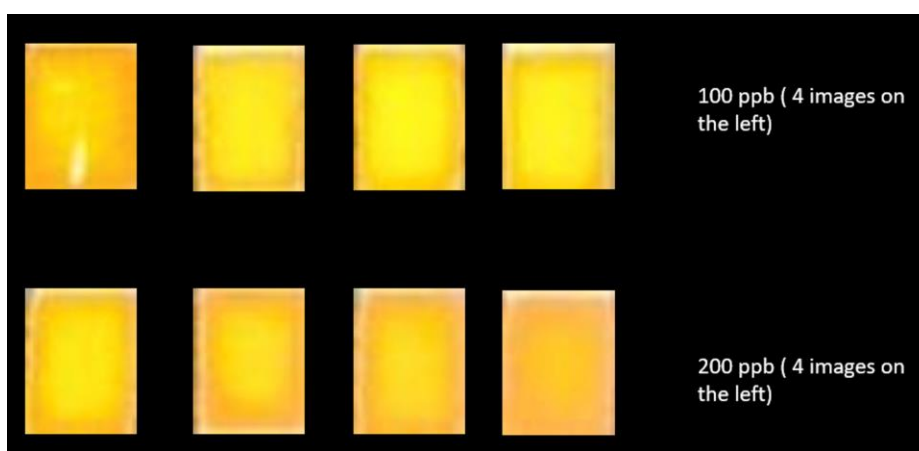


Figure 5.15: 100 ppb vs 200 ppb (Arsenic in water) side by side comparison

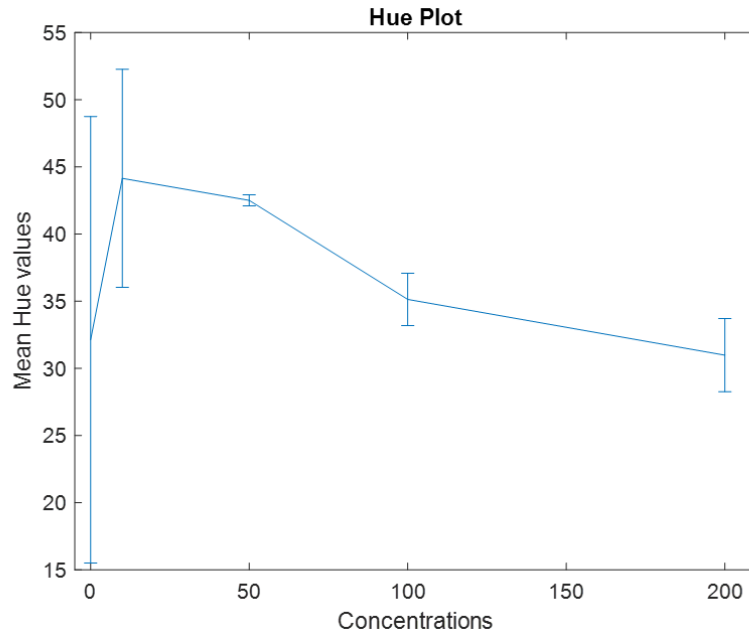


Figure 5.16: Boxplot of mean hue data, commercial kit testing (four replicates per concentration)  
*Error bars are plotted using standard deviation data*

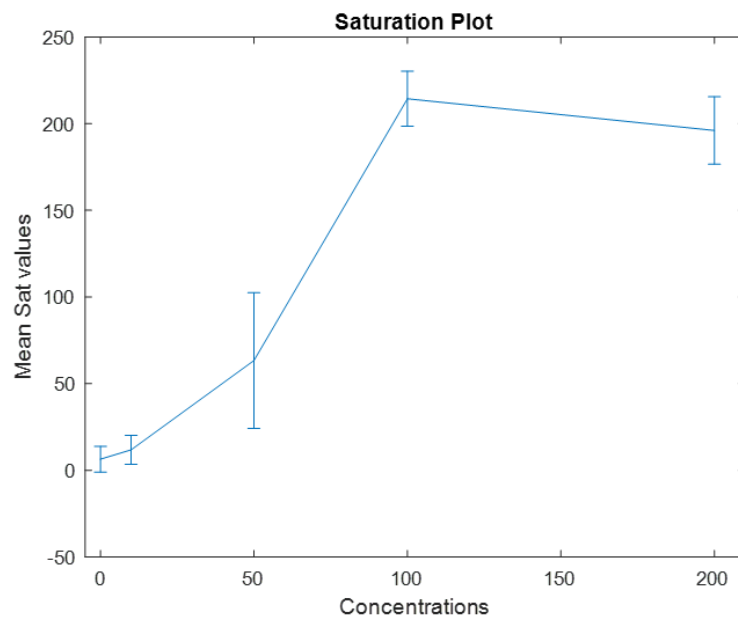


Figure 5.17: Box plot of mean saturation data, commercial kit testing (four replicates per concentration)  
*Error bars are plotted using standard deviation data*

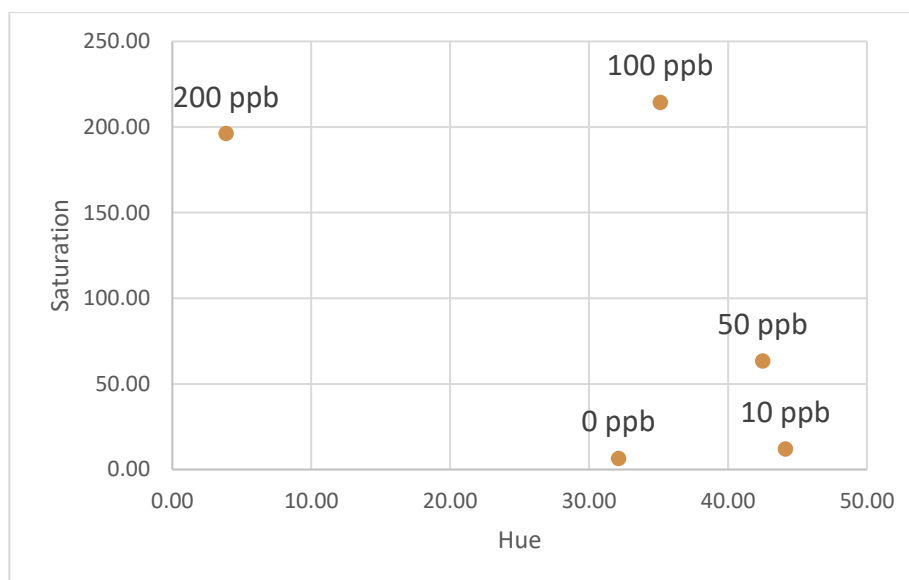


Figure 5.18: Mean hue vs mean saturation scatter plot

## 5.7 Tables

Table 5.1: Visual snapshot - within concentration analysis (4 replicates/concentration)

Concentration (ppb)	Comment on the similarity within a concentration	% Similarity ( $V_{sc}$ )
0	3 of the 4 replicates are similar, one replicate is yellowish	$V_{sc} = 75\%$
10	2 of the 4 replicates are whitish and 2 others are yellowish	$V_{sc} = 50\%$
50	3 of the 4 replicates are similar but one is totally different	$V_{sc} = 75\%$
100	3 of the 4 replicates are similar but one is different and has an unexpected white patch	$V_{sc} = 75\%$
200	3 replicates have a darker border, one replicate does not	$V_{sc} = 75\%$

Table 5.2: Visual snapshot - pairwise concentration analysis

Concentrations	% Pairwise Differentiability (PDIF)	Comments
0 vs 10	$C_{0,10} = 0\%$	Six of the eight test strips look whitish. The two other strips look pale yellow but are not differentiable.
10 vs 50	$C_{10,50} = 62.5\%$	The 10 ppb is differentiable 2 of 4 times. The 50 ppb is differentiable 3 out of 4 times.
50 vs 100	$C_{50,100} = 75\%$	50 ppb and 100 ppb are very different in color intensity. Except for 50 ppb R3, all other test strips appear clearly different
100 vs 200	$C_{100,200} = 0\%$	Three of the 100 ppb replicates (R2, R3, R4) look like one of the 200 ppb (R1). Overall, it is hard to distinguish 100 and 200 ppb apart.

Table 5.3: Mean values of the hue data – all replicates

Parameters	0	10 ppb	50 ppb	100 ppb	200 ppb
R1_hue	48.85	51.94	41.98	35.18	34.30
R2_hue	9.82	44.57	42.97	32.37	30.99
R3_hue	30.67	32.86	42.52	36.76	30.98
R4_hue	39.18	47.20	42.56	36.21	27.64

Table 5.4: Mean and SD of replicates – hue data

Parameters	0 ppb	10 ppb	50 ppb	100 ppb	200 ppb
Mean of all replicates	32.13	44.14	42.51	35.13	3.89
SD of all replicate means	16.62	8.12	0.41	1.95	2.72

*SD = Standard Deviation*

Table 5.5: Mean values of the saturation data – all replicates

Parameters	0	10 ppb	50 ppb	100 ppb	200 ppb
R1_Sat	17.34	6.33	6.39	199.64	221.53
R2_Sat	0.98	22.65	80.37	234.57	196.21
R3_Sat	3.24	4.25	71.78	219.26	192.14
R4_Sat	4.03	14.02	94.78	203.94	174.36

Table 5.6: Mean and SD of replicates – saturation data

Parameters	0 ppb	10 ppb	50 ppb	100 ppb	200 ppb
Mean of all replicates	6.40	11.81	63.33	214.35	196.06
SD of all replicate means	7.41	8.36	39.13	15.89	19.45

Table 5.7: Confusion matrix of Euclidean distances

	0 ppb	10 ppb	50 ppb	100 ppb	200 ppb
0 ppb	0.0	13.2	57.9	208.0	191.8
10 ppb	13.2	0.0	51.5	202.7	188.6
50 ppb	57.9	51.5	0.0	151.2	138.2
100 ppb	208.0	202.7	151.2	0.0	36.2
200 ppb	191.8	188.6	138.2	36.2	0.0

## 5.8 References

- [1] G. E. Arteel, "Hepatotoxicity," in *Arsenic*, John Wiley & Sons, Ltd, 2015, pp. 249–265. doi: 10.1002/9781118876992.ch11.
- [2] B. Wahlang *et al.*, "Toxicant-associated Steatohepatitis," *Toxicol. Pathol.*, vol. 41, no. 2, pp. 343–360, Feb. 2013, doi: 10.1177/0192623312468517.
- [3] G. Azeh Engwa, P. Udoka Ferdinand, F. Nweke Nwalo, and M. N. Unachukwu, "Mechanism and Health Effects of Heavy Metal Toxicity in Humans," in *Poisoning in the Modern World - New Tricks for an Old Dog?*, O. Karciglu and B. Arslan, Eds. IntechOpen, 2019. doi: 10.5772/intechopen.82511.
- [4] O. US EPA, "National Primary Drinking Water Regulations," *US EPA*, Nov. 30, 2015. <https://www.epa.gov/ground-water-and-drinking-water/national-primary-drinking-water-regulations> (accessed Nov. 01, 2020).
- [5] World Health Organization, Ed., *Guidelines for drinking-water quality*, 4th ed. Geneva: World Health Organization, 2011.
- [6] O. US EPA, "Basic Information about Your Drinking Water," Jun. 17, 2013. <https://www.epa.gov/ground-water-and-drinking-water/basic-information-about-your-drinking-water> (accessed Apr. 14, 2022).
- [7] Industrial Test Systems, "Quick Arsenic Test Kit," Dec. 2020. Accessed: Jan. 17, 2021. [Online]. Available: <https://sensafe.com/content/481396.pdf>
- [8] "Image Processing Toolbox - MATLAB." <https://www.mathworks.com/products/image.html> (accessed Apr. 09, 2022).
- [9] Suranjan Panigrahi, "Algorithmic framework for analysis of the images of the test strips/display paper linked with heavy metal sensor kit.," Purdue University, Internal Document, 2022.
- [10] "2-D Gaussian filtering of images - MATLAB imgaussfilt." <https://www.mathworks.com/help/images/ref/imgaussfilt.html> (accessed Apr. 09, 2022).
- [11] A. R. Smith, "Color gamut transform pairs," in *Proceedings of the 5th annual conference on Computer graphics and interactive techniques - SIGGRAPH '78*, Not Known, 1978, pp. 12–19. doi: 10.1145/800248.807361.
- [12] K. Choi, G. Lee, Y. J. Han, and J. M. Bunn, "Tomato Maturity Evaluation Using Color Image Analysis," *Trans. ASAE*, vol. 38, no. 1, pp. 171–176, 1995, doi: 10.13031/2013.27827.
- [13] A. Vadivel, S. Sural, and A. K. Majumdar, "Human color perception in the HSV space and its application in histogram generation for image retrieval," San Jose, CA, Jan. 2005, p. 598. doi: 10.1117/12.586823.

- [14] I. S. Ahmad and J. F. Reid, "Evaluation of Colour Representations for Maize Images," *J. Agric. Eng. Res.*, vol. 63, no. 3, pp. 185–195, Mar. 1996, doi: 10.1006/jaer.1996.0020.
- [15] K. Cantrell, M. M. Erenas, I. de Orbe-Payá, and L. F. Capitán-Vallvey, "Use of the Hue Parameter of the Hue, Saturation, Value Color Space As a Quantitative Analytical Parameter for Bitonal Optical Sensors," *Anal. Chem.*, vol. 82, no. 2, pp. 531–542, Jan. 2010, doi: 10.1021/ac901753c.
- [16] L. Georgieva, T. Dimitrova, and N. Angelov, "RGB and HSV colour models in colour identification of digital traumas images," in *International conference on computer systems and technologies*, 2005, vol. 12, no. 1.



## APPENDIX E. P4 - CODE

### 1. CODE TO RESIZE ALL IMAGES TO A STANDARD SIZE

```
%%%%%%%%%%
% Created on 010/25/2021
% Input: raw images
% Output: resized images
% Author: Ridhi Deo
% File name: obj3b_matlab_1.m (R2020b [8])
% Description: This code was written to resize all the images to a standard reference size
%%%%%%%%%%

clc;
clear all;
close all;
%% Load data
input_img = imread('input image');
%% Set up a standard size

% Load the standard image (image with minimum size)
std_img = imread('standard image');
[rowsstd, colstd, numberOfColorChannelsstd] = size(std_img);

%% Resize the image %each of the 20 images need to be read, one at a time

    out_img = imresize(input_img, [rowsstd, colstd]); %Lowest image size, found manually
    for the  cropped squares (50 ppb, R2)

%% Display
figure, imshow(input_img), figure, imshow(out_img)

%% Save image
imwrite(out_img, 'Output path')
```

### 2. CODE TO SMOOTH IMAGES

```
%%%%%%%%%%
% Created on 010/26/2021
% Input: resized images
% Output: smoothed images
% Author: Ridhi Deo
% File name: obj3b_matlab_2.m (R2020b [8]) )
% Description: This code was written to smooth all the resized images using a gaussian filter
```

```

%%%%%%%%%%
%% Apply a smoothing effect to the image
clc;
clear all;
close all;
%% Load data
% The input to this step is the resized image
input_img = imread(input_image');

%% Apply smoothing filter
out_img = imgaussfilt(input_img,2);

%% Display
figure, imshow(input_img), figure, imshow(out_img)

%% Save the smoothed image
imwrite(out_img,output_path)

```

### 3. CODE TO CONVERT RGB IMAGES TO HSV

```

%%%%%%%%%%
% Created on: 10/26/2021
% Input: Smoothed RGB images
% Output: smoothed HSV images
% Author: Ridhi Deo
% File name: obj3b_matlab_3.m (R2020b [8]) )
% Description: This code was written to convert the default RGB images into HSV scale
%%%%%%%%%%

clc;
clear all;
close all;

%% Load data
% Input to this function will be the smoothed images
input_img = imread(input_image); %each of the 20 images need to be read, one at a time

%% Convert to HSV
out_img = rgb2hsv(input_img);

%% Display
figure, imshow(input_img), figure, imshow(out_img)

%% Save image
imwrite(out_img,output_path)

```

#### 4. CODE TO COMPUTE MEANS AND SD

```
%%%%%%%%%%
% Created on: 11/08/2021
% Input: Smoothed HSV images
% Output: statistics related to each image and box plots
% Author: Ridhi Deo
% File name: obj3b_matlab_4
% Description: This code was written extract statistics and make box plots
%%%%%%%%%%

clear all;

%% Extracting Hue data
clear all;

R1_0_hue = xlsread(excel sheet path);(excel sheet path);
R1_0_mean_hue = mean(R1_0_hue,'all');
R1_0_SD_hue = std(R1_0_hue,0,'all');

R2_0_hue = xlsread(excel sheet path);(excel sheet path);
R2_0_mean_hue = mean(R2_0_hue,'all');
R2_0_SD_hue = std(R2_0_hue,0,'all');

R3_0_hue = xlsread(excel sheet path);
R3_0_mean_hue = mean(R3_0_hue,'all');
R3_0_SD_hue = std(R3_0_hue,0,'all');

R4_0_hue = xlsread(excel sheet path);
R4_0_mean_hue = mean(R4_0_hue,'all');
R4_0_SD_hue = std(R4_0_hue,0,'all');

R1_10_hue = xlsread(excel sheet path);
R1_10_mean_hue = mean(R1_10_hue,'all');
R1_10_SD_hue = std(R1_10_hue,0,'all');

R2_10_hue = xlsread(excel sheet path);
R2_10_mean_hue = mean(R2_10_hue,'all');
R2_10_SD_hue = std(R2_10_hue,0,'all');

R3_10_hue = xlsread(excel sheet path);
R3_10_mean_hue = mean(R3_10_hue,'all');
R3_10_SD_hue = std(R3_10_hue,0,'all');
```

```
R4_10_hue = xlsread(excel sheet path);  
R4_10_mean_hue = mean(R4_10_hue,'all');  
R4_10_SD_hue = std(R4_10_hue,0,'all');
```

```
R1_50_hue = xlsread(excel sheet path);  
R1_50_mean_hue = mean(R1_50_hue,'all');  
R1_50_SD_hue = std(R1_50_hue,0,'all');
```

```
R2_50_hue = xlsread(excel sheet path);  
R2_50_mean_hue = mean(R2_50_hue,'all');  
R2_50_SD_hue = std(R2_50_hue,0,'all');
```

```
R3_50_hue = xlsread(excel sheet path);  
R3_50_mean_hue = mean(R3_50_hue,'all');  
R3_50_SD_hue = std(R3_50_hue,0,'all');
```

```
R4_50_hue = xlsread(excel sheet path);  
R4_50_mean_hue = mean(R4_50_hue,'all');  
R4_50_SD_hue = std(R4_50_hue,0,'all');
```

```
R1_100_hue = xlsread(excel sheet path);  
R1_100_mean_hue = mean(R1_100_hue,'all');  
R1_100_SD_hue = std(R1_100_hue,0,'all');
```

```
R2_100_hue = xlsread(excel sheet path);  
R2_100_mean_hue = mean(R2_100_hue,'all');  
R2_100_SD_hue = std(R2_100_hue,0,'all');
```

```
R3_100_hue = xlsread(excel sheet path);  
R3_100_mean_hue = mean(R3_100_hue,'all');  
R3_100_SD_hue = std(R3_100_hue,0,'all');
```

```
R4_100_hue = xlsread(excel sheet path);  
R4_100_mean_hue = mean(R4_100_hue,'all');  
R4_100_SD_hue = std(R4_100_hue,0,'all');
```

```
R1_200_hue = xlsread(excel sheet path);  
R1_200_mean_hue = mean(R1_200_hue,'all');  
R1_200_SD_hue = std(R1_200_hue,0,'all');
```

```
R2_200_hue = xlsread(excel sheet path);
R2_200_mean_hue = mean(R2_200_hue,'all');
R2_200_SD_hue = std(R2_200_hue,0,'all');
```

```
R3_200_hue = xlsread(excel sheet path);
R3_200_mean_hue = mean(R3_200_hue,'all');
R3_200_SD_hue = std(R3_200_hue,0,'all');
```

```
R4_200_hue = xlsread(excel sheet path);
R4_200_mean_hue = mean(R4_200_hue,'all');
R4_200_SD_hue = std(R4_200_hue,0,'all');
```

```
Param_hue = {'0-mean','0-Std','10-mean','10-Std','50-mean','50-Std','100-mean','100-Std','200-mean','200-Std'};
```

```
R1_hue =
{R1_0_mean_hue;R1_0_SD_hue;R1_10_mean_hue;R1_10_SD_hue;R1_50_mean_hue;
R1_50_SD_hue;R1_100_mean_hue;R1_100_SD_hue;R1_200_mean_hue;R1_200_SD_h
ue};
R2_hue =
{R2_0_mean_hue;R2_0_SD_hue;R2_10_mean_hue;R2_10_SD_hue;R2_50_mean_hue;
R2_50_SD_hue;R2_100_mean_hue;R2_100_SD_hue;R2_200_mean_hue;R2_200_SD_h
ue};
R3_hue =
{R3_0_mean_hue;R3_0_SD_hue;R3_10_mean_hue;R3_10_SD_hue;R3_50_mean_hue;
R3_50_SD_hue;R3_100_mean_hue;R3_100_SD_hue;R3_200_mean_hue;R3_200_SD_h
ue};
R4_hue =
{R4_0_mean_hue;R4_0_SD_hue;R4_10_mean_hue;R4_10_SD_hue;R4_50_mean_hue;
R4_50_SD_hue;R4_100_mean_hue;R4_100_SD_hue;R4_200_mean_hue;R4_200_SD_h
ue};
```

```
Stats_hue = table(Param_hue, R1_hue,R2_hue,R3_hue, R4_hue);
Stats_hue = rows2vars(Stats_hue);
Stats_hue(1,:) = [];
Stats_hue.Properties.VariableNames = {'Param', '0-mean', '0-Std', ...
    '10-mean', '10-Std', '50-mean', '50-Std', '100-mean', ...
    '100-Std', '200-mean', '200-Std'};
```

```
%% Creating box plots of Hue means of replicates
```

```

Avg_0_hue = mean([R1_0_mean_hue, R2_0_mean_hue, R3_0_mean_hue,
R4_0_mean_hue]);
Avg_10_hue = mean([R1_10_mean_hue, R2_10_mean_hue, R3_10_mean_hue,
R4_10_mean_hue]);
Avg_50_hue = mean([R1_50_mean_hue, R2_50_mean_hue, R3_50_mean_hue,
R4_50_mean_hue]);
Avg_100_hue = mean([R1_100_mean_hue, R2_100_mean_hue, R3_100_mean_hue,
R4_100_mean_hue]);
Avg_200_hue = mean([R1_200_mean_hue, R2_200_mean_hue, R3_200_mean_hue,
R4_200_mean_hue]);

SD_0_hue = std([R1_0_mean_hue, R2_0_mean_hue, R3_0_mean_hue,
R4_0_mean_hue]);
SD_10_hue = std([R1_10_mean_hue, R2_10_mean_hue, R3_10_mean_hue,
R4_10_mean_hue]);
SD_50_hue = std([R1_50_mean_hue, R2_50_mean_hue, R3_50_mean_hue,
R4_50_mean_hue]);
SD_100_hue = std([R1_100_mean_hue, R2_100_mean_hue, R3_100_mean_hue,
R4_100_mean_hue]);
SD_200_hue = std([R1_200_mean_hue, R2_200_mean_hue, R3_200_mean_hue,
R4_200_mean_hue]);

x_hue = [0, 10, 50, 100, 200];
y_hue = [Avg_0_hue, Avg_10_hue, Avg_50_hue, Avg_100_hue, Avg_200_hue];
err_hue = [SD_0_hue, SD_10_hue, SD_50_hue, SD_100_hue, SD_200_hue];
figure;errorbar(x_hue,y_hue,err_hue), xlim([-5,210])
xlabel('Concentrations'), ylabel('Mean Hue values'), title('Hue Plot');

%% Extracting Sat data

R1_0_Sat = xlsread(excel sheet path);
R1_0_mean_Sat = mean(R1_0_Sat,'all');
R1_0_SD_Sat = std(R1_0_Sat,0,'all');

R2_0_Sat = xlsread(excel sheet path);
R2_0_mean_Sat = mean(R2_0_Sat,'all');
R2_0_SD_Sat = std(R2_0_Sat,0,'all');

R3_0_Sat = xlsread(excel sheet path);
R3_0_mean_Sat = mean(R3_0_Sat,'all');
R3_0_SD_Sat = std(R3_0_Sat,0,'all');

