

**PERFORMANCE EVALUATION OF UNIVARIATE TIME SERIES AND  
DEEP LEARNING MODELS FOR FOREIGN EXCHANGE MARKET  
FORECASTING: INTEGRATION WITH UNCERTAINTY MODELING**

by

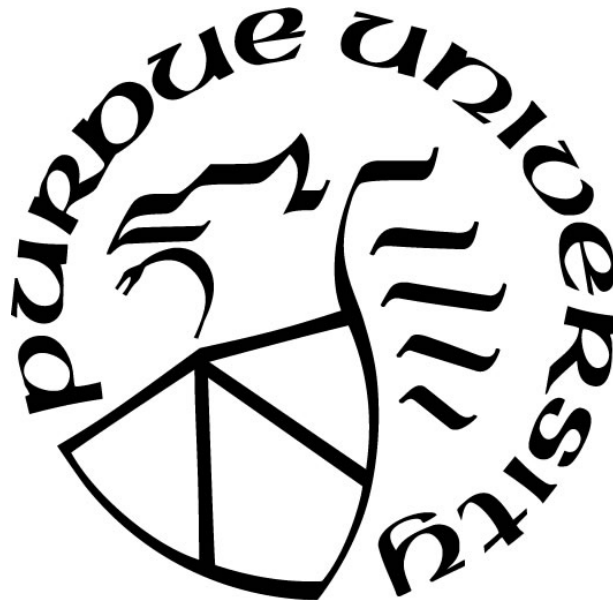
**Wajahat Waheed**

**A Thesis**

*Submitted to the Faculty of Purdue University*

*In Partial Fulfillment of the Requirements for the degree of*

**Master of Science in Electrical and Computer Engineering**



Department of Electrical and Computer Engineering

Hammond, Indiana

December 2021

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

**Dr. Colin Elkin, Chair**

Electrical and Computer Engineering

**Dr. Quamar Niyaz**

Electrical and Computer Engineering

**Dr. Xiaoli Yang**

Electrical and Computer Engineering

**Approved by:**

Dr. Colin Elkin, Chair

## ACKNOWLEDGMENTS

I would like to express the deepest appreciation to Dr. Colin Elkin who supervised this research. Dr. Elkin's areas of expertise include Machine Learning, Artificial Intelligence, Wireless Networks Modeling and Simulation Software Engineering. He continually supported, provided all the resources, and conveyed a spirit of adventure regarding the research. While there were some roadblocks, his guidance really helped me explore more possibilities and get a holistic view.

I would also like to thank Dr Xiaoli Yang and Dr. Quamar Niyaz for being part of my thesis committee and being flexible, which made the process smoother for me. Dr Xiaoli Yang has also helped me through her guidance in various aspects and Dr Quamar Niyaz has inspired me through his own body of research and the gift of helping me visualize a similar trajectory for myself.

I would also like to thank Dr. Sam L. Savage who is an adjunct Professor at Stanford University and Executive Director of [probabilitymanagement.org](http://probabilitymanagement.org), a non-profit, with a mission to make uncertainty actionable. Dr Savage is also the author of *The Flaw of Averages* and the inventor of the Stochastic Information Packet (SIP), a standardized data array for conveying uncertainty. His contribution, collaboration, and validation in this research regarding integration of deep learning models' predictions with uncertainty modeling using the SIP Math Modeler tool, Monte-Carlo simulations, and ChancCalc tool is highly appreciated.

# TABLE OF CONTENTS

LIST OF TABLES-----	6
LIST OF FIGURES-----	7
ABSTRACT-----	8
1. INTRODUCTION-----	9
2. LITERATURE REVIEW -----	12
3. THEORETICAL BACKGROUND-----	15
3.1 Macroeconomic Indicators-----	15
3.2 Recurrent Neural Networks (RNN)-----	17
3.3 Long Short-Term Memory Network (LSTM)-----	18
3.4 Gated Recurrent Unit (GRU)-----	20
3.4 Auto Regressive Integrated Moving Average (ARIMA) -----	22
3.5 Moving Average (MA)-----	22
3.6 Uncertainty Modeling -----	22
4. METHODOLOGIES-----	24
4.1 Datasets-----	24
4.2 Data Preparation-----	24
4.2.1 Upsampling and handling missing values -----	24
4.2.2 Data Transformation -----	24
4.3 Training and Testing-----	25
4.4 Hyperparameters' Optimization-----	25
4.5 Metrics for Evaluation-----	28
4.5.1 Mean Absolute Error (MAE)-----	28
4.5.2 Mean Absolute Percentage Error (MAPE)-----	29
4.5.3 Mean Squared Error (MSE)-----	29
4.6 Time period for forecasting: 1-day, 1-week, 2-weeks-----	30
4.7 Integration of models' predictions with Uncertainty Modeling-----	30
5. RESULTS -----	31
5.1 USD/CAD Forecasts-----	31
5.1.1 1-Day Forecasts-----	31

5.1.2	1-Week Forecasts-----	32
5.1.3	2-Weeks Forecasts-----	34
5.2	USD/AUD Forecasts -----	35
5.2.1	1-Day Forecasts-----	35
5.2.2	1-Week Forecasts-----	36
5.2.3	2-Weeks Forecasts-----	38
5.3	Uncertainty Modeling Example Results: -----	39
6.	DISCUSSION-----	41
7.	CONCLUSION -----	43
8.	FUTURE WORK-----	44
	APPENDIX-----	45
	REFERENCES-----	65

## LIST OF TABLES

Table 1: Hyperparameters for USD/CAD 1-day prediction model. -----	25
Table 2: Hyperparameters for USD/CAD 1-week prediction model-----	26
Table 3: Hyperparameters for USD/CAD 2-weeks prediction model-----	26
Table 4: Hyperparameters for USD/AUD 1-day prediction model. -----	27
Table 5: Hyperparameters for USD/AUD 1-week prediction model. -----	27
Table 6: Hyperparameters for USD/AUD 2-weeks prediction model -----	28
Table 7: Evaluation metrics of all models on test dataset for the USD/CAD 1-day predictions. -	31
Table 8: Error Distribution for all models for USD/CAD 1-day predictions. -----	32
Table 9: Evaluation metrics of all models on test dataset for the USD/CAD 1-week predictions. -----	33
Table 10: Error Distribution for all models for USD/CAD exchange rate 1-week predictions. --	33
Table 11: Evaluation metrics of all models on test dataset for the USD/CAD 2-weeks predictions. -----	34
Table 12: Error Distribution for all models for USD/CAD exchange rate 2-weeks predictions. -	35
Table 13: Evaluation metrics of all models on test dataset for the USD/AUD 1-day predictions. -----	36
Table 14: Error Distribution for all models for USD/AUD exchange rate 1-day predictions. ----	36
Table 15: Evaluation metrics of all models on test dataset for the USD/AUD exchange rate 1- week predictions.-----	37
Table 16: Error Distribution for all models for USD/AUD exchange rate 1-week predictions. --	38
Table 17: Evaluation metrics of all models on test dataset for the USD/AUD 2-weeks predictions. -----	38
Table 18: Error Distribution for all models for USD/AUD 2-weeks predictions. -----	39

## LIST OF FIGURES

Figure 1: An unrolled recurrent neural network [20].	17
Figure 2: Design Architecture of LSTM [20].	19
Figure 3: GRU and LSTM Architecture Comparison [22].	21
Figure 4: Overall comparison of architecture of LSTM and GRU [22].	21
Figure 5: 1-Day Predictions for USD/CAD Exchange Rate Value.	31
Figure 6: 1-Week Predictions for USD/CAD Exchange Rate Value.	32
Figure 7: 2-Weeks' Time Series Forecasting Prediction for USD/CAD Exchange Rate Value.	34
Figure 8: 1-Day Time Series Forecasting Prediction for USD/AUD Exchange Rate Value.	35
Figure 9: 1-Week Time Series Forecasting Prediction for USD/AUD Exchange Rate Value.	36
Figure 10: 1-Week Time Series Forecasting Prediction for USD/AUD Exchange Rate Value.	37
Figure 11: 2-Weeks' Time Series Forecasting Prediction for USD/AUD Exchange Rate Value.	38
Figure 12: Chance of prediction greater than or less than the actuals.	39
Figure 13: Chance of prediction being greater than or less than the target value.	40

## **ABSTRACT**

Foreign exchange market is the largest financial market in the world and thus prediction of foreign exchange rate values is of interest to millions of people. In this research, I evaluated the performance of Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Autoregressive Integrated Moving Average (ARIMA) and Moving Average (MA) on the USD/CAD and USD/AUD exchange pairs for 1-day, 1-week and 2-weeks predictions. For LSTM and GRU, twelve macroeconomic indicators along with past exchange rate values were used as features using data from January 2001 to December 2019. Predictions from each model were then integrated with uncertainty modeling to find out the chance of a model's prediction being greater than or less than a user-defined target value using the error distribution from the test dataset, Monte-Carlo simulation trials and ChancCalc excel add-in. Results showed that ARIMA performs slightly better than LSTM and GRU for 1-day predictions for both USD/CAD and USD/AUD exchange pairs. However, when the period is increased to 1-week and 2-weeks, LSTM and GRU outperform both ARIMA and moving average for both USD/CAD and USD/AUD exchange pair.



## 1. INTRODUCTION

Each day billions or trillions of dollars are traded on multiple foreign exchange markets where foreign currencies are exchanged. In terms of the volume, this makes it the largest financial market in the whole world. A foreign exchange rate value is the value of a country's currency compared to that of another country. Most of the major monetary bodies have embraced floating exchange rates. It is a system in which currencies are exchanged or traded. When referencing a foreign exchange rate pair as USD/CAD or USD/AUD with a value of 1.4 or 1.38 it is meant that how much Canadian or Australian dollars need to be paid to get 1 United States Dollar.

Trading foreign currencies used to be very hard for an individual before the evolution of the internet. Foreign exchange (forex) market trading also required a minimum amount in the past and thus only large companies, funds, or individuals with a lot of capital used to trade in the foreign exchange market. Due to the evolution of the internet, retail markets for individuals have been created by brokers or banks, making it very easy to trade in the forex market. Also, forex markets constitute not only spot (cash) markets but also the derivatives markets which offer futures, forwards, options, and currency swaps [1]. Individuals in this market, hedge against a currency and more commonly interest rate risk, to estimate or predict on specific events, and to diversify their investing portfolios [1].

At the same time, business owners who are in the import or export business are heavily dependent on the currency exchange values. Exchange rate's volatility along with the value is an extremely important factor in the business owners' daily business decisions. As an example, if you are living in the United States and want to buy chocolate made in Canada, then directly or indirectly the company you are buying from will be paying the Canadians for the chocolate in Canadian dollars. This essentially means that the company importing in the United States will be exchanging the United States Dollar to Canadian Dollar based on the current USD/CAD exchange value.

Moreover, people who send or receive foreign remittance are also very interested in the exchange value of the currencies. Foreign remittance is when a foreign worker sends money to their family or any other individuals residing in another country. For a lot of countries in the world, a lot of the portion of GDP is due to foreign remittance.

While the exchange rate value is of interest to traders, foreign remittances' receivers and senders, imports/exports business owners, it is also of interest to countries themselves about to obtain loan from organizations like International Monetary Fund (IMF). IMF, as an example, has started to give loans to help countries have enough space to adjust their policies and restore good economic conditions for a long-term growth [2]. IMF provides loan on the current dollar rate; however, when the country wants to pay back the installment of the loans, the exchange rate is different. Since the earnings of a country are in their local currencies, while negotiating the lending settlement, the country wants to know what their exchange value is going to be in the future.

