PREDICTING DELAYS IN DELIVERY PROCESS USING MACHINE LEARNING-BASED APPROACH

by

Shehryar Shahid

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science

Department of Computer Science

Fort Wayne, Indiana

December 2020

THE PURDUE UNIVERSITY GRADUATE SCHOOL

STATEMENT OF COMMITTEE APPROVAL

Dr. Adolfo Coronado, Chair

Department of Computer Science

Dr. Jin Soung Yoo

Department of Computer Science

Dr. Mohammadreza Hajiarbabi

Department of Computer Science

Approved by:

Dr. Jin Soung Yoo

Dedicated to my family.

ACKNOWLEDGMENTS

I would like to thank my family and friends for their love and support. I would also like to thank my advisor Dr. Adolfo Coronado for his guidance and mentorship.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

There has been a great interest in applying Data Science, Machine Learning, and AI-related technologies in recent years. Industries are adopting these technologies very rapidly, which has enabled them to gather valuable data about their businesses. One such industry that can leverage this data to improve their business's output and quality is the logistics and transport industry. This phenomenon provides an excellent opportunity for companies who rely heavily on air transportation to leverage this data to gain valuable insights and improve their business operations. This thesis is aimed to leverage this data to develop techniques to model complex business processes and design a machine learning-based predictive analytical approach to predict process violations.

This thesis focused on solving delays in shipment delivery by modeling a prediction technique to predict these delays. The approach presented here was based on real airfreight shipping data, which follows the International Air and Transport Association industry standard for airfreight transportation, to identify shipments at risk of being delayed. By leveraging the shipment process structure, this research presented a new approach that solved the complex event-driven structure of airfreight data that made it difficult to model for predictive analytics.

By applying different data mining and machine learning techniques, prediction techniques were developed to predict delays in delivering airfreight shipments. The prediction techniques were based on random forest and gradient boosting algorithms. To compare and select the best model, the prediction results were interpreted in the form of six confusion matrix-based performance metrics. The results showed that all the predictors had a high specificity of over 90%, but the sensitivity was low, under 44%. Accuracy was observed to be over 75%, and a geometric mean was between 58% – 64%.

The performance metrics results provided evidence that our approach could be implemented to develop a prediction technique to model complex business processes. Additionally, an early prediction method was designed to test predictors' performance if complete process information was not available. This proposed method delivered compelling evidence suggesting that early prediction can be achieved without compromising the predictor’s performance.

CHAPTER 1. INTRODUCTION

With the advancements in internet connectivity, a massive amount of digital data is collated from multiple sources every day. Social media, web browsing, inventory and order management systems, enterprise resource planning (ERP) systems, and hardware-based data sources such as global positioning systems (GPS), mobile and sensors, radio frequency identification (RFID) systems, and many others are examples of data sources available to companies nowadays (Govindan et al., 2018). These sources produce large quantities and various types, such as structured and unstructured, referred to as big data.

All this collected data contains information that can be analyzed using data analytics techniques to extract valuable information. Many new data analysis techniques and advanced algorithms, such as machine learning and AI, have effectively performed this analysis. Industries are now adopting these advanced data analytics tools and technologies rapidly, which has enabled them to gather valuable data about their business (Ben Ayed et al., 2015). A recent study by International Data Corporation (IDC) showed that the global spending in Data Science, Machine Learning, and AI-related technologies in 2020 had reached $50.1 billion and forecasted that this would double in the next four years. These data analytics techniques have transformed healthcare, consumer goods, governments, and many other industries (Cukier & Mayer-Schoenberger, 2013) by enabling them to make better decisions.

One such industry that could benefit from predictive and data analytics technology is the supply chain and logistics industry. A research study conducted by Schoenherr & Speier-Pero (2015) on the current application of predictive analytics in the supply chain, its challenges, and its future potential revealed that 40% of the companies they surveyed were using some analytics, 8.7% had the plan to use analytics in future, and only 22% had no plan to implement any form of analytics at all. The most common reason for adopting predictive analytics in the supply chain industry is that it would add value to the supply chain process. Another research study by Govindan et al. (2018) found that there has been a considerable increase in research publications related to the application of data analytics in the supply chain and logistics industry since 2012, particularly in the US, UK, and China. There are many reasons behind the industrial application of predictive analytics, including improved decision-making abilities and supply chain operations, better risk management, efficient demand planning and forecasting capabilities, and reduced costs.

Predictive analytic is a branch of data engineering that predicts future occurrences based on historical data or events analysis using a combination of data mining and machine learning techniques (Mishra & Silakari, n.d.). Predictive analytics application requires an understanding of the business domain, data, and analytical techniques. Predictive analysis has found its applications in various fields, including medical and healthcare, finance, telecommunication, insurance, customer relations, and many other fields.

Descriptive analytical techniques such as process and data mining techniques have been widely used by the telecommunication and healthcare industries to improve their operations and provide better customer support. Senderovich et al. (2014) extended this descriptive-analytical perspective to include predictive perspective by developing a novel queue mining approach based on a combination of snapshot principle and machine learning-based predictors. The results provided evidence that this combined snapshot-based predictor approach performed far better than the individual models in predicting delay times for queued customers in a bank’s call center. Husband and Roberts (2017) applied predictive analysis techniques in human resource management to predict the time to fill a job position. They combined the survival analysis technique to analyze the survival rate after surgery with machine learning tree-based models to develop a unique prediction approach.

Within the supply chain industry, one such domain that can leverage this data to improve their business's output and quality is logistics and transportation. Globally, industries rely heavily on the logistics and transport industry to ensure their supply chain remains uninterrupted. Over the years, industries have shifted towards air freight transportation, due to which there has been an increase in the air freight transportation industry. This provides an excellent opportunity to leverage air transport data to gain valuable insights and help companies, directly and indirectly, affect this industry to improve their business operations. To improve business performance and ensure competitiveness, the logistics and transportation industry has to shift to advanced data analytics techniques, such as predictive and proactive analytics, to make decisions quickly and efficiently by leveraging the full protentional of data analytics (Ben Ayed et al., 2015).

The logistics and transport industry blends multiple functions, such as infrastructure, networks, IT, and different stakeholders. This makes the whole process very complicated and creates issues associated with shipping goods efficiently around the world. Technology enables logistic companies to respond to evolving requirements and to keep up with the growing demand.

Companies working in the transportation industry are trying to optimize their business with advanced data analytics technologies to analyze resources, improve operational efficiency, and demand forecasting. Predictive analytics will help these companies make better-informed decisions (Iovan, n.d.). In recent years, the transport and logistics industries have seen rapid growth, causing companies to struggle for efficient transport modes and reducing risks.

Logistics companies, such as air cargo companies, experience several issues in managing their shipments. One such problem faced by these companies is the delay in delivery of airfreight shipments. Any delay in shipment results in an interruption in their supply chains and incurs a substantial monetary loss due to the high air transportation cost. Due to the complicated shipping process structure and high level of uncertainties, early prediction of any shipping process delays is challenging. The air cargo industry caters to time-sensitive and higher value goods that need to be transported to longer distances.

Literature shows that complex event processing has been the most popular analytics choice than other techniques. Alias et al. (2016) investigated research publications to understand the research trends related to complex event processing and predictive analytics in the transport and logistics industry and found that complex event processing was the most popular choice due to its long history of the application. Complex event processing is a rule-based technique where events trigger certain actions according to predefined rules, based on systematic importance (Alias et al., 2016). The study also highlighted the untapped potential that predictive techniques could bring to the transportation industry.

A limited number of research publications have been published highlighting the application of analytics in the logistics and transportation industry. Complex event processing has been a widely used analytics technique for the transport industry due to its past applications. Feldman et al. (n.d.) addressed the issue commonly faced by airfreight companies of incorrect freight volume and weight by leveraging real-time cargo data. They implemented a predictive and proactive approach using complex event processing to predict discrepancies in shipment's weight. The predictive complex event processing-based technique differs from predictive analytics because it uses a rule-based engine for prediction.

In contrast, the latter uses a prediction model developed using historical data for prediction. Metzger et al. (2015) studied ways to implemented predictive monitoring of business processes to resolve problems in the business process so that proactive action can be taken to resolve them.

This was accomplished by implementing and comparing three different predictive analytical techniques that included machine learning, constraint satisfaction, and Quality of Service aggregation.

This dissertation provides a solution for one major problem faced by air cargo companies: the delay in airfreight shipment arrival. Delays in shipment delivery incur a substantial financial loss to these companies and their stakeholders. The airfreight shipping process's complex nature poses a significant challenge to predict delays in shipment delivery efficiently. This dissertation focuses on providing a solution for this issue by developing a predictive analytical approach to identify shipments at risk of being delayed. This approach was based on real airfreight shipping data based on the IATA industry standard, called cargo 2000, for air freight transportation. This research aims to predict delays in airfreight delivery and help improve the shipping process.

1. Perform exploratory data analysis on cargo 2000 to understand the complexities in the shipment process
2. Study different machine learning techniques and develop a model to predict shipments that were delayed.
3. Perform experiments and compare the performance of different models to find the best model.

The literature related to applying different was analyzed to identify possible machine learning techniques for the prediction solution. Tree-based machine learning models provide an excellent opportunity for their application in developing a prediction model for predicting delivery delays in airfreight shipments. Gal et al. (2017) explored a novel method to predict a bus journey's traveling time using a combination of queueing theory and machine learning. The proposed solution combined the queue mining snapshot principle with tree-based machine learning algorithms, including random forest, Adaboost, and gradient boosting models. These models have also found their application in predicting energy usage. Ahmad et al. (2017) compared the performance of two well-known machine learning algorithms, random forest and artificial neural networks, to predict a hotel's energy consumption.

Based on the research goals introduced earler, the two research questions this thesis would answer are:

> RQ1. Can complex business process data be modeled to predict process violations such as delays?
>
> RQ2. Will the performance of prediction models decrease if the process violations were predicted early?

The rest of the thesis is organized into four sections. Section 3 provides the details of the literature reviewed to achive the first two research goals. It provides an overview of current state of application of predictive analytics in supply chain and logistics industry, and studies its application in other research areas. Section 4 introduces the airfreight dataset and describes the exploratory analysis performed on the data. It details indepth the process of modeling the data for prediction to achieve the first two goals. This section also introduces the performance metrics that were used to achieve the third goal. The results of prediction models and their performances are compared and discussed in Section 5. Conclusion and future research opportunities are presented in Section 6.

CHAPTER 2. LITERATURE REVIEW

With the advancements in the internet and technology, a vast amount of data is collected from multiple sources, such as social media, industrial operations, enterprise resource planning systems, logistics and transport, customer data, and many other technological sources. The development of big data analytics tools and software provides an excellent opportunity to extract valuable information from this data. Analyzing this data, especially in real-time, provides organizations the ability to make quicker decisions, thus reducing costs, improving operations, and providing better customer services. Govindan et al. (2018) conducted research to study papers published on big data analytics applications in supply chain and logistics and found evidence that there was an increasing interest in this research area. The USA, China, and the UK were the top 3 highest contributors to big data analytics papers. The authors collected evidence of a considerable increase in big data analytics research publications between 2012 and 2018.

2.1 Predictive Analytics

Schoenherr and Speier-Pero (2015) researched through a large-scale survey on the current application of predictive analytics in supply chain management, existing barriers in its application, and discussed its future potential. It provided insights into the current application of predictive analytics being used in the supply chain industry. One question posed in the study was how to leverage this data to gain insights and apply predictive analytics. The study found that 40% of their respondents were using some analytics, 8.7% had the plan to use analytics in the future, while 22% had no plan to implement any form of analytics. They also found that the most common factor behind the adoption of predictive analytics in supply chain management was the belief that it would add value to the supply chain process.

Another objective of this study was to highlight the benefits and barriers to applying predictive analytics in supply chain management. The main benefits that the companies believed that predictive analytics would bring included improved decision-making abilities, supply chain operations, risk management, demand planning capabilities, and reduction in costs. The main barriers identified in the study were employees' inexperience with predictive analytics, time constraints, integration between analytics software and existing systems, cost of predictive analytics solutions, and availability of predictive analytics solutions for the supply chain management.

Companies use forecasting to optimize their resources and forecast their demands to make their operations more efficient. The transportation industry's rapid growth has created companies' need to utilize their resources while reducing associated risks effectively. Iovan (n.d.) highlighted the transportation industry's evolution, analyzed the logistic process activities, and identified its associated risks. The vast amount of data collected from sources such as global positioning systems, time stamping, barcoding, and many other sources, too much information makes decision making complicated. This phenomenon creates a need for streamlining analytics processes for more accurate predictive analytics.

The air cargo industry caters to time-sensitive and higher value goods that need to be transported to longer distances. This industry, as compared to rail and road transport, is more cost and technology intensive. Airfreight companies have been striving to use technology to make their operation more efficient. According to Iovan (n.d.), predictive analytics will help these companies make better-informed decisions. These companies can implement a model-based approach for better forecasting and identify challenges and opportunities.