```

```
R4_0_Sat = xlsread(excel sheet path);  
R4_0_mean_Sat = mean(R4_0_Sat,'all');  
R4_0_SD_Sat = std(R4_0_Sat,0,'all');
```

```
R1_10_Sat = xlsread(excel sheet path);  
R1_10_mean_Sat = mean(R1_10_Sat,'all');  
R1_10_SD_Sat = std(R1_10_Sat,0,'all');
```

```
R2_10_Sat = xlsread(excel sheet path);  
R2_10_mean_Sat = mean(R2_10_Sat,'all');  
R2_10_SD_Sat = std(R2_10_Sat,0,'all');
```

```
R3_10_Sat = xlsread(excel sheet path);  
R3_10_mean_Sat = mean(R3_10_Sat,'all');  
R3_10_SD_Sat = std(R3_10_Sat,0,'all');
```

```
R4_10_Sat = xlsread(excel sheet path);  
R4_10_mean_Sat = mean(R4_10_Sat,'all');  
R4_10_SD_Sat = std(R4_10_Sat,0,'all');
```

```
R1_50_Sat = xlsread(excel sheet path);  
R1_50_mean_Sat = mean(R1_50_Sat,'all');  
R1_50_SD_Sat = std(R1_50_Sat,0,'all');
```

```
R2_50_Sat = xlsread(excel sheet path);  
R2_50_mean_Sat = mean(R2_50_Sat,'all');  
R2_50_SD_Sat = std(R2_50_Sat,0,'all');
```

```
R3_50_Sat = xlsread(excel sheet path);  
R3_50_mean_Sat = mean(R3_50_Sat,'all');  
R3_50_SD_Sat = std(R3_50_Sat,0,'all');
```

```
R4_50_Sat = xlsread(excel sheet path);  
R4_50_mean_Sat = mean(R4_50_Sat,'all');  
R4_50_SD_Sat = std(R4_50_Sat,0,'all');
```

```
R1_100_Sat = xlsread(excel sheet path);  
R1_100_mean_Sat = mean(R1_100_Sat,'all');  
R1_100_SD_Sat = std(R1_100_Sat,0,'all');
```

```
R2_100_Sat = xlsread(excel sheet path);
R2_100_mean_Sat = mean(R2_100_Sat,'all');
R2_100_SD_Sat = std(R2_100_Sat,0,'all');
```

```
R3_100_Sat = xlsread(excel sheet path);
R3_100_mean_Sat = mean(R3_100_Sat,'all');
R3_100_SD_Sat = std(R3_100_Sat,0,'all');
```

```
R4_100_Sat = xlsread(excel sheet path);
R4_100_mean_Sat = mean(R4_100_Sat,'all');
R4_100_SD_Sat = std(R4_100_Sat,0,'all');
```

```
R1_200_Sat = xlsread(excel sheet path);
R1_200_mean_Sat = mean(R1_200_Sat,'all');
R1_200_SD_Sat = std(R1_200_Sat,0,'all');
```

```
R2_200_Sat = xlsread(excel sheet path);
R2_200_mean_Sat = mean(R2_200_Sat,'all');
R2_200_SD_Sat = std(R2_200_Sat,0,'all');
```

```
R3_200_Sat = xlsread(excel sheet path);
R3_200_mean_Sat = mean(R3_200_Sat,'all');
R3_200_SD_Sat = std(R3_200_Sat,0,'all');
```

```
R4_200_Sat = xlsread(excel sheet path);
R4_200_mean_Sat = mean(R4_200_Sat,'all');
R4_200_SD_Sat = std(R4_200_Sat,0,'all');
```

```
Param_Sat = {'0-mean';'0-Std';'10-mean';'10-Std';'50-mean';'50-Std';'100-mean';'100-Std';'200-mean';'200-Std'};
```

```
R1_Sat =
{R1_0_mean_Sat;R1_0_SD_Sat;R1_10_mean_Sat;R1_10_SD_Sat;R1_50_mean_Sat;R1_50_S
D_Sat;R1_100_mean_Sat;R1_100_SD_Sat;R1_200_mean_Sat;R1_200_SD_Sat};
R2_Sat =
{R2_0_mean_Sat;R2_0_SD_Sat;R2_10_mean_Sat;R2_10_SD_Sat;R2_50_mean_Sat;R2_50_S
D_Sat;R2_100_mean_Sat;R2_100_SD_Sat;R2_200_mean_Sat;R2_200_SD_Sat};
R3_Sat =
{R3_0_mean_Sat;R3_0_SD_Sat;R3_10_mean_Sat;R3_10_SD_Sat;R3_50_mean_Sat;R3_50_S
D_Sat;R3_100_mean_Sat;R3_100_SD_Sat;R3_200_mean_Sat;R3_200_SD_Sat};
```



```

R4_Sat =
{R4_0_mean_Sat;R4_0_SD_Sat;R4_10_mean_Sat;R4_10_SD_Sat;R4_50_mean_Sat;R4_50_S
D_Sat;R4_100_mean_Sat;R4_100_SD_Sat;R4_200_mean_Sat;R4_200_SD_Sat};

Stats_Sat = table(Param_Sat, R1_Sat,R2_Sat,R3_Sat, R4_Sat);
Stats_Sat = rows2vars(Stats_Sat);
Stats_Sat(1,:) = [];
Stats_Sat.Properties.VariableNames = {'Param', '0-mean', '0-Std', ...
    '10-mean', '10-Std', '50-mean', '50-Std', '100-mean', ...
    '100-Std', '200-mean', '200-Std'};
%% Creating box plots of Sat means of replicates

    Avg_0_Sat = mean([R1_0_mean_Sat, R2_0_mean_Sat, R3_0_mean_Sat,
    R4_0_mean_Sat]);
    Avg_10_Sat = mean([R1_10_mean_Sat, R2_10_mean_Sat, R3_10_mean_Sat,
    R4_10_mean_Sat]);
    Avg_50_Sat = mean([R1_50_mean_Sat, R2_50_mean_Sat, R3_50_mean_Sat,
    R4_50_mean_Sat]);
    Avg_100_Sat = mean([R1_100_mean_Sat, R2_100_mean_Sat, R3_100_mean_Sat,
    R4_100_mean_Sat]);
    Avg_200_Sat = mean([R1_200_mean_Sat, R2_200_mean_Sat, R3_200_mean_Sat,
    R4_200_mean_Sat]);

SD_0_Sat = std([R1_0_mean_Sat, R2_0_mean_Sat, R3_0_mean_Sat, R4_0_mean_Sat]);
SD_10_Sat = std([R1_10_mean_Sat, R2_10_mean_Sat, R3_10_mean_Sat,
    R4_10_mean_Sat]);
SD_50_Sat = std([R1_50_mean_Sat, R2_50_mean_Sat, R3_50_mean_Sat,
    R4_50_mean_Sat]);
SD_100_Sat = std([R1_100_mean_Sat, R2_100_mean_Sat, R3_100_mean_Sat,
    R4_100_mean_Sat]);
SD_200_Sat = std([R1_200_mean_Sat, R2_200_mean_Sat, R3_200_mean_Sat,
    R4_200_mean_Sat]);

x_Sat = [0, 10, 50, 100, 200];
y_Sat = [Avg_0_Sat, Avg_10_Sat, Avg_50_Sat, Avg_100_Sat, Avg_200_Sat];
err_Sat = [SD_0_Sat, SD_10_Sat, SD_50_Sat, SD_100_Sat, SD_200_Sat];
figure;errorbar(x_Sat,y_Sat,err_Sat), xlim([-5,210])
xlabel('Concentrations'), ylabel('Mean Sat values'), title('Saturation Plot');

%% Exporting results into an excel table
writetable(Stats_hue, 'Stats_Hue_Comkit.xlsx');
writetable(Stats_Sat, 'Stats_Sat_Comkit.xlsx');

```

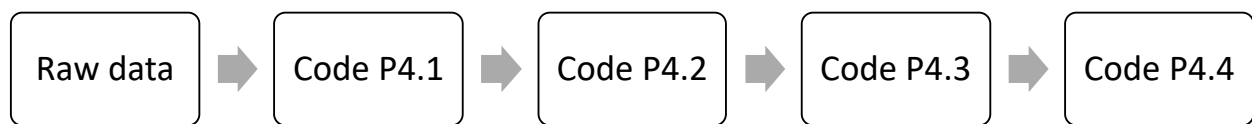


Figure P4.1: Figure outlining the flow of code used in this research objective

## GENERAL CONCLUSIONS

In this research, models were developed in paper 1 to predict Hepatic Steatosis (HS) using ML-techniques. In developing decision support systems based on ML for HS prediction, a hierarchical approach was used in exploring different input parameters. Models developed as part of objective 1A used only six physiological parameters. The models in objective 1B used seven physiological and liver biochemistry parameters. In paper 2, three selected models (each for male and female specific populations) from objective 1B were evaluated from an interpretability perspective to include model transparency.

In paper 3, ML-based models were developed to test the impact of including vs excluding a cluster of heavy metal exposure data (Lead, Iron, and Cadmium). In paper 4, a commercially available Arsenic detection kit was tested, and the results were examined visually and using a customized image analysis algorithm.

Based on the work described above, the following are concluded:

1. Physiological parameters alone can predict HS using 79% accuracy, 76% sensitivity and 82% sensitivity.
2. Models with only seven parameters (vital and liver biochemical) led screening models with sensitivities of 82.6% for male-specific models and 86% for female specific models. It is logical to use both physiological and liver biochemical parameters to maximize the sensitivity and therefore, screening capability of these models.
3. The best performing models from an explainability perspective were identified as Quadratic SVM in male population and Gaussian SVM scale 1 in female population.
4. The top three independent predictors for male and female data were identified using the mean of the partial dependencies. In both sexes, ALT, AST, and Glucose were found to be the most individually contributing features. These three parameters were found to be individually contributing highly to HS prediction.
5. Results for male and female populations were found to vary slightly with male models outperforming the female models in terms of alignment with clinical normal values.
6. A theoretical framework for developing hybrid models is provided in sections 3.5.1 d and 3.5.2 d.

7. Inclusion of heavy metal exposure (Lead, Iron, and Cadmium) did not have a numerically significant impact on the model performance in predicting HS.
8. The minimum concentration identifiable using the commercial kit was 100 ppb (under lab conditions).
9. The commercial kit was not able to differentiate the following concentrations based on our testing: 0 vs 10 ppb, 10 vs 50 ppb and 100 vs 200 ppb. The 50-ppb vs 100-ppb concentrations (Arsenic in water) were the most differentiable at 75% PDIF.
10. The increase in Euclidean distance metric (computed using Hue and saturation) was not linear with increase in concentration of Arsenic in water.

## GENERAL REFERENCES

- [1] Suranjan Panigrahi, “A comprehensive system-based representation of the liver disease and it associated components. Internal document.” Purdue University, West Lafayette, IN, 2019.
- [2] Z. M. Younossi, A. B. Koenig, D. Abdelatif, Y. Fazel, L. Henry, and M. Wymer, “Global epidemiology of nonalcoholic fatty liver disease-Meta-analytic assessment of prevalence, incidence, and outcomes,” *Hepatology*, vol. 64, no. 1, pp. 73–84, Jul. 2016, doi: 10.1002/hep.28431.
- [3] D. L. Wyness, “The four stages of Non-Alcoholic Fatty Liver Disease (NAFLD),” *liver-health-uk*, Oct. 09, 2017. <https://www.liverhealthuk.com/post/the-four-stages-of-naflD> (accessed Mar. 11, 2022).
- [4] “OPTN/SRTR 2018 Annual Data Report: Liver.” [https://srtr.transplant.hrsa.gov/annual\\_reports/2018/Liver.aspx](https://srtr.transplant.hrsa.gov/annual_reports/2018/Liver.aspx) (accessed Aug. 28, 2020).
- [5] N. Chalasani *et al.*, “The diagnosis and management of nonalcoholic fatty liver disease: Practice guidance from the American Association for the Study of Liver Diseases: Hepatology,” *Hepatology*, vol. 67, no. 1, pp. 328–357, Jan. 2018, doi: 10.1002/hep.29367.
- [6] J.-Y. Chung, S.-D. Yu, and Y.-S. Hong, “Environmental Source of Arsenic Exposure,” *J. Prev. Med. Pub. Health*, vol. 47, no. 5, pp. 253–257, Sep. 2014, doi: 10.3961/jpmph.14.036.
- [7] Hopenhayn-Rich C, Biggs M L, Smith A H, Kalman D A, and Moore L E, “Methylation study of a population environmentally exposed to arsenic in drinking water,” *EnvIron. Health Perspect.*, vol. 104, no. 6, pp. 620–628, Jun. 1996, doi: 10.1289/ehp.96104620.
- [8] H. Ali, E. Khan, and I. Ilahi, “Environmental Chemistry and Ecotoxicology of Hazardous Heavy Metals: Environmental Persistence, Toxicity, and Bioaccumulation,” *J. Chem.*, vol. 2019, pp. 1–14, Mar. 2019, doi: 10.1155/2019/6730305.
- [9] G. E. Arteel, “Hepatotoxicity,” in *Arsenic*, John Wiley & Sons, Ltd, 2015, pp. 249–265. doi: 10.1002/9781118876992.ch11.
- [10] B. Wahlang *et al.*, “Toxicant-associated Steatohepatitis,” *Toxicol. Pathol.*, vol. 41, no. 2, pp. 343–360, Feb. 2013, doi: 10.1177/0192623312468517.
- [11] M. Cave *et al.*, “Toxicant-associated steatohepatitis in vinyl chloride workers,” *Hepatology*, vol. 51, no. 2, pp. 474–481, Feb. 2010, doi: 10.1002/hep.23321.

- [12] A. R. Murali and W. D. Carey, "Liver Test Interpretation - Approach to the Patient with Liver Disease: A Guide to Commonly Used Liver Tests," *Cleveland Clinic - Center for Continuing Education*, Apr. 2014.  
<https://www.clevelandclinicmeded.com/medicalpubs/diseasemanagement/hepatology/guide-to-common-liver-tests/> (accessed Oct. 27, 2020).
- [13] L. C. Bertot *et al.*, "Nonalcoholic fatty liver disease-related cirrhosis is commonly unrecognized and associated with hepatocellular carcinoma: Hepatology Communications, Month 2017," *Hepatol. Commun.*, vol. 1, no. 1, pp. 53–60, Feb. 2017, doi: 10.1002/hep4.1018.
- [14] R. Loomba, "Role of imaging-based biomarkers in NAFLD: Recent advances in clinical application and future research directions," *J. Hepatol.*, vol. 68, no. 2, pp. 296–304, Feb. 2018, doi: 10.1016/j.jhep.2017.11.028.
- [15] L. A. Adams *et al.*, "The Natural History of Nonalcoholic Fatty Liver Disease: A Population-Based Cohort Study," *Gastroenterology*, vol. 129, no. 1, pp. 113–121, Jul. 2005, doi: 10.1053/j.gastro.2005.04.014.
- [16] S. Mitra, A. De, and A. Chowdhury, "Epidemiology of non-alcoholic and alcoholic fatty liver diseases," *Transl. Gastroenterol. Hepatol.*, vol. 5, pp. 16–16, Apr. 2020, doi: 10.21037/tgh.2019.09.08.
- [17] A. Ofosu, "Non-alcoholic fatty liver disease: controlling an emerging epidemic, challenges, and future directions," *Ann. Gastroenterol.*, 2018, doi: 10.20524/aog.2018.0240.
- [18] E. Carey, A. Wieckowska, and W. D. Carey, "Nonalcoholic Fatty Liver Disease," *Cleveland Clinic - Center for Continuing Education*, Mar. 2013.  
<https://www.clevelandclinicmeded.com/medicalpubs/diseasemanagement/hepatology/non-alcoholic-fatty-liver-disease-march-13/> (accessed Oct. 22, 2020).
- [19] S. C. Lin *et al.*, "Noninvasive Diagnosis of Nonalcoholic Fatty Liver Disease and Quantification of Liver Fat Using a New Quantitative Ultrasound Technique," *Clin. Gastroenterol. Hepatol.*, vol. 13, no. 7, pp. 1337–1345.e6, Jul. 2015, doi: 10.1016/j.cgh.2014.11.027.
- [20] J. S. Paige *et al.*, "A Pilot Comparative Study of Quantitative Ultrasound, Conventional Ultrasound, and MRI for Predicting Histology-Determined Steatosis Grade in Adult Nonalcoholic Fatty Liver Disease," *Am. J. Roentgenol.*, vol. 208, no. 5, pp. W168–W177, May 2017, doi: 10.2214/AJR.16.16726.
- [21] A. Han *et al.*, "Inter-platform reproducibility of ultrasonic attenuation and backscatter coefficients in assessing NAFLD," *Eur. Radiol.*, vol. 29, no. 9, pp. 4699–4708, Sep. 2019, doi: 10.1007/s00330-019-06035-9.