Exchange rate value has multiple indicators and influences. The State Bank of a certain country like Canada does not have the right to influence the exchange rate directly [2]. However, the state bank can adjust interest rates or print more money for stability. Fundamentally, foreign exchange rate value is influenced by demand and supply of the foreign and domestic currencies. Currency values tend to rise with the increase in demand and tend to fall with the decrease in demand [3]. The demand and supply are then influenced by multiple macroeconomic indicators, most of which are not accessible easily by the public. I will thus be forecasting multiple exchange currencies which have relatively reliable and sufficient data.

Past foreign exchange rates and the two countries' macroeconomic indicators will help us to build a better forecasting model. Nasdaq Data Link provides around 8000+ datasets for 200+ countries' macroeconomic indicators. Since there has been a lot of improvement in processing, it has become easier to use machine learning in time series forecasting. Some of the big challenges include formatting the data correctly, getting enough samples, removing noise of one indicator which is relatively much larger than others. Thus, I will be performing a comparative analysis of univariate traditional time series forecasting models with deep learning models like LSTM and

GRU using a combination of macroeconomic indicators and past exchange rate values. I will then be integrating the models' predictions with uncertainty modeling to add uncertainties from the error distribution in the test dataset using metalog distribution generator, Monte-Carlo simulations and ChancCalc excel add-in tool to move away from "single point averages" and also answer the most common but unanswered question: "What is the chance the exchange rate value will be greater/less than X in Y days"?

## 2. LITERATURE REVIEW

In this chapter, I will be providing the literature review of multiple univariate and multivariate time series forecasting models built for time series data predictions.

Zheng & Khushi [4] hybridized multiple models like Recurrent Neural Network (RNN) and Autoregressive Integrated Moving Average (ARIMA). Interestingly, they could remove a lot of noise from the time series data of USD/JPY exchange rate. Their methodology can model dynamic systems in general and specifically the forex market. Their results show a directional accuracy of around 76 % on five-minute frequency time series data. Ariyo et al.,[5] built a stock prediction ARIMA model. They obtained their data from New York Stock Exchange (NYSE) and Nigeria Stock Exchange (NSE). Their results depicted that ARIMA performs good for short-term predictions.

Chantarakasemchit et al., [6] investigated the performance of Multilayer Perceptron (MLP) and Linear regression for EUR/USD exchange rate predictions. They took a novel approach and added moving average, interest rate, GDP, and dollar index as features. Their results showed that adding moving average and the financial factors improved the models' performance significantly. Nootyaskool & Choengtong [7] predicted USD/THB exchange rate using Hidden Markov Models (HMM). Their data ranged from 2002 till 2013 and the main features for the model was the interest rate, growth, inflation rate and dollar index. Their novel technique was to encode these four features into one observation sequence to be able to train the HMM. They were able to achieve a mean absolute percentage error of around 0.2 % for next day predictions.

Aguilar et al., [8] performed a survey on multiple techniques like machine learning, data mining, and natural language processing. Their survey results depicted that foreign exchange rates are non-stationary and have a lot of noise and thus researchers are prone to using machine learning and data mining techniques to predict foreign exchange rates. Also, they identified that there are no official benchmarks set and the evaluation metrics are inconsistent. This leads to difficulties in comparing the performance of the models.

Oncharoen & Vateekul [9] used deep learning to improve stock market predictions using event embedding vectors which were obtained from the headlines of certain news sources. Along with these vectors, they also included the past time series data and technical indicators as features. They built Convolutional Neural Network (CNN) and Long Short-term Memory (LSTM) models and used accuracy and annualized return as their evaluation metrics. Using data from three different news sources, a comparative analysis was performed. Their results depicted that enhancing the vectors, numerical and textual information as features to the deep learning architectures significantly improved their model performance.

Kadilar & Adla [10] used Autoregressive Integrated Moving Average along with multiple neural network architecture-based models to predict USD/TRY exchange pair. Their results depicted that neural network performed better for the daily predictions than ARIMA. Kamruzzaman et al., [11] built neural networks model to predict AUD/USD and AUD/EUR exchange pairs. Their results showed that back propagation is very slow and tedious in a real-life depiction. Naeini et al., [12] predicted stock values using time series data and feed-forward neural network. Their model depicted that RNN performed much better than Multi-Layer Perceptron (MLP) in terms of accuracy. Nilesh et al., [13] performed a comparative analysis of multiple machine learning regression models. Their predictions were run on 59 exchange pairs, and they found out that SVM works best for relatively smaller data when evaluating accuracies. Alexiei & Karl [14] also performed a comparative analysis for stock market time series forecasting. Their CNN model achieved an accuracy of 65 % when forecasting next month's prediction while only 60 % for the next week's prediction. Their study concluded that CNN doesn't perform better than industry-leading techniques like SVM and Logistic Regression.

Ahmed et al., [15] performed a comparative analysis on multiple machine learning models' time series forecasting using monthly data. Multi-layer perceptron (MLP), neural networks, k-nearest neighbor regression, support vector regression, and Bayesian processes were used. Their research showed that MLP and Bayesian processes performed much better. Their study also showed that different data preprocessing techniques had different implications on the accuracy of the models.

Papacharalampous et al., [16] performed a detailed analysis on multiple problems that are linked with univariate time series forecasting models. To summarize, their results favor the usage of less recent lagged variables, hyperparameter optimization won't likely improve model's performance and ML and classical algorithms' performance is similar. Sudimanto et al., [17] performed a comparative analysis on SVM, Tree model and Ensemble model. The accuracy for SVM, tree and ensemble models ranged from 86-88 %. Livieris et al., [18] used CNN and LSTM to predict gold prices. They concluded that when using LSTM layers with a combination of convolutional layers, model's overall performance can be significantly improved.

### **3. THEORETICAL BACKGROUND**

In this chapter I will be providing theoretical background and defining the macroeconomic features, time series forecasting models, and some of the concepts used in uncertainty modeling.

#### **3.1 Macroeconomic Indicators**

While some of indicators are well-known, there are many which need a little context to be able to get a clear picture. Following are the descriptions for each macroeconomic indicator and their possible effect on the foreign exchange value:

1. Current Account:

Current account records the value of imports, exports and international transfers of capital and is amongst the three balances of payments for a given country. An increase in current account volume is a positive indicator for a currency's exchange value [19].

2. Foreign Exchange Reserves:

Foreign exchange reserves refer to currency deposits/assets of a country's State/Central Bank used to support the liabilities

3. Foreign Direct Investment (FDI):

FDI refers to investment/ purchase of interest by a foreign entity in a country.

4. Gross Domestic Product (GDP):

GDP is total value of everything that was produced in a single year. Increase in GDP refers to positive economic growth having a direct or indirect effect on the currency exchange value.

5. Government Debt to GDP:

Debt-to-GDP ratio is ratio of country's debt and its gross domestic product (explained above). It refers to a country's ability to produce and sell goods and services enough to pay back the debts incurred.

6. Interest Rate:

Interest rate of a country heavily influences the foreign exchange rate. In general, it is believed that foreign investment is tied to a higher interest rate and high interest rate will increase the demand for the specific currency and increase the exchange rate [3].

7. Inflation Rate:

Inflation rate refers to decline in the purchasing power of a currency. It is the rate at which the value of currency falls, and prices of goods & services rise. The increase in prices means that the currency can buy less than it could buy in the past [3].

8. Imports:

When a country's imports increase, the supply of the currency increases which then decreases the value of the currency.

9. Exports:

When a country's exports increase, the demand of the currency increases which increases the value of the currency.

10. Consumer Spending:

Consumer Spending refers to the total money spent on goods and services by individuals and households. Lower consumer spending impacts the currency exchange value negatively.



### 11. Unemployment Rate:

Unemployment rate is the percentage of labor force that is currently not employed. Labor force refers to the people actively job hunting a job in the last month(s). High unemployment is typically linked to a decrease in foreign exchange value.

### 12. Consumer Pricing Index:

The most used inflation index

## 3.2 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNN) are a kind of artificial neural network where the output from the previous step is passed onto the current step as an input. RNNs have internal memory are recurrent because they perform the same operation for every input and the output of the current input is dependent on the last operation [20]. RNN takes into account the current input and the output which it learnt from the previous input [20]. In contrast to the feedforward neural networks, RNNs utilize the internal memory to be able to process multiple sequences of inputs where the inputs are dependent on each other. Figure 1 shows the design architecture of RNN as shown below:

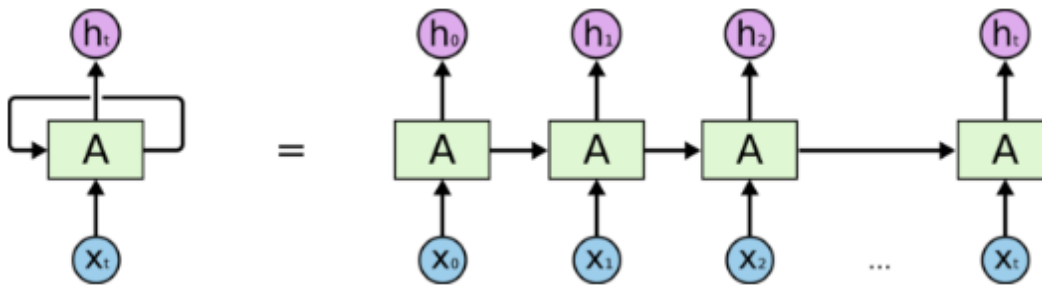


Figure 1: An unrolled recurrent neural network [20].

If we look at Figure 1 we can see that the recurrent neural network takes  $x_0$  from the sequence of the inputs resulting in  $h_0$ . This combination together with  $x_1$  becomes the input for the following step and so on. While training, following this process the RNN is able to keep remembering the context [20].

Following is the formula for the current state:

$$h_t = f(h_{t-1}, x_t)$$

Following formula shows how an activation is applied:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

Where  $W$  = weight,  $W_{hh}$  = weight at previous hidden state,  $W_{xh}$  = weight at the current input state,  $\tanh$  = activation function to transform the activations to squeeze in to values from -1 to 1.

Following is the resultant output:

$$y_t = W_{yh}h_t$$

Where  $y$  = output state,  $W_{yh}$  = weight at the output state

With this structure RNN has some advantages and disadvantages. One major advantage is that RNN is able to learn in such a way that we can assume that the samples/data points are dependant on the previous samples. Disadvantages on the other hand include gradient exploding and vanishing problem, difficult training, inability to process very long sequences [20].