2.2 Transport Analytics

Alias et al. (2016) investigated the research literature related to complex event processing and predictive analytics in the transport and logistics industry to better understand the research currently being done in this area. The author discussed the different techniques of event processing, predictive analytics and monitoring, and their differences. Complex event processing was mostly implemented in application areas as compared to all other areas. This was primarily due to the long history of applying complex event processing in those areas instead of the other technologies. Another reason was its benefit in the application in the logistics and transportation processes. It was also highlighted that complex event processing and predictive analytical techniques go head-on, indicating the substantial untapped potential predictive analytics can bring to the transport and logistics industry.

The logistics sub-domain had the highest number of papers with forty-two publications in this sub-domain, compared to supply chain management and value-creation sub-domains, in different applications. According to the author, the reason behind this is the perceived benefit that businesses are expected to show an increase in their gains and improvements in efficiencies by introducing these technologies. A more in-depth analysis showed that these technologies are applied to as many as twenty-one different sub-domains, reinforcing the benefit of applying predictive technologies in supply chain, logistics, and transportation. Most papers examined showed that predictive techniques within the logistics domain were focused on a specific area. In contrast, the distribution of papers showed significant variance in the technologies applied. It showed great diversity in the use of different predictive technologies concerning different application sub-domains.

The domains and sub-domains with a higher number of complex event processing applications are not similar to those that employ predictive analytics and monitoring techniques or proactive monitoring applications. Sub-domains such as road transport, general transport, and supply chain integration have numerous complex event processing applications. On the other hand, predictive analytics and monitoring techniques are applied in air transport, road transport, and supply chain management. Solutions based on proactive monitoring techniques, supply chain integration, multimodal transport, and warehousing are areas with upcoming applications. The trends show a clear pattern that all three techniques focused on generic applications in specific domains. With time, the author believes that there will be wider adoption of these analytics and monitoring techniques since it is in an early phase of research, implementation, and testing. All techniques, complex event processing, predictive and proactive technologies have found their application in specific sub-domains and show the potential of their adoption in other domains.

Feldman et al. (n.d.) addressed the airfreight volume and weight issue by leveraging real-time cargo data. Their results indicated that their novel approach could be used to predict the weight of freight to enable proactive action for a better plan of shipment. Through their proactive approach, the authors demonstrated that predictions about shipments' weight could be made in advance of a shipment. Event processing is the connection between events and subsequent reactions required by the situation that generally deals with detecting event patterns and identifying reactive situations. These situations are generated by noticing patterns in the flow of an event. An event processing solution gets these events and transforms them into alerts to act human or autonomous. In this paper, the authors have augmented proactive computing to event processing. They refer to this proactive approach to alleviate undesired states or take advantage of predicted opportunities. Traditional event processing techniques do not deal with forecasting and predictions.

The authors suggested a new proactive event processing approach that leverages decision-making capabilities for future events forecasted from ongoing events. The authors have extended this reactive pattern approach to their novel approach that consists of four stages. First, detecting events using conventional event processing techniques, second, predicting future states using predictive analytics, third, deciding how to handle predicted events, and fourth, acting in the best way possible by enacting the decision.

According to the author, the motivation behind proactive computing came from economic and social factors. The paper presented a method of proactive event processing of run-time process management of logistics and transport events. Proactive applications have previously been developed, such as proactive security systems, routing in mobile networks, quality prediction, failure, and network management. In this paper, the authors proposed a single framework to augment both predictive and proactive behavior. This approach was divided into two modules: a forecasting module that predicts future states, and second, a near real-time decision-making module. The forecasting module uses predictive analytics that combines pattern recognition to predict future events. These future events will trigger the real-time decision-making module to carry out proactive actions.

The event-driven platform was based on the concept of an event processing network (EPN). They applied three types of event processing agents (EPA's). A filter agent takes an incoming event and applies a test to check whether to discard it or pass it for processing to the next agent. This split agent takes an incoming event and splits it into a collection of event objects. An enrich agent that takes an incoming event uses it to query data and creates a forecasted event that contains features from the original event. Each agent acts as the onset of event objects specified by a context. They can be assigned to one or more context partitions. The authors then extended the EPA concept to include a proactive event processing agent that allows the EPA to process patterns. The proactive agent analyzes the incoming events in a particular context and derives either an operative decision or a future event. These proactive agents are equipped with predictive and decision-making models. The predictive models were used to predict probabilities, numerical and categorical attributes.

The authors used airfreight cargo transportation data to put their novel approach to the test. The cargo process has numerous kinds of deviations, such as cancellations, volume, and weight violations. In this paper, the authors focused on the proactive management of weight discrepancies. These discrepancies occurred when there was a difference between the booked and the actual shipment weight. Two situations that occur are over-weight load and under-weight load. They first evaluated the probability of a discrepancy through the available dataset. To determine the predictive model, they have analyzed past shipments that have attributes including planned weight, number of packages, source, destination, and airlines. The actual weight was computed from milestone completion events. Their analysis found that the relationship between planned weight and actual weight was independent of any other attribute.

Using the weight forecasting model, the authors then developed a decision model that gives alerts based on two objectives, first, to capture as many violations as possible in advance (high recall), and second, to avoid false alarms as much as possible (high precision). Throughout the experiment, they maintained four sets: Alerted Under, Alerted Over, Actual Under, and Actual Over. These sets were used to calculate two performance metrics, recall, and precision for both overweight and underweight shipments. Evidence showed that overload alerts could be made with outstanding precision (almost 100%) while keeping the recall at 70%. A lower precision was observed (75-80%) at a high recall rate (95%). The authors had also examined the time between when an alert was made for a particular flight.

Metzger et al. (2015) studied ways to implemented predictive monitoring of business processes to identify problems in the business process. Proactive action can be taken to resolve them by implementing and comparing three different predictive analytical techniques. This includes machine learning, constraint satisfaction problem, and Quality of Service aggregation. The research is supported through evidence by implementing the proposed method on a case study in logistics and transport. They assessed the lead-time required for accurate prediction of the business process problem. The paper presented a case study-based approach to analyze the proposed prediction techniques to predict process violations, compare each model's accuracy depending on process lead-time, and combine these techniques to improve prediction accuracy.

Amongst well-known machine learning techniques, the authors have selected Artificial Neural Network (ANN). To apply ANN for prediction monitoring, they have identified checkpoints in the business process. These checkpoints are process data on which they have carried out the predictions. They have used the execution time of each checkpoint of the service. Whenever an instance passes a checkpoint, a prediction is generated based on that particular instance. Then it compares the generated prediction with the planned checkpoint performance to decide the occurrence of the violation.

The second prediction model used by the authors is the constraint satisfaction problem (CSP). This is a problem-solving technique that has conditions that must be met. The solution to CSP are values assigned to variables that satisfy all the constraints. The CSP formulated by the authors is related to the execution time of each checkpoint. For their application of CSP, the authors have divided each instance into a sequence of service execution, and the execution time of each instance is derived from its components. The duration of a sequence of service execution s1 and s2 have time T1 and T2 respectively, so their sequence time is Tseq = T1 + T2 (this represents multiple hops within a leg). While the duration of their parallel execution (duration of multiple legs within an instance, it is also referred to as and-join) is Tand = max(T1, T2). The authors have used the business process structure to create a set of equations that needs to be satisfied. This was the first step in their implementation of CSP. In the second step, they solved the CSP with end-to-end quality targets, which, in their case, is the execution time limit. They created two cases: the required time limit is not exceeded and the second, where the time limit was exceeded. The CSP implemented in this paper was through the ECLIPSE constraint logic programming system.

The third technique used here is Quality-of-Service (QoS) aggregation. It is a rule-based reduction process model that determines QoS values. QoS aggregation rules are defined for the business process and include sequence time, parallel execution, and loop. At the time of execution, QoS rules are evaluated using process data and planned QoS values for the remaining service instance as an input. The authors have implemented their version of the QoS aggregation technique using the BOGOR model-checker system. The research was focused on answering three main research questions. First, what accuracy can be achieved when applying the three predictive models on business process instances? Second, how much accuracy depends upon the lead time for accurate prediction? Third, can these models be combined for more accurate predictions?

To answer these questions, the authors employed binary indicators for the prediction: violation and non-violation. Furthermore, the performance of predictions was measured by using the indicators to compute precision, recall, specificity, and accuracy. A case study implementation based on operational data from an international forwarding company was used to validate the hypothesis. The data was collected through the company's Cargo 2000 monitoring system. The data was used to predict violations in service delivery and analyze the accuracy of their prediction techniques. Two-thirds of the dataset was used as a training data set, while the remaining one-third of the dataset was used as a testing dataset. Each technique used in this paper predicts either true or false based on each checkpoint in the business instance. To visualize the lead-time prediction, the authors have used a normalized scale to compensate for the different number of service execution within an instance. They have represented this as a position in their contingency table. This position ranges between 0% to 100%, and 50% represents the point where all incoming legs are consolidated.

The data was engineered to create 21 checkpoints for each instance. The ANN was then trained for each checkpoint using their training dataset and used the test dataset to calculate the contingency matrix. All parameters used by the authors for their ANN were default proposed by WEKA. Their results show that the ANN prediction is less accurate up to the 50% position mark, after which the accuracy increases substantially. The constraint satisfaction technique used the training dataset to set the lower and upper bound of acceptable execution time. At each checkpoint, the CSP looks for the remaining service and predicts whether a violation can be ruled out or not. This approach can only predict violation and non-violation of checkpoints that are closer to the point of prediction. A similar contingency table was developed to represent the results. This approach gives the highest levels of precision, recall, accuracy, and specificity but only for the earliest predictor at 100%.

The QoS aggregation technique does not require historical data, so only the test dataset was used. They used the planned checkpoint data as an estimate of service QoS. This method only starts prediction once it observes a violation of individual service. They observed that the precision increases gradually towards the end of process execution, but there is a sudden drop in the recall at 60% position. According to the authors, this is observed because the DEP service shows an 84% violation rate, which substantially impacts recall. Specificity in QoS performs better than the rest of the techniques because the planned service checkpoints are chosen too optimistically.

The authors explain that their application of the three prediction techniques and their results provide substantial evidence that these techniques can be applied to business process instances and yield excellent prediction accuracy results. The prediction accuracy is more significant than 0.7 for predictions that are made after 50% of the service execution. Their prediction accuracy increases for all techniques towards the end of the process execution. To answer their last research question, the authors combine these techniques. They used ensemble learning to analyze if there is an increase in accuracy when these techniques are combined as compared to them individually. The first voting technique they used is called the majority. It predicts a violation of at least two techniques predicts a violation. Depending on the actual setting, this technique can improve the accuracy of the prediction of violations or non-violations.

The authors then used the majority voting to define three specific voting strategies further. This first strategy is aimed at increasing the recall by combining two techniques that delivered the best recall. In their setting, they combined machine learning and constraint satisfaction. To increase recall, they predicted a true-positive if at least one of the techniques predicted a real positive. Their next strategy was to increase the precision; thus, they combined CSP and QoS and defined their strategy. If both predicted a violation only, then the final prediction will be a violation. This helped in reducing false violations. Their last strategy was to increase the specificity by combining CSP and QoS techniques. This voting technique is designed such that a non-violation is predicted if at least one of the techniques predicts a non-violation. These combined techniques showed improvements in contingency table matrices. Their techniques provide substantial evidence that combining techniques can perform better than individual techniques.

2.3 Predictor Based Approaches

Senderovich et al. (2014) implemented a queuing perspective in operational process mining. They have gained motivation from the recent work on process mining that showed how the management of these processes could be guided by models extracted from the event logs recorded during process operation—the paper presented techniques for queue mining using two different approaches. First, a traditional approach to process mining is implemented that enhances an existing time prediction system to consider queues and system loads. Second, the queueing theory was applied to the model, analyze, and improve the service process. In addition to defining mining techniques, a comprehensive experimental evaluation using real-world logs was also presented. The results showed that the proposed techniques improved predictions.

The authors illustrated the need for a queueing perspective in the operating model for service processing using a bank's call center. Queue mining theory was implemented to solve the problem of online delay prediction. To solve the delay prediction problem, a mining technique for three classes of delay predictors for proposed. The first predictor included integrating queueing information into operational process mining techniques by considering queues as separate activities. This approach was implemented by considering the queueing phase and service phase as two different steps. Other cases on each case were considered by extending the state construct to include customers' system load in the queue. The system load was divided into high, moderate, and low load regions to account for customer load variation.