- [22] J. Choi *et al.*, “Mercury Exposure in Association With Decrease of Liver Function in Adults: A Longitudinal Study,” *J. Prev. Med. Pub. Health*, vol. 50, no. 6, pp. 377–385, Nov. 2017, doi: 10.3961/jpmph.17.099.
- [23] M. Colombo *et al.*, “EASL Clinical Practice Guideline: Occupational liver diseases,” *J. Hepatol.*, vol. 71, no. 5, pp. 1022–1037, Nov. 2019, doi: 10.1016/j.jhep.2019.08.008.
- [24] I. Palma-Lara *et al.*, “Arsenic exposure: A public health problem leading to several cancers,” *Regul. Toxicol. Pharmacol.*, vol. 110, p. 104539, Feb. 2020, doi: 10.1016/j.yrtph.2019.104539.
- [25] O. US EPA, “Learn about Lead,” *US EPA*, Feb. 12, 2013.  
<https://www.epa.gov/lead/learn-about-lead> (accessed Nov. 28, 2020).
- [26] World Health Organization, “Chemical fact Sheets,” in *Guidelines for drinking -water quality*, 4th ed., Geneva: World Health Organization, 2011. Accessed: Nov. 01, 2020. [Online]. Available: [https://www.who.int/water\\_sanitation\\_health/publications/2011/9789241548151\\_ch12.pdf](https://www.who.int/water_sanitation_health/publications/2011/9789241548151_ch12.pdf)
- [27] H. S. Kim, Y. J. Kim, and Y. R. Seo, “An Overview of Carcinogenic Heavy Metal: Molecular Toxicity Mechanism and Prevention,” *J. Cancer Prev.*, vol. 20, no. 4, pp. 232–240, Dec. 2015, doi: 10.15430/JCP.2015.20.4.232.
- [28] World Health Organization, Ed., *Guidelines for drinking-water quality*, 4th ed. Geneva: World Health Organization, 2011.
- [29] M. Balali-Mood, K. Naseri, Z. Tahergorabi, M. R. Khazdair, and M. Sadeghi, “Toxic Mechanisms of Five Heavy Metals: Mercury, Lead, Chromium, Cadmium, and Arsenic,” *Front. Pharmacol.*, vol. 12, p. 643972, Apr. 2021, doi: 10.3389/fphar.2021.643972.
- [30] Suranjan Panigrahi, “Verbal discussion of the need of heavy metal detection sensor at the household level.” Purdue University, West Lafayette, IN, 2020.
- [31] S. Panigrahi, “A System-Based Analysis and Approach for Water-Linked Health and Wellness in Low-Resource Setting,” p. 7.
- [32] T. D. Ellington, B. Momin, R. J. Wilson, S. J. Henley, M. Wu, and A. B. Ryerson, “Incidence and Mortality of Cancers of the Biliary Tract, Gallbladder, and Liver by Sex, Age, Race/Ethnicity, and Stage at Diagnosis: United States, 2013 to 2017,” *Cancer Epidemiol. Biomarkers Prev.*, vol. 30, no. 9, pp. 1607–1614, Sep. 2021, doi: 10.1158/1055-9965.EPI-21-0265.
- [33] S. U. Khan, A. Y. Zomaya, and A. Abbas, Eds., *Handbook of Large-Scale Distributed Computing in Smart Healthcare*. Cham: Springer International Publishing, 2017. doi: 10.1007/978-3-319-58280-1.

- [34] K. Ganapathy, “Artificial Intelligence and Healthcare Regulatory and Legal Concerns,” *Telehealth Med. Today*, Apr. 2021, doi: 10.30953/tmt.v6.252.
- [35] C. for D. and R. Health, “Artificial Intelligence and Machine Learning in Software as a Medical Device,” *FDA*, Oct. 2020, Accessed: Nov. 30, 2020. [Online]. Available: <https://www.fda.gov/medical-devices/software-medical-device-samd/artificial-intelligence-and-machine-learning-software-medical-device>
- [36] W. Nicholson Price II, “Artificial Intelligence in Health Care: Applications and Legal Implications,” *SciTech Lawyer*, vol. 14, no. 1, p. 5, 2017.
- [37] O. Asan, A. E. Bayrak, and A. Choudhury, “Artificial Intelligence and Human Trust in Healthcare: Focus on Clinicians,” *J. Med. Internet Res.*, vol. 22, no. 6, p. e15154, Jun. 2020, doi: 10.2196/15154.
- [38] S. N. Payrovnaziri *et al.*, “Explainable artificial intelligence models using real-world electronic health record data: a systematic scoping review,” *J. Am. Med. Inform. Assoc.*, vol. 27, no. 7, pp. 1173–1185, Jul. 2020, doi: 10.1093/jamia/ocaa053.
- [39] A. Shaban-Nejad, M. Michalowski, and D. L. Buckeridge, “Explainability and Interpretability: Keys to Deep Medicine,” in *Explainable AI in Healthcare and Medicine*, vol. 914, A. Shaban-Nejad, M. Michalowski, and D. L. Buckeridge, Eds. Cham: Springer International Publishing, 2021, pp. 1–10. doi: 10.1007/978-3-030-53352-6\_1.
- [40] F. K. Došilović, M. Brčić, and N. Hlupić, “Explainable artificial intelligence: A survey,” in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2018, pp. 0210–0215. doi: 10.23919/MIPRO.2018.8400040.
- [41] G. Vilone and L. Longo, “Explainable Artificial Intelligence: a Systematic Review,” *ArXiv200600093 Cs*, Oct. 2020, Accessed: Feb. 24, 2022. [Online]. Available: <http://arxiv.org/abs/2006.00093>
- [42] M. Ghassemi, L. Oakden-Rayner, and A. L. Beam, “The false hope of current approaches to explainable artificial intelligence in health care,” *Lancet Digit. Health*, vol. 3, no. 11, pp. e745–e750, Nov. 2021, doi: 10.1016/S2589-7500(21)00208-9.
- [43] C. for D. and R. Health, “Software as a Medical Device (SAMD): Clinical Evaluation,” *U.S. Food and Drug Administration*, Mar. 02, 2020. <https://www.fda.gov/regulatory-information/search-fda-guidance-documents/software-medical-device-samd-clinical-evaluation> (accessed Nov. 30, 2020).
- [44] T. W. Kim and B. R. Routledge, “Informational Privacy, A Right to Explanation, and Interpretable AI,” in *2018 IEEE Symposium on Privacy-Aware Computing (PAC)*, Washington, DC, Sep. 2018, pp. 64–74. doi: 10.1109/PAC.2018.00013.



- [45] L. Gordon, T. Grantcharov, and F. Rudzicz, “Explainable Artificial Intelligence for Safe Intraoperative Decision Support,” *JAMA Surg.*, vol. 154, no. 11, pp. 1064–1065, Nov. 2019, doi: 10.1001/jamasurg.2019.2821.
- [46] S. M. Lauritsen *et al.*, “Explainable artificial intelligence model to predict acute critical illness from electronic health records,” *Nat. Commun.*, vol. 11, no. 1, p. 3852, Jul. 2020, doi: 10.1038/s41467-020-17431-x.
- [47] N. Mirchi, V. Bissonnette, R. Yilmaz, N. Ledwos, A. Winkler-Schwartz, and R. F. Del Maestro, “The Virtual Operative Assistant: An explainable artificial intelligence tool for simulation-based training in surgery and medicine,” *PLOS ONE*, vol. 15, no. 2, p. e0229596, Feb. 2020, doi: 10.1371/journal.pone.0229596.
- [48] J. Peng *et al.*, “An Explainable Artificial Intelligence Framework for the Deterioration Risk Prediction of Hepatitis Patients,” *J. Med. Syst.*, vol. 45, no. 5, p. 61, May 2021, doi: 10.1007/s10916-021-01736-5.
- [49] Erik J. Tokar, Windy A. Boyd, Jonathan H. Freedman, and Michael P. Waalkes, “Chapter 23: Toxic Effects of Metals,” in *Essentials of Toxicology*, 3rd ed., McGraw Hill Professional, 2015.
- [50] O. Wada, “What are Trace Elements ? — Their deficiency and excess states —,” *undefined*, 2004, Accessed: Feb. 23, 2022. [Online]. Available: <https://www.semanticscholar.org/paper/What-are-Trace-Elements-%E2%80%94-Their-deficiency-and-%E2%80%94-Wada/1ab535fdde462592695589f05280de9ba8cffab0>
- [51] A. P. Ebokaiwe *et al.*, “Assessment of heavy metals around Abakaliki metropolis and potential bioaccumulation and biochemical effects on the liver, kidney, and erythrocyte of rats,” *Hum. Ecol. Risk Assess. Int. J.*, vol. 24, no. 5, pp. 1233–1255, Jul. 2018, doi: 10.1080/10807039.2017.1410695.
- [52] D. N. G. Mazumder *et al.*, “Chronic arsenic toxicity from drinking tubewell water in rural West Bengal,” p. 8.
- [53] H. Lee, Y. Kim, C.-S. Sim, J.-O. Ham, N.-S. Kim, and B.-K. Lee, “Associations between blood mercury levels and subclinical changes in liver enzymes among South Korean general adults: Analysis of 2008–2012 Korean national health and nutrition examination survey data,” *EnvIron. Res.*, vol. 130, pp. 14–19, Apr. 2014, doi: 10.1016/j.envres.2014.01.005.
- [54] R. Khan *et al.*, “Toxicological effects of toxic metals (Cadmium and mercury) on blood and the thyroid gland and pharmacological intervention by vitamin C in rabbits,” *EnvIron. Sci. Pollut. Res.*, vol. 26, no. 16, pp. 16727–16741, Jun. 2019, doi: 10.1007/s11356-019-04886-9.
- [55] A. Sharma and S. Nagalli, “Chronic Liver Disease,” in *StatPearls*, Treasure Island (FL): StatPearls Publishing, 2022. Accessed: Feb. 26, 2022. [Online]. Available: <http://www.ncbi.nlm.nih.gov/books/NBK554597/>

- [56] P. B. Tchounwou, C. G. Yedjou, A. K. Patlolla, and D. J. Sutton, “Heavy Metal Toxicity and the Environment,” in *Molecular, Clinical and Environmental Toxicology*, vol. 101, A. Luch, Ed. Basel: Springer Basel, 2012, pp. 133–164. doi: 10.1007/978-3-7643-8340-4\_6.
- [57] G. Flora, D. Gupta, and A. Tiwari, “Toxicity of lead: a review with recent updates,” *Interdiscip. Toxicol.*, vol. 5, no. 2, pp. 47–58, Nov. 2012, doi: 10.2478/v10102-012-0009-2.
- [58] S. T. Cherng, J. Tam, P. J. Christine, and R. Meza, “Modeling the Effects of E-cigarettes on Smoking Behavior: Implications for Future Adult Smoking Prevalence,” *Epidemiology*, vol. 27, no. 6, pp. 819–826, Nov. 2016, doi: 10.1097/EDE.0000000000000497.
- [59] International Agency for Research on Cancer and Weltgesundheitsorganisation, Eds., *IARC monographs on the evaluation of carcinogenic risks to humans, volume 100 C, arsenic, metals, fibres, and dusts: this publication represents the views and expert opinions of an IARC Working Group on the Evaluation of Carcinogenic Risks to Humans, which met in Lyon, 17 - 24 March 2009*. Lyon: IARC, 2012.
- [60] V. M. Nurchi, A. Buha Djordjevic, G. Crisponi, J. Alexander, G. Bjørklund, and J. Aaseth, “Arsenic Toxicity: Molecular Targets and Therapeutic Agents,” *Biomolecules*, vol. 10, no. 2, p. 235, Feb. 2020, doi: 10.3390/biom10020235.
- [61] “ATSDR - Medical Management Guidelines (MMGs): Arsine.” <https://www.atsdr.cdc.gov/MMG/MMG.asp?id=1199&tid=278> (accessed Nov. 28, 2020).
- [62] “Arsenic Toxicity | Winchester Hospital.” <https://www.winchesterhospital.org/health-library/article?id=120795> (accessed Nov. 28, 2020).
- [63] Cave Matt, Appana Savitri, Patel Mihir, Falkner Keith Cameron, McClain Craig J., and Brock Guy, “Polychlorinated Biphenyls, Lead, and Mercury Are Associated with Liver Disease in American Adults: NHANES 2003–2004,” *EnvIron. Health Perspect.*, vol. 118, no. 12, pp. 1735–1742, Dec. 2010, doi: 10.1289/ehp.1002720.
- [64] A. Michailova, T. Kuneva, and T. Popov, “A comparative assessment of liver function in workers in the petroleum industry,” *Int. Arch. Occup. EnvIron. Health*, vol. 71 Suppl, pp. S46–49, Sep. 1998.
- [65] H. P. Cotrim, Z. A. Andrade, R. Parana, M. Portugal, L. G. Lyra, and L. A. R. Freitas, “Nonalcoholic steatohepatitis: a toxic liver disease in industrial workers,” *Liver Int.*, vol. 19, no. 4, pp. 299–304, Aug. 1999, doi: 10.1111/j.1478-3231.1999.tb00053.x.
- [66] O. US EPA, “Potential Well Water Contaminants and Their Impacts,” May 06, 2015. <https://www.epa.gov/privatewells/potential-well-water-contaminants-and-their-impacts> (accessed Mar. 14, 2022).

- [67] H. Zhai *et al.*, “Blood lead level is associated with non-alcoholic fatty liver disease in the Yangtze River Delta region of China in the context of rapid urbanization,” *Environ. Health*, vol. 16, no. 1, p. 93, Aug. 2017, doi: 10.1186/s12940-017-0304-7.
- [68] T. Luo *et al.*, “Chronic exposure to low doses of Pb induces hepatotoxicity at the physiological, biochemical, and transcriptomic levels of mice,” *Environ. Toxicol.*, vol. 34, no. 4, pp. 521–529, 2019, doi: 10.1002/tox.22706.
- [69] E. Georgieva *et al.*, “Histological and biochemical changes in liver of common carp (*Cyprinus carpio* L.) under metal exposure,” p. 10.
- [70] A. Santra *et al.*, “Hepatic Damage Caused by Chronic Arsenic Toxicity in Experimental Animals,” *J. Toxicol. Clin. Toxicol.*, vol. 38, no. 4, pp. 395–405, Jan. 2000, doi: 10.1081/CLT-100100949.
- [71] A. H. Welch, D. B. Westjohn, D. R. Helsel, and R. B. Wanty, “Arsenic in Ground Water of the United States: Occurrence and Geochemistry,” *Groundwater*, vol. 38, no. 4, pp. 589–604, 2000, doi: 10.1111/j.1745-6584.2000.tb00251.x.

# APPENDIX F. NON-ALCOHOLIC FATTY LIVER DISEASE (NAFLD) FROM MULTIPLE SCIENTIFIC PERSPECTIVES AND CONTEMPORARY REVIEWS

*This manuscript was submitted to a peer review journal – IEEE EMB – Reviews in Biomedical Engineering in April 2021. The manuscript is currently under-review.*

Ridhi Deo, Suranjan Panigrahi, and Edward Liechty

**Abstract:** In the past three decades, the prevalence of NAFLD has been rising consistently across the globe, including in the USA. The progression of NAFLD into cirrhosis, fibrosis, and other complications is predicted to cause a high clinical burden in the upcoming years. Although NAFLD prevalence is increasing along with the increasing co-morbidities of obesity and diabetes, the exact cause of NAFLD is unknown. Non-invasive and low-cost screening options for NAFLD are limited to ultrasound-based imaging, while liver biopsy is the required benchmark for diagnosis. Lack of specific biomarkers makes it challenging to identify and screen the disease early-on. Moreover, chronic heavy metal exposure (Arsenic, Lead, Mercury, and Cadmium) was found to be associated with NAFLD occurrence. Considering the complex pathway and multiple factors associated with NAFLD, it is appropriate to conduct a benchmark literature review related to this disease (NAFLD) and its associated factors. Thus, this mini review identifies: 1) Recent research related to the enzymes indicative of NAFLD 2) Overlap between NAFLD and heavy metal exposure and, 3) Contemporary tools and techniques being researched for NAFLD detection. This review also reflects on NAFLD as an associated risk factor for other diseases like PCOS, cardiovascular diseases, and hepatocellular carcinoma.

## I. INTRODUCTION

The liver plays a key role in xenobiotic metabolism while maintaining the body's metabolic homeostasis and synthesizing carbohydrates and lipids. It is the most important detoxifying organ in the body and is therefore highly susceptible to toxicity [1],[2].

There are multiple sources of liver problems, caused by external or internal agents.

The liver's functionality can be impaired by external agents like viruses or toxins. Additionally, liver functionality could also be impaired due to cancer or other genetic conditions (e.g.: Wilson's disease).

Viruses cause liver diseases like hepatitis A, B, and C in humans. Toxins, on the other hand, can cause liver diseases by accumulation. The toxins that accumulate in the liver can stem from various sources like medicinal drugs, alcohol consumption, or heavy metal exposure. In this review, the focus is on heavy metal-induced toxicity and its impact on liver functionality. Heavy metals: Arsenic (As), Lead (Pb), Mercury (Hg), and Cadmium (Cd) have been previously researched in the context of toxicity and their impact on the human body [1]–[13], [13]–[31].

The major source of heavy metal exposure in humans occurs via drinking water [11], food [26] and, environmental exposure - particularly due to occupational hazards [32], [33]. These heavy metals usually accumulate over time in the liver, causing it to inflame and eventually leading to liver dysfunction [1], [15].

Liver diseases initiated by exposure to toxins or other pollutants like heavy metals, tend to progress with time. Chronic liver diseases either caused by or due to excessive

alcohol consumption, exposure to heavy metals or other toxic chemicals, etc. lead to fibrosis, cirrhosis, and eventually cancer, when left undiagnosed and/or untreated. Non-alcoholic fatty liver disease (NAFLD) is a chronic disease, similar in progression to alcoholic fatty liver disease (ALD) [34].

The scope of this review paper is limited to the consequences of exposure to heavy metals on the liver, leading up to or progressing the NAFLD condition.

It is important to point out that the exact cause for NAFLD is unknown [35], however, heavy metal exposure is a known risk factor for NAFLD [1], [11], [36]. Recent studies have indicated the occurrence of NAFLD in individuals with the exposure of heavy metals Hg [21], As [1], and Pb [14]. Moreover, one of the causes of liver enzyme modification was found to be linked with the presence of heavy metals (in adult mice) [26]. Thus, this paper focuses on the impact of specific heavy metals (Hg, As, Cd, and Pb) on the liver, with a special emphasis on NAFLD.

The prevalence of NAFLD has been increasing in the recent years [37]. Increased urbanization led by industrialization has encouraged a sedentary lifestyle in many parts of the world [14], [38], [39]. This shift has resulted in increasing cases of obesity worldwide. One of the outcomes of this shift, combined with rising obesity, has been an increase in the prevalence of NAFLD [40]. Younossi et al. identified a 10% increase in the global prevalence of NAFLD between 2005 to 2010 [37]. The prevalence of NAFLD by country is shown in figure 1 [37]. The prevalence estimate (as of 2016) included studies from 1989 – 2015, with a sample size of 8,515,431 adults, in a meta-analysis by Younossi et al. [37].

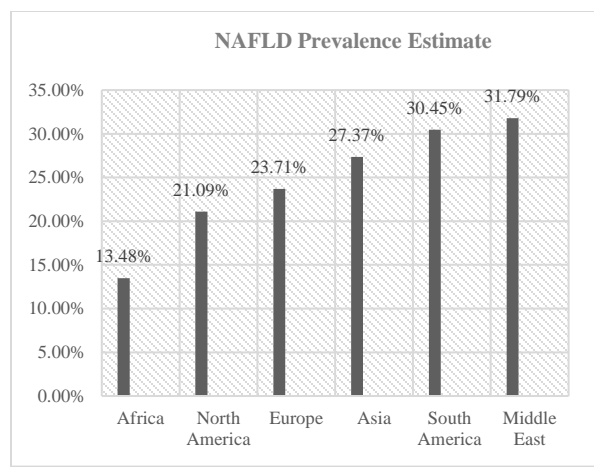


Fig 1. Prevalence estimate of NAFLD around the world [37]

Considering that an exact cause for NAFLD is not known [35], [41], and its increasing prevalence, a systems perspective is needed, accounting the different factors linked to the liver and its functionality. For instance, it is important to know the pathway of heavy metals from the environment into the human system and further impacting the liver. Thus, we justify conducting a state-of-art literature review related to the topics described below:

- 1) The enzymes indicative of non-alcoholic fatty liver disease (NAFLD) and related liver diseases
- 2) Effect of heavy metal exposure on biomarkers/enzymes indicative of liver functionality
- 3) Emerging biomarkers and techniques for NAFLD detection

Typical biomarkers related to NAFLD and more specific biomarkers in the context of heavy metal toxicity are reviewed in this paper. Biomarkers at different levels of abstraction – protein biomarkers, gene mutations, and enzyme biomarkers are included in this paper. Detection strategies based on these biomarkers are also discussed. Further, detection devices based on specific biomarkers are part of the review as well.

In this review, three databases (PubMed, Engineering Village, and Web of Science) were used to find relevant literature. Combinations of keywords: heavy metal toxicity, liver functionality, liver enzymes, liver dysfunction, NAFLD biomarkers and, mathematical models were used. Papers published in the last six years (2014- 20) were included. A total of 676 abstracts were initially obtained. After abstract review and de-duplication, 97 papers were selected for further review. Of these 97 papers, selected studies that are related to this review's research objectives are cited in our paper, along with other cross-references that stemmed from the literature. The organization of the review is shown in fig 2.