### 3.3 Long Short-Term Memory Network (LSTM)

LSTM is a type of Recurrent Neural Network (RNN) which can learn long-term dependencies. Originally introduced in 1997, they have been refined and made popular over the years. In comparison to RNNs, which have the design structure like a chain of modules that are repetitions of artificial neural networks, LSTMs have a similar chain structure, however, the repeating module has four neural network layers instead of one. It was specifically introduced to solve the

vanishing gradient problem [20]. Thus, LSTM is very good at learning information for much longer periods of time and suited to process and predict time series given the time lags and past data. LSTM uses backpropagation to train the model. Figure 2 below shows the three gates in LSTM and the overall design architecture:

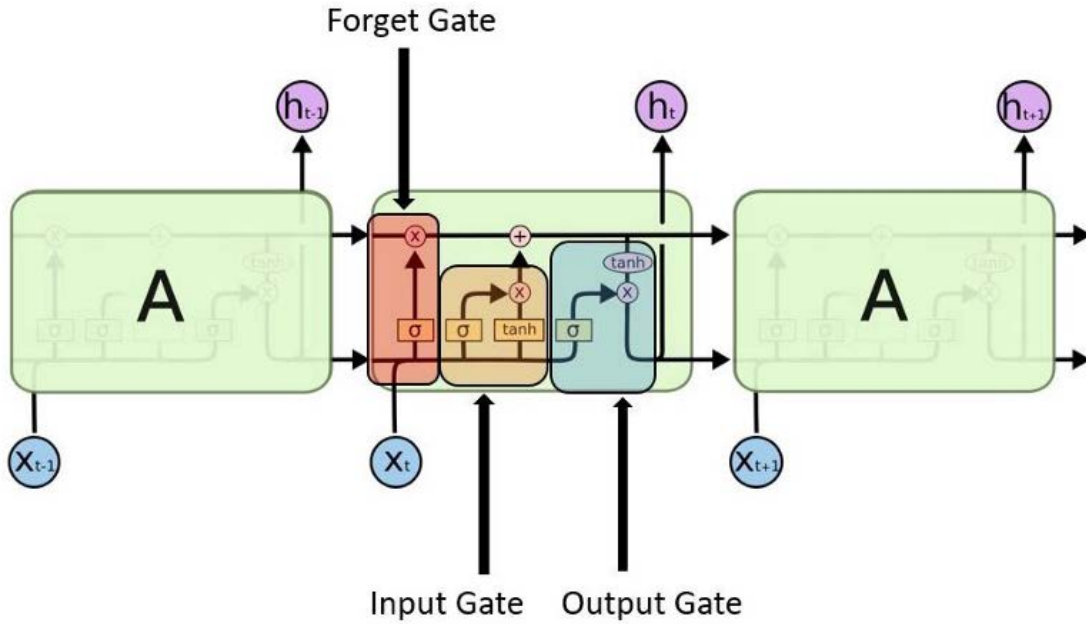


Figure 2: Design Architecture of LSTM [20].

As can be seen from Figure, there are three gates in LSTM. I will briefly explain each of the gates:

1. **Input Gate:** This gate's job is to find which value from the input should be utilized to be able to update the memory. This gate has a sigmoid function which makes the decision on which specific values are going to be let through and a tanh function which links a weight to the passed values from -1 to 1. Following formulas depict the equations involved at the input gate:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$C_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

2. **Forget gate:** This gate's job is to find out what information needs be removed from the block. This is done through sigmoid function which looks at the previous state( $h_{t-1}$ ) and input

( $x_t$ ) resulting in a number 0 and 1 for each of the number in the cell state ( $C_{t-1}$ ). Following formulas depict the equations involved at the forget gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

3. Output Gate: This gate is where the input and the memory block is utilized to decide the output where the sigmoid function again chooses which specific values are going to be let through. The tanh function gives a weight from -1 to 1 based on significance which is then multiplied with sigmoid function's output. Following formulae depict the equations involved at the output gate:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \cdot \tanh(C_t)$$

### 3.4 Gated Recurrent Unit (GRU)

GRUs are also very similar to LSTM. They were also introduced to solve the short-term memory problem faced using RNNs. Just like LSTM, GRU also has gates that control the flow of information. Compared to LSTM, GRU is relatively new, were designed to improve LSTM and simpler in design architecture and thus much faster to train as compared to LSTM [21].

Figure 3 below shows the comparison in the architecture of LSTM and GRU:

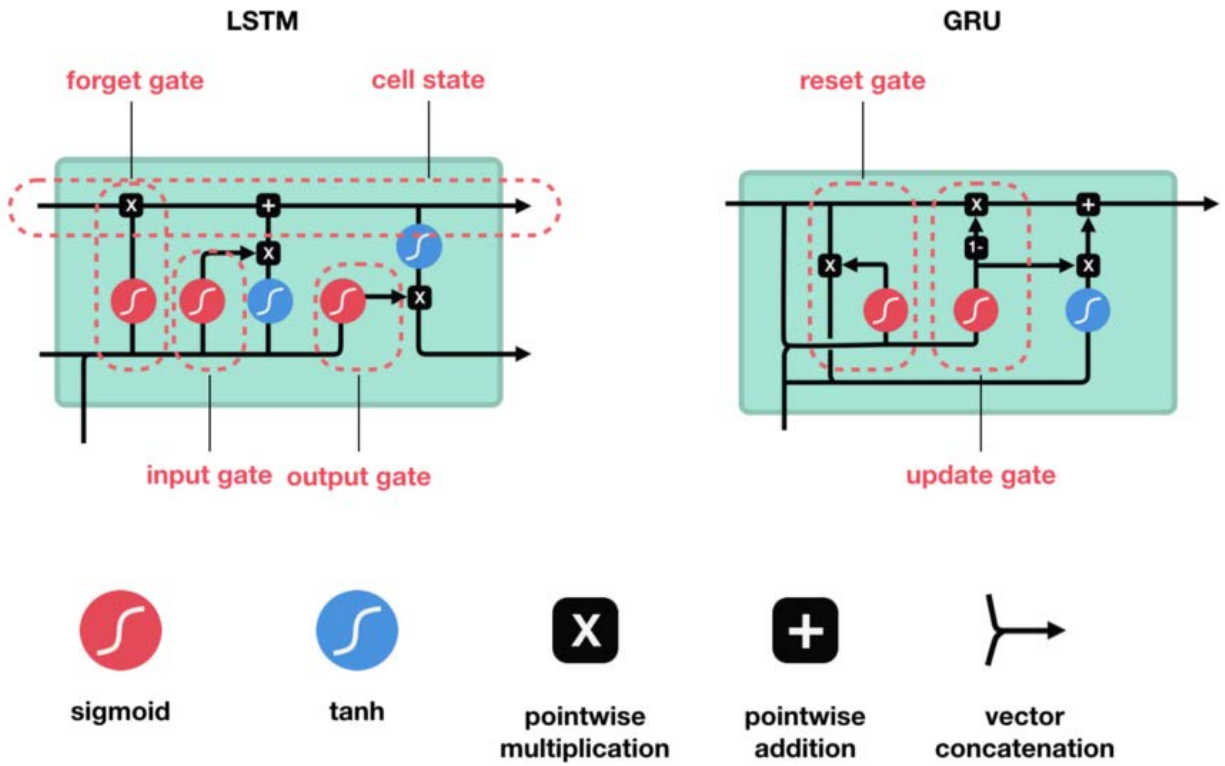


Figure 3: GRU and LSTM Architecture Comparison [22].

Following image also shows the comparison in the overall design architecture of LSTM and GRU:

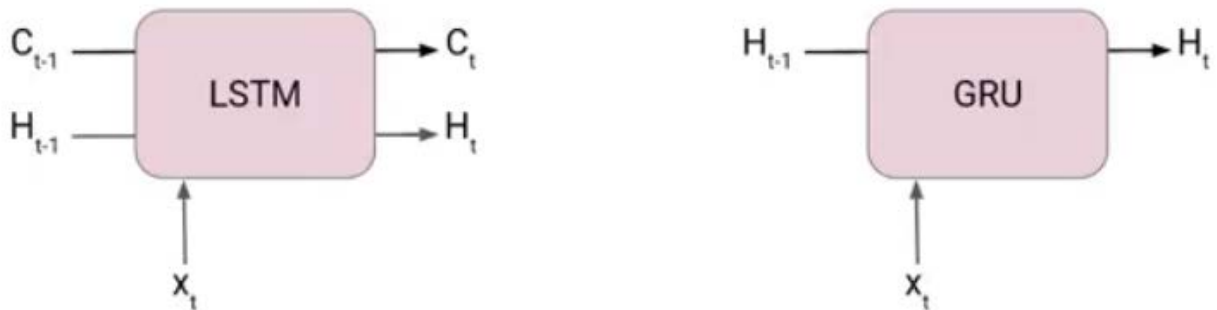


Figure 4: Overall comparison of architecture of LSTM and GRU [22].

Another key difference between GRU and LSTM is the separate cell state denoted as  $C_t$ . While LSTM has a separate cell state, GRU has a hidden state denoted as  $H_t$ . GRU has only reset gate and update gate while has Input gate, Forget gate, and Output gate.

### **3.4 Auto Regressive Integrated Moving Average (ARIMA)**

ARIMA is a univariate time series forecasting model which uses the past time series values to predict the future. ARIMA is based on the lagging values and the lagged forecast errors. ARIMA works for all kinds of time series data except seasonal. ARIMA has three important hyperparameters which need to be tuned to get the best results. The three hyperparameters are  $p$ ,  $d$  and  $q$ .  $p$  is the order of the Auto Regressive term,  $q$  is the order of the moving average term and  $d$  is the total number of differencing needed to make the time series stationary [23].

### **3.5 Moving Average (MA)**

MA is another univariate time series forecasting model that smooths out the time series data by producing a constantly updated average time series value. The average can be taken for any period. Moving averages are best when analyzing short-term or long-term trends. The algorithm runs the fastest and has only one parameter and thus can be adjusted easily for any time frame. [24]

### **3.6 Uncertainty Modeling**

Have we ever thought about how our model's error distribution on unseen data can help us predict how the model's predictions can vary in the future and how we can improve them? This is where uncertainty modeling comes into the picture. There are multiple approaches for uncertainty modeling; however, for the purpose of this research, I will only be focusing on the probabilistic and possibilistic approach [25]. I will do that using Probability Management, which is the representation of uncertainties as data arrays called SIPs which adhere to not only laws of arithmetic and but also the laws of probability [26]. SIPMath modeler tools along with ChancCalc can be used together for adding uncertainties to future predictions. ChancCalc is an Excel add-in which performs the operations of uncertainty using the same keystrokes as those for

numerical calculations [26]. This approach solves the Flaw of Averages, which is a systematic error that occurs when uncertain predictions are replaced by single “average” numbers [26].

## **4. METHODOLOGIES**

### **4.1 Datasets**

The datasets were obtained from the Nasdaq data link website which has 8000+ datasets of 200+ countries' macroeconomic variables including currency exchange values [27]. While each variable's data points are not equal, each variable can be visualized on the platform and data can be accessed through an API. Due to some countries' currency exchanges being affected by unstable political situations, volatility, variance, frequency, and percentage of null values, I chose only USD/CAD and USD/AUD exchange pairs for modeling and analysis. The data obtained is from 1st January 2000 till 31<sup>st</sup> December 2019. The reason for not adding 2020 and 2021 is because of the unusual and uncertain circumstances originated due to the COVID pandemic.

### **4.2 Data Preparation**

#### **4.2.1 Upsampling and handling missing values**

A few of the variables had weekly data available. Since I am dealing with daily data throughout, to be able to solve the problem to some extent, I used upsampling. Upsampling, an opposite operation of downsampling, is a technique used to handle time series data. The goal of upsampling is to resample the dataset from a larger time frame to a smaller time frame. However, the additional data created has a lot of null values which are filled using forward filling. The point of resampling and forward filling is to propagate forward from the day of the week. However, if the first day of the week has missing values, then the whole week would have missing (null) values. Thus, before re-sampling and forward filling the null values created, I first performed forward filling to fill null values for all the variables.

#### **4.2.2 Data Transformation**

Indicators like unemployment rate, inflation rate, interest rates are already in percentage values and have a limited variance. These variables' true values were normalized from 0 to 1. However, indicators like GDP, FDI and FER are very large numbers compared to other country's same



indicator. Thus, I only took percent changes instead of the original value and then normalized them so that all variables have a similar weightage for the deep learning models to run. This also results in better performance for all deep learning models employed.