The second class of predictors based on queueing theory was developed. Based on this, two predictors were designed: queue length-based predictor (QLP) and Markovian queue length-based predictor (QLMP). These predictors use the queue lengths to predict the expected delay. These predictors were applied to real-world customer queue data from a bank call center to test the proposed predictors. Two performance matrices, absolute bias to measure accuracy and root-mean-squared error to measure precision, were used to evaluate the predictors' performance. RMSE was more significant because it penalizes for any deviation from the actual delay in time compared to absolute bias. Thus, RMSE was considered as the sole indicator to measure the performance of the predictors.

The prediction model was considered the control variable in this research. The training data was mine to cluster the queue into three categories: high load, moderate load, and typical load. Both the queue length-based predictors (QLP and QLMP) were found to be biased. For some instances, the snapshot-based predictors showed negligible systematic errors, making them ideal for their delay prediction applicability.

The authors divided their findings of the predictors into three sections. The first is about the transition system method where both of the methods, PTS and KTS, compare past delays of customers with similar path history to predict the delay. KTS method performed significantly better because it captures three different steady states. The second section compared the queue length predictors, QLP, and QLMP. QLP predictor failed for most cases due to a lack of reasonable assumptions, while QLMP outperforms PTS for most of the scenarios in both accuracy and prediction. However, QLMP was found to be inferior to KTS and the snapshot predictors. The final section compared the snapshot-based predictors. All the snapshot-based predictors showed better performance consistently than the rest of them. All the evidence leads to the authors' conclusion that the snapshot-based predictors are well suited for the application to predict delay time for a newly queued customer.

2.4 Random Forest

Decision Trees and Random Forests are two of the most popular Machine Learning Algorithms used for classification. Datla (2015) made a comparison of classification results from these algorithms based on bike-sharing and Titanic datasets. The method proposed was divided into two parts. First, feature engineering was used to eliminate noise in data, and new variables were created based on existing variables. Second, to ensure optimum results, classification parameters were tailored based on precision and accuracy before training models with the data.

The authors used two types of datasets, a large data set and a small dataset, to compare prediction models' results. To avoid overfitting, a 10-fold cross-validation method was employed. The research was aimed at determining the efficiency of both the models on varying dataset sizes. Based on the data result, the authors found that classifiers based on Decision Trees performed far better on smaller data sets than Random Forest-based classifiers. Whereas Random Forest-based classifiers performed better on more massive datasets using the same number of variables.

Ahmad et al. (2017) performed a comparison of performance between Random Forest, an ensemble-based tree model, and artificial neural networks, a feed-forward back-propagation model to predict a hotel's energy consumption. Different approaches have been proposed to predict building energy consumption classified into three categories: numerical, analytical, and predictive. Artificial neural network collects information from historical data in many hidden layers. They work as a black-box model, generating a relationship between the input and output. Feed-forward is the most popular neural network model generally used. The authors have implemented a feed-forward neural network coupled with a back-propagation algorithm to predict energy consumption. Another method used in this paper was a random forest, an ensemble-based tree method that uses a voting scheme to vote for the most popular class.

The research was conducted on energy data collected from the hotel's energy management system. Data from external factors such as the number of guests and outside weather information was also collected and considered when building the models. Four evaluation matrices, including root mean square error (RMSE), mean absolute percentage error (MAPE), coefficient of variation (CV), and mean absolute deviation (MAD), were used. An artificial neural network's sensitivity for the different number of hidden layer neurons was studied, after which ten neurons were selected to reduce the model's complexity. The random forest model's depth was also studied and found that random forest performance decreased as maximum tree depth increased to more than ten.

Comparing the performance matrices for both the models showed that the artificial neural network had a slightly better performance than the random forest-based model. Although there was a slight difference in the performance, both models showed nonlinear mapping capability, making them ideal for predicting energy consumption. An advantage of using random forest over the neural network is its ability to handle missing values. Due to the ensemble-based algorithm, the random forest can quite accurately predict even when missing values are present. The experiments showed that the random forest-based model performed better when predicting lower energy consumption values, while the neural network-based model performed better in predicting higher values.

Comparing the models, the authors found that random forest took much less time to train, a couple of seconds, while the neural networks took a couple of minutes to train. While training time depends on multiple factors like the algorithm's implementation, the number of variables, model complexity, etc., there was substantial evidence suggesting that random forests are quicker than neural networks.

Husband and Roberts (2017) developed a set of quantifiable job features and then applied the random survival forest to build a predictor that would assess a job's probability of remaining open beyond its time-to-fill. The number of days from the date a new job is posted to the date it is filled. A predictor based on a random survival forest was developed. Survival analysis is a technique used to analyze patients' survival rates after surgery based on each patient's risk factors. The time-to-fill predictor follows the same principle, except here jobs are considered instead of patients and filled state of a job instead of death. Quantifiable job features such as job type, location, time of year, number of candidates interviewed, etc. from historical filled jobs data are used as risk factors.

The authors created a random survival forest based on these features from the historical data. They used 2/3 of the dataset to draw "n" bootstrap samples. The remaining 1/3 of the dataset was used for testing. For each bootstrap sample, they grew a survival tree from a randomly selected subset of features. Finally, they created a cumulative hazard function for each survival tree and used the test dataset to calculate the prediction error.

They made two adjustments to the typical random survival forest. Their time-to-fill predictor has categorized newly opened positions as high, medium, or low risk of missing the time-to-fill target. Secondly, they created three random survival forest-based predictors. Predictor A uses all available features, predictor B uses all features except candidate interviewed, and predictor C excludes candidate interviewed and candidate submitted. For jobs that have been opened for more than 21 days, they have used predictor A. For jobs that were open for days between 14 and

21, they have used predictor B. For jobs opened for less than 14 days, they have used predictor C. Using the predictors, they calculated the probabilities and categorized them into three risk categories. Jobs with p <= 0.5 were categorized as low risk, jobs with 0.5 < p <= 0.75 were categorized as medium risk and jobs with p > 0.75 were categorized as at high risk. All random survival forests were created in R using the randomForestSRC package.

The authors used historical data of 19,000 filled job positions. The prediction error was calculated from the randomForestSRC algorithm that uses Harrell's concordance index. Their goal was to flag jobs that missed the target time-to-fill. Further, they also tracked the degree to which a job missed its target using categories 1-15 days, 16-30 days, 31-60 days, and 61+ days. Using predictors on test data, they categorized it into high, medium, and low-risk groups. Then, for each of these, they further categorized each risk group into the miss target categories. Plotting this information showed that 85.7% of the high-risk group missed their time-to-fill target while only 24.8% of the low-risk group missed theirs.

2.5 Gradient Boosting

Gal et al. (2017) explored a novel method for traveling time prediction using a combination of queueing theory and machine learning. A prediction mechanism to predict a bus journey's traveling time using historical data, journey routes, and source/destination data is proposed. This research is based on Dublin city's bus network data and provides substantial evidence that predictions based on the queueing theory suffer from outliers. This issue was overcome by combining queueing theory with machine learning to assist in the identification of outliers. The proposed model segments the data according to intermediate stops to overcome outlier identification by implementing a machine learning technique on historical data to improve the queueing theory-based prediction results.

The prediction model suggested uses historical data of scheduled journeys and real-time information about vehicle movement. The information is stored in the journey log and contains sequences of recorded journey events and attributes for each bus trip. The problem of real-time travel time prediction for a bus journey has been addressed such that the journey events indicate the progress of the bus on its route. A novel solution to predict the traveling time for the next step based on their journey pattern has been implemented. To analyze the accuracy of their predictions, performance metrics such as root-mean-squared error (RMSE), mean absolute relative error

(MARE), and median absolute relative error (MdARE) were employed. A two-step approach was used to implement the model of travel time prediction. The first step modifies the input data to construct a model based on journey patterns. The second step uses this model to predict the actual travel-time. The predictions were based on a travel journey log and used either the most recent information, historical data, or a combination of both.

The prediction methods were divided into two categories. The first category used recent events to predict future traveling times using a model based on the queueing theory and approximates systems in heavy traffic. The second category was based on machine learning's decision tree approach. The first prediction model uses a heavy traffic approximation that, also known as the snapshot principle, is implemented. When a bus passes through a segment, it will experience the same traveling time as another bus that has just passed through that segment. A predictor was developed with this method and thus defined a multi-segment snapshot predictor Last-Bus-to-Travel-Network (LBTN). The hypothesis was stated that the snapshot predictor performs better whenever busses are close to a journey.

The second technique proposed uses a regression tree-based approach to predict travel time. Unlike the snapshot method, the regression tree method uses historical data to train a prediction model. This model was then used to predict the traveling time for new instances of an ongoing journey. The predictor uses features including the prediction time and estimated time for the bus that enters the segment and predicts the time bus will travel through the segment. This predictor approach consists of two parts. First, a feature set is defined, and then they use this feature set to construct a prediction model. The feature set considered for modeling the predictor includes the bus's travel time that last used the segment, the interval between the time the last bus left the segment, the day of the week, and the time of the day. The authors implemented different ensemble-based regression trees. All of them take the feature set as input and produces an output that predicts the traveling time. A collection of multiple regression tree models was used since they produce better accuracy than a single model. This collection of regression trees included Random Forest (RF), Extremely Randomized Trees (ET), AdaBoost (AB), Gradient Tree Boosting (GB), and a robust version of Gradient Tree (GBLAD). Boosted tree algorithms were selected to construct models sequentially and combined them with the snapshot model. Three combined algorithms were proposed: S+AB, S+GB, and S+GBLAD.

The models were implemented using regression tree modeling in scikit-learn. Experiments showed that the RMSE increased as the trip length increased, while MARE decreased. This was observed for all prediction methods. Evaluating their techniques, the authors found that the combined prediction models outperformed the individual models in terms of prediction quality when comparing their performance. The prediction error increased as the number of stops in a journey increased. However, the relative error remained stable over the length of the trips. Snapshot-based predictors did not show any deterioration in performance for longer trips, thus contradicting the hypothesis that the snapshot predictor would be more precise for journeys with higher temporal proximity to the current journey. It was also found that the prediction accuracy correlated negatively with the number of busses traveling through the city.

Georganos et al. (2018) implemented extreme gradient boosting (Xgboost) based classifier for land use-land cover urban classification. They further evaluated its efficiency based on sample size and feature selection. Finally, they compared their gradient boosting-based classifier's performance with random forest- and support vector machine-based classifiers. Extreme gradient boosting belongs to the classification and regression tree (CART) family and is an extension of the traditional gradient boosting technique. The traditional gradient boosting machine model was prone to overfitting due to a lack of proper regularization aspect. Extreme boosting tackles this problem by having a regularization framework that overcomes the problem of overfitting. The authors employed a correlation-based feature selection, a method that uses correlation and co-variance to inspect the relation between the dependent and independent variables. This method allows the reduction in features by looking for features with high dependence on the dependent variable. The authors also optimized the parameters of the Xgboost classifier using Bayesian optimization.

The results showed that Xgboost performed better than both random forest and support vector machine-based models for larger sample sizes. The highest accuracy was observed when Xgboost was trained with the maximum possible number of features. However, it showed reduced accuracy when correlation-based feature selection was applied, while SVM and random forest showed a better performance. This research provides evidence that Xgboost with Bayesian optimization can outperform random forest and support vector machines, but it comes at the cost of increased computation time.

Batra and Jawa (2016) implemented multiple classification algorithms to detect arrhythmia using a combination of classification methods and electrocardiogram diagnostic techniques. The proposed solution was implemented using a clinical dataset from the UCI data repository and machine learning algorithms that included Decision trees, Random Forest, Gradient Boosting, Support Vector Machine, and Neural Networks. Matrices such as confusion matrix, kappa-score, confidence interval, the area under the curve, and accuracy were employed to measure and compare models' performance.

75% of the dataset was used for training, and the remaining 25% was used for testing. The dataset was divided into sixteen classes to represent normal and different types of arrhythmia. The arithmetic mean values were used to replace the missing feature data, and features that had the same values were removed for all instances. All continuous values were normalized using the Z-score normalization technique. The principal component analysis was employed to convert correlated features into linearly uncorrelated features. For classes with very few instances, the authors used ECG analysis criteria to identify them using machine learning algorithms accurately. Doing this showed 100% classification accuracy for some class labels. The dataset was then used to train and test all the machine learning models. The neural network showed better results for all performance matrices when a reduced feature set using principal component analysis was used instead of all the features. On the other hand, Decision trees, Random Forest, and Gradient boosting showed better classification results when all features were used to train the model. The random forest and gradient boosting models were implemented in R. XGBoost R package was used to develop the gradient boosting prediction model.

Finally, the support vector machine model was implemented using all feature sets, reduced PCA features, top 60 features from gradient boosting, and the random forest model's top features. The results showed that using top features from the gradient boosting model gave the best results for all performance matrices compared to other methods. The authors then combined the EGC diagnostic analysis technique with this to improve the overall accuracy even further.