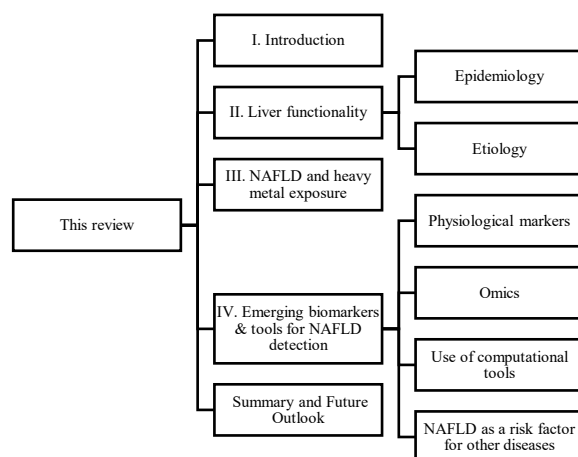


Fig 2. Organization of this review

## II. LIVER FUNCTIONALITY

*Liver's role in detoxification:* The specificity of the liver's role in detoxifying can be highlighted using the results of research, specifically with liver tissues in animal models [9], [42].

Geng et al. researched five specific tissues of fish species in Shanghai [42]. They investigated the nutritional and toxic contents in the dorsal, abdominal and tail muscles and also liver, and abdominal fat tissues [42]. Hg levels in all tissues were found to be under the limit of 500 µg/kg, except in the liver tissue; Hg levels were very high (range: < 500 – 1567 µg/kg) in the liver compared to other muscle tissues in their study [42]. The disproportionate amount of toxicity in the liver tissue compared to other muscles highlights the burden of toxic exposure specifically on the liver.

Another research group explored the contamination levels in fish from a reservoir, in an area with rigorous copper mining [9]. The heavy metal concentrations in the livers of fish in the reservoir (N = 45, 15 per season) vs water heavy metal concentrations in the reservoir itself (three seasons) were measured. They found the metal concentrations in the fish livers to be higher than in the reservoir ( $p < 0.05$ ) [9]. The aspartate aminotransferase (AST) activity was also found to be significantly different ( $p < 0.05$ ) [9]. Furthermore, hepatic histological alterations were found in the fish livers, including degenerations, necrosis and, blood vessel changes, indicating liver dysfunctionality at the tissue and cellular level when exposed to metal-contaminated water [9].

The association between liver health and heavy metal exposure was also explored in human models by some researchers. These are elaborated on in section III of this paper.

As mentioned in section I of this paper, a liver's functionality can be impaired by various external agents like viruses (cause hepatitis A, B, C), toxins (cause fatty liver disease (FLD), cirrhosis), or genetic conditions (that cause Wilson's disease) [43]. This paper focuses on Non-Alcoholic Fatty Liver Disease (NAFLD). NAFLD is

described by the storage of additional liver fat, leading to either a simple fatty liver or Non-Alcoholic Steatohepatitis (NASH) [41]. NASH is defined as the presence of liver inflammation and liver damage in addition to fat accumulation [41]. NASH can lead to further complications over time. Liver cirrhosis and liver cancer are complications associated with NASH [41]. For patients with severe liver cirrhosis leading to organ failure, a liver transplant is usually required for survival [41]. The incidence of NAFLD is increasing in both developed and developing countries [37].

In a multiethnic study of 106,458 individuals living in the USA, NAFLD was found to be the major cause (52%) of chronic liver diseases (CLD) [44].

Current clinical practices define the normal ranges of clinical parameters (enzymes, proteins) from the liver function test as follows:

- i) ALT (alanine aminotransferase): 29-33 U/L for men, 19-25 U/L for women [45]
- ii) AST (aspartate aminotransferase): 0-35 U/L [46]
- iii) ALP (alkaline phosphatase): 30-120 U/L [46]
- iv) Albumin: 4.0-6.0 g/dL [46]
- v) GGT (gamma-glutamyl transferase): 0-30 U/L [46]
- vi) Bilirubin: 2-17  $\mu$ M/L [46]

In this review, the guidelines and references related to liver diseases in the USA are included. Other guidelines from Europe, Asia Pacific etc. are not a part of this review. They can be found in the references [8], [47], [48]. Although clinical practitioners conduct imaging tests or biopsies for confirmed NAFLD and NASH diagnoses, if liver enzymes are elevated, the hepatocellular pattern for NAFLD and NASH screening is AST/ALT ratio  $< 1$  [49]. Note that the liver enzymes are not always elevated in the case of NAFLD and NASH [49]. Additional details about albumin, bilirubin, total protein etc. are not covered in this paper. Details related to these proteins/enzymes can be found in the references [50], [51].

#### A. Epidemiology

A 10% increase in the global prevalence of NAFLD between 2005 – 10 was identified in an epidemiological study [37]. This finding is consistent with other studies that have also found increasing prevalence rates both globally and specifically in the USA [52], [53]. Although the primary target region of this mini-review is the USA, it is pertinent to highlight the magnitude of the NAFLD prevalence in other regions around the globe.

Fig 1 shows the prevalence of NAFLD around the world (from 1989 – 2015) [37]. The Middle East and South America were identified as highly prevalent regions at 31.8% and 30.4% respectively [37]. A similar statistic was observed in the USA based on National Health and Nutrition Examination Survey (NHANES) data [52]. The prevalence of NAFLD in the USA rose from 5.5% to 11% from 1988 to 2008. [52], [54].

In Asia, the epidemiology of NAFLD has continued to rise since the year 1990 [55]. Japan, for instance, experienced an increase in NAFLD prevalence by 17% from 1990 to 1998 – with varying sex prevalence, as of 2008 (~32% male vs 17% female) [55]. Similarly, the NAFLD prevalence ranged from 15 – 45% in southeast Asia (includes India, Sri Lanka, Malaysia, Singapore, and Indonesia), Korea, Japan, and Taiwan [55]. China (20%) and Hong Kong (27%) were also found to show an increasing prevalence of NAFLD during 2003-13 [55].

Translating these prevalence rates into absolute numbers (individuals), in highly populated countries like China and India highlights the significance of addressing this disease (prevention, early diagnosis, and management).

A global meta-analysis of 49,419 individuals (from 80 studies in 20 countries) revealed that 55.5% and 37.3% of type 2 diabetes (T2DM) patients had NAFLD and NASH (non-alcoholic steatohepatitis) respectively, between January 1989 to September 2018 [56]. A combined NAFLD prevalence (within group of type 2 diabetes patients) of 59.20% was reported in Iran, Saudi Arabia, and Turkey (via meta-analysis based on four separate reports) [56]. The same meta-analysis further reported a prevalence of 57.46% in India and Pakistan (within group of type 2 diabetes patients, based on six separate reports). Interestingly, the NAFLD prevalence rose to 68.82% in type 2 diabetic patients in Europe (based on 26 reports) [56].

Another recent review paper outlined the increasing prevalence of NAFLD risk factors like insulin resistance, hypertension, and dyslipidemia alongside increasing NAFLD prevalence [54]. Aside from high prevalence with obese, overweight, and diabetic populations, approximately 5-15% of the lean population (based on BMI) were found to have NAFLD [55], [57]. Further, it was found that older adults, specifically in the range of 40-50 have been affected by NAFLD, NASH risk, and fibrosis [54]. Greater prevalence in patients with Hispanic ethnicity when compared to non-Hispanic Caucasians and African Americans was also found [58].

According to the United States Organ Procurement and Transplantation Network's (OPTN) annual report, the number of liver transplants for NASH continued to rise in 2018 [59]. The number of candidates on the waitlist for liver transplant in 2018 were increasingly found to have NAFLD [59]. Obesity (defined as BMI  $\geq 30$  kg/m<sup>2</sup>) and diabetes were found to be consistent with adult liver recipients at 34.6% and 29.2% respectively in 2018 [59]. Further, the United Network for Organ Sharing (UNOS) recorded that obesity and diabetes increased by approximately 1.7% and 4.3% respectively over the past decade [59].

#### 1) NAFLD and sex disparities:

Sex disparities have been observed in NAFLD prevalence around the globe. In a recent paper, Lonardo et al. summarized population-based studies (10 studies on NAFLD in adult populations) that report higher prevalence in men, than in women [60]. It is important to note that multiple diverse investigations led to a consistent finding of men being more prone to NAFLD than women are [60]–

[62]. The possible linkage of estrogen as a protective factor for NAFLD has been reported [61]. The report further alluded that men and women (post-menopause) have a larger risk of NAFLD compared to risk in pre-menopausal women [61].

NAFLD prevalence was found to increase with the increase in BMI for both men and women [63]. However, in 445 normal BMI (18.5 to 23.9) individuals, NAFLD prevalence was higher among men than that for women (14.4%; for men vs 11.9%; for women;  $p = 0.0156$ , 95% CI). Although NAFLD prevalence, in general, is lower among women than among men, the progression of NAFLD to advanced fibrosis is faster among women as compared to that for men [62].

Therefore, we argue that a comparatively lower NAFLD prevalence percentage among women needs to be carefully interpreted. In a populated country(ies), this percentage can translate to a high, concerning, absolute number of women affected by NAFLD.

Overall, the literature does not indicate any specific reason for the difference in NAFLD prevalence among men and women. Additional investigation is needed to address this disparity.

Aside from the traditional risk factors for NAFLD, toxicant-associated risk factors for NAFLD have been identified to induce NAFLD and NASH like pathology. These conditions are named Toxicant-induced fatty liver disease (TAFLD) and Toxicant-associated steatohepatitis (TASH) [36], [64].

## 2) TAFLD and TASH:

These conditions are induced via occupational hazards in jobs that involve chronic exposure to heavy metals. TAFLD is similar in pathology to NAFLD and ALD, while NASH and TASH are similar, pathologically [36].

Upon evaluation of liver biopsies of 25 highly exposed vinyl chloride workers, Cave et al. found an 80% prevalence of steatohepatitis [64]. It is important to note that the 25 workers were not obese and had no other identifiable risk factor for steatohepatitis, other than occupational exposure [64].

In an attempt to estimate the prevalence of unexplained NAFLD in the US adult population, a population study based on 4,582 individuals was conducted (using NHANES data) and compared with their Pb, Hg, and Cd exposure [21]. Upon extrapolation using sample weights, an estimated 10.6% of the US adult population was found to have an unexplained elevation of the liver enzyme – ALT [21]. Elevated ALT was defined as - Men: 18-20 years:  $\geq 37$  IU/L,  $\geq 21$  years:  $\geq 48$  IU/L; Women: 18-20 years:  $\geq 30$  IU/L,  $\geq 21$  years:  $\geq 31$  IU/L and was used as a proxy for unexplained NAFLD [21]. Blood Pb concentration and total Hg concentration were associated with elevated ALTs at  $p$ -values of 0.006 and 0.010 respectively (95% CI) [21]. However, Cd was not found to correlate with the elevated ALTs [21]. Researchers find this unexplained NAFLD to be possibly associated with exposure to toxicants [21]. Other such studies that evaluate the prevalence of NAFLD or NASH in workers with occupational exposure to toxicants

(volatile petrochemicals, vinyl chloride) have shown similar results [65], [66].

Overall, the American Liver Foundation estimates that 25% of the US population has NAFLD [67]. NAFLD was also found to be the most common chronic liver condition in the USA by the American Liver Foundation [67]. However, research shows that a majority of the NAFLD diagnoses are incidental [68]–[70]. In a study with 100 adults with NAFLD (from west Australia between 2009 – 15), it was found that 66% of NAFLD-related cirrhosis was diagnosed incidentally, of which 74% was further diagnosed with NAFLD incidentally [68]. This finding is indicative of a lack of systematic screening and diagnostic patterns for an increasingly prevalent disease like NAFLD. The various intricate factors impacting NAFLD, and their interactions are outlined in the etiology section below.

## B. Etiology

Multiple complex parameters like inflammation, insulin resistance, diabetes, obesity, diet, and lifestyle play a role in the etiology of NAFLD [25], [71], [72]. The interaction of these parameters promotes disease progression. These parameters are analyzed in detail in the sections below.

### 3) NAFLD and diabetes:

A global meta-analysis study of 49,419 type-2 diabetes patients was conducted. 55.5% of them had NAFLD and 37.3% had NASH [56]. This observation indicates the co-existence of diabetes with NAFLD/NASH.

Similar to NAFLD, the prevalence of diabetes is also different for men and women. According to the International Diabetes Foundation, an additional 17.2 million men were diagnosed with diabetes in 2019, as compared to that among women [73]. Although obesity is observed in both diabetic [74] and NAFLD patients [71], the contribution of NAFLD to diabetes hazard factor (3.59) is higher than that (1.99) contributed by ‘over-weight’ (BMI  $\geq 23$ ) condition [75].

### 4) NAFLD and Metabolic Syndrome (MS):

Researchers have found a strong association between NAFLD and MS [71], [72], [76]. Multiple agencies around the globe like the World Health Organization (WHO), European Group for the Study of Insulin Resistance (EGIR), National Cholesterol Education Program (NCEP) Adult Treatment Panel III (ATP III), and others have defined MS [27]–[29]. The more recent definition by NCEP-ATP III in 2005 identifies MS when three of the five criteria in table 1 apply [77], [78]:

Parameter	Men	Women
Abdominal obesity	> 40 inches	> 35 inches
Triglycerides	> 150 mg/dL	> 150 mg/dL
HDL	< 40 mg/dL	< 50 mg/dL
Blood Pressure	> 130/> 85 mm of Hg	> 130/> 85 mm of Hg
Fasting Glucose	> 110 mg/dl	> 110 mg/dl

Although there isn't a unanimous opinion about NAFLD manifesting itself as MS or NAFLD as a precursor for MS, some researchers have found data that supports the latter [58], [71], [79]. While MS is a common risk factor for NAFLD patients, differences in prevalence were found based on race and ethnicity [71]. Further, exposure to heavy metals was found to contribute to MS in individuals – although the available data is conflicting [43]. MS is defined as the co-occurrence of several known factors; the central pathophysiology being insulin resistance (IR) [78].

#### 5) NAFLD and IR:

IR was reported as a stand-alone parameter associated with risk of NAFLD, with or without MS [80]. Cases of IR and hyperinsulinemia are both persistent in NAFLD patients but in the case of NAFLD patients who develop IR, high circulating free fatty acids (FFAs) are created [71]. Hepatic uptake of FFAs results in reduced glycogen storage and increased gluconeogenesis [71]. The combined effect of IR and hyperinsulinemia have been linked with steatosis and excessive release of triglycerides [71]. These triglycerides are in the form of very-low-density particles that assist oxidative stress in the liver and prompt atherosclerosis [71], [72].

#### 6) NAFLD and hormone dysfunction:

Various hormones are related to the promotion of obesity, inflammation, and lifestyle habits. Two such hormones derived in the gut and related to NAFLD are 1.) glucagon-like peptide – 1 (GLP-1) and 2.) Ghrelin [72]. GLP-1 is the hormone responsible for the activation of reward centers of the brain upon the consumption of macronutrients like fructose. Ghrelin concentrations promote hunger. Weakened GLP – 1 secretion and reduced receptors for GLP-1 have been found in the livers of NAFLD patients, damaging the hepatic glucose and lipid metabolism [72]. In NAFLD patients, the concentration of acylated/deacylated Ghrelin was found to be elevated [72]. The association of dysregulated hormones and NAFLD needs to be further explored.

Largely, NAFLD was found to be one of the emerging etiologies of chronic liver disease. For example, NAFLD contributed to 22% of all chronic liver diseases in young adults in the USA [82] and 39.7% among adults in India [25]. NASH, is a fast-growing etiology of end-stage liver disease [83]. In a nutshell, NAFLD and NASH are rapidly growing both globally and regionally (in the USA). Early detection and intervention are not only critical but also urgent.

### III. HEAVY METAL EXPOSURE AND NAFLD

#### A. Impact of heavy metal exposure on the liver's functionality:

Various studies have shown the impacts of heavy metal exposure to general human health [1], [8], [11], [32], [33], [84]–[87]. Heavy metal exposure leads to various types of skin diseases, kidney malfunctions, and liver problems [88], [89]. Almost all heavy metals are categorized as carcinogens

[84]. Occupational exposure was found to occur via environmental pollutants like industrial waste, mining activities, and ore smelting [32], [33]. Drinking water standards for As, Cd, Pb, and Hg as defined by the US – Environmental Protection Agency (US EPA) and the World Health Organization (WHO) are shown in table 2. The routes and sources of exposure of As, Pb, Hg, and Cd are outlined below.

TABLE II  
HEAVY METAL LIMITS IN DRINKING WATER

Element	US EPA limit [90]	WHO limit [91]
Arsenic	0.01 mg/L	0.01 mg/l
Cadmium	0.005 mg/L	0.003 mg/l
Lead	Action level* = 0.015 mg/L	Action level** = 0.01 mg/l
Mercury	0.002 mg/L	0.006 mg/l

\* "Lead and copper are regulated by a treatment technique that requires systems to control the corrosiveness of their water. If more than 10% of tap water samples exceed the action level, water systems must take additional steps. For copper, the action level is 1.3 mg/L, and for lead is 0.015 mg/L" [90].

\*\* The guideline value is provisional based on treatment performance and analytical achievability.

Arsenic exists in three forms: Organic, Inorganic, and Arsenic gas [92]. Inorganic As is the most toxic form of As and is a confirmed carcinogen [93]. Arsenic exposure to human beings can occur from multiple routes: ingestion (consumption), inhalation, and dermal exposure. However, the primary route of As exposure is mostly through consumption of contaminated food and water [88], [92]. Geological characteristics in certain regions of the world lead to higher Arsenic levels in drinking water. Population living in that area is vulnerable to exposure [88]. Contaminated groundwater flows through rivers and is used directly for consumption or irrigation purposes. Food that is grown in arsenic polluted regions also leads to exposure via diet [32]. Moreover, multiple food products are contaminated with Arsenic. Some examples of food contaminated with Arsenic are rice, seafood, food and vegetables, meats, cereals, and dairy products [92], [94]. Rice is a primary staple for a large population in the world. In a study involving rice and other grains, white rice grown in Thailand, India, and Italy had higher median heavy metal concentrations compared to white rice from the USA



(Arsenic 155 vs 131  $\mu\text{g/kg}$ , Pb 3.6 vs 2.8  $\mu\text{g/kg}$  and Cd 17.4 vs 6.5  $\mu\text{g/kg}$ ) [95]. Populations consuming white rice as a staple could be chronically exposed to heavy metals like As, Pb, and Cd.

Lead, another heavy metal, exists in three forms: elemental lead, inorganic and organic lead. The organic form of lead is highly toxic [96]. Pb occurs in the environment (soil, water, dust) and industrial waste as well as in household products [89]. It also has multiple exposure routes into aquatic life, seafood, food produce, and ultimately into the human body [26]. Seafood (canned and fresh samples of fish, mussels, and other seafood) from different geographical regions were found to have varying degrees of inorganic lead in them [97]. Chronic consumption of seafood or vegetables farmed in contaminated soil or water can lead to lead exposure.

Mercury exists in three main forms – elemental Hg, organic Hg (methyl Hg), and inorganic mercury (in the mercurous and mercuric types) [2]. In humans, the most common exposure route of Hg (methyl Hg) is via seafood consumption [98]. A study comparing the Asian and non-Asian population in the USA found that Asian adults (> 50 years old) had higher methyl Hg compared to their non-Asian equivalents (mean concentrations: 1.69  $\mu\text{g/L}$  vs 0.58  $\mu\text{g/L}$ , respectively) [99]. The impact of chronic consumption of seafood can lead to methyl Hg exposure.

Cadmium exists in the environment but not in its refined form. Cadmium in the environment occurs as an ore in two forms: Cadmium sulfide, Cadmium with zinc [100]. It is refined during zinc production [100]. Cadmium has various industrial applications like batteries (Nickel-Cadmium batteries), fertilizers, pigments, plastics, and coatings [100]. The human use of Cadmium enables widespread Cadmium dispersion into the air. Transfer of Cd from the air into the rain and soil introduces Cd into drinking water, food, and finally into the human system [100]. Certain animal meats, shellfish, mushrooms, and plant produce such as rice, grain, potatoes were found to contain Cadmium [100]. The rates of Cd in food are especially high when farmed in contaminated areas close to mines and smelters [100]. Populations living closer to Cd contaminated areas are at a higher risk of chronic Cd exposure via inhalation of contaminated air and ingestion of food and water contaminated with Cd.

Further, exposure to a mixture of heavy metals was found to have a larger impact on the liver cell function, susceptibility to liver injury and, histopathological changes in the liver, compared to individual heavy metal exposure [28], [33], [43]. The individual impact of the above four heavy metals (As, Pb, Hg, and Cd) on NAFLD are outlined below.

#### 1) NAFLD and Arsenic:

NAFLD occurrence and heavy metal exposure were found to be overlapping based on recent studies. Arteel et al. found striking similarities in ‘demographic and mechanistic overlap’ between As exposure and NAFLD occurrence in individuals [1]. States in the USA with higher levels of Arsenic in their drinking water found higher incidences of obesity and NAFLD in their residents [1], [101], [102].

Effects of chronic Arsenic exposure were researched in prior studies involving animal models [1]. It was found that Arsenic exposure for longer than nine months induced hepatic steatosis, potentially leading to diabetes and insulin resistance when exposure is combined with an MCD (methionine choline-deficient diet) diet [1], [103], [104]. Increased hepatic injury through low-grade inflammation, was found to increase lipid accumulation in the liver [1].