### 4.3 Training and Testing

I divided the dataset into training and testing, where training dataset contains 85 % of the entire dataset. This results in training dataset which starts from 2000 to 2016 while the remaining dataset i.e., from 2017 to 2019 is thus reserved for testing purposes. While training the deep learning models, I also reserved 10-15 % of the training set to validation set to improve the model's performance and pointing out where the model tends to overfit.

### 4.4 Hyperparameters' Optimization

Keras deep learning library was used to build the LSTM and GRU models using Python in Jupyter Notebook. Since the training time for LSTM and GRU is high, the hyperparameters were chosen based on trial-and-error method and trying all the possible combinations is not possible due to time constraints. Following table depicts the hyperparameters used in the final LSTM and GRU model which produced the best results on the testing set. Table 1 to 6 depict the parameters used for training the LSTM and GRU model. Table 1 below depicts the hyperparameters used for building the LSTM and GRU model for the USD/CAD 1-day prediction model:

Table 1: Hyperparameters for USD/CAD 1-day prediction model.

Parameter	<b>LSTM</b>	<b>GRU</b>
Neurons	25	30
Input shape	Step size = 30, features = 23	Step size = 35, features = 23
Activation Function	Tanh	Relu
Loss Function	Mean squared error	Mean squared error
Layers	2	3
Dropout	0.25	0.2
Optimizer	Adam	Adam
Epochs for training	80	80
Output neuron	1	1

Table 2 depicts the two models for the USD/CAD 1-week predictions. The step size and number of neurons increase for both the models.

Table 2: Hyperparameters for USD/CAD 1-week prediction model

Parameter	<b>LSTM</b>	<b>GRU</b>
Neurons	40	50
Input shape	Step size = 40, features = 23	Step size = 45, features = 23
Activation Function	Tanh	Relu
Loss Function	Mean squared error	Mean squared error
Layers	2	3
Dropout	0.25	0.2
Optimizer	Adam	Adam
Epochs for training	100	100
Output neuron	1	1

Table 3 depicts the two models for the USD/CAD 1-week predictions. The activation function for LSTM changes and step size, epochs and number of neurons increase for both the models.

Table 3: Hyperparameters for USD/CAD 2-weeks prediction model

Parameter	<b>LSTM</b>	<b>GRU</b>
Neurons	30	40
Input shape	Step size = 60, features = 23	Step size = 75, features = 23
Activation Function	Tanh	Tanh
Loss Function	Mean squared error	Mean squared error
Layers	2	3
Dropout	0.25	0.2
Optimizer	Adam	Adam
Epochs for training	150	150
Output neuron	1	1

Table 4 to 6 depict the models created for the USD/AUD exchange pair. For the USD/AUD exchange pair models, CPI and GDP-to-Debt ratio features were not considered due to high number of null values. Thus, for the USD/AUD models I will only be considering 9 macroeconomic indicators for each country:

Table 4 depicts the two models for the USD/AUD 1day predictions. The parameters are a bit different than the 1-day model for USD/CAD:

Table 4: Hyperparameters for USD/AUD 1-day prediction model.

Parameter	<b>LSTM</b>	<b>GRU</b>
Neurons	40	50
Input shape	Step size = 35, features = 19	Step size = 45, features = 19
Activation Function	Tanh	Relu
Loss Function	Mean squared error	Mean squared error
Layers	2	3
Dropout	0.15	0.1
Optimizer	Adam	Adam
Epochs for training	100	100
Output neuron	1	1

Table 5 depicts the two models for the USD/AUD 1-week predictions. The step size, epochs and number of neurons increase for both the models.

Table 5: Hyperparameters for USD/AUD 1-week prediction model.

Parameter	<b>LSTM</b>	<b>GRU</b>
Neurons	50	60
Input shape	Step size = 45, features = 19	Step size = 60, features = 19
Activation Function	Tanh	Relu
Loss Function	Mean squared error	Mean squared error
Layers	2	3
Dropout	0.1	0.2
Optimizer	Adam	Adam
Epochs for training	130	130
Output neuron	1	1

Table 6 depicts the two models for the USD/CAD 1week predictions. The step size and number of neurons increase for both the models.

Table 6: Hyperparameters for USD/AUD 2-weeks prediction model

Parameter	<b>LSTM</b>	<b>GRU</b>
Neurons	60	75
Input shape	Step size = 60, features = 19	Step size = 75, features = 19
Activation Function	Tanh	Tanh
Loss Function	Mean squared error	Mean squared error
Layers	2	3
Dropout	0.1	0.2
Optimizer	Adam	Adam
Epochs for training	120	120
Output neuron	1	1

The decision for the number of epochs was taken based on the training and validation errors. I stopped the training after a certain number of epochs because the model starts overfitting. This is when the training errors keep decreasing but the validation errors start increasing.

As a starting point or a general rule of thumb, one hidden layer works with simpler problems while two layers are mostly enough to train complex features [28]. Moreover, each LSTM and GRU layer was followed by a Dropout which helps in preventing the models from overfitting. The dropout layer does that by ignoring neurons randomly. This lets the model reduce its sensitivity to certain weights of individual neurons [28].

## 4.5 Metrics for Evaluation

The model was then tested on the testing set and the evaluation metrics used are as follows:

### 4.5.1 Mean Absolute Error (MAE)

The Mean absolute error (MAE) is the average of the absolute difference between the actual exchange values and predicted exchange values.

Following is the equation for finding out MAE:

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - y_m|$$

Where,

$y$  = predicted value of  $y$

$N$  = number of total samples

$y_m$  = mean value of  $y$

#### 4.5.2 Mean Absolute Percentage Error (MAPE)

This metric is much easier to understand for a comparative analysis.

$$MAPE = \frac{100}{N} \sum_{i=1}^N \frac{|y_i - y_m|}{y_i}$$

Where,

$y$  = predicted value of  $y$

$N$  = number of total samples

$y_m$  = mean value of  $y$

#### 4.5.3 Mean Squared Error (MSE)

MSE is the average of the squared difference between the actual exchange values and predicted exchange values. This metric gives us an idea about the variance of the residuals.

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - y_m)^2$$

Where,

$y$  = predicted value of  $y$

$N$  = number of total samples

$y_m$  = mean value of  $y$

#### **4.6 Time period for forecasting: 1-day, 1-week, 2-weeks**

Time periods used for future predictions are for 1 day, 1 week and 2 weeks which essentially means that I will be predicting the foreign exchange value for USD/CAD and USD/AUD for a single day which is 1 day, 1 week and 2 weeks far from the last available date.

#### **4.7 Integration of models' predictions with Uncertainty Modeling**

For our use case, I used both the SIPMath modeler tools and ChancCalc to add uncertainties into our predictions and answer one of the most unanswered questions: “What is the chance the exchange value will be greater/less than X after Y days?”.

After training our models, I tested the model's performance on the unseen data from 2017 to 2019. I then obtained the error distribution of the test dataset and calculated specifically the 25<sup>th</sup> %, 50<sup>th</sup> % and 75<sup>th</sup> % percentile. Using these percentiles, I created a metalog distribution of the errors and simulated the errors on 1000 Monte-Carlo simulation trials using the SIPMath modeler tool. For statistical correctness, the test dataset should be large enough or representative to be able to employ Monte-Carlo simulations. Since our test dataset is for 3 years, we can consider this period large enough to be representative of the model's possibilities of errors in the future predictions. These 1000 trials represent the possibilities of the possible future error values for a possible future prediction. These 1000 trials are saved as a SIP and converted into a library. This library/SIP is then imported into the ChancCalc excel add-in along with our average predictions. Using the ChancCalc tool, I simulated the average error/uncertainty-added predictions and calculate the percentage chance of the possibility of a target exchange value.

## 5. RESULTS

In this chapter, I will be depicting the predictions, evaluation metrics and error distribution of all the models for USD/CAD and USD/AUD for each of the 1-day, 1-week, and 2-weeks forecasts.

### 5.1 USD/CAD Forecasts

#### 5.1.1 1-Day Forecasts

Figure 5 below shows the results of all the 4 models' prediction along with the actual exchange values:



Figure 5: 1-Day Predictions for USD/CAD Exchange Rate Value.

Table 7: Evaluation metrics of all models on test dataset for the USD/CAD 1-day predictions.

USD/CAD 1-DAY				
	LSTM	GRU	ARIMA	Moving Avg
MSE	7.455191e-05	1.286733e-04	2.711388e-05	4.928905e-04
MAE	0.0069	0.009319	0.003912	0.016502
MAPE	0.527593	0.709541	0.300806	2.666454

Table 7 shows that ARIMA performs the best while moving average performs the worst for 1-day prediction of the USD/CAD exchange pair on 3 years of test dataset.

To be able to add uncertainties into our model predictions for the test dataset and also for future prediction, and adjust our forecasts due to any bias of the model, I will now calculate the error distrubution of the erros resulted from the test predictions. Table 8 below shows the error distrubution for all of the 4 models.

Table 8: Error Distribution for all models for USD/CAD 1-day predictions.

Error Description	LSTM	GRU	ARIMA	Moving Avg
Std	0.007300	0.008078	0.005211	0.022182
Min	-0.020309	-0.049200	-0.020573	-0.077037
25 %	0.000069	0.003304	-0.002990	-0.012195
50 %	0.004557	0.008061	0.000197	0.001907
75 %	0.009402	0.012200	0.002912	0.011298
Max	0.025480	0.040328	0.017748	0.049630

### 5.1.2 1-Week Forecasts

Figure 6 below shows the results of all the 4 models' prediction for 1-week along with the actual exchange values:

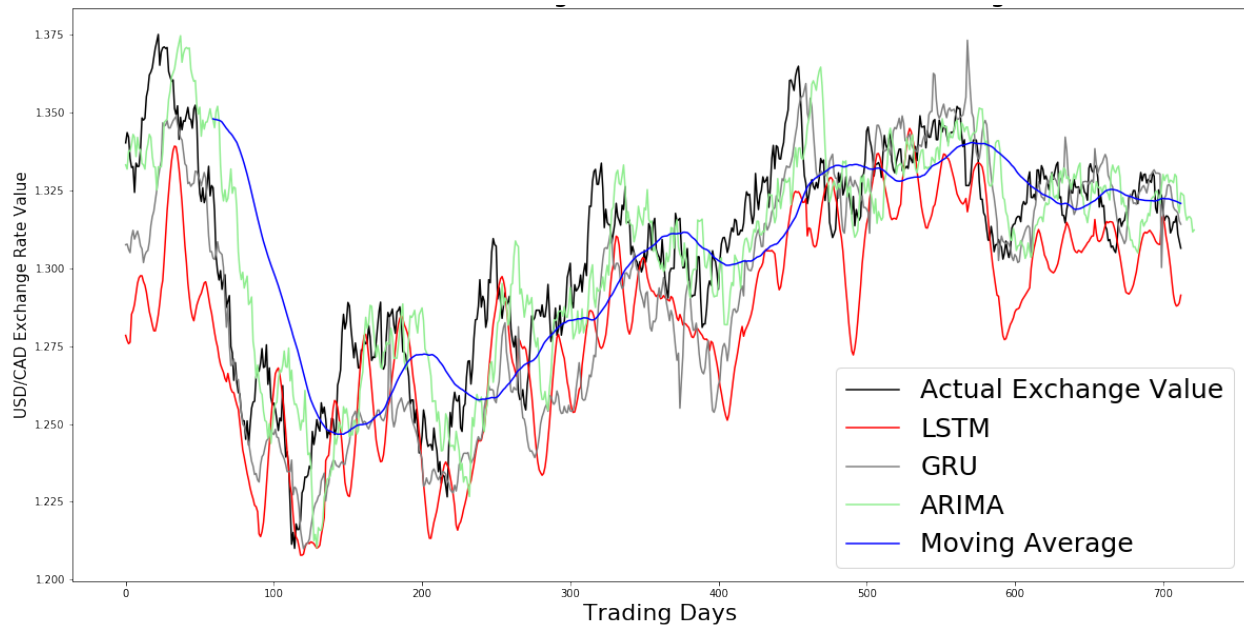


Figure 6: 1-Week Predictions for USD/CAD Exchange Rate Value.