The classification has found various applications in numerous fields. It uses training data to develop a classification model that predicts the class of new data. Assessing performance is a critical factor in evaluating a classification model and selecting the best classification problem model. Tharwat (2020) conducted a detailed study of different classification assessment techniques to understand the basics of these techniques. This study explained the confusion matrix for binary and multi-class classification problems and explained the influence of balanced and imbalanced data on multiple performance metrics. Furthermore, a comparison of various scalar and graph performance metrics presented a detailed analysis of each method's robustness against imbalance data.

Classification problems can be divided into binary and multi-class, depending on the number of classes. Binary has only two classes, while multi-class has more than two. Classification output can be either discrete or continuous. Discrete output predicts the class label, whereas the continuous output estimates the probability of class membership. In binary classification, there are four possible outcomes: true-positive, true-negative, false-positive, and false-negative. An imbalanced dataset occurs when members of one class outnumber those of the other class. Geometric Mean and Youden's index are two metrics that are not affected by a class imbalance, while accuracy and precision are affected by an imbalanced dataset. The author introduced multiple scalar performance matrices for binary classification, including Accuracy and Error Rate, Sensitivity and Specificity, False-Positive and False-Negative Rates, Predictive Values, Likelihood Ratio, Youden's Index, Matthews Correlation Coefficient, Discriminant Power, F1-Score, Markedness, Geometric Mean, Optimized Precision and Jaccard. Out of these matrices, Sensitivity, and Specificity, False-Positive and False-Negative-Rates, Likelihood Ratio, Youden's Index, Discriminant Power, Markedness, and Geometric Mean are not affected or slightly effected by imbalanced data.

The other performance assessment methods include graphical techniques such as Receiver Operating Characteristics (ROC) curve, Area under ROC (AUR), and Precision-Recall Curve. The ROC curve is a graphical technique representing true-positive rate and false-positive rate on the y-axis and x-axis. The area under the ROC curve is a technique to compare the performance of different classifiers. The precision-recall curve is similar to the ROC curve, except that it represents the relation between precession and recall. In contrast, the ROC curve represents the relation between true-positive and false-positive rates.

A key issue when selecting the best model measures is its performance on an imbalanced dataset. Imbalance datasets consist of an unequal number of elements in each class. Some models perform better in one class but provide poor performance in other classes. The impact of this on model performance has shown to be a significant problem. Luque et al. (2019) conducted an extensive study to develop and analyze the effect of imbalanced datasets on different performance matrices and determine the most suitable performance metrics through results obtained by using binary classifiers. A new method to characterize class disparity was proposed to measure imbalances that surpass the commonly used imbalance ratio. This was achieved by developing functions and numerical indicators to compare the imbalanced dataset's behavior on multiple performance matrices based on the confusion matrix. This research focused on the impact of an imbalanced dataset on the performance of the binary classification model. The author considered ten performance matrices based on confusion matrix, which included Sensitivity, Specificity, Precision, Negative Predictive Value, Accuracy, F1 Score, Geometric Mean, Matthews Correlation Coefficient, Bookmaker Informedness, and Markedness.

Their results showed sensitivity and specificity were the best performing metrics with no bias due to imbalance in the dataset. These are considered as partial performance metrics since they only consider positive or negative classes. Geometric mean and Bookmaker Informedness also showed no bias towards imbalanced datasets due to them depending on specificity and sensitivity. However, they did solve the problem of partiality by considering both the classes. Both Geometric mean and Bookmaker Informedness had their drawbacks. They only consider the classification success rate and do not consider the error rates. The second group of performance matrices includes accuracy, the Matthews correlation coefficient, and markedness. These matrices showed medium biases towards an imbalanced dataset. All three consider both positive and negative classes, so they are considered two-dimensional performance metrics. Accuracy has shown to have the highest biasedness and also only considers the success rate. The rest of the two metrics showed lower bias than accuracy, and they also considered both the success and the error rates.

The last group had consisted of the precision, negative predictive value, and F1 score. This group showed a high bias. While they all are two-dimensional and consider both success and error rates, the authors recommended avoiding its use on imbalanced datasets due to high bias. The research showed Geometric mean and Bookmaker informedness to the best performing metrics for applications that focus just on the classification success rates. For those applications where classification error rates are also significant, Matthews correlation coefficient proved to be the best option.

2.6 Summary

The literature review provides evidence that the supply chain and logistics industry can benefit significantly by applying predictive analytics. Govindan et al. (2018) and Schoenherr and Speier-Pero (2015) suggest that supply chain and logistics industries show a growing interest in predictive analytics primarily due to its benefits in enabling better decision making, providing better risk and operational management, reducing operational costs, and improving forecasting capabilities. Literature also suggests that most predictive analytics in the supply chain industry were in value creation, forecasting, or operational management sub-domains. Alias et al. (2016) found evidence that predictive analytics techniques were applied mostly in the supply chain industry.

In contrast, the logistics and transportation industry relied mostly on predictive monitoring techniques such as complex event processing. This is mostly due to the long history of its application in the logistics and transportation industry instead of the other technologies. A closer look at the application of predictive monitoring techniques in the logistics and transport industry showed that most research areas were mostly focused on road transportation or general transportation. This provided an opportunity to study predictive monitoring techniques in the air transport industry. Feldman et al. (n.d.) implemented a predictive monitoring technique called event processing to develop an event processing network to monitor and predict violations in volume and weight of air cargo.

The remaining literature reviewed was focused on the current application of predictive analytics and machine learning techniques. Senderovich et al. (2014) applied queue mining theory and the snapshot principle to predict the time to serve customers in a bank. Decision trees, and more specifically, random forests, have found great success in their application in different industries. Ahmad et al. (2017) compared machine learning techniques, random forest, and artificial neural network to predict energy consumption and found a random forest-based approach to run much faster than a neural network-based approach while having almost the same prediction performance. Datla (2015) researched to compare the performance of two of the most popular classification machine learning algorithms, decision trees, and random forest, and found the random forest to perform better on more massive datasets. Husband and Roberts (2017) applied the survival theory and developed a random forest-based survival predictor to predict the time to fill an open job position.

Another tree-based algorithm, gradient boosting, has shown the great potential of its application for predictive analytics. Gal et al. (2017) combine queueing theory and different tree-based techniques to predict travel time and found that gradient boosting-based predicting results in better predicting travel time than all other tree-based predictors. Gradient boosting trees have shown better performance in classification applications than other machine learning techniques. Georganos et al. (2018) compared gradient boosting, random forest, and support vector machine to build a prediction model for urban land classification and found gradient booting to perform better than the other two.

CHAPTER 3. METHODOLOGY

A significant issue faced by companies that rely upon heavily on-air freight is the delay in the timely arrival of their shipment—any delay in shipment results in an interruption in their supply chains, which incurs an extensive monetary loss. The shipping process's complex structure and the high level of uncertainties pose a significant challenge in predicting any delays. This research proposes a prediction approach to identify shipments that are at risk of being delayed. Our approach is based on real air freight shipping data from a freight forwarding company. This section introduces the prediction techniques, data samples, and detailed process of implementing those prediction techniques on the dataset.

3.1 Airfreight Dataset

To standardize the shipment process monitoring, the International Air Transport Association (IATA) implements a new quality management system, Cargo 2000, for the air transport industry. The goal was to introduce a system where all stakeholders involved in the transportation process can share shipment information. This includes the planning, re-planning, and completion of shipment events. Each shipment is structured into incoming and outgoing legs. A shipment is planned into two sections, incoming leg and outgoing leg. Each shipment can consist of multiple incoming legs ranging from one to three independent of each other, and so all incoming legs are transported in parallel. Each leg of a shipment might pass through multiple segments that can range from one to four. A segment replicates a layover. Each shipment gets a plan with predefined milestones of planned completion time for that service. The transport legs consist of four physical transport services, as shown in the table below:

Table 1. Physical transport services

RCS	Service when the shipment is checked in at the departure airport
DEP	Service when the shipment is on-board, and the aircraft has departed
RCF	Service when the shipment is received at the arrival airport
DLV	Service when the shipment is delivered at the destination airport

Similar to the incoming legs, all the shipments have an outgoing leg. All the incoming legs arrive at the outgoing leg's departure airport, combined and sent out as one shipment in this leg. Unlike the incoming legs, each shipment has just one outgoing leg. The outgoing leg also consists of multiple segments ranging from one to three and consists of the four physical transport services. The delivery at the outgoing leg is the final delivery of the shipment. Table 2 shows the complete list of features of leg 1. Note that the unique id of the airport location has been masked for confidentiality. Each leg has a unique leg ID (i1_legid), which remains the same throughout the shipment transport in leg 1. Each leg consists of all four transport services, and each service has a planned and actual duration. For example, i1_rcs_p and i1_rcs_e are the planned and actual timings for the RCS service, respectively.

Similarly, all other transport services (DEP, RCF, and DLV) have actual and planned service timings. Since each leg consists of up to three segments, three layovers, departure, and arrival information (DEP and RCF) are also available for those segments. i1_dlv_p and i1_dlv_e represent the planned and actual delivery time of the shipment's first leg at the outgoing leg.

Table 2. Incoming Leg 1 Feature Overview

i1_legid	unique id across all transport legs of incoming transport leg 1
i1_rcs_p	planned duration (minutes) of incoming transport leg 1 (RCS: Freight Check-in)
i1_rcs_e	effective (i.e., actual) duration (minutes) of incoming transport leg 1 (RCS: Freight Check in)
i1_dep_1_p	planned duration (minutes) of incoming transport leg 1 (DEP: Departure Segment 1)
i1_dep_1_e	actual duration (minutes) of incoming transport leg 1 (DEP: Departure Segment 1)
i1_dep_1_place	unique id for the airport (original IATA codes have been masked due to confidentiality reasons) of incoming transport leg 1 (DEP: Departure Segment 1)
i1_rcf_1_p	planned duration (minutes) of incoming transport leg 1 (RCF: Arrival Segment 1)
i1_rcf_1_e	actual duration (minutes) of incoming transport leg 1 (RCF: Arrival Segment 1)
i1_rcf_1_place	unique id for the airport (original IATA codes have been masked due to confidentiality reasons) of incoming transport leg 1 (RCF: Arrival Segment 1)
i1_dep_2_p	planned duration (minutes) of incoming transport leg 1 (DEP: Departure Segment 2)
i1_dep_2_e	actual duration (minutes) of incoming transport leg 1 (DEP: Departure Segment 2)
i1_dep_2_place	unique id for the airport (original IATA codes have been masked due to confidentiality reasons) of incoming transport leg 1 (DEP: Departure Segment 2)
i1_rcf_2_p	planned duration (minutes) of incoming transport leg 1 (RCF: Arrival Segment 2)
i1_rcf_2_e	actual duration (minutes) of incoming transport leg 1 (RCF: Arrival Segment 2)
i1_rcf_2_place	unique id for the airport (original IATA codes have been masked due to confidentiality reasons) of incoming transport leg 1 (RCF: Arrival Segment 2)
i1_dep_3_p	planned duration (minutes) of incoming transport leg 1 (DEP: Departure Segment 3)
i1_dep_3_e	actual duration (minutes) of incoming transport leg 1 (DEP: Departure Segment 3)
i1_dep_3_place	unique id for the airport (original IATA codes have been masked due to confidentiality reasons) of incoming transport leg 1 (DEP: Departure Segment 3)
i1_rcf_3_p	planned duration (minutes) of incoming transport leg 1 (RCF: Arrival Segment 3)
i1_rcf_3_e	actual duration (minutes) of incoming transport leg 1 (RCF: Arrival Segment 3)
i1_rcf_3_place	unique id for the airport (original IATA codes have been masked due to confidentiality reasons) of incoming transport leg 1 (RCF: Arrival Segment 3)
i1_dlv_p	planned duration (minutes) of incoming transport leg 1 (DLV: Freight Delivery)
i1_dlv_e	effective (i.e., actual) duration (minutes) of incoming transport leg 1 (DLV: Freight Delivery)
i1_hops	number of segments (hops) in the transport leg of incoming transport leg 1

Since each shipment can consist of up to three incoming legs, all three legs have the same service information as leg 1 shown in Table 2. All the legs are transported in parallel, so the shipments from each of these legs are combined at the departure airport of the outgoing leg and shipped out in this last transport leg. The DLV service of the outgoing leg (o_dlv_p and o_dlv_e) provides information on whether the shipment was delivered on time or whether it was delayed. Similar to the incoming legs, the outgoing leg might have more than one segment. Since not all the shipment processes are the same, not all instances have the same number of legs or segments. Due to this, there is a lot of missing information. The sub-section below provided a more detailed view of the dataset and introduced the techniques used to cater to missing data and inconsistent shipment data.

3.2 EDA and Feature Engineering

The dataset consists of five months of shipment data from an air freight forwarding company. There are 3,942 shipment instances, 1,318 instances with just a single incoming leg, 1,258 instances with two incoming legs, and 1,366 instances with three incoming legs. In total, there are 7,932 incoming legs. Since each instance has just one outgoing leg, thus there are only 3,942 outgoing legs. Each instance consists of the planned execution time and the actual execution time of all four transport services.