In humans, low dose exposure to Arsenic can result in Arsenosis while chronic exposure can cause liver injury. The toxicokinetic mechanism of As upon exposure results in absorption of 70-90% of As by the gastrointestinal tract (GI tract). From the GI tract, As mainly spreads to the liver, kidneys, lungs, and bladder via the bloodstream, and the highest accumulation of As occurs in the liver [88]. The majority of As metabolism also occurs in the liver [88]. A portion of absorbed As is eventually eliminated via urine but the remainder bioaccumulates inside the body [94], [105]. The major detoxification pathway of inorganic As from the human body is via methylation [106]. However, intermediate metabolites released during methylation were found to have toxic effects and cause DNA damage [106]. Most of the biochemical processes in the human system involve proteins and enzymes [107]. DNA damage caused by heavy metal exposure (via induction of reactive oxygen species) can impact the DNA repair pathways and the maintenance of cell health in the human body [107]. Increased oxidative stress in the liver cells of zebrafish was observed when exposed to chronic Arsenic concentrations (six months, 50 ppb to 300 ppb As) [108]. Therefore, chronic exposure to heavy metals might also affect the human liver enzymes which are the basis of liver functionality.

#### 2) NAFLD and Mercury:

A research group hypothesized that the body burden of Hg exposure is apparent in the functionality of the liver based on urine and blood Hg values ( $N = 3,769$ ) [30]. Blood Hg was indirectly measured as MeHg in their work [30]. They found serum and urinary Hg levels to be correlated ( $r = 0.54$ ,  $p < 0.001$ ) [30]. To determine the liver functionality, enzymes AST, ALT, and GGT were used from the dataset [30]. They defined liver enzyme values as follows: ALT: greater than 47 U/L and 30 U/L for men and women, respectively; AST: greater than 33 U/L both for men and women; GGT: greater than 65 U/L and 36 U/L for men and women, respectively [30]. In cases of elevated AST, ALT, and GGT together, lower urine Hg levels at a given Hg exposure were found [30]. This relationship, however, was only marginally significant ( $p = 0.06 - 0.08$ ) when ALT, AST and GGT were used together in the multivariate regression analyses [30]. They found that urinary Hg analysis may not be optimal because of the possible dependence of urinary Hg on liver functionality [30]. On the other hand, they found MeHg concentrations to be higher in individuals with all three liver enzymes elevated ( $p=0.01$ ), which was found to be consistent with other studies [30]. Based on their study results, they suggest that increased MeHg levels may be due to decreased demethylation, as indicated by elevated liver enzymes [30].

In an elderly population study in Seoul, Korea (> 60 years, N = 560), Lee et al. also found the presence of Hg in the blood (mean: 2.81 µg/L (2.73, 2.89), 95% CI) to be associated with abnormal liver enzyme levels (AST, ALT, and GGT) at  $p < 0.05$ , 95% CI, leading to reduced liver functionality [22]. Their definition of abnormal liver enzymes was: ALT > 35 U/L for men and women; AST > 34 U/L men and >40 U/L women; GGT > 48 U/L for men and > 29 U/L for women [22]. Despite adjusting for age, sex, smoking, drinking, and other lifestyle and clinical habits, the liver enzyme concentrations were abnormal in the high blood Hg group [22]. However, when blood Hg levels were combined with alcohol consumption data, the liver functionality deteriorated further in the patients who consumed alcohol regularly [22]. The specific role of Hg exposure on the development of metabolic syndrome was also explored in a review paper [3]. However, the scope of this paper is limited to NAFLD and heavy metal exposure.

### 3) NAFLD and Lead

Luo et al. found that chronic exposure of Pb in mice led not only to hepatotoxicity but also influenced multiple signaling pathways, fatty acid metabolism, and drug metabolism [26]. The levels of three liver enzymes (AST, ALT, and ALP) were found to rise with an increase in Pb exposure in adult mice [26]. The obtained result is aligned with that from Georgieva et al.'s work with common carp undergoing chronic exposure in the Topolnitsa reservoir [9]. In both these animal models, it was found that the interference caused by heavy metal exposure in liver enzyme concentrations lead to hepatic injury at the tissue and cellular level, interrupting several signaling pathways [9], [26].

High levels of Pb in the blood (Median value: 4.49 µg/dL (2.97–6.59) for women; 5.29 µg/dL (3.60–7.28) for men) were also found to be related to NAFLD occurrence in China [14]. Although ALT blood levels were measured in this study, abdominal ultrasound (US) was used to diagnose liver health [14]. Blood Pb levels were found to positively correlate with NAFLD, independent of ALT levels. The correlation was more pronounced in women ( $p$  for trend < 0.001, 95% CI, N = 610, vs control N = 876) than in men ( $P$  for trend = 0.033, 95% CI, N = 214 vs control N = 311).

Exposure to Pb was found to increase liver injury, via an intermediate agent – Glutathione [109]. Glutathione is a detoxifying agent synthesized from amino acids, which is crucial for detoxification and cell physiology [110]. Exposure to Pb impacts the Glutathione levels in the body up to 40% [85]. Further, the regeneration time for glutathione slowed down to 40 mins, from the standard 20 minutes [85].

### 4) NAFLD + Cadmium and NAFLD + mixture of heavy metals

Prystupa et al. conducted a study in Lublin (Southeastern Poland), a region majoring in agriculture [4]. Study subjects were farmers and unemployed people who had advanced alcoholic liver cirrhosis (N = 62 with 46 Male, 16 Female). Essential trace elements required for the proper functioning of the body - (copper (Cu), Zinc (Zn), Nickel (Ni), and

Cobalt (Co)) were found to be lower in study subjects compared to controls (N = 18). Contrastingly, they found serum Cd concentrations to be higher in advanced cirrhosis patients (Control:  $0.0054 \pm 0.0007$  mg/L vs advanced cirrhosis:  $0.0078 \pm 0.0044$  mg/L) [4]. Cd is a toxic heavy metal, capable of causing hepatocyte damage [4].

A study conducted in Taiwan found up to 26.5% prevalence of fatty liver disease in men (n = 1,137) who are exposed to higher levels of soil heavy metals [111]. The study investigated the presence of As, Hg, Cd, Cr, Cu, Ni, Pb, and Zn in soil and their association with fatty liver disease [111]. They used abdominal sonography to identify fatty liver instead of liver biopsy [111]. The association was highest for men with BMI < 24 kg/m<sup>2</sup> [111]. However, in comparison with the Framingham Steatosis Index (FSI), they found that heavy metal exposure in FSI was also positively correlated with men (with BMI < 24).

However, this result needs to be validated in the context of pre-and post-menopausal women to understand the impact of heavy metals on the liver for a diverse population. The liver enzymes and heavy metal exposure relationship need to be researched further to assist with the early detection of heavy metal related liver toxicity.

Studies have also found an association between significant quantities of other heavy metals like Iron and zinc with liver dysfunction in various populations [112]–[114]. The scope of the present study only focuses on As, Pb, Hg, and Cd. Therefore, the impact of other heavy metals is out of the scope of this review. Researchers have also explored the association between NAFLD and metabolic syndrome, as detailed in section 2(b) in this paper [71], [72], [76], [80], [115].

Although these studies need to be extensively validated in human models with diverse populations and demographics, the initial impact on the liver in human and animal models for liver damage is crucial. The inability of the liver to metabolize heavy metals, especially in advance diseased stages indicates the importance of early diagnosis and treatment of liver disease for individuals living with chronic exposure.

As mentioned earlier in section 2a, NAFLD progresses with time, making it crucial to identify the disease early enabling intervention and care. Current evidence-based practice in NAFLD screening is the use of circulating enzyme concentration in blood plasma [49], [117], [118]. However, it is important to note that the detection of NAFLD based on liver enzymes alone can be misleading [49]. All cases of NAFLD do not exhibit elevated liver enzymes, the prevalence of NAFLD was under-estimated consistently when liver enzymes were used for diagnoses instead of imaging [68]–[70], [119]–[121]. Ultrasound (US) imaging of the liver is conducted when clinicians suspect NAFLD/NASH in a patient. However, due to lack of sensitivity with US (in case of obese patients or liver cells with fat droplets), computer tomography (CT) or magnetic resonance imaging (MRI) are also used [118]. The current standard for confirmatory diagnosis of steatosis, steatohepatitis, and fibrosis is liver biopsy [118].

Although these methods can detect NAFLD, US tests lack the sensitivity to diagnose the disease, and liver biopsies are expensive, invasive, and risky [69]. In the search for alternate, specific biomarkers for NAFLD, researchers have found potential genetic, proteomic, blood-based, enzyme-based, and urine-based biomarkers.

TABLE 3  
IMPACT OF HEAVY METAL EXPOSURE ON THE HUMAN SYSTEM

Heavy metal	The affected biological entity in the human body (Organ, enzyme, cell, etc.)	Reference
Arsenic	Liver toxicity	[1]
	Skin cancer, bladder cancer, and other cancers	[102]
	Chronic Arsenical dermatosis and hepatomegaly	[11]
	Skin lesions, respiratory and nervous system problems, cancer	[32]
	Elevated ALT correlated with (p = 0.07) higher urinary Arsenic values	[15]
Cadmium exposure	Reduced levels of essential trace elements (Cu, Zn, Ni, and Co).	[4]
	High serum Cd levels correlated with advanced cirrhosis patients.	[4]
	Liver cell damage, liver carcinoma	[19]
	Elevated B-Pb correlated with NAFLD, especially in women (probability < 0.001, 95% CI). For men: probability = 0.033	[14]
Lead exposure	Hematopoietic and renal toxicology	[12]
	Slowed glutathione regeneration and reduced glutathione levels	[85]
	Elevated ALT, AST & GGT (p = 0.06 - 0.08)	[30]
Mercury exposure		[22][112][112][111][110][109][105][104][103][102][101][100][99][98][97][96][95][94][92][91][90][89][88][87][86][84]
	Abnormal AST, ALT & GGT (p < 0.05, 95% CI)	
Mixture effects	Pb and Hg exposure correlated with ALT elevation (p for trend = 0.006 and 0.010 respectively)	[21]

#### IV. EMERGING BIOMARKERS AND TOOLS FOR NAFLD DETECTION

This section focuses on emerging and contemporary methods or tools for NAFLD (and HS, NASH) detection. It

is important to note that these findings are currently in the research investigation/exploratory phase. Although most of these methods are not currently used in the clinical setting, they show potential for use in the future and have therefore been reviewed below.

##### 1) Physiological biomarkers

Previously, adipose tissue was considered to be a passive storage unit for excess energy, but current research shows the ability of the adipose tissue to synthesize and release multiple hormones and cytokines that circulate throughout the body [122]. Adipose tissue dysfunction has since been researched in the context of NAFLD association [122]. Adipokines like “leptin, adiponectin, ghrelin, interleukin-6, and, tumor necrosis factor- $\alpha$ ” have shown an association with NASH [123], [124]. Imbalanced adipokines initiate a proinflammatory and insulin-resistant response which exasperate the progression of NAFLD, eventually leading to NASH in severe NAFLD patients (N = 82) [124]. A formula that includes the use of serum levels of adipokines to determine NASH was developed. The value of:  $\text{adiponectin (ng/ml)} / (2 \times \text{leptin (ng/ml)} \times \text{ghrelin (pg/ml)}) < 0.31$  resulted in an AUROC of 0.789 with 81.8% sensitivity and 76.1% specificity with a negative predictive value of 96.4% and positive predictive value of 34.6% for diagnosing NASH [124]. However, it is important to note that their patient demographic was limited to “morbidly obese individuals, with biopsy-proven NAFLD” [124].

The relationship of BMI with various adipokines was also researched. It was found that leptin had positive correlation ( $r=0.45$ ,  $P<0.001$ ) whereas ghrelin had negative correlation ( $r=-0.28$ ,  $P=0.012$ ) with BMI [124]. Adiponectin on the other had negative correlation with waist to hip ratio ( $r=-0.34$ ,  $P=0.007$ ) and showed no correlation with BMI [124]. However, BMI does not consider the fat percentage in the body, so conclusions from the above study should be drawn with caution. In short, ghrelin was found to be associated with diabetes occurrence whereas adiponectin was associated with “insulin resistance, hypertension, dyslipidemia, and metabolic syndrome” [124]. A review paper also found that circulating adipokines are related to NAFLD pathogenesis and NASH progression [122].

The correlation of NAFLD and leptin was used for NAFLD detection by researchers [125]. Cai et al. developed an immunosensor using “porous graphene functionalized black phosphorus (PG-BP)” [125]. Anti-leptin was fixed firmly on the surface of the electrode which led to high sensitivity (LOD: 0.036 pg/ml) and reduced interferences with other enzymes [125]. In their approach, a label-free and environment-friendly leptin sensor were developed for very low levels of leptin detection (up to 0.036pg/ml, with a linear range of detection in 0.150–2500 pg/mL) [125]. The results of the immunosensor based detection were found to surpass those of ELISA (enzyme-linked immunosorbent assay) or other previous electrochemical methods [125]. Upon further validation,

such sensors can be effectively used for early detection of NAFLD, thereby allowing early intervention and care.

Further, fatty acid-binding proteins like adipokine binding protein (A-FABP), retinol-binding protein (RBP4), and lipocalin-2 “are associated with obesity, insulin resistance, and metabolic syndrome” [126]. A study based in a South Korean hospital found A-FABP levels were found to be high in the NAFLD group (N = 73) when compared to the normal group (N = 67): “ $18.42 \pm 7.24$  ng/mL vs.  $15.74 \pm 7.02$  ng/mL vs.,  $p = 0.022$ ” [126]. Upon conducting a logistic regression analysis, patients in the highest quartile of A-FABP levels corresponded to three times the risk of NAFLD than that of those in the lowest quartile (p-trend: 0.039, 95% CI) [126]. The explanation of this association could be the modulation of inflammatory responses based on A-FABP levels in the body [126], [127]. However, their study presented contradictory results regarding the serum RBP4 and lipocalin-2 levels compared to other studies [126]. Although their results need to be validated with a diverse population, they were obtained independent of age and sex, indicative of the potential A-FABP could have in NAFLD diagnosis.

A hormone secreted by the liver called fibroblast growth factor 21 (FGF21) has been implicated to indicate “lowering blood glucose, lipids, and insulin levels, reversing hepatic steatosis, and increasing insulin sensitivity” and therefore, as an early biomarker for NAFLD [128]. Gong et al. developed a field-effect transistor (FET) which is also label-free and very sensitive to FGF21 levels in human serum samples (1 fg/ml) [128]. They used a “molybdenum disulfide (MoS<sub>2</sub>)” surface for non-aqueous environment detection of FGF21 and achieved a limit of detection of 10 fg/mL [128].

## 2) Biomarkers related to NASH:

Long term NAFLD prognosis has implicated the prevalence of non-alcoholic steatohepatitis (NASH) in patients [118]. NASH prediction via blood biomarkers was reviewed by Gomez et al. In their research, the gap in specifically and sensitively identifying NASH from NAFLD is highlighted. They suggest combining blood-based biomarkers with existing diagnostics [123].

Methionine choline-deficient diet (MCD diet) was used by researchers performing studies in animal models in this domain to induce NASH like symptoms. After the NASH symptoms were induced in mice, Clarke et al. analyzed the serum levels of microRNA – 122: RNA specific to the liver [129]. They found a 40-fold average increase in the levels of miRNA-122 in mice induced with an MCD diet for three days [129]. ALT and AST levels were only 4.8 fold and 3.3 fold elevated compared to miRNA-122 (at 40- fold) [129].

## 3) Omics:

Research towards specific biomarkers for NAFLD or other liver diseases (FLD, NASH, etc.) has led to the identification of multi-omics compounds associated with the disease. Urinary steroid metabolome, multi-omics, serum-based omics, and a combination of clinical features with omics are being studied concerning liver diseases [130]–[132].

Researchers used volatile organic compounds from the NAFLD breath sample to develop indirect, sensitive detection methods for NAFLD [133]. A micro gas chromatography column was used by researchers to separate the VOCs in the breath condensate [133]. The breath condensate was separated from the gas pentane, considered to be a potential biomarker for NAFLD based on its ability to predict fibrosis [133]. Their findings indicate that the micro GC column was able to separate gases C5 – C12 within 5 minutes [133]. However, their micro-GC column needs to be installed inside a conventional GC instrument, requiring high temperatures and expensive equipment. The reliability of pentane as a potential biomarker for NAFLD remains to be researched.

## 4) Use of computational tools:

With the increasing availability of data and computational power, research in the field of health care analytics and model-based disease prediction has accelerated. Machine learning (ML) and Artificial Neural Network (ANN) based models have been created in various disease domains to assist with diagnosis and screening [131], [132], [134]–[148].

A computational approach to NAFLD screening or diagnosis, liver fat quantification, fibrosis pattern detection, assessment of the severity of the liver disease can be classified into four different modalities, per our literature search.

- i. Physiological parameters (serum triglyceride levels, BMI, age, etc.)
- ii. Imaging modalities (US, MRI)
- iii. Omics to find associations with liver disease (genetics, transcriptomics, metabolomics, etc.)
- iv. Images of liver biopsies

The scope of this review is limited to the initial benchmarking of minimally invasive or non-invasive techniques for quantifying liver diseases. Hence, we did not include liver biopsy related work in this review.

## 5) Physiological parameters & Machine Learning:

Use of physiological parameters as ‘risk factors’ to predict a disease has been applied to several disease domains. In this approach, researchers use electronic medical health records, or physiological data like age, BMI, blood glucose, etc. in combination with machine learning or deep learning algorithms [134], [135], [141], [142].

In this regard, our previous research was to predict the occurrence of fatty liver (HS) based on previously established risk factors [134]. We used data from NHANES III (N = 12,719), and the model was able to classify HS and no-HS with an accuracy of 79.03% using a gentle boosted tree algorithm. Another research group developed ML models to predict FLD using data from a hospital (N = 577) in Taiwan [135]. They used nine predictor variables (similar to risk factors) as model inputs. Their best performing model

(Random forest, 10-fold validated) had an accuracy of 86.48% [135]. Similarly, other models to support NAFLD diagnosis and to assess NAFLD severity were also developed by other researchers [141], [142]. These models show potential as screening tools for fatty liver disease, especially in populations with low or no alcohol consumption. However, the results from such models need validation from large, diverse datasets.

#### 6) *Imaging modality:*

Imaging modalities have significant scope in NAFLD detection and quantification. In general, three imaging modalities are of importance in the context of NAFLD and they are: 1) Ultrasound (US), 2) MRI and 3) Computerized tomography (CT) [149]. Ultrasound scans are of two types: qualitative or conventional ultrasound (CUS) and, quantitative US (QUS) [150]. Although MRI is emerging as a quantitative imaging biomarker, US-based studies were more commonly used by researchers, per our literature search. Computer tomography can also be used to detect liver fat but is more commonly used for tumor diagnosis. Further, the radiation exposure associated with CT makes it an uncommon tool for NAFLD detection [151].

A meta-analysis of 49 studies using conventional ultrasound (CUS) from October 1967 to March 2010 was conducted to assess the diagnostic performance of CUS [152]. They reported the performance metrics for use of CUS to detect “moderate to severe fatty liver” in the absence of steatosis. The values for sensitivity (84.8%) as well as specificity (93.6%) were reported [152]. However, the use of CUS as a differentiator of fatty liver, hepatitis, fibrosis, or normal liver had slightly different metrics for sensitivity (87.2%) and specificity (79.2%) [152]. CUS is safe, and generally widely accessible, as compared to MRI. CUS is relatively less expensive than other imaging modalities. However, CUS scans lack sensitivity and specificity to quantify liver fat [153]. As the CUS is qualitative, it suffers from the subjectivity of the interpreter, the operator, and the sensitivity/capability of the machine, thus contributing to lower accuracy [154]. Therefore, the use of quantitative ultrasound (QUS) with backscatter coefficients has been explored in the context of hepatic fat quantification and NAFLD detection, as an alternative to CUS [150], [151], [154].

A preliminary comparison study consisting of ultrasonic scans from 60 different subjects found quantitative ultrasound (QUS) to be more accurate (68.3%) when compared to CUS (51.7%) [151]. They used histologic steatosis grading as a reference standard for NAFLD detection [151]. The use of QUS instead of CUS to quantify steatosis has also been supported by other studies [148], [150], [154]. In recent years, researchers are exploring the capabilities of machine learning (ML) or advanced pattern recognition techniques (i.e., deep learning (DL) and artificial neural networks (ANN)) techniques for NAFLD diagnosis [136], [139], [140], [143]–[148]. Selected example applications of ML, DL, and ANN techniques to US images are discussed below.