Table 9: Evaluation metrics of all models on test dataset for the USD/CAD 1-week predictions.

USD/CAD 1-WEEK				
	LSTM	GRU	ARIMA	Moving Avg
MSE	8.420169e-04	4.656462e-04	1.667207e-04	6.090170e-04
MAE	0.023833	0.017048	0.010077	0.018162
MAPE	1.817861	1.307865	2.904946	2.674599

Table 9 above shows that GRU performs the best with 1.3 % MAPE, with LSTM at second. ARIMA and moving average performance is similar and much worse than GRU. Table 10 shows the distribution of the errors from the test dataset for 1-week:

Table 10: Error Distribution for all models for USD/CAD exchange rate 1-week predictions.

Error Description	LSTM	GRU	ARIMA	Moving Avg
Mean	0.011562	0.021872	0.0015	-0.001397
Std	0.019083	0.018233	0.021921	0.024654
Min	-0.020067	-0.046633	-0.049370	-0.086663
25 %	-0.009017	-0.000932	-0.008025	-0.012499
50 %	0.021575	0.010718	0.000119	0.002230
75 %	0.032986	0.024070	0.008051	0.013117
Max	0.090724	0.060917	0.038422	0.051077

### 5.1.3 2-Weeks Forecasts

Figure 7 below shows the results of all the 4 models' prediction for 2-weeks along with the actual exchange values:



Figure 7: 2-Weeks' Time Series Forecasting Prediction for USD/CAD Exchange Rate Value.

Table 11: Evaluation metrics of all models on test dataset for the USD/CAD 2-weeks predictions.

USD/CAD 2-WEEKS				
	LSTM	GRU	ARIMA	Moving Avg
MSE	9.464108e-04	8.235780e-04	2.931645e-04	6.174470e-04
MAE	0.0266	0.022296	0.01373	0.018493
MAPE	2.027219	1.703849	2.907438	2.69363

Table 11 shows that, just like for the 1-week predictions, GRU again performs the best with 1.7 % MAPE. LSTM comes at second, while ARIMA and moving average have similar MAPE.

Table 12 below shows the error distribution calculated from the predictions and actual values:

Table 12: Error Distribution for all models for USD/CAD exchange rate 2-weeks predictions.

Error Description	LSTM	GRU	ARIMA	Moving Avg
Std	0.018191	0.021106	0.017134	0.024796
Min	-0.029900	-0.026577	-0.045422	-0.085713
25 %	0.014141	0.004055	-0.011666	-0.013829
50 %	0.025845	0.016401	0.001345	0.001278
75 %	0.036330	0.032813	0.011374	0.012650
Max	0.078402	0.088656	0.042184	0.057567

## 5.2 USD/AUD Forecasts

### 5.2.1 1-Day Forecasts

Figure 8 below shows the results of all the 4 models' prediction on the test dataset for 1-day along with the actual exchange values:

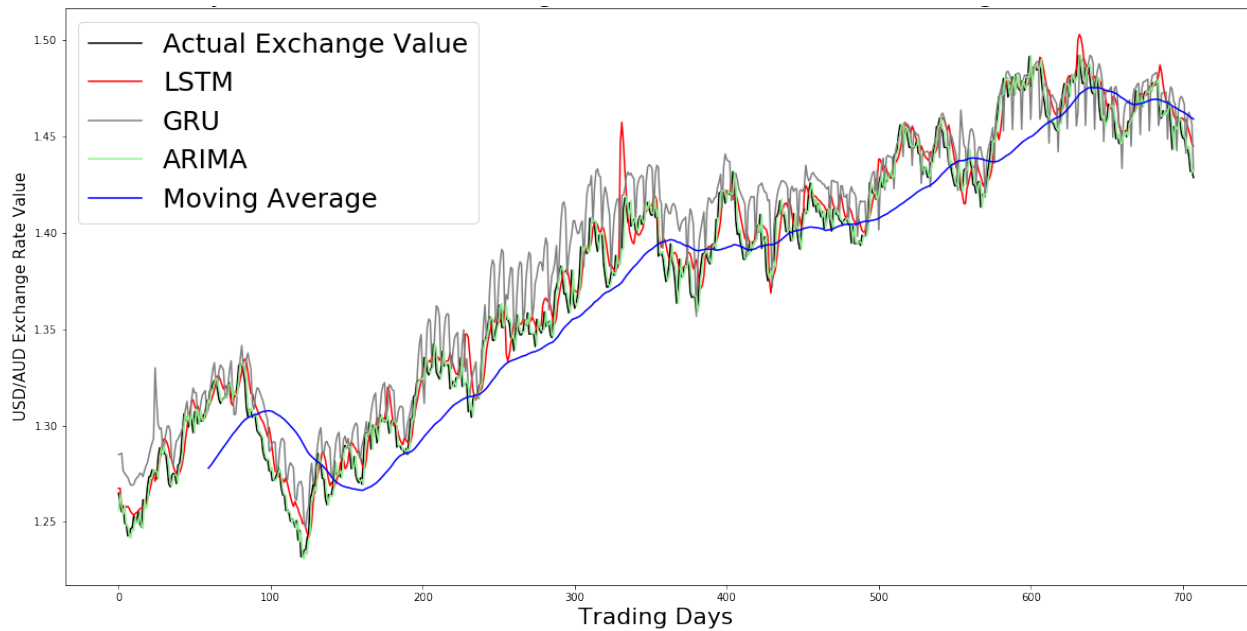


Figure 8: 1-Day Time Series Forecasting Prediction for USD/AUD Exchange Rate Value.

Table 13: Evaluation metrics of all models on test dataset for the USD/AUD 1-day predictions.

USD/AUD 1-DAY				
	LSTM	GRU	ARIMA	Moving Avg
MSE	1.118955e-04	3.710354e-04	3.811704e-05	4.452420e-04
MAE	0.008174	0.015922	0.004543	0.017224
MAPE	0.597375	1.167506	0.331319	5.337073

Table 13 above shows that ARIMA performs the best while LSTM is also close. GRU comes at third while the moving average performs bad. This is due to higher volatility in this exchange pair. Table 14 below depicts the error distribution for the models:

Table 14: Error Distribution for all models for USD/AUD exchange rate 1-day predictions.

Error Description	LSTM	GRU	ARIMA	Moving Avg
Std	0.009975	0.013850	0.006178	0.019693
Min	-0.065893	-0.053477	-0.024554	-0.060635
25 %	-0.009826	-0.022614	-0.003269	-0.004676
50 %	-0.003378	-0.013393	-0.000060	0.009449
75 %	0.002320	-0.004336	0.003412	0.023041
Max	0.027050	0.030605	0.025370	0.046180

## 5.2.2 1-Week Forecasts

Figure 9 below shows the results of all the 4 models' prediction for 1-week along with the actual exchange values:

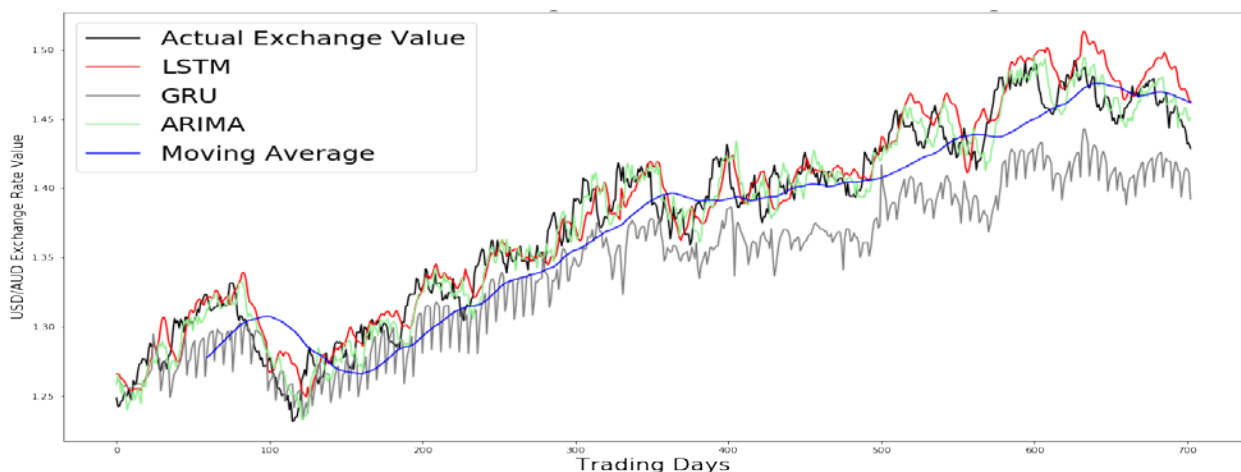


Figure 9: 1-Week Time Series Forecasting Prediction for USD/AUD Exchange Rate Value.

Based on the error percentage mean value of the GRU model, it looks like our GRU model is always under-estimating the exchange value. Thus, we can adjust our forecast by “adding” the mean of the error. Figure 10 below shows how our adjusted GRU model’s forecast after the change:

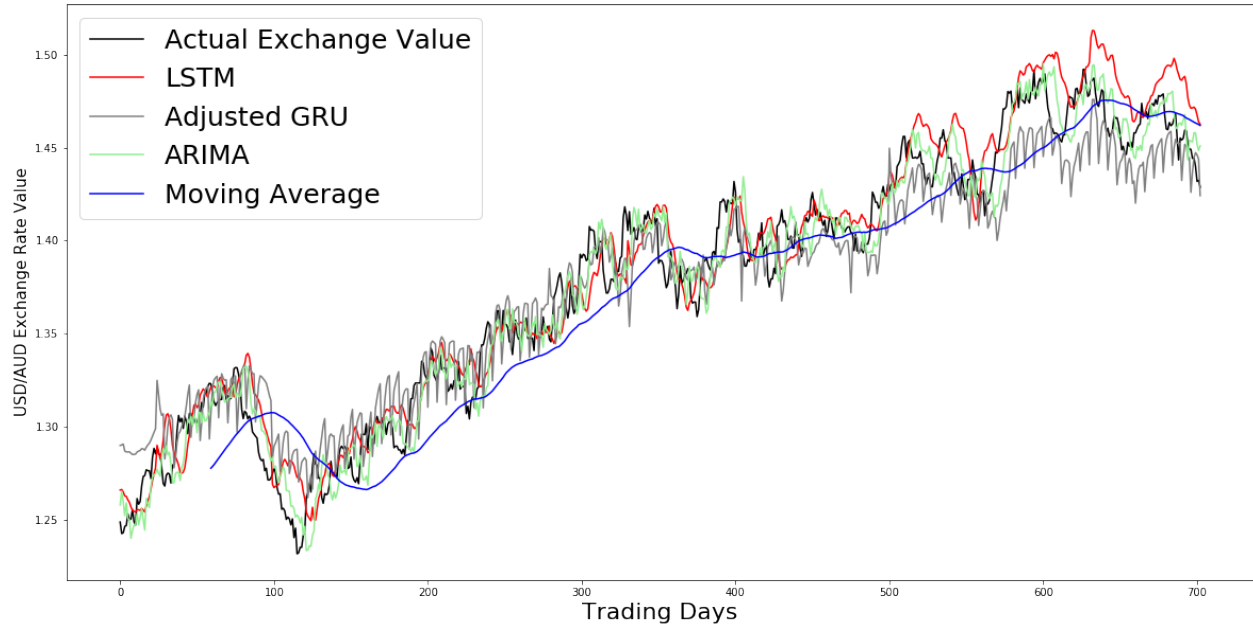


Figure 10: 1-Week Time Series Forecasting Prediction for USD/AUD Exchange Rate Value.