Table 3. Breakdown of instances based on the number of legs

Total Instances	3,942
Instances with 1 incoming leg	1,318
Instances with 2 incoming leg	1,258
Instances with 3 incoming leg	1,366
Total Incoming legs	7,932
Total Outgoing legs	3,942

The dataset has a total of 98 variables. These include the planned and actual execution of four transport services, leg ID, location, number of hops in each leg, and legs in each instance. Some of these variables do not give us much information about the delay, for example, leg ID or location, so those variables were dropped from the dataset used in the implementation of random forest and XGboost. These included leg ID, departure place, arrival place, leg hops, and legs. Another major problem with the dataset was missing values. Since not all instances had the same number of legs, instances with one or two legs had missing values for the third leg variables. Similarly, legs with one or two segments have no values for the second and third segments, respectively. Table 4 below shows this issue for an instance that had just one leg.

Table 4. Sample of a shipment's Leg 1 data showing missing values

nr	i1_legid	i1_rcs_p	i1_rcs_e	i1_dep_1_p	i1_dep_1_e	i1_dep_1_place	i1_rcf_1_p	i1_rcf_1_e	i1_rcf_1_place	i1_dep_2_p	i1_dep_2_e	i1_dep_2_place	i1_rcf_2_p	i1_rcf_2_e	i1_rcf_2_place	i1_dep_3_p	i1_dep_3_e	i1_dep_3_place	i1_rcf_3_p	i1_rcf_3_e	i1_rcf_3_place	i1_dlv_p	i1_dlv_e
0	5182	199	218	210	215	609	935	736	256	?	?	?	?	?	?	?	?	?	?	?	?	840	1539

As shown in Table 4, the shipment's first leg had just one segment (layover). The remaining segment information for this leg was left blank. To overcome inconsistencies in a leg's segments, all the segment's transport service planned and actual execution times were combined. This was done because the segments occur in the series. For example, the instance shown in Table 4 has just one segment, so DEP and RCF service for the second and third segments are missing. A new service variable (i1_dep_p) was created that combined all the planned departure times for the leg as shown by the equation below:

$$\text{i1\_dep\_p} = \text{i1\_dep\_1\_p} + \text{i1\_dep\_2\_p} + \text{i1\_dep\_3\_p} \qquad (1)$$

$$\text{i1\_dep\_e} = \text{i1\_dep\_1\_e} + \text{i1\_dep\_2\_e} + \text{i1\_dep\_3\_e} \qquad (2)$$

Removing all the unimportant variables and combining the segment services allowed to reduce the first leg variables from twenty-four to eight. The Table 5 below shows the reduced set of variables for leg one information shown in Table 4, thus, removing the segment missing data. This was done for all three incoming legs and the outgoing leg to remove any missing segment data—this data cleansing approach allowed to reduce the total number of variables from ninety-eight to thirty-three.

Table 5. Reduced set of variables for incoming leg one

I1_rcs_p	**i1_rcs_e**	**i1_dep_p**	**i1_dep_e**	**i1_rcf_p**	**i1_rcf_e**	**i1_dlv_p**	**i1_dlv_e**
199	218	210	215	935	736	840	1539

Another problem with the dataset was that not all instances had the same number of legs. As seen from Table 3, 1,318 instances have just one incoming leg, 1,258 have two, and 1,366 have three. The dataset was divided into three datasets based on the number of incoming legs to solve missing leg data. These three datasets were used to create three separate predictors. The section below discusses the approaches implemented in developing these predictors. The table below shows the number of variables in each of these derived datasets. It can be seen that the total number of variables is different for all the legs, with Leg 1 predictor having half as many variables as that of Leg 3 for prediction.

Table 6. Number of variables in each leg

	Number of Incoming Leg's Variables	**Number of Outgoing Leg's Variables**	**Total Number of Variables**
Leg 1	8	8	16
Leg 2	16	8	24
Leg 3	24	8	32

The biggest problem with airfreight shipment is the delay in delivering the shipment. This could happen due to multiple reasons. For instance, a delay could occur due to a delay in delivering the shipment at the departure airport (RCS) or delayed departure of flight (DEP and RCF) due to weather conditions. While these sources of delays cannot be identified beforehand, the delay in the final delivery (dlv_final) can be predicted based on delays in a shipment's physical transport service status.

The original dataset did not contain any variable for the final delivery, so a new variable was derived using the planned and actual delivery variables of the outgoing leg (o_dlv_p and o_dlv_e). The new variable is called the final delivery variable (dlv_final), was used to represent the delay in final delivery. It is defined as the difference between the planned and actual delivery time and is represented by equation (3) below:

$$\text{dlv\_final} = \text{o\_dlv\_p} - \text{o\_dlv\_e} \qquad (3)$$

To better understand this new variable, we took a more in-depth look into this variable. Table 7 and 8 below provides a better understanding of the final delivery variables and delays corresponding to each leg. Instances with dlv_final > 0 were considered as delayed. Approximately one-third, 1,167, of the instances had some sort of delay. All the legs had an almost similar number of instances with delayed delivery. Table 8 shows that the minimum value for dlv_final was -6.6 days, which means that the shipment arrived almost six days early. A maximum of 387 days delay in shipment delivery was observed. A median of -1.5 days means that most of the shipments were delivered before their planned delivery time. This means that the planned delivery times are overestimated when booking to ensure that the shipments are delivered on or before that time.

Table 7. Delays in shipment delivery

Total Number of Instances	**Instances with delayed final delivery**
3942	1,167

Table 8. Summary of dlv_final variable

Minimum	**Maximum**	**Median**
-6.6 days	387 days	-1.5 days

Quantile			
25%	**50%**	**75%**	**95%**
-2.24 days	-1.5 days	0.33 days	21 days

Looking at the summary of the final delay for each leg in Table 9, we can see that all the legs had similarly delayed shipments. The rate of delays on-time/delayed instances was almost the same: 41.4% for Leg 1, 41.3% for Leg 2, and 43.3% for Leg 3. Finally, a new variable is introduced that serves as the class label for each shipment. This new variable was labeled as "Violation" and derived from the dlv_final variable. The Violation variable had two factors: Yes and No. Instances in which delays were observed that is dlv_final > 0 were labeled Yes, while all those for which dlv_final < 0 were labeled No. This variable served as the dependent variable in our prediction approaches. Succeeding sub-sections will detail the dependent and independent variables and their implementation in our tree-based predictors.

Table 9. Summary of delays in final delivery for each leg

	LEG 1	LEG 2	LEG 3
Total instances	1,318	1,258	1,366
On-time	932	890	953
Delayed	386	368	413
Minimum delay	109 days	387 days	364 days
Maximum delay	- 6.5 days	- 6.6 days	- 6.4 days

3.3 Decision Trees

Tree-based approaches have been widely used to solve classification problems. To predict whether a shipment will be delayed or on-time, decision tree-based classification predictors have been designed. Tree-based classification techniques, such as Decision Tree, possesses the ability to model the complicated relationship between variables without having strong model assumptions (Zhao & Zhang, 2008). They can classify features and limits that best splits data into distinct groups. This splitting ability occurs recursively until all the data divide into similar groups. Decision Tree is a handy classification algorithm because of its ability to detect important features immediately. One most significant advantage provided by Decision Trees is the readability of its classification rules. Decision trees do not require long training methods so, it saves much time when modeling large datasets. Some key advantages of decision trees are listed below:

1. Decision trees are easy to interpret and visualize
2. Decision trees are easy to reproduce since they are transformed easily into production rules;
3. Decision trees can handle both categorical and numerical data, but the output attribute have to be categorical
4. Decision trees are extremely fast and possess extensive data handling capabilities.

Decision trees come with disadvantages. Multiple output attributes are not allowed in decision trees. Deep trees are prone to the problem of overfitting. Decision trees select the optimum option at each node without considering the global optimum, which does not ensure the optimum choice when the tree reaches the leaf node.

To give an example of how a decision tree can be grown on the dataset, suppose that we divide the dataset into delayed and not delayed. These delayed and not delayed labels are represented in the dataset by the Violation variable. This variable contains two factors: Yes and No, representing delayed and not delayed, respectively. The dataset is divided into two parts, training and testing datasets. The training dataset will be used to train the decision tree and the testing dataset to test the final classifier.

Each instance, or shipment, in this case, has a set of variables called features that characterize the instance between delayed and not delayed. For each instance, one variable is selected and split into two parts, depending on the variable's value. A simplified schematic of a decision tree is shown in Figure 1. Here, the first variable picked is the outgoing leg's planned delivery time. A splitting value for it greater than 982 is selected (o_dlv_p > 982). It gives the best separation to split the dataset into delayed and not delayed. This is repeated for all the variables and for both the branches until a leaf node is reached, that is delayed or not delayed. This is repeated for all the instances in the training dataset to train the model, and then the test dataset is used on this model to gauge its performance. This is a simplified example of applying the decision tree model to our dataset. The models implemented here are much more complex decision tree-based models, random forest, and XGboost. The sub-sections below explain how these models were implemented on this dataset to classify whether the shipments were on time or delayed.

3.4 Random Forest-Based Prediction Method

The ensemble-based technique is based on the collection of results from multiple trees instead of a single tree. Bagging and boosting are the two most well-known techniques used in ensemble-based trees. In the boosting technique, the extra weight given to the predicted features incorrectly and voting based on weight produces the final prediction. In the bagging technique, each tree is developed using a bootstrap sample from the dataset, and each tree is independent of the other tree. The final prediction is based on the majority votes from each tree.

Breiman (2001) proposed a new technique based on the bagging ensemble technique called random forest. In Random Forest, during learning, the tree nodes are split by creating a random feature subset. Each tree is grown by a randomly drawn bootstrap sample, sampling with replacement. Random forests also have another source of randomness in them. At each split in a tree, the features are also selected at random. In a random forest, the prediction from all trees is combined to give the final prediction. Each tree votes for a class prediction, and the class with maximum votes becomes the final answer. Thus, random forest work is an unbiased predictor.

The random forest also has additional methods of checking prediction. Like other models, random forest uses a separate test dataset for testing model performance, but it also comes with an out-of-bag prediction feature. When trees, built using bootstrapping during the learning phase, some training dataset instances do not include building trees. These excluded observations are called out-of-bag samples and used for computing the performance of the built model. According to Breiman (2001), the random forest performed better than a single classification tree and other machine learning techniques such as support vector machine or neural networks.

A simple tree grown using the dataset, shown in Figure 1, was visualized to show how an instance is divided at each node. The leaf nodes provide the final class prediction. This is a simplified version of a tree; the actual random forest tree has a much more complicated structure. A random forest grows multiples such trees and selects a class prediction based on majority voting. Also, it picks a random set of features to grow each tree.