A study used 63 conventional ultrasound (CUS) images

from a hospital in Portugal for fatty liver detection and risk stratification [136]. They used features extracted from a region of interest (ROI) (128 x 128 pixels) in the CUS images [136]. No other clinical features (aside from those obtained via analysis ROI) were used [136]. Their best performing model used a deep learning technique with an accuracy of 100% [136]. They reported better results using deep learning paradigm compared to machine learning models (Support Vector Machine (SVM) and Extreme learning machine (ELM) at 82% and 92% accuracies, respectively) [136]. Although the above study shows encouraging results, it does not specify the type of fatty liver disease. Therefore, we assess that the use of ML on CUS requires validation with larger datasets, for potential NAFLD, hepatic steatosis (HS), and NASH screening.

Another study used radiofrequency (RF) signals from quantitative ultrasound of 140 NAFLD patients and 64 controls [147]. They used one dimensional artificial neural network (ANN) classifier with equally split training and test datasets. The classifier showed 96% accuracy with 97% sensitivity and 94% specificity (95% CI) in diagnosing NAFLD [147]. Additionally, they also developed a fat fraction estimator using RF signals as input to a one-dimensional ANN [147]. Their estimates correlated with MRI- proton density fat fraction (PDFF) at  $r = 0.85$  (Pearson’s correlation) [147]. However, this study did not use histological steatosis as reference grade. Thus, we assess that this method needs to be further evaluated for adaption in clinical settings.

More recently, a combination of imaging and non-imaging (physiological) parameters has been implemented with machine learning (ML) and artificial neural network (ANN) tools [139]. The study used clinical data (HDL, LDL, triglycerides, fasting blood sugar, BMI, Forns score [155]) and ultrasound images from 726 patient to extract rules for FLD detection using ANN model [139]. Their derived rules were able to detect FLD with an accuracy ranging from 80.58% - 100%, based on different model parameters. They used reference standards from FibroScan (transient elastography) instead of using liver biopsy. It is to be noted here that the use of FibroScan is limited by the demographic it can be performed on. For instance, FibroScan is not recommended for subjects with a history of ascites, those with morbid obesity, and/or with significant quantities of fat in the chest wall [156]. Thus, the results from the study [139] reported above that used FibroScan as a reference, needs additional careful evaluation for the adaption in clinical practice. Moreover, as the study [139] also used a small dataset, further validation is recommended.

Alternatively, magnetic resonance imaging– (MRI)-proton density fat fraction (PDFF) measures the liver fat by computing the ratio of liver fat signals to total signals [153]. A study comparing conventional ultrasound (CUS), quantitative ultrasound (QUS) and MRI found MRI-PDFF to be more accurate (76.7%) in NAFLD detection when compared to US techniques (QUS: 68.3%, CUS: 51.7%), against a histological grading reference [151].

Previous studies have also demonstrated the accuracy,

repeatability and precision of MRI – PDFF compared to histological reference for NAFLD detection [149], [151], [153], [157]–[159]. The sensitivity and specificity of some of these studies over all grades of steatosis ranged between 0.64–1.0 and 0.76 – 0.96, respectively [149], [151], [157]–[159]. Further, MRI-PDFF enables volumetric steatosis assessment of the liver which is not possible with ultrasound scans [153]. Although MRI-PDFF has many advantages for non-invasive detection of NAFLD, some of the pressing limitations are its cost and limited access to instrument and expertise, in low-resource settings, where NAFLD is prevalent. Additionally, for certain patients with metal implants, MRI-PDFF needs additional preparation and precautions. Current research efforts in developing and validating portable, cost-effective MRI systems [160] might contribute to improve the limitations described above in near future.

In the context of non-invasive and rapid detection of NAFLD, MRI-PDFF is the preferred imaging modality for resource-intensive locations. The use of advanced pattern recognition techniques with MRI-PDFF interpretation can increase the adoption of this technology for NAFLD detection. In low-resource settings, where NAFLD is prevalent, the lack of access to instruments and expertise is a problem. In such cases, QUS with advanced pattern recognition techniques using physiological parameters (detection with comparable or acceptable accuracy for initial screening) are promising compared to those provided by MRI-PDFF and CUS.

In summary, each imaging modality (ultrasound, MRI, CT) has its own merits and limitations, in varying degrees, for NAFLD detection. Additional analysis is required to assess the breadth and depth of each imaging modality. In this review, we intend to highlight the importance of the three imaging modalities in the context of NAFLD detection. However, the details of each modality or their combination is out of scope of the defined objective of this paper.

#### 7) *Machine learning integrated omics:*

Recently developed omics-based studies that use ML or DL paradigms are included in this review [130]–[132].

A study with LASSO (least absolute shrinkage and selection operator) select and a random forest model obtained an ROAUC of 0.84 (95% CI,  $p < 0.001$ ) [131]. Multi-omics (genetic, transcriptomic, proteomic, metabolomic) and clinical (liver enzymes, serum biomarkers, lifestyle etc.) data comprised the key input variables for fatty liver disease [131]. Although their model has the potential to avoid liver biopsies, they require data through RNA sequencing, protein-coding and metabolomic assays which makes diagnosis complicated from a patient/clinical perspective. Further, their study was limited in their demographics to northern European population.

We anticipate that sensor-based systems integrated with computational models to explain predictor contribution of heavy metals (ingested in the body) on liver disease will be useful to further understand the dynamics of the disease.

#### 8) *NAFLD as a risk factor for other diseases*

Aside from leading the liver to failure in its' advanced stages (like liver scarring and liver failure), the occurrence of NAFLD also leads to increased risk for other diseases like PCOS (polycystic ovary syndrome), diabetes, cardiovascular diseases and is linked highly to diabetes type II and obesity [115], [161], [162].

In a study on PCOS patients, it was found that 48 out of a total 88 patients (55%) had NAFLD along with a high insulin resistance score; highly associating PCOS with NAFLD, high BMI, and high insulin resistance [40], [162]. In a review paper linking PCOS and NAFLD, Kelly et al. propose screening of high-risk PCOS women to identify NAFLD [115].

The link between NAFLD and cardiovascular diseases like heart valve calcification (in the mitral and aortic valves) was researched in a cross-sectional study of diabetic patients ( $N = 247$ ) [161]. Approximately 71% of these patients were found to have NAFLD (via US tests) [161]. They found NAFLD to be linked with aortic valve sclerosis and/or mitral annulus calcification with unadjusted-odds ratio: 3.51 (95% CI,  $p < 0.001$ ) [161].

The progression of NAFLD with time can lead to liver cancer or hepatocellular carcinoma (HCC) [163]. 14.1% of the HCC cases were related to NAFLD in a study consisting of 4,929 HCC cases (with 14,937 controls) [163]. The development of NAFLD to HCC further reduces patient's survival rate along with a decreased chance for liver transplant [163].

NAFLD is linked with many metabolic parameters in the body and therefore it is further associated with conditions like diabetes, insulin resistance, PCOS and increased cardiovascular risk. However, specific interactions between NAFLD and other conditions is out of scope for this paper.

### V. SUMMARY AND FUTURE OUTLOOK

A contemporary scientific review is conducted on non-alcoholic fatty liver disease (NAFLD) between 2014 – 2020 in this paper. In the recent years, more emphasis has been placed on NAFLD, therefore relevant literature from the past six years was reviewed. The focus of this review is on heavy metal toxicity and NAFLD. Three main objectives are elaborated in this paper: major enzymes and biomarkers indicative of chronic liver conditions (particularly NAFLD), the effect of heavy metal exposure on liver health and, emerging biomarkers and techniques for NAFLD detection.

The increasing prevalence of NAFLD globally, as elaborated in section II a, indicates the urgent need to address NAFLD diagnosis and management. Further, it is important to note that NAFLD progresses with time to cause fibrosis, cirrhosis and potentially leading to hepatocellular carcinoma. NAFLD etiology is complex and associated with multiple other conditions (i.e., metabolic syndrome, diabetes type-II and insulin resistance etc.) – as outlined in section II b. Current evidence-based practice uses liver enzyme ratios

for NAFLD and NASH screening. However, enzyme ratios do not identify all NAFLD cases. While imaging tools (like ultrasound and MRI) and liver biopsies (required for confirmatory for NAFLD diagnosis) are used, a majority of NAFLD cases are diagnosed incidentally. A gap in NAFLD screening and potential diagnosis is therefore noted.

The relationship of NAFLD with heavy metal exposure is explored in section III. Chronic heavy metal exposure (As, Pb, Hg and Cd) is found to correlate with abnormal liver enzyme values, with NAFLD and, with liver damage in general. The inability of the liver to metabolize heavy metals warrants the screening, diagnosis, and early intervention for individuals living with chronic heavy metal exposure. Toxicant-induced fatty liver disease (TAFLD) and Toxicant-associated steatohepatitis (TASH) are similar in pathology to NAFLD and NASH, respectively. TAFLD and TASH are induced due to chronic heavy metal exposure and need further investigation as part of future research.

Potential biomarkers for NAFLD detection: adipokines (leptin, adiponectin etc.), fatty acid-binding proteins (A-FABP) and hormones (fibroblast growth factor 21 (FGF21)) are being researched. More details regarding these biomarkers are introduced in section IV (a, b, c) of this paper. Recent implementations of potential biomarkers via computational tools for liver disease diagnosis are reviewed in section IV d. Artificial intelligence (AI) tools like machine learning, deep learning, and artificial neural networks are being implemented to detect NAFLD using physiological parameters, imaging data, and omics data (section IV e). The merits and limitations of conventional ultrasound (CUS), quantitative ultrasound (QUS), and MRI-PDFF were discussed in detail in IV f. This review highlighted the implementation of quantitative ultrasound (QUS) instead of conventional ultrasound (CUS) to be promising, especially when QUS data is combined with AI tools for increased detection accuracy.

As mentioned earlier, NAFLD is a widely prevalent disease of significance in the global domain. Therefore, it is important to reiterate the significance of NAFLD by citing a few recent statistics. For example, as of 2020, the global estimated prevalence of NAFLD was at 25% and that in the USA was at 30% [164]. The prevalence of NASH in the USA was estimated to be at 5% [164]. Similar estimates for NAFLD are found in Asia, where the estimated pooled prevalence is at 27.4% [164].

From a broader perspective, NAFLD is a silent disease and can progress over time to cirrhosis and other complications or liver failure. NASH progressing into cirrhosis is estimated to have caused 3.3 million cases of advanced fibrosis in the USA in 2015 [164]. In regions, where regular health checkups are not conducted or available, the health outcomes of NAFLD incidence can be fatal. Further, the etiology and the pathobiology for the disease is connected with heterogeneous factors (including contaminated food and water). This paper has reflected on multiple such relevant factors.

In recent times, the emphasis on the prevention and management of NAFLD has increased. As per our

knowledge, the cure for the disease, at the time of writing this paper, is not available. The current treatment methods emphasize on management of the disease or delaying progression of the disease. Additional understanding (via research and development) of different factors contributing to NAFLD and other chronic liver diseases is needed towards effective treatment and management of the disease.

#### REFERENCES:

- [1] G. E. Arteel, "Hepatotoxicity," in *Arsenic*, John Wiley & Sons, Ltd, 2015, pp. 249–265.
- [2] J. Choi *et al.*, "Mercury Exposure in Association With Decrease of Liver Function in Adults: A Longitudinal Study," *J. Prev. Med. Pub. Health*, vol. 50, no. 6, pp. 377–385, Nov. 2017, doi: 10.3961/jpmph.17.099.
- [3] A. A. Tinkov *et al.*, "Mercury and metabolic syndrome: a review of experimental and clinical observations," *BioMetals*, vol. 28, no. 2, pp. 231–254, Apr. 2015, doi: 10.1007/s10534-015-9823-2.
- [4] A. Prystupa, A. Błażewicz, P. Kiciński, J. J. Sak, J. Niedzialek, and W. Załuska, "Serum Concentrations of Selected Heavy Metals in Patients with Alcoholic Liver Cirrhosis from the Lublin Region in Eastern Poland," *Int. J. Environ. Res. Public Health*, vol. 13, no. 6, p. 582, Jun. 2016, doi: 10.3390/ijerph13060582.
- [5] A. P. Ebokaibe *et al.*, "Assessment of heavy metals around Abakaliki metropolis and potential bioaccumulation and biochemical effects on the liver, kidney, and erythrocyte of rats," *Hum. Ecol. Risk Assess. Int. J.*, vol. 24, no. 5, pp. 1233–1255, Jul. 2018, doi: 10.1080/10807039.2017.1410695.
- [6] C. S. Carvalho, H. S. M. Utsunomiya, T. Pasquoto, R. Lima, M. J. Costa, and M. N. Fernandes, "Blood cell responses and metallothionein in the liver, kidney and muscles of bullfrog tadpoles, *Lithobates catesbeianus*, following exposure to different metals," *Environ. Pollut.*, vol. 221, pp. 445–452, Feb. 2017, doi: 10.1016/j.envpol.2016.12.012.
- [7] D. A. Omran *et al.*, "Serum Zinc Deficiency and its Relation to Liver Fibrosis in Chronic HCV: a Real-Life Egyptian Study," *Biol. Trace Elem. Res.*, vol. 179, no. 1, pp. 1–7, Sep. 2017, doi: 10.1007/s12011-017-0938-x.
- [8] M. Colombo *et al.*, "EASL Clinical Practice Guideline: Occupational liver diseases," *J. Hepatol.*, vol. 71, no. 5, pp. 1022–1037, Nov. 2019, doi: 10.1016/j.jhep.2019.08.008.
- [9] E. Georgieva *et al.*, "Histological and biochemical changes in liver of common carp (*Cyprinus carpio* L.) under metal exposure," p. 10.
- [10] G. Malaguarnera, "Toxic hepatitis in occupational exposure to solvents," *World J. Gastroenterol.*, vol. 18, no. 22, p. 2756, 2012, doi: 10.3748/wjg.v18.i22.2756.
- [11] D. N. G. Mazumder *et al.*, "Chronic arsenic toxicity from drinking tubewell water in rural West Bengal," p. 8.
- [12] H. Wan, J. Wu, P. Sun, and Y. Yang, "Investigation of delta-aminolevulinic acid dehydratase polymorphism affecting hematopoietic, hepatic and renal toxicity from lead in Han subjects of southwestern China,"

- Acta Physiol. Hung.*, vol. 101, no. 1, pp. 59–66, Mar. 2014, doi: 10.1556/APhysiol.101.2014.1.7.
- [13] H. Lee, Y. Kim, C.-S. Sim, J.-O. Ham, N.-S. Kim, and B.-K. Lee, "Associations between blood mercury levels and subclinical changes in liver enzymes among South Korean general adults: Analysis of 2008–2012 Korean national health and nutrition examination survey data," *Environ. Res.*, vol. 130, pp. 14–19, Apr. 2014, doi: 10.1016/j.envres.2014.01.005.
- [14] H. Zhai *et al.*, "Blood lead level is associated with non-alcoholic fatty liver disease in the Yangtze River Delta region of China in the context of rapid urbanization," *Environ. Health*, vol. 16, no. 1, p. 93, Aug. 2017, doi: 10.1186/s12940-017-0304-7.
- [15] J. K. Frediani, E. A. Naioti, M. B. Vos, J. Figueroa, C. J. Marsit, and J. A. Welsh, "Arsenic exposure and risk of nonalcoholic fatty liver disease (NAFLD) among U.S. adolescents and adults: an association modified by race/ethnicity, NHANES 2005–2014," *Environ. Health*, vol. 17, no. 1, Dec. 2018, doi: 10.1186/s12940-017-0350-1.
- [16] J. Marmur *et al.*, "Hepcidin levels correlate to liver iron content, but not steatohepatitis, in non-alcoholic fatty liver disease," *BMC Gastroenterol.*, vol. 18, no. 1, Dec. 2018, doi: 10.1186/s12876-018-0804-0.
- [17] K. Karunanidhi, R. Rajendran, D. Pandurangan, and G. Arumugam, "First report on distribution of heavy metals and proximate analysis in marine edible puffer fishes collected from Gulf of Mannar Marine Biosphere Reserve, South India," *Toxicol. Rep.*, vol. 4, pp. 319–327, 2017, doi: 10.1016/j.toxrep.2017.06.004.
- [18] K. C. Makris *et al.*, "Association between exposures to brominated trihalomethanes, hepatic injury and type II diabetes mellitus," *Environ. Int.*, vol. 92–93, pp. 486–493, Jul. 2016, doi: 10.1016/j.envint.2016.04.012.
- [19] L. Zhang, Y. Huang, Y. Zhu, Z. Yu, M. Shao, and Y. Luo, "Identification and Characterization of Cadmium-Related Genes in Liver Carcinoma," *Biol. Trace Elem. Res.*, vol. 182, no. 2, pp. 238–247, Apr. 2018, doi: 10.1007/s12011-017-1106-z.
- [20] M. Stepień *et al.*, "Circulating copper and zinc levels and risk of hepatobiliary cancers in Europeans," *Br. J. Cancer*, vol. 116, no. 5, pp. 688–696, Feb. 2017, doi: 10.1038/bjc.2017.1.
- [21] Cave Matt, Appana Savitri, Patel Mihir, Falkner Keith Cameron, McClain Craig J., and Brock Guy, "Polychlorinated Biphenyls, Lead, and Mercury Are Associated with Liver Disease in American Adults: NHANES 2003–2004," *Environ. Health Perspect.*, vol. 118, no. 12, pp. 1735–1742, Dec. 2010, doi: 10.1289/ehp.1002720.
- [22] M.-R. Lee, Y.-H. Lim, B.-E. Lee, and Y.-C. Hong, "Blood mercury concentrations are associated with decline in liver function in an elderly population: a panel study," *Environ. Health*, vol. 16, no. 1, p. 17, Mar. 2017, doi: 10.1186/s12940-017-0228-2.
- [23] R. Khan *et al.*, "Toxicological effects of toxic metals (Cadmium and mercury) on blood and the thyroid gland and pharmacological intervention by vitamin C in rabbits," *Environ. Sci. Pollut. Res.*, vol. 26, no. 16, pp. 16727–16741, Jun. 2019, doi: 10.1007/s11356-019-04886-9.
- [24] S. Rajeshkumar, Y. Liu, J. Ma, H. Y. Duan, and X. Li, "Effects of exposure to multiple heavy metals on biochemical and histopathological alterations in common carp, *Cyprinus carpio* L.," *Fish Shellfish Immunol.*, vol. 70, pp. 461–472, Nov. 2017, doi: 10.1016/j.fsi.2017.08.013.
- [25] G. Choudhuri, S. Chaudhari, D. Pawar, and D. S. Roy, "Etiological Patterns, Liver Fibrosis Stages and Prescribing Patterns of Hepato-Protective Agents in Indian Patients with Chronic Liver Disease," *J. Assoc. Physicians India*, vol. 66, no. 12, pp. 58–63, Dec. 2018.
- [26] T. Luo *et al.*, "Chronic exposure to low doses of Pb induces hepatotoxicity at the physiological, biochemical, and transcriptomic levels of mice," *Environ. Toxicol.*, vol. 34, no. 4, pp. 521–529, 2019, doi: 10.1002/tox.22706.
- [27] U. K. Singh, A. L. Ramanathan, and V. Subramanian, "Groundwater chemistry and human health risk assessment in the mining region of East Singhbhum, Jharkhand, India," *Chemosphere*, vol. 204, pp. 501–513, Aug. 2018, doi: 10.1016/j.chemosphere.2018.04.060.
- [28] X. Lin, Y. Gu, Q. Zhou, G. Mao, B. Zou, and J. Zhao, "Combined toxicity of heavy metal mixtures in liver cells," *J. Appl. Toxicol.*, vol. 36, no. 9, pp. 1163–1172, 2016, doi: 10.1002/jat.3283.
- [29] J. Yin *et al.*, "A transcriptomics study on hepatic lipid metabolism in mice exposed to contaminated drinking water," *Int. J. Environ. Sci. Technol.*, vol. 12, no. 3, pp. 847–856, Mar. 2015, doi: 10.1007/s13762-013-0424-8.
- [30] Y.-S. Lin *et al.*, "Association of body burden of mercury with liver function test status in the U.S. population," *Environ. Int.*, vol. 70, pp. 88–94, Sep. 2014, doi: 10.1016/j.envint.2014.05.010.
- [31] G. Cano-Sancho, S. Marin, A. J. Ramos, J. Peris-Vicente, and V. Sanchis, "Occurrence of aflatoxin M1 and exposure assessment in Catalonia (Spain)," *Rev. Iberoam. Micol.*, vol. 27, no. 3, pp. 130–135, Jul. 2010, doi: 10.1016/j.riam.2010.05.003.
- [32] J.-Y. Chung, S.-D. Yu, and Y.-S. Hong, "Environmental Source of Arsenic Exposure," *J. Prev. Med. Pub. Health*, vol. 47, no. 5, pp. 253–257, Sep. 2014, doi: 10.3961/jpmph.14.036.
- [33] Hopenhayn-Rich C, Biggs M L, Smith A H, Kalman D A, and Moore L E, "Methylation study of a population environmentally exposed to arsenic in drinking water.," *Environ. Health Perspect.*, vol. 104, no. 6, pp. 620–628, Jun. 1996, doi: 10.1289/ehp.96104620.
- [34] N. Toshikuni, "Clinical differences between alcoholic liver disease and nonalcoholic fatty liver disease," *World J. Gastroenterol.*, vol. 20, no. 26, p. 8393, 2014, doi: 10.3748/wjg.v20.i26.8393.
- [35] "Symptoms & Causes of NAFLD & NASH | NIDDK." <https://www.niddk.nih.gov/health-information/liver-disease/naflid-nash/symptoms-causes> (accessed Oct. 19, 2020).