Table 15: Evaluation metrics of all models on test dataset for the USD/AUD exchange rate 1-week predictions.

USD/AUD 1-WEEK				
	LSTM	GRU	ARIMA	Moving Avg
MSE	3.369981e-04	1.535412e-03	2.196540e-04	5.364913e-04
MAE	0.014779	0.034062	0.011826	0.019014
MAPE	1.073506	2.428072	5.87329	5.32763

Table 15 shows that LSTM performs the best with 1.07 % MAPE. ARIMA model which had a 0.33 % MAPE for the 1-day predictions for the same exchange pair, now performs the worst among all other models.

Table 16: Error Distribution for all models for USD/AUD exchange rate 1-week predictions.

Error Description	LSTM	GRU	ARIMA	Moving Avg
Mean	-0.005868	0.032531	0.000366	0.008840
Std	0.017407	0.021859	0.014827	0.021480
Min	-0.045870	-0.027159	-0.042661	-0.066275
25 %	-0.018887	0.017238	-0.009675	-0.003902
50 %	-0.005827	0.034295	-0.000473	0.010178
75 %	0.005761	0.047836	0.010112	0.026499
Max	0.038789	0.092235	0.041348	0.049827

### 5.2.3 2-Weeks Forecasts

Figure 11 below shows the results of all the 4 models' prediction for 2-weeks along with the actual exchange values:

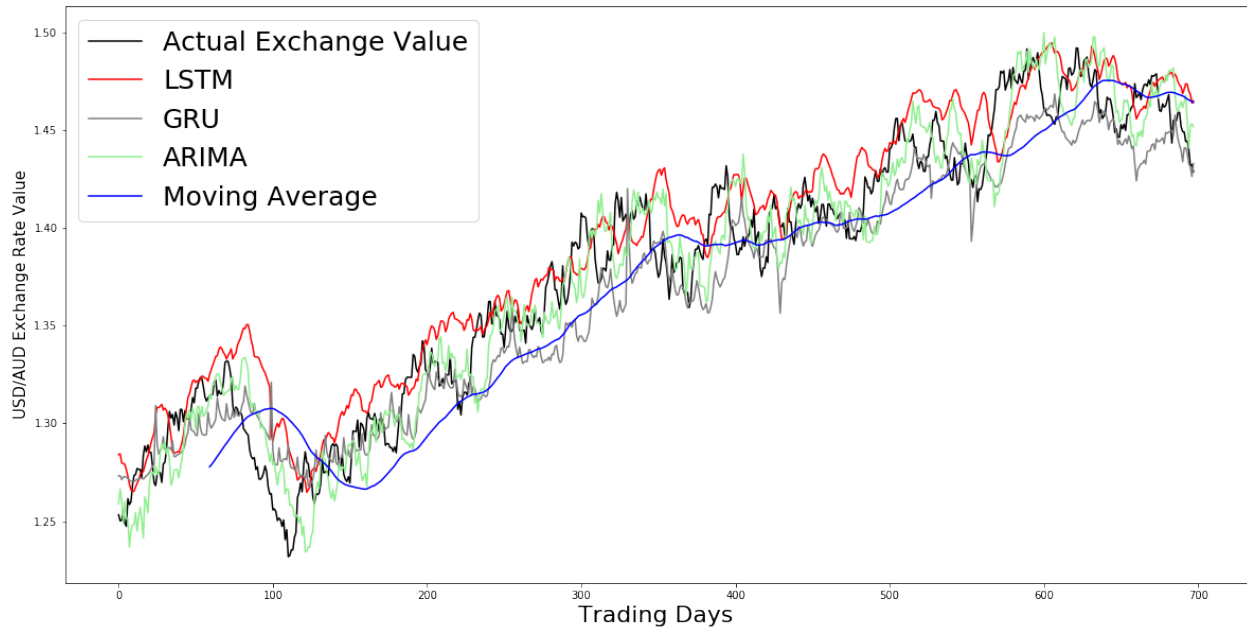


Figure 11: 2-Weeks' Time Series Forecasting Prediction for USD/AUD Exchange Rate Value.

Table 17: Evaluation metrics of all models on test dataset for the USD/AUD 2-weeks predictions.

USD/AUD 2-WEEKS				
	LSTM	GRU	ARIMA	Moving Avg
MSE	5.647630e-04	5.407281e-04	3.904476e-04	6.247325e-04
MAE	0.019266	0.018671	0.016136	0.02066
MAPE	1.4142	1.349716	5.85679	5.314694

Table 17 above shows that GRU and LSTM performance is similar and best amongst the 4 models. ARIMA and moving average perform similarly worse just like for 1-week predictions for the same exchange pair.

Table 18: Error Distribution for all models for USD/AUD 2-weeks predictions.

Error Description	LSTM	GRU	ARIMA	Moving Avg
Mean	-0.013778	0.011620	0.000679	0.010059
Std	0.019377	0.020156	0.019762	0.022923
Min	-0.061788	-0.057166	-0.044581	-0.070535
25 %	-0.027849	-0.001374	-0.013891	-0.004087
50 %	-0.013689	0.011999	-0.000776	0.012527
75 %	-0.000579	0.025114	0.015800	0.029267
Max	0.044337	0.064008	0.053705	0.052084

### 5.3 Uncertainty Modeling Example Results:

After obtaining the 25<sup>th</sup>, 50<sup>th</sup> and 75<sup>th</sup> % percentiles for each of the models, creating a metalog distribution of the errors and then ending up with 1000 trials of simulated errors, I obtain a SIP (data array) of the 1000 error possibilities. For the purposes of interpretability, I am only showing the USD/CAD 1-day predictions' average prediction and errors for the GRU model on past data. Figure 12 below shows the average prediction for each of the 7 days, the predicted interval (average prediction plus error distribution), the actual value and the chance of the predicted interval being either greater than or less than actual value:








Day	Average Prediction	Prediction with uncertainty added	Actual	Chance Greater than	Chance Lower than
day 1	1,371	 1,373490602	1,364924	72%	28%
day 2	1,3705	 1,372990602	1,364592	72%	28%
day 3	1,3708	 1,373290602	1,364369	73%	27%
day 4	1,3631	 1,365590602	1,364285	48%	52%
day 5	1,3614	 1,363890602	1,36297	46%	54%
day 6	1,3599	 1,362390602	1,361116	48%	53%
day 7	1,3604	 1,362890602	1,359086	57%	44%

Figure 12: Chance of prediction greater than or less than the actuals.

While we can check the chance for past day and our model's predictions with the addition of uncertainties, the more important example question to ask is "What is the chance the USD/CAD will be 1.33 tomorrow"? Figure 13 shows the 4 different models' average prediction, the predicted interval (average prediction plus error distribution), the target, chance of being greater than the target and chance of being less than the target:





Model Name	Average Prediction	Prediction with uncertainty added	Target	Chance Greater than	Chance Lower than
LSTM	1,2462	 1,248690602	1,25	39%	61%
GRU	1,2545	 1,256990602	1,25	69%	31%
ARIMA	1,2517	 1,254190602	1,25	58%	42%
MA	1,2455	 1,247990602	1,25	36%	64%

Figure 13: Chance of prediction being greater than or less than the target value.



## 6. DISCUSSION

I will first discuss results of the USD/CAD exchange pair. The results showed that ARIMA slightly performs better than LSTM, GRU and MA for 1-day predictions. For the same exchange pair, when I increased the time frame to 1 week and 2 weeks, ARIMA performed the worst. Moving average consistently performs with a similar mean absolute percentage error and comes at 3<sup>rd</sup> or 4<sup>th</sup> for all time periods. Also, looking at the predictions, ARIMA seems to copy past day, week, 2 weeks' values which is mainly due to the high autocorrelation between the lags of the time series data. This is not the case with the deep learning models which learn much better than ARIMA and MA when the period is increased. GRU performed the best for 1-week and 2-week predictions with a MAPE of 1.3 % and 1.7 % respectively. It is also of importance to point out that LSTM is also close in terms of performance to GRU for the 1-week and 2-week predictions with a MAPE of 1.8 % and 2.0 %.

I will now discuss the results of the USD/AUD exchange pair. Looking at the actual values, this pair is comparatively more volatile than the USD/CAD pair. The results for the USD/AUD had a similar picture as for the USD/CAD with some key differences. ARIMA performed the best for 1-day predictions with a MAPE of 0.3 %. However, when I increased the time period, ARIMA went from 0.3 % MAPE to 5.9 % which shows a significant difference in terms of performance. Moving average consistently performs the worst for this pair, mainly because it is not able to catch up the volatility in the test set especially. LSTM performed the best for the 1-week predictions with a MAPE of 1.1 % compared to GRU with a MAPE of 2.4 %. However, for the 2-week predictions, GRU performed the best with a MAPE of 1.3 % and LSTM with a MAPE of 1.4 %. This shows that there is some potential to improve the results for both the models with better hyperparameter tuning and feature selection.

Another important aspect of this research was the addition of uncertainties into our model's predictions. Firstly, just calculating the mean of the errors can show the bias of our model. For the USD/CAD pair for 1-week forecasts, the error mean of the GRU model was high. I used this knowledge to remove the bias of the model and the adjusted forecast performed much better. Secondly, using our models' error distribution on the test set of 3 years, I was able to answer two

important questions which were not answered before in the literature: “What is the chance our model’s predictions would have been greater than the actual past values?” and “What is the chance exchange rate is going to be greater/less than  $X$  in  $Y$  days?”

## 7. CONCLUSION

In conclusion, this research constituted 3 time periods, 2 currencies and 4 different time series forecasting models and thus 24 different combinations to evaluate for comparison. For the univariate time series forecasting models, I used Autoregressive Integrated Moving Average (ARIMA) and simple moving average (MA). For the deep learning models, I used LSTM and GRU which are well-known due to their ability to learn long sequences. Twelve macroeconomic indicators of each country and the past exchange rate values were used as features for LSTM and GRU. After performing upsampling, handling missing values, data transformation these features were used to build the LSTM and GRU model. For each period and exchange pair, the hyperparameters for these two models were tweaked to get the best performance. All 4 models were then tested for 1-day, 1-week and 2-weeks time periods and for USD/CAD and USD/AUD exchange rates separately.

Looking at the mean absolute percentage error for all the models, it can be concluded that ARIMA performs slightly better than LSTM and GRU for 1-day predictions regardless of the exchange pair. However, when the period is increased to 1 or 2-weeks, LSTM and GRU outperform ARIMA and Moving Average. This shows that the deep learning models do not have a strong dependance on the immediate past values and the macroeconomic indicators improve the overall performance of the model. While for USD/CAD pair, GRU performs better than LSTM, for USD/AUD pair LSTM performs slightly better than GRU. This also shows that there is a lot of potential in improving both these models using even better feature selection, transformation and hyperparameter tuning.