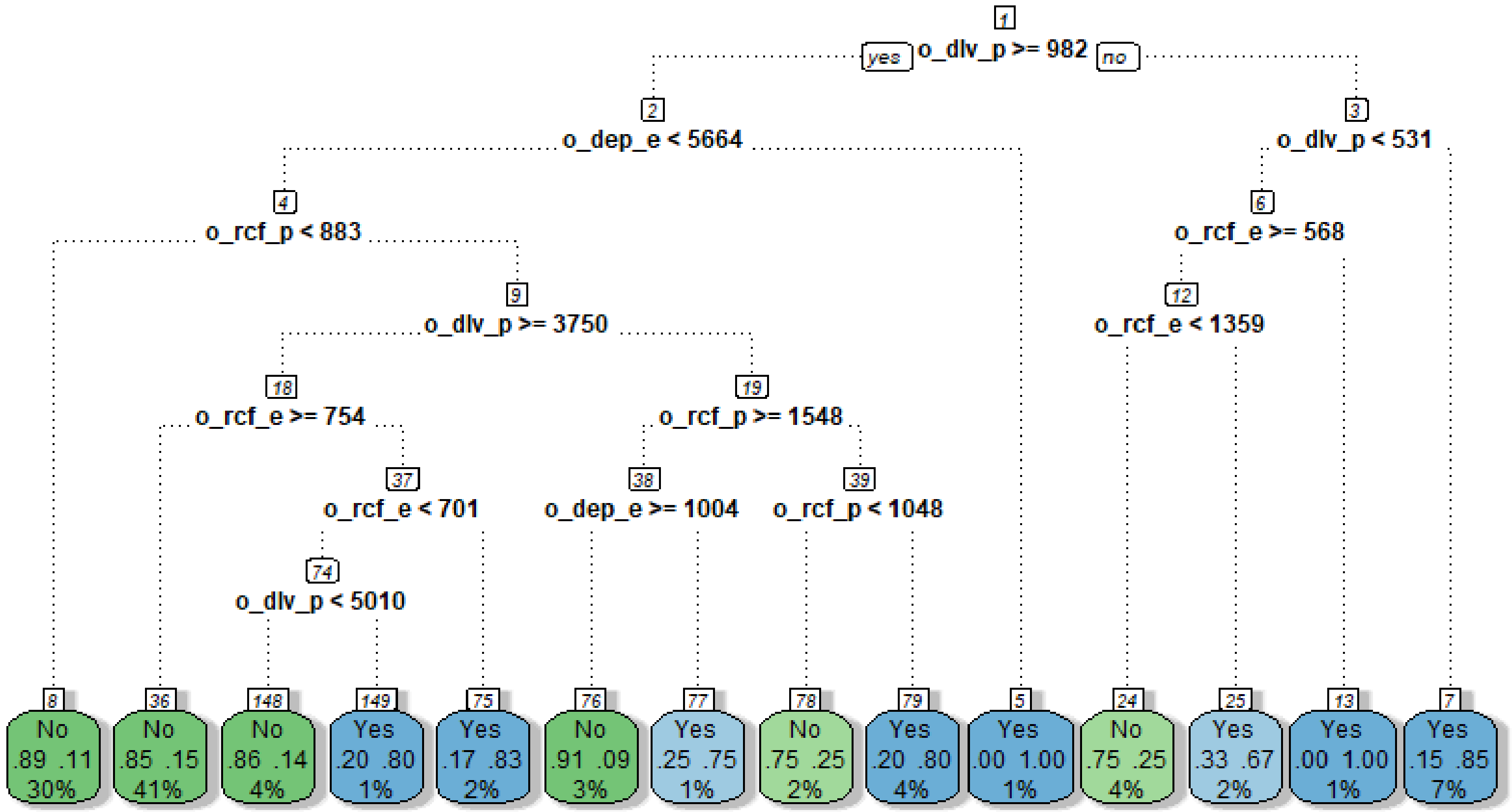

Figure 1. Tree Simple Tree Representation of Dataset

The critical advantage of single classification tree methods, such as decision trees, is interpreting and visualizing them quickly. However, interpreting and visualizing ensemble-based techniques like a random forest is not easy because random features are selected from the feature set to grow each tree. Each feature may appear at different positions. This gives random forest an advantage in that different features are selected each time a tree is grown and can have a complex effect on the tree prediction. Thus, random forest computes the variable's importance to assess their importance in generating all trees. This importance is given in the form of Mean Decreasing Accuracy, which permutates out-of-box samples to calculate each feature's importance. Variable importance is measured by finding the difference between out-of-box prediction error before permuting and out-of-box prediction error after permuting averaged over all trees. The figure below shows the variable importance after implementing random forest on our dataset

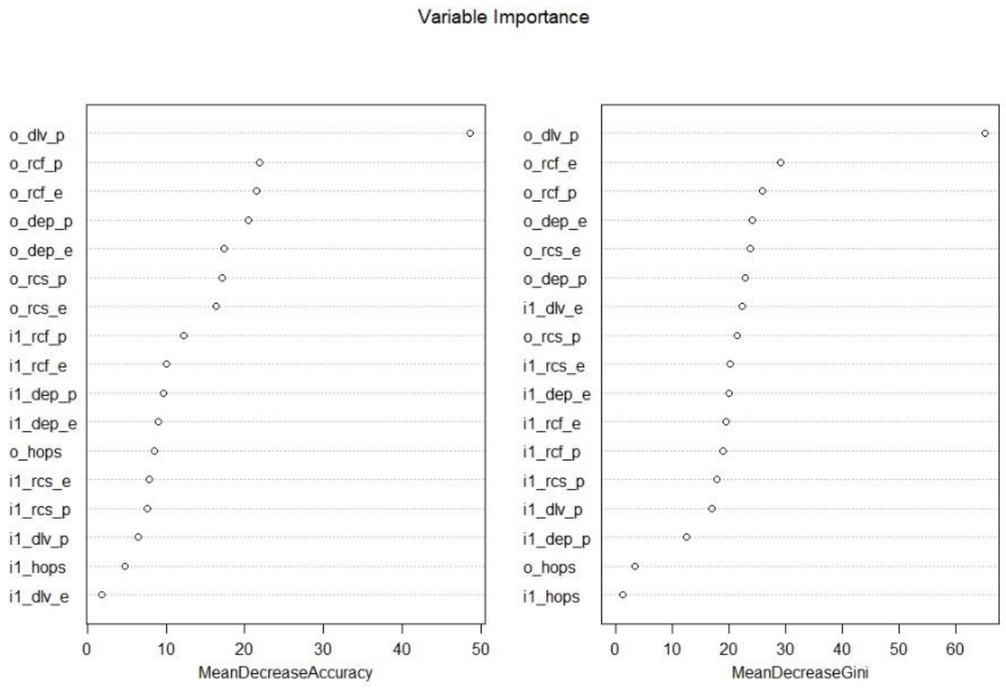

Figure 2. Variable Importance of Leg 1 variables represented in terms of Mean Decrease Accuracy and Mean Decrease Gini

All exploratory data analysis, feature engineering, and modeling were performed in R. R's randomForest package was used to generate the random forest-based predictors. As mentioned in the preceding sub-section, three separate predictors were developed, one each leg. Leg 1 predictor had seventeen variables in its feature set, nine variables of the incoming and eight of that from the outgoing legs. The outgoing leg's actual delivery time (o_dlv_e) was dropped from the feature set because the dependent variable, violation, was derived. The planned delivery time was kept because all the planned transport services are known before starting the shipping process. Two key parameters, the number of variables selected at each split (mtree) and the number of trees grown (ntree), affect a random forest model's stability. By default, the ntree is set to 500, and mtree is taken as the square root of the total number of variables in the feature set, which in our case is equal to four.

A 70/30 split was used to split the dataset into training and test data. 70% of the dataset was selected randomly for the training dataset, and the remaining 30% were reserved for the testing dataset. The training dataset was used to model a random forest-based predictor using the default parameters. Figure 3 below shows the change in each class's error rate and out of box error rate. It was observed that the predictor showed its optimum performance ntree = 600 and mtree = 9. The predictor showed a classification accuracy of 75% on the test dataset and 77% out-of-box accuracy at these parameter settings.

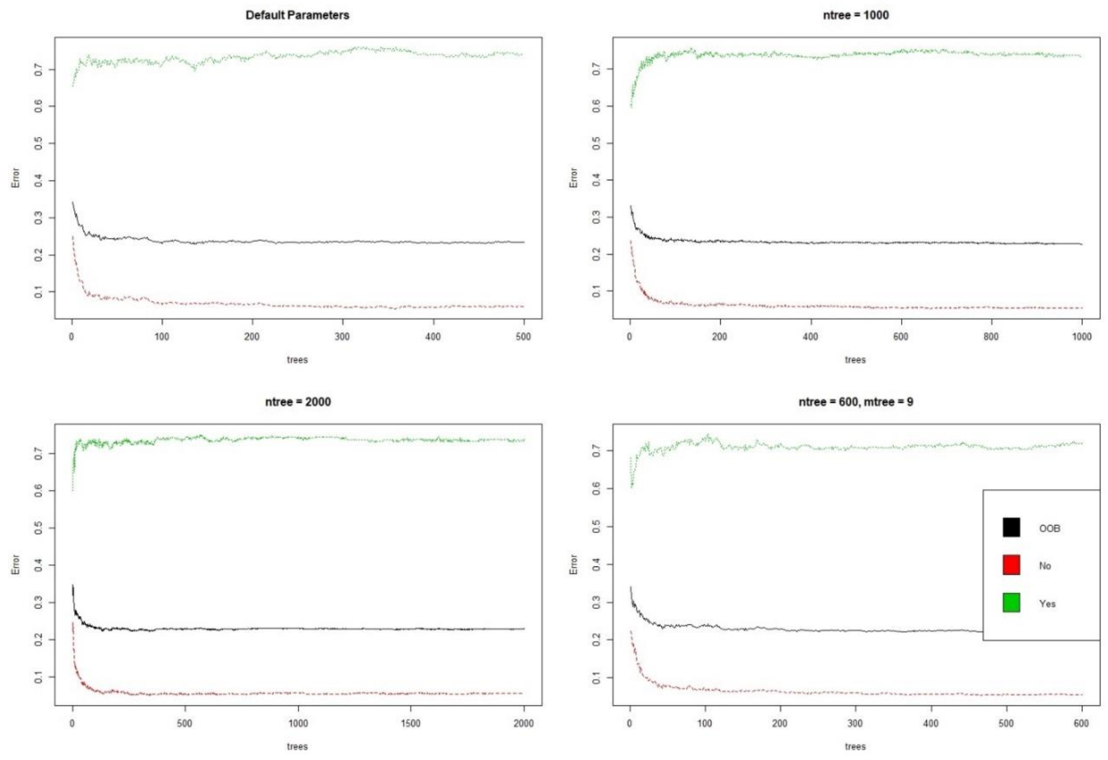

Figure 3. Prediction model error-rate vs number of trees plot for Leg 1 predictor (default mtree = 4, ntree = 500)

Similarly, predictors for Leg 2 and Leg 3 were implemented. For both of these predictors, mtree was set to 9, and ntree was set to 600. The leg 2 based predictor produced 80% accuracy on the test dataset and 77% on the out-of-box data. Leg 3 showed 76% accuracy on the test dataset and 77% on the out-of-box dataset. A detailed comparison of accuracy and other performance metrics are presented in the next section. The preliminary results provide evidence that random forest-based predictor approaches can predict delays in shipment delivery.

3.5 Gradient Boosting-Based Prediction Method

Another well-known ensemble-based technique is called boosting. It is considered one of the most powerful learning techniques introduced in recent times (Roe et al., 2005). In boosting algorithms, incorrectly predicted features or weak classifiers are given extra weight, and voting is based on weight to produce the final prediction. In boosting algorithm, a tree model is built from the training data. Then a second model is built that tries to correct the classification errors made by the first model. This is repeated until the models correctly classify the predicting dataset.

AdaBoost was one of the first boosting ensemble method proposed by Freund and Schapire (1997) to boost decision trees' performance. Boosting refers to a general method of improving the performance of any algorithm. In gradient boosting, this technique uses weak classifiers to reduce error rather than randomly guessing the outcome. A weak classifier is developed using weighted samples from the training dataset, initially having the same weights. The misclassification rate is then evaluated, which is then used to calculate the new weights. The next tree is grown on this new weighted data. These weak models are then added until a given number of weak models are generated. Finally, the weighted average of these weak models is calculated for each instance in the test dataset to give the final prediction.

Chen and Guestrin (2016) introduced XGBoost, which is one of the fastest and scalable implementations of gradient tree boosting. This sub-section implements the XGBoost library in R on the cargo dataset to predict a delay in shipment delivery. Similar to the random forest implementation, three separate predictors were developed for instances with one, two, and three incoming legs. The same 70/30 split for the training and testing dataset was used. The random forest predictor, 70/30 split for the training and testing dataset, was used here.

Unlike random forest, which has minimal tuning parameters, XGBoost provides many parameters for tuning the model. These parameters are divided into three parts: general parameters, booster parameters, and learning parameters. The general parameters include the type of booster (tree, linear, or dart), the verbosity of printing message, and the number of threads. "gbtree" booster was used to model our predictor running on a single thread. Table 10 below represents a list of general parameters of XGBoost that were used in building our model.

Table 10. XGBoost General Parameters

booster	Type of booster ("*gbtree*", "*gblinear*" and "*dart*")
verbosity	Default = 1 (warning)
nthread	1

The second type of parameter controls the boosting method used. Each type of boosting method has its boosting parameters. Table 11 details the tree boosting parameters that were used in building the models. Gamma represents the minimum loss reduction that is necessary for further splitting at the leaf node. A higher gamma value means that the model will be more conservative. Eta, or learning rate, represents the step size shrinkage used to prevent overfitting. Max depth represents the maximum depth of the grown tree. Subsample allows random sampling of data before growing a tree to prevent overfitting. Lambda and alpha are L2 and L1 regularization terms, respectively. The default value for lambda is one, while that of alpha is zero. Increasing the values for both of these parameters makes the model more conservative.

Table 11. Tree Booster Parameters for XGBoost

gamma (min_split_loss)	Minimum loss reduction required to make a further partition on a leaf node
max_depth	Maximum depth of the tree
Subsample	Subsample ratio of the training instances
eta (learning_rate)	Step size shrinkage used to shrink the feature weights
lambda	L2 regularization on weights
alpha	L1 regularization on weights

Three predictors were built, one for each one, two, and three incoming leg datasets using 70% of data to train the models and the remaining 30% to test the model by building a confusion matrix. Predictor built for one incoming leg shipment predicted delayed and on-time shipments with 75% accuracy. The predictor modeled to predict delays in shipments with two incoming legs showed a prediction accuracy of 80%. In comparison, that designed to predict delays in shipments with three incoming legs showed a prediction accuracy of 75%. All the predictors were modeled with the same tuning parameters for XGBoost. The preliminary results are compared in terms of accuracy, but these results are not conclusive. The subsection below introduces the different performance models selected and the rationale behind their selection to compare the prediction models. The next section will discuss the results obtained from all the predictions and compare the models to select the best performing one. The effect of parameters used in tuning the XGBoost model and their effect on different performance metrics have also been discussed.