- [36] B. Wahlang *et al.*, "Toxicant-associated Steatohepatitis," *Toxicol. Pathol.*, vol. 41, no. 2, pp. 343–360, Feb. 2013, doi: 10.1177/0192623312468517.
- [37] Z. M. Younossi, A. B. Koenig, D. Abdelatif, Y. Fazel, L. Henry, and M. Wymer, "Global epidemiology of nonalcoholic fatty liver disease—meta-analytic assessment of prevalence, incidence, and outcomes," *Hepatology*, vol. 64, no. 1, pp. 73–84, 2016.
- [38] B. M. Jarrar and Z. N. Mahmoud, "Histochemical demonstration of changes in the activity of hepatic phosphatases induced by experimental lead poisoning in male white rats (*Rattus norvegicus*)," *Toxicol. Ind. Health*, vol. 16, no. 1, pp. 7–15, Feb. 2000, doi: 10.1177/074823370001600102.
- [39] R. C. Patra, D. Swarup, and S. K. Dwivedi, "Antioxidant effects of  $\alpha$  tocopherol, ascorbic acid and l-methionine on lead induced oxidative stress to the liver, kidney and brain in rats," *Toxicology*, vol. 162, no. 2, pp. 81–88, May 2001, doi: 10.1016/S0300-483X(01)00345-6.
- [40] G. Vernon, A. Baranova, and Z. Younossi, "Systematic review: the epidemiology and natural history of non-alcoholic fatty liver disease and non-alcoholic steatohepatitis in adults," *Aliment. Pharmacol. Ther.*, vol. 34, no. 3, pp. 274–285, 2011.
- [41] "Definition & Facts of NAFLD & NASH | NIDDK," *National Institute of Diabetes and Digestive and Kidney Diseases*. <https://www.niddk.nih.gov/health-information/liver-disease/nafl-d-nash/definition-facts> (accessed Aug. 29, 2020).
- [42] J.-J. Geng *et al.*, "Nutrients and contaminants in tissues of five fish species obtained from Shanghai markets: risk–benefit evaluation from human health perspectives," *Sci. Total Environ.*, vol. 536, pp. 933–945, 2015.
- [43] B. O. Anyanwu, A. N. Ezejiofor, Z. N. Igweze, and O. E. Orisakwe, "Heavy Metal Mixture Exposure and Effects in Developing Nations: An Update," *Toxics*, vol. 6, no. 4, p. 65, Dec. 2018, doi: 10.3390/toxics6040065.
- [44] V. W. Setiawan, D. O. Stram, J. Porcel, S. C. Lu, L. Le Marchand, and M. Noureddin, "Prevalence of chronic liver disease and cirrhosis by underlying cause in understudied ethnic groups: The multiethnic cohort," *Hepatology*, vol. 64, no. 6, pp. 1969–1977, Dec. 2016, doi: 10.1002/hep.28677.
- [45] P. Y. Kwo, S. M. Cohen, and J. K. Lim, "ACG Clinical Guideline: Evaluation of Abnormal Liver Chemistries," *Am. J. Gastroenterol.*, vol. 112, no. 1, pp. 18–35, Jan. 2017, doi: 10.1038/ajg.2016.517.
- [46] V. Lala, A. Goyal, P. Bansal, and D. A. Minter, "Liver Function Tests," in *StatPearls*, Treasure Island (FL): StatPearls Publishing, 2020.
- [47] A. Lonardo *et al.*, "AISF position paper on nonalcoholic fatty liver disease (NAFLD): Updates and future directions," *Dig. Liver Dis.*, vol. 49, no. 5, pp. 471–483, May 2017, doi: 10.1016/j.dld.2017.01.147.
- [48] S. Chitturi *et al.*, "The Asia-Pacific Working Party on Non-alcoholic Fatty Liver Disease guidelines 2017-Part 2: Management and special groups," *J. Gastroenterol. Hepatol.*, vol. 33, no. 1, pp. 86–98, Jan. 2018, doi: 10.1111/jgh.13856.
- [49] A. R. Murali and W. D. Carey, "Liver Test Interpretation - Approach to the Patient with Liver Disease: A Guide to Commonly Used Liver Tests," *Cleveland Clinic - Center for Continuing Education*, Apr. 2014. <https://www.clevelandclinicmeded.com/medicalpubs/diseasemanagement/hepatology/guide-to-common-liver-tests/> (accessed Oct. 27, 2020).
- [50] A. Dasgupta and A. Wahed, "Chapter 10 - Liver Diseases and Liver Function Tests," in *Clinical Chemistry, Immunology and Laboratory Quality Control*, A. Dasgupta and A. Wahed, Eds. San Diego: Elsevier, 2014, pp. 177–195.
- [51] N. Chalasani *et al.*, "The diagnosis and management of nonalcoholic fatty liver disease: Practice guidance from the American Association for the Study of Liver Diseases: Hepatology, Vol. XX, No. X, 2017," *Hepatology*, vol. 67, no. 1, pp. 328–357, Jan. 2018, doi: 10.1002/hep.29367.
- [52] Z. M. Younossi *et al.*, "Changes in the Prevalence of the Most Common Causes of Chronic Liver Diseases in the United States From 1988 to 2008," *Clin. Gastroenterol. Hepatol.*, vol. 9, no. 6, pp. 524–530.e1, Jun. 2011, doi: 10.1016/j.cgh.2011.03.020.
- [53] S. Kojima, N. Watanabe, M. Numata, T. Ogawa, and S. Matsuzaki, "Increase in the prevalence of fatty liver in Japan over the past 12 years: analysis of clinical background," *J. Gastroenterol.*, vol. 38, no. 10, pp. 954–961, Oct. 2003, doi: 10.1007/s00535-003-1178-8.
- [54] U. Iqbal, B. Perumpail, D. Akhtar, D. Kim, and A. Ahmed, "The Epidemiology, Risk Profiling and Diagnostic Challenges of Nonalcoholic Fatty Liver Disease," *Medicines*, vol. 6, no. 1, p. 41, Mar. 2019, doi: 10.3390/medicines6010041.
- [55] G. C. Farrell, V. W.-S. Wong, and S. Chitturi, "NAFLD in Asia—as common and important as in the West," *Nat. Rev. Gastroenterol. Hepatol.*, vol. 10, no. 5, pp. 307–318, May 2013, doi: 10.1038/nrgastro.2013.34.
- [56] Z. M. Younossi *et al.*, "The global epidemiology of NAFLD and NASH in patients with type 2 diabetes: A systematic review and meta-analysis," *J. Hepatol.*, vol. 71, no. 4, pp. 793–801, Oct. 2019, doi: 10.1016/j.jhep.2019.06.021.
- [57] Z. M. Younossi *et al.*, "Nonalcoholic Fatty Liver Disease in Lean Individuals in the United States," *Medicine (Baltimore)*, vol. 91, no. 6, pp. 319–327, Nov. 2012, doi: 10.1097/MD.0b013e3182779d49.
- [58] A. Lonardo *et al.*, "Epidemiological modifiers of non-alcoholic fatty liver disease: Focus on high-risk groups," *Dig. Liver Dis.*, vol. 47, no. 12, pp. 997–1006, Dec. 2015, doi: 10.1016/j.dld.2015.08.004.

- [59] "OPTN/SRTR 2018 Annual Data Report: Liver." [https://srtr.transplant.hrsa.gov/annual\\_reports/2018/Liver.aspx](https://srtr.transplant.hrsa.gov/annual_reports/2018/Liver.aspx) (accessed Aug. 28, 2020).
- [60] A. Lonardo *et al.*, "Sex Differences in Nonalcoholic Fatty Liver Disease: State of the Art and Identification of Research Gaps," *Hepatology*, vol. 70, no. 4, pp. 1457–1469, Oct. 2019, doi: 10.1002/hep.30626.
- [61] S. Ballestri, F. Nascimbeni, E. Baldelli, A. Marrazzo, D. Romagnoli, and A. Lonardo, "NAFLD as a Sexual Dimorphic Disease: Role of Sex and Reproductive Status in the Development and Progression of Nonalcoholic Fatty Liver Disease and Inherent Cardiovascular Risk," *Adv. Ther.*, vol. 34, no. 6, pp. 1291–1326, 2017, doi: 10.1007/s12325-017-0556-1.
- [62] M. Balakrishnan *et al.*, "Women Have a Lower Risk of Nonalcoholic Fatty Liver Disease but a Higher Risk of Progression Vs Men: A Systematic Review and Meta-analysis," *Clin. Gastroenterol. Hepatol.*, p. S1542356520306121, Apr. 2020, doi: 10.1016/j.cgh.2020.04.067.
- [63] L. Wang, J. Guo, and J. Lu, "Risk factor compositions of nonalcoholic fatty liver disease change with body mass index in males and females," *Oncotarget*, vol. 7, no. 24, pp. 35632–35642, Jun. 2016, doi: 10.18632/oncotarget.9691.
- [64] M. Cave *et al.*, "Toxicant-associated steatohepatitis in vinyl chloride workers," *Hepatology*, vol. 51, no. 2, pp. 474–481, Feb. 2010, doi: 10.1002/hep.23321.
- [65] A. Michailova, T. Kuneva, and T. Popov, "A comparative assessment of liver function in workers in the petroleum industry," *Int. Arch. Occup. Environ. Health*, vol. 71 Suppl, pp. S46–49, Sep. 1998.
- [66] H. P. Cotrim, Z. A. Andrade, R. Parana, M. Portugal, L. G. Lyra, and L. A. R. Freitas, "Nonalcoholic steatohepatitis: a toxic liver disease in industrial workers," *Liver Int.*, vol. 19, no. 4, pp. 299–304, Aug. 1999, doi: 10.1111/j.1478-3231.1999.tb00053.x.
- [67] "NASH Definition & Prevalence — American Liver Foundation." <https://liverfoundation.org/for-patients/about-the-liver/diseases-of-the-liver/nonalcoholic-steatohepatitis-information-center/nash-definition-prevalence/> (accessed Aug. 28, 2020).
- [68] L. C. Bertot *et al.*, "Nonalcoholic fatty liver disease-related cirrhosis is commonly unrecognized and associated with hepatocellular carcinoma: Hepatology Communications, Month 2017," *Hepatol. Commun.*, vol. 1, no. 1, pp. 53–60, Feb. 2017, doi: 10.1002/hep4.1018.
- [69] R. Loomba, "Role of imaging-based biomarkers in NAFLD: Recent advances in clinical application and future research directions," *J. Hepatol.*, vol. 68, no. 2, pp. 296–304, Feb. 2018, doi: 10.1016/j.jhep.2017.11.028.
- [70] L. A. Adams *et al.*, "The Natural History of Nonalcoholic Fatty Liver Disease: A Population-Based Cohort Study," *Gastroenterology*, vol. 129, no. 1, pp. 113–121, Jul. 2005, doi: 10.1053/j.gastro.2005.04.014.
- [71] R. M. Carr, A. Oranu, and V. Khungar, "Nonalcoholic Fatty Liver Disease," *Gastroenterol. Clin. North Am.*, vol. 45, no. 4, pp. 639–652, Dec. 2016, doi: 10.1016/j.gtc.2016.07.003.
- [72] S. Petta *et al.*, "Pathophysiology of Non Alcoholic Fatty Liver Disease," *Int. J. Mol. Sci.*, vol. 17, no. 12, p. 2082, Dec. 2016, doi: 10.3390/ijms17122082.
- [73] International Diabetes Foundation, "IDF Diabetes Atlas, Ninth Edition, 2019," pdf, 2019. Accessed: Nov. 24, 2020. [Online]. Available: <https://www.idf.org/e-library/epidemiology-research/diabetes-atlas/159-idf-diabetes-atlas-ninth-edition-2019.html>.
- [74] A. Kautzky-Willer, J. Harreiter, and G. Pacini, "Sex and Sex Differences in Risk, Pathophysiology and Complications of Type 2 Diabetes Mellitus," *Endocr. Rev.*, vol. 37, no. 3, pp. 278–316, 2016, doi: 10.1210/er.2015-1137.
- [75] T. Fukuda *et al.*, "The impact of non-alcoholic fatty liver disease on incident type 2 diabetes mellitus in non-overweight individuals," *Liver Int.*, vol. 36, no. 2, pp. 275–283, Feb. 2016, doi: 10.1111/liv.12912.
- [76] Z.-J. Xu, J.-P. Shi, D.-R. Yu, L.-J. Zhu, J.-D. Jia, and J.-G. Fan, "Evaluating the Relationship Between Metabolic Syndrome and Liver Biopsy-Proven Non-Alcoholic Steatohepatitis in China: A Multicenter Cross-Sectional Study Design," *Adv. Ther.*, vol. 33, no. 11, pp. 2069–2081, Nov. 2016, doi: 10.1007/s12325-016-0416-4.
- [77] S. M. Grundy, H. B. Brewer, J. I. Cleeman, S. C. Smith, and C. Lenfant, "Definition of Metabolic Syndrome: Report of the National Heart, Lung, and Blood Institute/American Heart Association Conference on Scientific Issues Related to Definition," *Circulation*, vol. 109, no. 3, pp. 433–438, Jan. 2004, doi: 10.1161/01.CIR.0000111245.75752.C6.
- [78] P. L. Huang, "A comprehensive definition for metabolic syndrome," *Dis. Model. Mech.*, vol. 2, no. 5–6, pp. 231–237, May 2009, doi: 10.1242/dmm.001180.
- [79] A. Chowdhury and Z. M. Younossi, "Global Epidemiology and Risk Factors for Nonalcoholic Fatty Liver Disease," in *Alcoholic and Non-Alcoholic Fatty Liver Disease: Bench to Bedside*, N. Chalasani and G. Szabo, Eds. Cham: Springer International Publishing, 2016, pp. 21–40.
- [80] A. Lonardo, S. Ballestri, G. Marchesini, P. Angulo, and P. Loria, "Nonalcoholic fatty liver disease: A precursor of the metabolic syndrome," *Dig. Liver Dis.*, vol. 47, no. 3, pp. 181–190, Mar. 2015, doi: 10.1016/j.dld.2014.09.020.
- [81] N. Chalasani and G. Szabo, Eds., *Alcoholic and Non-Alcoholic Fatty Liver Disease*. Cham: Springer International Publishing, 2016.
- [82] I. Doycheva *et al.*, "Increasing Burden of Chronic Liver Disease Among Adolescents and Young Adults in the USA: A Silent Epidemic," *Dig. Dis. Sci.*, vol. 62, no. 5, pp. 1373–1380, May 2017, doi: 10.1007/s10620-017-4492-3.
- [83] G. Cholaneril *et al.*, "Liver Transplantation for Nonalcoholic Steatohepatitis in the US: Temporal

- Trends and Outcomes,” *Dig. Dis. Sci.*, vol. 62, no. 10, pp. 2915–2922, Oct. 2017, doi: 10.1007/s10620-017-4684-x.
- [84] H. S. Kim, Y. J. Kim, and Y. R. Seo, “An Overview of Carcinogenic Heavy Metal: Molecular Toxicity Mechanism and Prevention,” *J. Cancer Prev.*, vol. 20, no. 4, pp. 232–240, Dec. 2015, doi: 10.15430/JCP.2015.20.4.232.
- [85] A. A. Hunaiti and M. Soud, “Effect of lead concentration on the level of glutathione, glutathione S-transferase, reductase and peroxidase in human blood,” *Sci. Total Environ.*, vol. 248, no. 1, pp. 45–50, Mar. 2000, doi: 10.1016/S0048-9697(99)00548-3.
- [86] I. A. Grasso, M. R. Blattner, T. Short, and J. W. Downs, “Severe Systemic Lead Toxicity Resulting From Extra-Articular Retained Shrapnel Presenting as Jaundice and Hepatitis: A Case Report and Review of the Literature,” *Mil. Med.*, vol. 182, no. 3, pp. e1843–e1848, Mar. 2017, doi: 10.7205/MILMED-D-16-00231.
- [87] X. Wu, S. J. Cobbina, G. Mao, H. Xu, Z. Zhang, and L. Yang, “A review of toxicity and mechanisms of individual and mixtures of heavy metals in the environment,” *Environ. Sci. Pollut. Res.*, vol. 23, no. 9, pp. 8244–8259, May 2016, doi: 10.1007/s11356-016-6333-x.
- [88] I. Palma-Lara *et al.*, “Arsenic exposure: A public health problem leading to several cancers,” *Regul. Toxicol. Pharmacol.*, vol. 110, p. 104539, Feb. 2020, doi: 10.1016/j.yrtph.2019.104539.
- [89] O. US EPA, “Learn about Lead,” *US EPA*, Feb. 12, 2013. <https://www.epa.gov/lead/learn-about-lead> (accessed Nov. 28, 2020).
- [90] O. US EPA, “National Primary Drinking Water Regulations,” *US EPA*, Nov. 30, 2015. <https://www.epa.gov/ground-water-and-drinking-water/national-primary-drinking-water-regulations> (accessed Nov. 01, 2020).
- [91] World Health Organization, “Chemical fact Sheets,” in *Guidelines for drinking -water quality*, 4th ed., Geneva: World Health Organization, 2011.
- [92] International Agency for Research on Cancer and Weltgesundheitsorganisation, Eds., *IARC monographs on the evaluation of carcinogenic risks to humans, volume 100 C, arsenic, metals, fibres, and dusts: this publication represents the views and expert opinions of an IARC Working Group on the Evaluation of Carcinogenic Risks to Humans, which met in Lyon, 17 - 24 March 2009*. Lyon: IARC, 2012.
- [93] “Arsenic.” <https://www.who.int/news-room/fact-sheets/detail/arsenic> (accessed Nov. 28, 2020).
- [94] H. Ali, E. Khan, and I. Ilahi, “Environmental Chemistry and Ecotoxicology of Hazardous Heavy Metals: Environmental Persistence, Toxicity, and Bioaccumulation,” *J. Chem.*, vol. 2019, pp. 1–14, Mar. 2019, doi: 10.1155/2019/6730305.
- [95] M. TatabMentan *et al.*, “Toxic and Essential Elements in Rice and Other Grains from the United States and Other Countries,” *Int. J. Environ. Res. Public. Health*, vol. 17, no. 21, p. 8128, Nov. 2020, doi: 10.3390/ijerph17218128.
- [96] ATSDR, “Lead (Pb) Toxicity: What is Lead? | ATSDR - Environmental Medicine & Environmental Health Education - CSEM.” <https://www.atsdr.cdc.gov/csem/csem.asp?csem=34&po=4> (accessed Nov. 29, 2020).
- [97] G. Chiocchetti, C. Jadán-Piedra, D. Vélez, and V. Devesa, “Metal(loid) contamination in seafood products,” *Crit. Rev. Food Sci. Nutr.*, vol. 57, no. 17, pp. 3715–3728, Nov. 2017, doi: 10.1080/10408398.2016.1161596.
- [98] Y. Liu, S. Buchanan, H. A. Anderson, Z. Xiao, V. Persky, and M. E. Turyk, “Association of methylmercury intake from seafood consumption and blood mercury level among the Asian and Non-Asian populations in the United States,” *Environ. Res.*, vol. 160, pp. 212–222, Jan. 2018, doi: 10.1016/j.envres.2017.09.031.
- [99] O. Adeoye *et al.*, “Recommendations for the Establishment of Stroke Systems of Care: A 2019 Update: A Policy Statement From the American Stroke Association,” *Stroke*, vol. 50, no. 7, Jul. 2019, doi: 10.1161/STR.0000000000000173.
- [100] ATSDR, “Cadmium (Cd) Toxicity: Where is Cadmium Found? | ATSDR - Environmental Medicine & Environmental Health Education - CSEM.” <https://www.atsdr.cdc.gov/csem/csem.asp?csem=6&po=5> (accessed Dec. 15, 2020).
- [101] A. H. Mokdad *et al.*, “Prevalence of Obesity, Diabetes, and Obesity-Related Health Risk Factors, 2001,” *JAMA*, vol. 289, no. 1, pp. 76–79, Jan. 2003, doi: 10.1001/jama.289.1.76.
- [102] A. H. Welch, D. B. Westjohn, D. R. Helsel, and R. B. Wanty, “Arsenic in Ground Water of the United States: Occurrence and Geochemistry,” *Groundwater*, vol. 38, no. 4, pp. 589–604, 2000, doi: 10.1111/j.1745-6584.2000.tb00251.x.
- [103] D. S. Paul, F. S. Walton, R. J. Saunders, and M. Stýblo, “Characterization of the Impaired Glucose Homeostasis Produced in C57BL/6 Mice by Chronic Exposure to Arsenic and High-Fat Diet,” *Environ. Health Perspect.*, vol. 119, no. 8, pp. 1104–1109, Aug. 2011, doi: 10.1289/ehp.1003324.
- [104] A. Santra *et al.*, “Hepatic Damage Caused by Chronic Arsenic Toxicity in Experimental Animals,” *J. Toxicol. Clin. Toxicol.*, vol. 38, no. 4, pp. 395–405, Jan. 2000, doi: 10.1081/CLT-100100949.
- [105] Erik J. Tokar, Windy A. Boyd, Jonathan H. Freedman, and Michael P. Waalkes, “Chapter 23: Toxic Effects of Metals,” in *Essentials of Toxicology*, 3rd ed., McGraw Hill Professional, 2015.
- [106] V. M. Nurchi, A. Buha Djordjevic, G. Crisponi, J. Alexander, G. Bjørklund, and J. Aaseth, “Arsenic Toxicity: Molecular Targets and Therapeutic Agents,” *Biomolecules*, vol. 10, no. 2, p. 235, Feb. 2020, doi: 10.3390/biom10020235.