## **8. FUTURE WORK**

In terms of the future work, there are lots of possibilities. I considered only USD/CAD and USD/AUD exchange pairs which are relatively less volatile and have enough data on the Nasdaq Data Link website; however, the research can be expanded on more volatile exchange pairs and different combinations. For this research, I only considered macroeconomic indicators; however, adding technical indicators as features could be tried since more information can be helpful for the deep learning models learning. Due to time constraints, hyperparameters' tuning was done using trial and error method, however, a more systematic approach could achieve much better results for the deep learning models. The errors used for the error distribution could also be changed to percentage values for easy interpretability and comparison for the results section. Last but not the least is to increase/decrease the size of training and test dataset.

## APPENDIX

### Python Code for USD/CAD 1-day Predictions

To remove redundancy, only the code for USD/CAD for 1-day predictions is added in this appendix. To be able to recreate the models for USD/AUD and other time periods, hyperparameters' tuning section in methodologies chapter can be referred to.

`## Obtaining the Foreign Exchange Value Data from 1_Jan_2001-30_Dec_2019 PRE-COVID`

```
import tensorflow as tf
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import pandas as pd
import h5py
import os
import pandas_datareader as pdr
import datetime
import statistics
import math
from scipy import stats
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten, Reshape
from keras.layers import Conv1D, MaxPooling1D, LeakyReLU
from keras.utils import np_utils
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, LSTM
from statsmodels.tsa.arima_model import ARIMA
from keras.layers import GRU, CuDNNGRU
from keras.callbacks import CSVLogger, ModelCheckpoint
from keras.backend.tensorflow_backend import set_session
from sklearn.metrics import mean_squared_error
```

```

from sklearn.metrics import mean_absolute_error
from sklearn.preprocessing import MinMaxScaler
import quandl

quandl.ApiConfig.api_key = "uTCxVQUPFy8T8xF-9BHA"

# Main dataframe which is going to be the input for the deep learning models
df_main = pd.DataFrame()

# defining the currency indicator for easy reusability
currency_pair_1 = "SGE/CAN" #Check Nasdaq data link website for codes
currency_pair_2 = "SGE/USA"

# Obtaining the forex data using the API
fx_value = quandl.get("SGE/CANCUR", authtoken="uTCxVQUPFy8T8xF-9BHA",
start_date="2000-12-31", end_date="2019-12-30")
print("fx_value before scaling: ", fx_value.Value.describe())
forexScaler = MinMaxScaler(feature_range=(0, 1))
forexScaler.fit(fx_value)
print(forexScaler)
print("fx_value after scaling: ", fx_value.Value.describe())
df_main = fx_value

# Upsampling method
def upsample(df_to_upsample):
    df_to_upsample = pd.DataFrame(df_to_upsample)
    df_to_upsample = df_to_upsample.fillna(method='bfill')
    upsampled = df_to_upsample.resample("B").ffill()
    df_to_upsample.replace([np.inf, -np.inf], np.nan)
    df_to_upsample = pd.DataFrame(upsampled)
    df_to_upsample = df_to_upsample.fillna(method='bfill')
    return df_to_upsample

```

```

# Upsampling with percent changes instead of actual values
def upsample_Percent_change(df_to_upsample_percent):
    df_to_upsample_percent = pd.DataFrame(df_to_upsample_percent)
    df_to_upsample_percent = df_to_upsample_percent.fillna(method='bfill')
    upsampled = df_to_upsample_percent.resample("B").ffill()
    df_to_upsample_percent.replace([np.inf, -np.inf], np.nan)
    df_to_upsample_percent = pd.DataFrame(upsampled)
    df_to_upsample_percent = df_to_upsample_percent.pct_change()
    df_to_upsample_percent = df_to_upsample_percent.fillna(method='bfill')
    return df_to_upsample_percent

# Obtaining 12 Macroeconomic indicators for Canada:
interest_Rate_CANADA = upsample(quandl.get(currency_pair_1 + "IR",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
inflationRate_CANADA = upsample(quandl.get(currency_pair_1 + "CPIC",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
Imports_CANADA = upsample_Percent_change(quandl.get(currency_pair_1 + "IMVOL",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31", end_date="2019-12-30"))
Exports_CANADA = upsample_Percent_change(quandl.get(currency_pair_1 + "EXVOL",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
GDP_CANADA = upsample_Percent_change(quandl.get(currency_pair_1 + "G",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
ConsumerSpending_CANADA = upsample_Percent_change(quandl.get(currency_pair_1 +
"CSP", authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
UnemploymentRate_CANADA = upsample_Percent_change(quandl.get(currency_pair_1 +
"UNR", authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
ConsumerPricingIndex_CANADA = upsample_Percent_change(quandl.get(currency_pair_1 +
"CPI", authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
current_account_CANADA = upsample_Percent_change(quandl.get(currency_pair_1 + "CA",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))

```

```

foreign_exchange_reserves_CANADA = upsample_Percent_change(quandl.get(currency_pair_1
+ "FER", authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
foreign_direct_investment_CANADA = upsample_Percent_change(quandl.get(currency_pair_1
+ "FDI", authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
#govtdebt_GDP_CANADA = upsample_Percent_change(quandl.get(currency_pair_1 + "GDG",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))

```

```

# # Transforming Data for Canada

```

```

df_main["InterestRate_CANADA"] = interest_Rate_CANADA
df_main["InflationRate_CANADA"] = inflationRate_CANADA
df_main['Imports_CANADA'] = Imports_CANADA
df_main["Exports_CANADA"] = Exports_CANADA
df_main["GDP_CANADA"] = GDP_CANADA
df_main["ConsumerSpending_CANADA"] = ConsumerSpending_CANADA
df_main["UnemploymentRate_CANADA"] = UnemploymentRate_CANADA
df_main["ConsumerPricingIndex_CANADA"] = ConsumerPricingIndex_CANADA
df_main["current_account_CANADA"] = current_account_CANADA
df_main["foreign_exchange_reserves_CANADA"] = foreign_exchange_reserves_CANADA
df_main["foreign_direct_investment_CANADA"] = foreign_direct_investment_CANADA
#df_main["govtdebt_GDP_CANADA"] = govtdebt_GDP_CANADA
percent_missing = df_main.isnull().sum() * 100 / len(df_main)
missing_value_df = pd.DataFrame({'column_name': df_main.columns,
                                'percent_missing': percent_missing})
print(missing_value_df)
print(df_main.count())
print(df_main.shape)
# feature_range=(0, 1) BY DEFAULT
iRCANADAScaler = MinMaxScaler()
iFRCANADAScaler = MinMaxScaler()
iCANADAScaler = MinMaxScaler()

```



```
eCANADAScaler = MinMaxScaler()
gCANADAScaler = MinMaxScaler()
cSCANADAScaler = MinMaxScaler()
uRCANADAScaler = MinMaxScaler()
cCANADAScaler = MinMaxScaler()
caCANADAScaler = MinMaxScaler()
ferCANADAScaler = MinMaxScaler()
fdiCANADAScaler = MinMaxScaler()
#gdgCANADAScaler = MinMaxScaler()
```

```
iR_CANADA = np.array(df_main["InterestRate_CANADA"]).reshape(-1, 1)
iF_CANADA = np.array(df_main["InflationRate_CANADA"]).reshape(-1, 1)
imp_CANADA = np.array(df_main["Imports_CANADA"]).reshape(-1, 1)
exp_CANADA = np.array(df_main["Exports_CANADA"]).reshape(-1, 1)
gdp_2_CANADA = np.array(df_main["GDP_CANADA"]).reshape(-1, 1)
cS_CANADA = np.array(df_main["ConsumerSpending_CANADA"]).reshape(-1, 1)
uR_CANADA = np.array(df_main["UnemploymentRate_CANADA"]).reshape(-1, 1)
cpi_CANADA = np.array(df_main["ConsumerPricingIndex_CANADA"]).reshape(-1, 1)
ca_CANADA = np.array(df_main["current_account_CANADA"]).reshape(-1, 1)
fer_CANADA = np.array(df_main["foreign_exchange_reserves_CANADA"]).reshape(-1, 1)
fdi_CANADA = np.array(df_main["foreign_direct_investment_CANADA"]).reshape(-1, 1)
#gdg_CANADA = np.array(df_main["govtdebt_GDP_CANADA"]).reshape(-1, 1)
```

```
iRCANADAScaler.fit(iR_CANADA)
iFRCANADAScaler.fit(iF_CANADA)
iCANADAScaler.fit(imp_CANADA)
eCANADAScaler.fit(exp_CANADA)
gCANADAScaler.fit(gdp_2_CANADA)
cSCANADAScaler.fit(cS_CANADA)
uRCANADAScaler.fit(uR_CANADA)
cCANADAScaler.fit(cpi_CANADA)
```

```

caCANADAScaler.fit(ca_CANADA)
ferCANADAScaler.fit(fer_CANADA)
fdiCANADAScaler.fit(fdi_CANADA)
#gdgCANADAScaler.fit(gdg_CANADA)

df_main["InterestRate_CANADA"] = iRCANADAScaler.transform(iR_CANADA)
df_main["InflationRate_CANADA"] = iFRCANADAScaler.transform(iF_CANADA)
df_main["Imports_CANADA"] = iCANADAScaler.transform(imp_CANADA)
df_main["Exports_CANADA"] = eCANADAScaler.transform(exp_CANADA)
df_main["GDP_CANADA"] = gCANADAScaler.transform(gdp_2_CANADA)
df_main["ConsumerSpending_CANADA"] = cSCANADAScaler.transform(cS_CANADA)
df_main["UnemploymentRate_CANADA"] = uRCANADAScaler.transform(uR_CANADA)
df_main["ConsumerPricingIndex_CANADA"] = cCANADAScaler.transform(cpi_CANADA)
df_main["current_account_CANADA"] = caCANADAScaler.transform(ca_CANADA)
df_main["foreign_exchange_reserves_CANADA"] =
ferCANADAScaler.transform(fer_CANADA)
df_main["foreign_direct_investment_CANADA"] =
fdiCANADAScaler.transform(fdi_CANADA)
#df_main["govtdebt_GDP_CANADA"] = gdgCANADAScaler.transform(gdg_CANADA)
percent_missing = df_main.isnull().sum() * 100 / len(df_main)
missing_value_df = pd.DataFrame({'column_name': df_main.columns,
                                'percent_missing': percent_missing})
print(missing_value_df)

# # Obtaining 12 Macroeconomic indicators for USA
interest_Rate_USA = upsample(quandl.get(currency_pair_2 + "IR",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-3"))
inflation_Rate_USA = upsample(quandl.get(currency_pair_2 + "CPIC",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
imports_USA = upsample_Percent_change(quandl.get(currency_pair_2 + "IMVOL",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))

```

```

exports_USA = upsample_Percent_change(quandl.get(currency_pair_2 + "EXVOL",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
consumer_Spending_USA = upsample_Percent_change(quandl.get(currency_pair_2 + "CSP",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
unemployment_Rate_USA = upsample_Percent_change(quandl.get(currency_pair_2 + "UNR",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
gdp_USA = upsample_Percent_change(quandl.get(currency_pair_2 + "G",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
consumer_Price_Index_USA = upsample_Percent_change(quandl.get(currency_pair_2 + "CPI",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
current_account_USA = upsample_Percent_change(quandl.get(currency_pair_2 + "CA",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
foreign_exchange_reserves_USA = upsample_Percent_change(quandl.get(currency_pair_2 +
"FER", authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
foreign_direct_investment_USA = upsample_Percent_change(quandl.get(currency_pair_2 +
"FDI", authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))
#govtdebt_GDP_USA = upsample_Percent_change(quandl.get(currency_pair_2 + "GDG",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31"))