3.6 Early Prediction Technique

The second research question for this thesis was to analyze the effect of predicting the shipments early on the performance of both the models. To address this, an early prediction approach was designed. Subsection 3.6 and 3.7 have detailed the complex structure of a shipping process in terms of legs and segments, and also explained the transport service information. This information was leveraged to develop an early prediction technique. To simulate a shipment in transit, three early prediction categories were designed. These categroies were used to show the start, haly way and near completion of a shipping sprocess. Since all the incoming legs were executed in parallel, the feature set of the dataset was divided into two parts: incoming and outgoing legs.

The start category represents a shipment process at the beginning of its journey. The actual transport service information may not be available at the start of a shipping process, so these features were dropped from the dataset for both the incoming and the outgoing legs. Similarly, when a shipment is half way through its journey, only half of actual transport service information may be available and when shipment reaches near completion, most of its actual service information is available. The dataset based on these early prediction categories were then used in training the random forest and gradient boosting-based predictors. Section 4.4 presents the results obtained from early prediction technique and discusses the effect of early prediction on the performance of the prediction models.

3.7 Performance Metrics

A critical factor in selecting the best classification model for a problem is assessing their performance. Different methods are used to measure the performance of classification algorithms. Most classification models split the dataset into training data, build the classification model, and test data to test the prediction model's performance. Our classification process was divided into two parts: the training phase and the testing phase. 70/30 split method was used with 70 percent of the dataset used for building the model in the training phase, while the remaining 30 percent for testing it in the testing phase. Classification outcomes can be either discrete or continuous, depending on the prediction model. The discrete classification can be binary, with two outcomes, or multi-class, having more than two outcomes. Here the classification output was discrete and binary in terms of whether there has been a delay in shipment delivery or not.

The output was then represented in the form of a 2 X 2 confusion matrix. The confusion matrix consists of four elements: True Positive, True Negative, False-Positive, False-Negative. The table below shows the possible outcomes as represented in the form of a confusion matrix. The actual values represent the number of correctly identified samples, while the false are samples incorrectly identified. This confusion matrix results were used to evaluate the performance in terms of scalar matrices like accuracy, recall, specificity, sensitivity, etc.

Table 12. Confusion Matrix

		Actual Class	
		Delay = Yes	**Delay = No**
Predicted Class	**Delay = Yes**	True Positive (TP)	False-Positive (FP)
	Delay = No	False-Negative (FN)	True Negative (TN)

Selecting the right performance metrics is crucial in comparing multiple classification models. Some of these matrices are sensitive to the issue of an imbalanced dataset. The problem of an Imbalanced dataset occurs when the number of one class label outnumber the others. Sensitivity, Specificity, Geometric Mean, and Bookmaker Informedness have shown to be unbiased towards imbalanced data. However, they only consider the success rate and do not consider the error rate (Luque et al., 2019), while Accuracy, Matthews correlation coefficient, and markedness have shown little bias towards imbalanced data. Matthews correlation coefficient and markedness consider both the success and error rates. Also, metrics such as False-Positive and False-Negative Rates, Likelihood Ratio, Youden's Index, and Discriminant Power are not affected by imbalance in data (Tharwat, 2020).

Table 13. Summary of class distribution of airfreight dataset

	DELAY		DELAY (%)	
	NO	YES	NO	YES
One incoming leg	932	386	70.71	29.29
Two incoming legs	890	368	70.75	29.25
Three incoming legs	953	413	69.77	30.23

A summary of the class distribution of the cargo dataset is shown in table 5. Approximately 70% of the data belongs to the class label. No, the represents shipments that are not delayed while the remaining 30% belong to the Yes class representing delayed shipments. Since the dataset used here has imbalanced class data, the number of instances belonging to the No class outnumber those in the Yes class, a mix of performance matrices that are not sensitive to imbalanced class data were selected. These include Accuracy, Geometric Mean, Sensitivity, Specificity, False-Positive, and False-Negative Rates. Accuracy is the ratio of correctly identifies classes (True Positive and True Negative) over the total number of instances. It is one of the most widely used performance measures used for classification. Accuracy is given by equation (1) below. Sensitivity is another performance metric used here and is the ratio of True Positive over the total number of Positive instances. It is also referred to as true-positive rate.

On the other hand, specificity, also known as the true-negative rate, is the true-negative ratio over the total number of negative instances. Equation (2) and (3) represents sensitivity and specificity, respectively. Geometric mean measures the balance between the classification performance of both classes and uses both sensitivity and specificity. Geometric Mean is represented by equation (4). The false-positive rate is the ratio of False-Negative over the total number of Negative instances.

Similarly, the false-negative rate is the ratio of False-Positive over the total number of Positive instances. They represent the portion of instances wrongly classified as negative or positive. Equations (5) and (6) represent False-Positive and False-Negative rates, respectively.

$$ACC = (TP+TN)/(TP+FP+TN+FN) \quad (1)$$

$$Sensitivity = TP/(TP+FN) \quad (2)$$

$$Specifity = TN/(TN+FP) \quad (3)$$

$$Geometric\ Mean = \sqrt{(TP/(TP+FN)*TN/(TN+FP))} \quad (4)$$

$$FPR = FP/(FP+TN) \quad (5)$$

$$FNR = FN/(FN+TP) \quad (6)$$

CHAPTER 4. RESULT ANALYSIS

The previous section introduced the air cargo dataset and the two tree-based predictive modeling methods that were employed to achieve the research goals. It also presented details of the feature engineering approaches applied to the dataset and the performance metrics used to compare the prediction models. Two ensemble tree-based classification models were implemented to predict a delay in final shipment delivery: random forest, a bagging ensemble model, and gradient boosting, a boosting ensemble model. Three predictors, each for one, two, and three incoming legs, were modeled for each prediction technique. The dataset was divided into a 70/30 split, and 70% of the dataset was used in training the models. At the same time, the remaining 30% were used for testing. The testing results were used to build a confusion matrix that was then used to derive the performance metrics' results. In this section, the empirical results that were obtained from two prediction techniques are presented in detail. Furthermore, a comparison is presented based on the results of the two models.

4.1 Results from Random Forest

Before training the models, the dataset was first divided into three sections to separate instances with one, two, and three incoming legs. These three new datasets went through a feature engineering process to make them ready for modeling. The table below shows the total number of instances and the portions of these datasets that were used for training and testing.

Table 14. Number of instances in training and testing dataset for each instance

Number of Incoming Legs	One	Two	Three
Total instances	1318	1258	1366
Training data	922	880	956
Testing data	396	378	410

During the random forest-based predictors training phase, two tuning parameters were set mtry = 9 and ntry = 600. The testing data was used to validate the model, and the confusion matrix for each predictor was generated. The confusion matrices are shown in Table 15 below. A confusion matrix is represented in terms of true-positive (TP), true-negative (TN), false-positive (FP), and false-negative (FN). Here no delay or "Delay = No" in shipment delivery is referred to as true-positive, and a delay or "Delay = Yes" is referred to as true-negative. All three of the predictors show quite similar true-positive and negative values. The confusion matrix itself does not give us much information about the models to understand the results better. The six-performance metrics introduced in Section 5 were derived from the confusion matrices.

Table 15. Confusion matrix representation of results from random forest-based predictors

One Incoming Leg			**Two Incoming Legs**			**Three Incoming Legs**		
Delay	**Yes**	**No**	**Delay**	**Yes**	**No**	**Delay**	**Yes**	**No**
Yes	46	22	**Yes**	44	20	**Yes**	48	22
No	77	251	**No**	55	259	**No**	72	267

Table 16 contains the results from the performance metrics for each of the predictors. Overall accuracy (ACC) gives us the ratio of correctly classified samples. The overall accuracy (ACC) of above 75% was observed for all three of the predictors. The predictor modeled to predict the data with two incoming legs performed better than the rest of the two predictors for all the performance metrics. The false-positive rate (FPR) and false-negative rate (FNR), specificity, and sensitivity are essential metrics in understanding the performance in terms of correct and incorrect identification of positive and negative classes. Two-incoming leg predictors showed the least false-positive (FPR) and false negative (FPN) rates and higher values of geometric mean, sensitivity, and specificity compared to the other two predictors, which means that it performed better than the rest in correctly predicting each class.

However, it was also observed that all the predictors had a false-positive rate under 8%, which means that the ratio of instances that belong to class no-delay (No) incorrectly identified as delayed (Yes) was below 8%. The specificity was over 92%; thus, the shipments with no-delays were correctly identified 92% of the time. In contrast, the ratio of shipments with a delay incorrectly identified as no-delay were wrongly identified almost 55-60% of the time. The sensitivity for all three predictors was between 37-44%, which means that almost 37-44% of shipments with a delay were correctly classified. These results show that all three of the predictors were biased towards correctly predicting a no-delay.

Table 16. Performance metrics' results for random forest predictors

One Incoming Leg		**Two Incoming Legs**		**Three Incoming Legs**	
ACC	75.0%	**ACC**	80.2%	**ACC**	76.8%
SPC	91.9%	**SPC**	92.8%	**SPC**	92.4%
SEN	37.4%	**SEN**	44.4%	**SEN**	39.7%
GM	58.6%	**GM**	64.2%	**GM**	60.5%
FNR	62.6%	**FNR**	55.6%	**FNR**	60.3%
FPR	8.1%	**FPR**	7.2%	**FPR**	7.6%

4.2 Results from Gradient Boosting

The XGBoost package in R was used to build predictors for the gradient boosting. The results obtained from these predictors are as following. The same dataset, shown in Table 14, was to train and test these models for the random forest. XGBoost allows a lot of more tuning parameters than random forest. "gbtree" booster parameter was used to model the gradient boosting-based predictors. A complete list of all the parameters used were presented in Table 10 and Table 11. The results obtained from test data validation are represented in the form of confusion metrics in Table 17 below. The true-positive and true-negative distributions are almost similar to those observed for the random forest-based predictors.

Table 17. Confusion matrix representation of results from XGBoost-based predictors

One Incoming Leg			Two Incoming Legs			Three Incoming Legs		
Delay	**Yes**	**No**	**Delay**	**Yes**	**No**	**Delay**	**Yes**	**No**
Yes	38	20	**Yes**	47	24	**Yes**	52	35
No	79	259	**No**	52	255	**No**	69	254

The results from the performance metrics for each of the predictors are presented in Table 18. Quite similar to the results from random forest-based predictors, the overall accuracy (ACC) was observed to be between 75-80% for all three predictors. Again, the two-incoming leg predictor outperformed the rest for all the performance metrics. The false-negative rates for two-incoming legs and three-incoming legs predictors were 52.5% and 57%, respectively. The highest false-negative rate was observed for the one-incoming leg, 67.5%, much higher than the rest of the two. However, the one-incoming leg predictor showed the least false-positive rate of 7% compared to the rest that showed 8% and 12%. The specificity was observed between 88-93%, while the sensitivity was between 32-47%. The results from the performance metrics showed similar trends to that of the random forest.

Table 18. Performance metrics' results for XGBoost-based predictors

One Incoming Leg		Two Incoming Legs		Three Incoming Legs	
ACC	75.0%	**ACC**	79.9%	**AC C**	74.6%
SPC	92.8%	**SPC**	91.4%	**SPC**	87.9%
SEN	32.5%	**SEN**	47.5%	**SEN**	43.0%
GM	54.9%	**GM**	65.9%	**GM**	61.5%
FNR	67.5%	**FNR**	52.5%	**FNR**	57.0%
FPR	7.2%	**FPR**	8.6%	**FPR**	12.1%

4.3 Comparison Between Results of Prediction Models

We have seen how the predictors individually performed in classifying delayed shipments from those that were not delayed. In this subsection, a comparison of both the predictors is presented for each of the different numbers of incoming legs. Table 19 contains the results of performance metrics for one-incoming leg predictors. The overall accuracy is the same for both predictors. Random forest-based predictor performed better in predicting delays as shown by higher sensitivity and lower false-negative rate than the XGBoost-based predictor. The XGBoost had higher specificity and a lower false-positive rate, which means it performed better in predicting the no-delay shipments.