- [107] M. E. Morales *et al.*, "Heavy Metal Exposure Influences Double Strand Break DNA Repair Outcomes," *PLOS ONE*, vol. 11, no. 3, p. e0151367, Mar. 2016, doi: 10.1371/journal.pone.0151367.
- [108] J. Hallauer, X. Geng, H.-C. Yang, J. Shen, K.-J. Tsai, and Z. Liu, "The Effect of Chronic Arsenic Exposure in Zebrafish," *Zebrafish*, vol. 13, no. 5, pp. 405–412, Oct. 2016, doi: 10.1089/zeb.2016.1252.
- [109] S. C. Lu, "Dysregulation of glutathione synthesis in liver disease," *Liver Res.*, vol. 4, no. 2, pp. 64–73, Jun. 2020, doi: 10.1016/j.livres.2020.05.003.
- [110] N. Kaplowitz, "The importance and regulation of hepatic glutathione," *Yale J. Biol. Med.*, vol. 54, no. 6, pp. 497–502, Dec. 1981.
- [111] Y.-C. Lin *et al.*, "Association between soil heavy metals and fatty liver disease in men in Taiwan: a cross sectional study," *BMJ Open*, vol. 7, no. 1, Jan. 2017, doi: 10.1136/bmjopen-2016-014215.
- [112] S. Eskreis-Winkler *et al.*, "IDEAL-IQ in an oncologic population: meeting the challenge of concomitant liver fat and liver Iron," *Cancer Imaging*, vol. 18, no. 1, p. 51, Dec. 2018, doi: 10.1186/s40644-018-0167-3.
- [113] S. S. Martinez *et al.*, "Low Plasma Zinc Is Associated with Higher Mitochondrial Oxidative Stress and Faster Liver Fibrosis Development in the Miami Adult Studies in HIV Cohort," *J. Nutr.*, vol. 147, no. 4, pp. 556–562, Apr. 2017, doi: 10.3945/jn.116.243832.
- [114] F. Albareda *et al.*, "Medical applications of Cu, Zn, and S isotope effects," *Metallomics*, vol. 8, no. 10, pp. 1056–1070, 2016, doi: 10.1039/C5MT00316D.
- [115] C. E. Kelley, "Review of nonalcoholic fatty liver disease in women with polycystic ovary syndrome," *World J. Gastroenterol.*, vol. 20, no. 39, p. 14172, 2014, doi: 10.3748/wjg.v20.i39.14172.
- [116] K. R. Lee, K. D. Ko, I. C. Hwang, H. S. Suh, and K. K. Kim, "Association between blood lead levels and blood pressures in a non-smoking healthy Korean population," *Postgrad. Med. J.*, vol. 93, no. 1103, pp. 513–518, Sep. 2017, doi: 10.1136/postgradmedj-2016-134208.
- [117] "Liver function tests - Mayo Clinic." <https://www.mayoclinic.org/tests-procedures/liver-function-tests/about/pac-20394595> (accessed Oct. 27, 2020).
- [118] E. Carey, A. Wieckowska, and W. D. Carey, "Nonalcoholic Fatty Liver Disease," *Cleveland Clinic - Center for Continuing Education*, Mar. 2013. <https://www.clevelandclinicmeded.com/medicalpubs/diseasemanagement/hepatology/nonalcoholic-fatty-liver-disease-march-13/> (accessed Oct. 22, 2020).
- [119] Z. M. Younossi, A. B. Koenig, D. Abdelatif, Y. Fazel, L. Henry, and M. Wymer, "Global epidemiology of nonalcoholic fatty liver disease—Meta-analytic assessment of prevalence, incidence, and outcomes: HEPATOLOGY, Vol. XX, No. X 2016," *Hepatology*, vol. 64, no. 1, pp. 73–84, Jul. 2016, doi: 10.1002/hep.28431.
- [120] S. Mitra, A. De, and A. Chowdhury, "Epidemiology of non-alcoholic and alcoholic fatty liver diseases," *Transl. Gastroenterol. Hepatol.*, vol. 5, pp. 16–16, Apr. 2020, doi: 10.21037/tgh.2019.09.08.
- [121] A. Ofosu, "Non-alcoholic fatty liver disease: controlling an emerging epidemic, challenges, and future directions," *Ann. Gastroenterol.*, 2018, doi: 10.20524/aog.2018.0240.
- [122] F. A. Cimini *et al.*, "Relationship between adipose tissue dysfunction, vitamin D deficiency and the pathogenesis of non-alcoholic fatty liver disease," *World J. Gastroenterol.*, vol. 23, no. 19, pp. 3407–3417, May 2017, doi: 10.3748/wjg.v23.i19.3407.
- [123] E. Vilar-Gomez and N. Chalasani, "Non-invasive assessment of non-alcoholic fatty liver disease: Clinical prediction rules and blood-based biomarkers," *J. Hepatol.*, vol. 68, no. 2, pp. 305–315, Feb. 2018, doi: 10.1016/j.jhep.2017.11.013.
- [124] M. V. Machado, J. Coutinho, F. Carepa, A. Costa, H. Proença, and H. Cortez-Pinto, "How adiponectin, leptin, and ghrelin orchestrate together and correlate with the severity of nonalcoholic fatty liver disease," *Eur. J. Gastroenterol. Hepatol.*, vol. 24, no. 10, p. 1166, Oct. 2012, doi: 10.1097/MEG.0b013e32835609b0.
- [125] J. Cai *et al.*, "Porous graphene-black phosphorus nanocomposite modified electrode for detection of leptin," *Biosens. Bioelectron.*, vol. 137, pp. 88–95, Jul. 2019, doi: 10.1016/j.bios.2019.04.045.
- [126] J.-B. Suh, S. M. Kim, G.-J. Cho, and K. M. Choi, "Serum AFBP levels are elevated in patients with nonalcoholic fatty liver disease," *Scand. J. Gastroenterol.*, vol. 49, no. 8, pp. 979–985, Aug. 2014, doi: 10.3109/00365521.2013.836754.
- [127] M. Furuhashi and G. S. Hotamisligil, "Fatty acid-binding proteins: role in metabolic diseases and potential as drug targets," *Nat. Rev. Drug Discov.*, vol. 7, no. 6, pp. 489–503, Jun. 2008, doi: 10.1038/nrd2589.
- [128] X. Gong *et al.*, "Membraneless reproducible MoS<sub>2</sub> field-effect transistor biosensor for high sensitive and selective detection of FGF21," *Sci. China Mater.*, vol. 62, no. 10, pp. 1479–1487, Oct. 2019, doi: 10.1007/s40843-019-9444-y.
- [129] J. D. Clarke, T. Sharapova, A. D. Lake, E. Blomme, J. Maher, and N. J. Cherrington, "Circulating microRNA 122 in the methionine and choline-deficient mouse model of non-alcoholic steatohepatitis: miRNA 122 and NASH," *J. Appl. Toxicol.*, vol. 34, no. 6, pp. 726–732, Jun. 2014, doi: 10.1002/jat.2960.
- [130] A. Moolla *et al.*, "Accurate non-invasive diagnosis and staging of non-alcoholic fatty liver disease using the urinary steroid metabolome," *Aliment. Pharmacol. Ther.*, vol. 51, no. 11, pp. 1188–1197, Jun. 2020, doi: 10.1111/apt.15710.
- [131] N. Atabaki-Pasdar *et al.*, "Predicting and elucidating the etiology of fatty liver disease: A machine learning modeling and validation study in the IMI DIRECT cohorts," *PLOS Med.*, vol. 17, no. 6, p. e1003149, Jun. 2020, doi: 10.1371/journal.pmed.1003149.

- [132] N. Perakakis *et al.*, “Non-invasive diagnosis of non-alcoholic steatohepatitis and fibrosis with the use of omics and supervised learning: A proof of concept study,” *Metabolism*, vol. 101, p. 154005, Dec. 2019, doi: 10.1016/j.metabol.2019.154005.
- [133] B. Han *et al.*, “A semi-packed micro GC column for separation of the NAFLD exhaled breath VOCs,” *Surf. Coat. Technol.*, vol. 363, pp. 322–329, Apr. 2019, doi: 10.1016/j.surfcoat.2019.02.049.
- [134] R. Deo and S. Panigrahi, “Prediction of Hepatic Steatosis (Fatty Liver) using Machine Learning,” in *Proceedings of the 2019 3rd International Conference on Computational Biology and Bioinformatics - ICCBB '19*, Nagoya, Japan, 2019, pp. 8–12, doi: 10.1145/3365966.3365968.
- [135] C.-C. Wu *et al.*, “Prediction of fatty liver disease using machine learning algorithms,” *Comput. Methods Programs Biomed.*, vol. 170, pp. 23–29, Mar. 2019, doi: 10.1016/j.cmpb.2018.12.032.
- [136] M. Biswas *et al.*, “Symtosis: A liver ultrasound tissue characterization and risk stratification in optimized deep learning paradigm,” *Comput. Methods Programs Biomed.*, vol. 155, pp. 165–177, Mar. 2018, doi: 10.1016/j.cmpb.2017.12.016.
- [137] C. L. Fanola *et al.*, “A novel risk prediction score in atrial fibrillation for a net clinical outcome from the ENGAGE AF-TIMI 48 randomized clinical trial,” *Eur. Heart J.*, vol. 38, no. 12, pp. 888–896, Mar. 2017, doi: 10.1093/eurheartj/ehw565.
- [138] M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang, “Disease Prediction by Machine Learning Over Big Data From Healthcare Communities,” *IEEE Access*, vol. 5, pp. 8869–8879, 2017, doi: 10.1109/ACCESS.2017.2694446.
- [139] M. Shahabi, H. Hassanpour, and H. Mashayekhi, “Rule extraction for fatty liver detection using neural networks,” *Neural Comput. Appl.*, vol. 31, no. 4, pp. 979–989, Apr. 2019, doi: 10.1007/s00521-017-3130-5.
- [140] U. R. Acharya *et al.*, “Decision support system for fatty liver disease using GIST descriptors extracted from ultrasound images,” *Inf. Fusion*, vol. 29, pp. 32–39, May 2016, doi: 10.1016/j.inffus.2015.09.006.
- [141] A. Canbay *et al.*, “Non-invasive assessment of NAFLD as systemic disease—A machine learning perspective,” *PLOS ONE*, vol. 14, no. 3, p. e0214436, Mar. 2019, doi: 10.1371/journal.pone.0214436.
- [142] P. Sorino *et al.*, “Selecting the best machine learning algorithm to support the diagnosis of Non-Alcoholic Fatty Liver Disease: A meta learner study,” *PLOS ONE*, vol. 15, no. 10, p. e0240867, Oct. 2020, doi: 10.1371/journal.pone.0240867.
- [143] G. I. Rajathi and G. W. Jiji, “Chronic Liver Disease Classification Using Hybrid Whale Optimization with Simulated Annealing and Ensemble Classifier,” *Symmetry*, vol. 11, no. 1, p. 33, Jan. 2019, doi: 10.3390/sym11010033.
- deep belief network parameters using grasshopper algorithm for liver disease classification,” *Int. J. Imaging Syst. Technol.*, vol. 30, no. 1, pp. 168–184, Mar. 2020, doi: 10.1002/ima.22375.
- [145] K. B. Kim, G. H. Kim, D. H. Song, H. J. Park, and C. W. Kim, “Automatic segmentation of liver/kidney area with double-layered fuzzy C-means and the utility of hepatorenal index for fatty liver severity classification,” *J. Intell. Fuzzy Syst.*, vol. 39, no. 1, pp. 925–936, Jul. 2020, doi: 10.3233/JIFS-191850.
- [146] Y. Huo *et al.*, “Fully automatic liver attenuation estimation combining CNN segmentation and morphological operations,” *Med. Phys.*, vol. 46, no. 8, pp. 3508–3519, Aug. 2019, doi: 10.1002/mp.13675.
- [147] A. Han *et al.*, “Noninvasive Diagnosis of Nonalcoholic Fatty Liver Disease and Quantification of Liver Fat with Radiofrequency Ultrasound Data Using One-dimensional Convolutional Neural Networks,” *Radiology*, vol. 295, no. 2, pp. 342–350, May 2020, doi: 10.1148/radiol.2020191160.
- [148] W. Cao, X. An, L. Cong, C. Lyu, Q. Zhou, and R. Guo, “Application of Deep Learning in Quantitative Analysis of 2-Dimensional Ultrasound Imaging of Nonalcoholic Fatty Liver Disease,” *J. Ultrasound Med.*, vol. 39, no. 1, pp. 51–59, Jan. 2020, doi: 10.1002/jum.15070.
- [149] A. E. Bohte, J. R. van Werven, S. Bipat, and J. Stoker, “The diagnostic accuracy of US, CT, MRI and 1H-MRS for the evaluation of hepatic steatosis compared with liver biopsy: a meta-analysis,” *Eur. Radiol.*, vol. 21, no. 1, pp. 87–97, Jan. 2011, doi: 10.1007/s00330-010-1905-5.
- [150] S. C. Lin *et al.*, “Noninvasive Diagnosis of Nonalcoholic Fatty Liver Disease and Quantification of Liver Fat Using a New Quantitative Ultrasound Technique,” *Clin. Gastroenterol. Hepatol.*, vol. 13, no. 7, pp. 1337–1345.e6, Jul. 2015, doi: 10.1016/j.cgh.2014.11.027.
- [151] J. S. Paige *et al.*, “A Pilot Comparative Study of Quantitative Ultrasound, Conventional Ultrasound, and MRI for Predicting Histology-Determined Steatosis Grade in Adult Nonalcoholic Fatty Liver Disease,” *Am. J. Roentgenol.*, vol. 208, no. 5, pp. W168–W177, May 2017, doi: 10.2214/AJR.16.16726.
- [152] R. Hernaez *et al.*, “Diagnostic accuracy and reliability of ultrasonography for the detection of fatty liver: A meta-analysis,” *Hepatology*, vol. 54, no. 3, pp. 1082–1090, Sep. 2011, doi: 10.1002/hep.24452.
- [153] Y. N. Zhang *et al.*, “Liver fat imaging—a clinical overview of ultrasound, CT, and MR imaging,” *Br. J. Radiol.*, p. 20170959, Jun. 2018, doi: 10.1259/bjr.20170959.
- [154] A. Han *et al.*, “Inter-platform reproducibility of ultrasonic attenuation and backscatter coefficients in assessing NAFLD,” *Eur. Radiol.*, vol. 29, no. 9, pp. 4699–4708, Sep. 2019, doi: 10.1007/s00330-019-06035-9.

- [155] X. Forns *et al.*, “Identification of chronic hepatitis C patients without hepatic fibrosis by a simple predictive model: Identification of chronic hepatitis C patients without hepatic fibrosis by a simple predictive model,” *Hepatology*, vol. 36, no. 4, pp. 986–992, Oct. 2002, doi: 10.1053/jhep.2002.36128.
- [156] N. H. Afdhal, “Fibroscan (transient elastography) for the measurement of liver fibrosis,” *Gastroenterol. Hepatol.*, vol. 8, no. 9, pp. 605–607, Sep. 2012.
- [157] I. S. Idilman *et al.*, “Hepatic Steatosis: Quantification by Proton Density Fat Fraction with MR Imaging versus Liver Biopsy,” *Radiology*, vol. 267, no. 3, pp. 767–775, Jun. 2013, doi: 10.1148/radiol.13121360.
- [158] M. S. Middleton *et al.*, “Agreement Between Magnetic Resonance Imaging Proton Density Fat Fraction Measurements and Pathologist-Assigned Steatosis Grades of Liver Biopsies From Adults With Nonalcoholic Steatohepatitis,” *Gastroenterology*, vol. 153, no. 3, pp. 753–761, Sep. 2017, doi: 10.1053/j.gastro.2017.06.005.
- [159] A. Tang *et al.*, “Accuracy of MR Imaging–estimated Proton Density Fat Fraction for Classification of Dichotomized Histologic Steatosis Grades in Nonalcoholic Fatty Liver Disease,” *Radiology*, vol. 274, no. 2, pp. 416–425, Feb. 2015, doi: 10.1148/radiol.14140754.
- [160] L. L. Wald, P. C. McDaniel, T. Witzel, J. P. Stockmann, and C. Z. Cooley, “Low-cost and portable MRI,” *J. Magn. Reson. Imaging*, vol. 52, no. 3, pp. 686–696, Sep. 2020, doi: 10.1002/jmri.26942.
- [161] A. Mantovani *et al.*, “Heart valve calcification in patients with type 2 diabetes and nonalcoholic fatty liver disease,” *Metabolism*, vol. 64, no. 8, pp. 879–887, Aug. 2015, doi: 10.1016/j.metabol.2015.04.003.
- [162] M. Gambarin–Gelwan, S. V. Kinkhabwala, T. D. Schiano, C. Bodian, H. Yeh, and W. Futterweit, “Prevalence of Nonalcoholic Fatty Liver Disease in Women With Polycystic Ovary Syndrome,” *Clin. Gastroenterol. Hepatol.*, vol. 5, no. 4, pp. 496–501, Apr. 2007, doi: 10.1016/j.cgh.2006.10.010.
- [163] Z. M. Younossi *et al.*, “Association of nonalcoholic fatty liver disease (NAFLD) with hepatocellular carcinoma (HCC) in the United States from 2004 to 2009: Hepatology, Vol. XX, No. X, 2015 Younossi et al.,” *Hepatology*, vol. 62, no. 6, pp. 1723–1730, Dec. 2015, doi: 10.1002/hep.28123.
- [164] T. G. Cotter and M. Rinella, “Nonalcoholic Fatty Liver Disease 2020: The State of the Disease,” *Gastroenterology*, vol. 158, no. 7, pp. 1851–1864, May 2020, doi: 10.1053/j.gastro.2020.01.052.

## APPENDIX G – IRB INFORMATION

**Subject:** PROPEL Determination Letter - PROPEL # 17975020

**Date:** Thursday, January 31, 2019, at 12:03:10 PM Eastern Standard Time

**From:** Purdue HRPP

**To:** Ridhi Deo

### HUMAN RESEARCH PROTECTION PROGRAM INSTITUTIONAL REVIEW BOARDS

To: Suranjan Panigrahi

From: Purdue University Human Research Protections Program (HRPP) Title: Prediction model for Fatty Liver Disease

Date: 2019-01-31

Re: Exempt on Determination

Through the answers you provided in response to questions in the Purdue Research Online Portal Exemption Logic (PROPEL), Purdue's HRPP has determined that the research project identified above qualifies as exempt from IRB review, under federal human subjects research regulations Exemption Category 4 [Existing, Deidentified Biospecimens or Data; e.g., 45 CFR 46.101(b)(4)].

#### **The answers provided in PROPEL indicate your plans to:**

Utilize **existing** data, documents records, or specimens that are either publicly available or were recorded in such a manner that the identity of the subjects cannot be identified directly or indirectly by you or your research team.

Follow the terms and conditions of any contracts or access agreements regarding data security measures.

#### **What are your responsibilities now, as you move forward with your research?**

You (and any staff collecting or analyzing data from this study) must renew your training in human subjects research via CITI ([www.citiprogram.org](http://www.citiprogram.org)) every 5 years, based on the date of your last CITI training certificate. CITI will notify you via email when your current training certification is close to expiration.

You must keep all study records for a minimum of 3 years following closure of the study.

You and the members of your research team acknowledge that this study is subject to review at any time by Purdue's HRPP staff, Institutional Review Board, and or Research Quality Assurance unit. At any time, this project may be subject to post exemption- determination monitoring by these Purdue entities to confirm the applicability of this exemption status.

This determination constitutes the Purdue HRPP assessment of regulations related to human subjects research protections. This determination does not constitute approval from any other Purdue campus department or outside agency. The Principal Investigator and all researchers are required to affirm that the research meets all applicable local, state, and federal laws that may apply.

Finally, if any changes occur with respect to this research project, recognize that such changes could result in change in need for review by HRPP/IRB. Therefore, it is important that you again complete PROPEL to ensure that your research remains exempt from IRB review.

Should you have any questions about your rights and responsibilities regarding conducting research with people, on this project or any other, please do not hesitate to contact Purdue's HRPP at [irb@purdue.edu](mailto:irb@purdue.edu). We are here to help! ■