Table 19. Comparison of one-incoming leg predictors

	One Incoming Leg Predictor					
	ACC	**SPC**	**SEN**	**GM**	**FNR**	**FPR**
Random Forest	75%	92%	37%	59%	63%	8%
XGBoost	75%	93%	32%	55%	68%	7%

Table 20 presents the comparison of both the predictors designed for shipments with two-incoming legs. Again, both the predictors had the same overall prediction accuracy, but the random forest had higher sensitivity and lower false-negative rate. XGBoost performed better in predicting delayed shipments, as evident by higher specificity and lower false-positive rate.

Comparing the predictors' prediction results for the three-incoming legs dataset, it was observed that the random forest-based predictor had better accuracy and specificity than the XGBoost-based predictor. However, the XGBoost-based predictor performed slightly better in classifying delayed shipments as represented by the higher specificity value. Both the predictors had the same geometric means.

Table 20. Comparison of two-incoming legs predictors

Two Incoming Legs Predictor

	ACC	SPC	SEN	GM	FNR	FPR
Random Forest	80%	93%	44%	64%	56%	7%
XGBoost	80%	91%	47%	66%	53%	9%

Table 21. Comparison of three-incoming legs predictors

Three Incoming Legs Predictor

	ACC	SPC	SEN	GM	FNR	FPR
Random Forest	77%	92%	40%	61%	60%	8%
XGBoost	75%	88%	43%	61%	57%	12%

4.4 Early Prediction of Delays

The results above provide compelling evidence that the proposed prediction models could predict with over 75% accuracy. RQ2: How accurately can a delay be predicted while in transit to address the second research question? Both the prediction models were trained with reduced features. The actual incoming and outgoing legs' checkpoint features were removed to mimic the limited checkpoint data available while in transit. The resulting datasets were then used in training random forest- and XGBoost-based predictors. To show the effect of early prediction on the predictors' performance, three early prediction categories were developed, each with different features. Since all the incoming legs are executed in parallel, the data was divided into incoming and outgoing legs. Table 22 represents the feature set in these three categories: near-completion, half-way, and the start of a shipment journey

Table 22. Features of incoming and outgoing legs are included in the feature set for each category.

	Incoming Legs	Outgoing Legs
Near-completion	All planned and actual checkpoints	All planned and actual checkpoints except actual delivery time (o_dlv_e)
Half-way	All planned and actual checkpoints	All planned checkpoints only
Start	All planned checkpoints only	All planned checkpoints only

It was observed that the prediction performance for the random forest-based predictor was not affected much by reducing the feature set. The change in accuracy remained with-in 2%. It was interesting to see that, for one- and two- incoming leg predictors, specificity increased by reducing the number of features. These results show that the random forest-based predictors can be used for early prediction without compromising on performance. The results of these predictors are presented in Table 23.

XGBoost-based predictors showed similar trends in performance results as the random forest-based predictors. The overall accuracy did not change much by reducing the number of features. One- and two-incoming leg predictors showed a decrease in Sensitivity and an increase in specificity. The overall performance did not show much change.

Table 23. Prediction performance for different categories for the random forest-based predictor

ONE-INCOMING LEG						
	ACC	**SPC**	**SEN**	**GM**	**FNR**	**FPR**
NEAR-COMPLETION	75%	92%	37%	59%	63%	8%
HALF-WAY	77.8%	91.3%	46.7%	65.3%	53.3%	8.7%
START	76.3%	91.8%	32.0%	54.2%	68.0%	8.2%
TWO-INCOMING LEGS						
	ACC	**SPC**	**SEN**	**GM**	**FNR**	**FPR**
NEAR-COMPLETION	80%	93%	44%	64%	56%	7%
HALF-WAY	77.2%	89.2%	48.2%	65.6%	51.8%	10.8%
START	79.1%	90.3%	51.8%	68.4%	48.2%	9.7%
THREE-INCOMING LEGS						
	ACC	**SPC**	**SEN**	**GM**	**FNR**	**FPR**
NEAR-COMPLETION	77%	92%	40%	61%	60%	8%
HALF-WAY	76.3%	95.7%	34.6%	57.6%	65.4%	4.3%
START	77.3%	95.4%	38.5%	60.6%	61.5%	4.6%

Table 24. Prediction performance for different categories for the XGBoost-based predictor

ONE-INCOMING LEG

	ACC	**SPC**	**SEN**	**GM**	**FNR**	**FPR**
NEAR-COMPLETION	75%	93%	32%	55%	68%	7%
HALF-WAY	76.5%	90.2%	45.0%	63.7%	55.0%	9.8%
START	74.2%	89.1%	32.0%	53.4%	68.0%	10.9%

TWO-INCOMING LEGS

	ACC	**SPC**	**SEN**	**GM**	**FNR**	**FPR**
NEAR-COMPLETION	80%	91%	47%	66%	53%	9%
HALF-WAY	77.2%	88.1%	50.9%	67.0%	49.1%	11.9%
START	77.0%	87.7%	50.9%	66.8%	49.1%	12.3%

THREE-INCOMING LEGS

	ACC	**SPC**	**SEN**	**GM**	**FNR**	**FPR**
NEAR-COMPLETION	75%	88%	43%	61%	57%	12%
HALF-WAY	72.4%	91.1%	32.3%	54.2%	67.7%	8.9%
START	76.3%	93.6%	39.2%	60.6%	60.8%	6.4%

CHAPTER 5. CONCLUSION AND FUTURE WORK

This research demonstrated the implementation of a predictive analytics-based approach to improve business processes. After examining the literature and thoroughly exploring the topic, it became evident that there was a need for new methods for applying predictive analytics in complex business processes such as logistics. The main challenges stemmed from developing a prediction technique revolved around the complex nature of the business process's events. The first research question was proposed to overcome this challenge. To address the second research question, an early prediction technique was developed.

RQ1: How can the complex nature of airfreight shipment be modeled for predicting delays?

Solution: A data transformation technique was proposed that simplified the shipping process's complex structure to prepare it for predictive modeling. The proposed technique divided the data based on the number of incoming legs: one, two, and three. Furthermore, the shipping process's serial structure of segments was leveraged to combine the segment data to overcome each leg's different number of segments. The result was three datasets each for different legs with no missing information and a simplified process structure. Two predictive modeling techniques were developed based on random forest and XGBoost using these datasets.

Additionally, six performance metrics were derived from the prediction results. It was observed that for all the prediction models, accuracy was between 75-80%, specificity was between 88-90%, sensitivity was between 32-47%, and geometric mean was between 55-66%. While the predictors performed very well in predicting no delays, they did not do very well in predicting delayed shipments, as evident by lower sensitivity score. However, these results provided compelling evidence that the proposed techniques could be used for predicting delays in a complex business process.

RQ 2: Will the performance of prediction models decrease if the process violations were predicted early?

Solution: Each shipment has planned and actual checkpoint information. The planned information is available before the start of the shipping process. The actual checkpoint information becomes available on completion of that checkpoint while the shipment is in transit. Three categories of datasets were derived to simulate shipments in transit: near completion, half-way, and the start of a shipment journey. The near completion category represented shipments that have completed the entire journey but have not yet been delivered. The second category represented shipments that were halfway through the journey; that is, they had completed their incoming leg journey. The last category represented shipments at the start of the journey process. Each category's dataset was used to model predictors, and their performance measured using the six metrics. The expectation was that the near completion category would show the best performance; however, the results showed this was not the case. Some predictors showed a slight decrease in performance, but others showed improved performance. This proves that early prediction is possible without compromising prediction accuracy.

Two tree-based algorithms were used to predict the delays in the shipment process. The predictors performed reasonably well in identifying shipments that did not have a delay, as shown by high specificity rates. However, they did not perform very well in identifying delayed shipments, as shown by the low sensitivity rate. This was due to the imbalance in the dataset, which had more than double the number of shipments with no delay than those that did. This has turned out to be a limiting factor for the prediction methods used here, but it also serves as an opportunity for future improvements. The prediction models were trained on shipment’s transport service data. This was another limitation of the dataset since it does not give us any information about the potential cause of delay. This limitation could be overcome if information of external souces of delay, such as temperature, weather conditions, etc., were available.

A new combined prediction model could be designed that could complement the existing model’s performance to improve their sensitivity. The early prediction technique could be extended to include a prescriptive approach by designing an intelligent decision-making model to prescribe actions based on the prediction outcome. This prescriptive analytical approach can be beneficial for companies that rely heavily on logistics.

The algorithms implemented here have found their application in various fields such as technology and healthcare. This research presented their application in a complex business setting. The results provide evidence that their application in other logistics industry fields, such as rail and road, is also possible.

REFERENCES

Ahmad, M. W., Mourshed, M., & Rezgui, Y. (2017). Trees vs Neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption. *Energy and Buildings*, *147*, 77–89. https://doi.org/10.1016/j.enbuild.2017.04.038

Alias, C., Rawet, V. L., & Neto, H. X. R. (2016). *Investigating into the Prevalence of Complex Event Processing and Predictive Analytics in the Transportation and Logistics Sector: Initial Findings From Scientific Literature*. 18.

Batra, A., & Jawa, V. (2016). *Classification of Arrhythmia Using Conjunction of Machine Learning Algorithms and ECG Diagnostic Criteria*. *1*, 7.

Ben Ayed, A., Ben Halima, M., & Alimi, A. M. (2015). Big data analytics for logistics and transportation. *2015 4th International Conference on Advanced Logistics and Transport (ICALT)*, 311–316. https://doi.org/10.1109/ICAdLT.2015.7136630

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Cukier, K., & Mayer-Schoenberger, V. (2013). The Rise of Big Data: How it's Changing the Way We Think about the World Essay. *Foreign Affairs*, *92*(3), [i]-40.

Datla, M. V. (2015). Bench marking of classification algorithms: Decision Trees and Random Forests - a case study using R. *2015 International Conference on Trends in Automation, Communications and Computing Technology (I-TACT-15)*, 1–7. https://doi.org/10.1109/ITACT.2015.7492647

Feldman, Z., Fournier, F., Franklin, R., & Metzger, A. (n.d.). *Proactive event processing in action: A case study on the proactive management of transport processes (industry article)*. 10.

Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139. https://doi.org/10.1006/jcss.1997.1504

Gal, A., Mandelbaum, A., Schnitzler, F., Senderovich, A., & Weidlich, M. (2017). Traveling time prediction in scheduled transportation with journey segments. *Information Systems*, *64*, 266–280. https://doi.org/10.1016/j.is.2015.12.001

Georganos, S., Grippa, T., Vanhuysse, S., Lennert, M., Shimoni, M., & Wolff, E. (2018). Very High Resolution Object-Based Land Use–Land Cover Urban Classification Using Extreme Gradient Boosting. *IEEE Geoscience and Remote Sensing Letters*, *15*(4), 607–611. https://doi.org/10.1109/LGRS.2018.2803259

Govindan, K., Cheng, T. C. E., Mishra, N., & Shukla, N. (2018). Big Data Analytics and Application for Logistics and Supply Chain Management. *Transportation Research Part E: Logistics and Transportation Review*, *114,* 343–349. https://doi.org/10.1016/j.tre.2018.03.011

Husband, S. M., & Roberts, J. (2017). Survival Random Forest to Predict Time to Fill. *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, 195–198. https://doi.org/10.1109/ICDMW.2017.32

Iovan, S. (n.d.). PREDICTIVE ANALYTICS FOR TRANSPORTATION INDUSTRY. *JOURNAL OF INFORMATION SYSTEMS*, 14.

Luque, A., Carrasco, A., Martín, A., & de las Heras, A. (2019). The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, *91*, 216–231. https://doi.org/10.1016/j.patcog.2019.02.023

Metzger, A., Leitner, P., Ivanović, D., Schmieders, E., Franklin, R., Carro, M., Dustdar, S., & Pohl, K. (2015). Comparing and Combining Predictive Business Process Monitoring Techniques. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *45*(2), 276–290. https://doi.org/10.1109/TSMC.2014.2347265

Mishra, N., & Silakari, D. S. (n.d.). *Predictive Analytics: A Survey, Trends, Applications, Oppurtunities & Challenges*.

Schoenherr, T., & Speier-Pero, C. (2015). Data Science, Predictive Analytics, and Big Data in Supply Chain Management: Current State and Future Potential. *Journal of Business Logistics*, *36*(1), 120–132. https://doi.org/10.1111/jbl.12082

Senderovich, A., Weidlich, M., Gal, A., & Mandelbaum, A. (2014). Queue Mining – Predicting Delays in Service Processes. In M. Jarke, J. Mylopoulos, C. Quix, C. Rolland, Y. Manolopoulos, H. Mouratidis, & J. Horkoff (Eds.), *Advanced Information Systems*

Engineering (pp. 42–57). Springer International Publishing. https://doi.org/10.1007/978-3-319-07881-6_4

Tharwat, A. (2020). Classification assessment methods. *Applied Computing and Informatics*, *ahead-of-print*(ahead-of-print). https://doi.org/10.1016/j.aci.2018.08.003