```

```

# # Transforming data for USA

```

```

df_main["interest_Rate_USA"] = interest_Rate_USA
df_main["inflation_Rate_USA"] = inflation_Rate_USA
df_main["imports_USA"] = imports_USA
df_main["exports_USA"] = exports_USA
df_main["consumer_Spending_USA"] = consumer_Spending_USA
df_main["unemployment_Rate_USA"] = unemployment_Rate_USA
df_main["gdp_USA"] = gdp_USA
df_main["consumer_Price_Index_USA"] = consumer_Price_Index_USA
df_main["current_account_USA"] = current_account_USA
df_main["foreign_exchange_reserves_USA"] = foreign_exchange_reserves_USA
df_main["foreign_direct_investment_USA"] = foreign_direct_investment_USA

```

```

#df_main["govtdebt_GDP_USA"] = govtdebt_GDP_USA
df_main = df_main.fillna(0)
# feature_range=(0, 1) By default
iR_USA_Scaler = MinMaxScaler()
iF_USA_Scaler = MinMaxScaler()
im_USA_Scaler = MinMaxScaler()
ex_USA_Scaler = MinMaxScaler()
cS_USA_Scaler = MinMaxScaler()
uR_USA_Scaler = MinMaxScaler()
g_USA_Scaler = MinMaxScaler()
c_USA_Scaler = MinMaxScaler()
ca_USA_Scaler = MinMaxScaler()
fer_USA_Scaler = MinMaxScaler()
fdi_USA_Scaler = MinMaxScaler()
#gdg_USA_Scaler = MinMaxScaler()

iR_USA = np.array(df_main["interest_Rate_USA"]).reshape(-1, 1)
iF_USA = np.array(df_main["inflation_Rate_USA"]).reshape(-1, 1)
imp_USA = np.array(df_main["imports_USA"]).reshape(-1, 1)
exp_USA = np.array(df_main["exports_USA"]).reshape(-1, 1)
cS_USA = np.array(df_main["consumer_Spending_USA"]).reshape(-1, 1)
uR_USA = np.array(df_main["unemployment_Rate_USA"]).reshape(-1, 1)
gdp_2_USA = np.array(df_main["gdp_USA"]).reshape(-1, 1)
cpi_USA = np.array(df_main["consumer_Price_Index_USA"]).reshape(-1, 1)
ca_USA = np.array(df_main["current_account_USA"]).reshape(-1, 1)
fer_USA = np.array(df_main["foreign_exchange_reserves_USA"]).reshape(-1, 1)
fdi_USA = np.array(df_main["foreign_direct_investment_USA"]).reshape(-1, 1)
#gdg_USA = np.array(df_main["govtdebt_GDP_USA"]).reshape(-1, 1)

iR_USA_Scaler.fit(iR_USA)
iF_USA_Scaler.fit(iF_USA)

```

```

im_USA_Scaler.fit(imp_USA)
ex_USA_Scaler.fit(exp_USA)
cS_USA_Scaler.fit(cS_USA)
uR_USA_Scaler.fit(uR_USA)
g_USA_Scaler.fit(gdp_2_USA)
c_USA_Scaler.fit(cpi_USA)
ca_USA_Scaler.fit(ca_USA)
fer_USA_Scaler.fit(fer_USA)
fdi_USA_Scaler.fit(fdi_USA)
##gdg_USA_Scaler.fit(gdg_USA)

df_main["interest_Rate_USA"] = iR_USA_Scaler.transform(iR_USA)
df_main["inflation_Rate_USA"] = iF_USA_Scaler.transform(iF_USA)
df_main["imports_USA"] = im_USA_Scaler.transform(imp_USA)
df_main["exports_USA"] = ex_USA_Scaler.transform(exp_USA)
df_main["consumer_Spending_USA"] = cS_USA_Scaler.transform(cS_USA)
df_main["unemployment_Rate_USA"] = uR_USA_Scaler.transform(uR_USA)
df_main["gdp_USA"] = g_USA_Scaler.transform(gdp_2_USA)
df_main["consumer_Price_Index_USA"] = c_USA_Scaler.transform(cpi_USA)
df_main["current_account_USA"] = ca_USA_Scaler.transform(ca_USA)
df_main["foreign_exchange_reserves_USA"] = fer_USA_Scaler.transform(fer_USA)
df_main["foreign_direct_investment_USA"] = fdi_USA_Scaler.transform(fdi_USA)
#df_main["govtdebt_GDP_USA"] = gdg_USA_Scaler.transform(gdg_USA)
df_main['Value'] = forexScaler.transform(quandl.get(currency_pair_1 + "CUR",
authtoken="uTCxVQUPFy8T8xF-9BHA", start_date="2000-12-31", end_date="2019-12-30"))

# # Setting the training size as 85 percent and testing size as 15 percent
train_set = np.array(df_main)
training_size = int(len(df_main) * 0.85)
print("training_size: ", training_size)

```

```

testing_size = len(df_main) - training_size
print("testing_size: ", testing_size)
training_data = train_set[0:training_size]
testing_data = train_set[: -training_size]
#training_data.describe()
# time_period length
time_period = 30
exchange_value = []
EV_to_Predict = []
for i in range(time_period, training_size):
    exchange_value.append(train_set[i - time_period:i]) # 0:29 , 1:30, 2:31
    EV_to_Predict.append(train_set[i, 0]) # 30

exchange_value, EV_to_Predict = np.array(exchange_value), np.array(EV_to_Predict)
print("exchange_value.shape: ",exchange_value.shape)
print("EV_to_Predict.shape: ",EV_to_Predict.shape)
input_shape=(exchange_value.shape[1], 23)
print("input_shape=(exchange_value.shape[1], 23): ", input_shape)
exchange_value = np.reshape(exchange_value, (exchange_value.shape[0],
exchange_value.shape[1], 23))
print("exchange_value.shape: ",exchange_value.shape)
inputs = np.array(df_main[training_size:])
print("inputs size ", inputs.shape)
print(exchange_value.shape)
print(EV_to_Predict.shape)

# # Building the LSTM Model
epochs= 100
batch_size = 30
validation_split = 0.10
model = Sequential()

```

```

model.add(LSTM(units=40, input_shape=(exchange_value.shape[1], 23)))
model.add(Dropout(0.15))
model.add(LSTM(units=40))
model.add(Dense(units = 1))
model.compile(optimizer = 'adam', loss = 'mean_squared_error')
print("MODEL COMPILATION COMPLETED")
model_history = model.fit(exchange_value, EV_to_Predict, epochs =epochs, batch_size =
batch_size, validation_split=validation_split)
#Plotting the training and validation losses for easy interpretability
plt.plot(model_history.history['loss'], color='green')
plt.plot(model_history.history['val_loss'], color = 'blue')
plt.title('Taining & Validation loss for the LSTM Model', fontsize=28)
plt.xlabel('Epochs', fontsize=19)
plt.ylabel('Loss', fontsize=19)
plt.legend(['Training', 'Validation'], loc='upper right')
plt.show()

# # TESTING LSTM PERFORMANCE
inputs = np.array(df_main[training_size:])
print("inputs size ", inputs.shape)
Array_test = []
Test_EV_to_Predict = []
for i in range(time_period, testing_size):
    Array_test.append(inputs[i - time_period:i])
    Test_EV_to_Predict.append(inputs[i, 0])
Array_test = np.array(Array_test)
print(Array_test.shape[0])
Array_test = np.reshape(Array_test, (Array_test.shape[0], Array_test.shape[1], 23))
predicted_normalized_exchange_value = model.predict(Array_test)
predicted_exchange_value =
forexScaler.inverse_transform(predicted_normalized_exchange_value)

```

```

buffer = training_size + time_period
fx_value = quandl.get(currency_pair_2 + "CUR", authtoken="uTCxVQUPFy8T8xF-9BHA",
start_date="2000-12-31", end_date="2019-12-30")
test_exchange_value = np.array(fx_value[buffer:])
real_exchange_value = np.array(fx_value)
print("test_exchange_value: ",test_exchange_value.shape)
print("real_exchange_value: ",real_exchange_value.shape)

```

# # Building the GRU Model

```

units= 50
batch_size = 25
epochs = 100
output_size=1
step_size = 60
num_features= 23
model_GRU = Sequential()
model_GRU.add(GRU(units=units,
input_shape=(step_size,num_features),return_sequences=True))
model_GRU.add(Activation('relu'))
model_GRU.add(GRU(units=units,return_sequences=False))
model_GRU.add(Activation('relu'))
model_GRU.add(Dropout(0.2))
model_GRU.add(Dense(64))
model_GRU.add(Activation('relu'))
model_GRU.add(Dropout(0.2))
model_GRU.add(Dense(output_size))
model_GRU.add(Activation('relu'))
model_GRU.compile(loss='mse', optimizer='adam')
print(len(exchange_value))
print(len(EV_to_Predict))

```



```

history_GRU = model_GRU.fit(exchange_value, EV_to_Predict,
batch_size=150,validation_split=0.10, epochs = epochs)
#Plotting the training and validation errors
plt.plot(history_GRU.history['loss'], color='green')
plt.plot(history_GRU.history['val_loss'], color = 'blue')
plt.title('Training & Validation loss for GRU', fontsize=28)
plt.xlabel('Epochs', fontsize=19)
plt.ylabel('Loss', fontsize=19)
plt.legend(['Training', 'Validation'], loc='upper right')
plt.show()

```

**## Testing GRU PERFORMANCE**

```

inputs = np.array(df_main[training_size:])
print("inputs size ", inputs.shape)
Array_test = []
GRU_Test_EV_to_Predict = []
for i in range(time_period, testing_size):
    Array_test.append(inputs[i - time_period:i])
    GRU_Test_EV_to_Predict.append(inputs[i, 0])
Array_test = np.array(Array_test)
Array_test = np.reshape(Array_test, (Array_test.shape[0], Array_test.shape[1], 23))
predicted_normalized_exchange_value_GRU = model_GRU.predict(Array_test)
predicted_exchange_value_GRU =
forexScaler.inverse_transform(predicted_normalized_exchange_value_GRU)
buffer = training_size + time_period
fx_value = quandl.get("SGE/CANCUR", authtoken="uTCxVQUPFy8T8xF-9BHA",
start_date="2000-12-31", end_date="2019-12-30")
test_exchange_value = np.array(fx_value[buffer:])
real_exchange_value = np.array(fx_value)

```

```

## UNI VARIATE TIME SERIES FORECASTING MODEL: ARIMA
def arima(Truth_vals, P, D, Q):
    arima_model = ARIMA(Truth_vals, order=(P, D, Q))
    arima_model_fitted = arima_model.fit(dispatch=0)
    Prediction_arima = arima_model_fitted.forecast()
    return Prediction_arima[0]
predicted_arima = []
Truth_vals = [x for x in train_set[0:training_size + time_period, 0]]
Truth_vals = np.array(real_exchange_value[:buffer])
for i in range(buffer, len(fx_value)):
    Truth_Value = real_exchange_value[i]
    Prediction_arima = arima(Truth_vals, 2, 2, 1)
    print('index = %f, Truth value = %f, Predicted value = %f, Error = %f' % (i, Truth_Value,
    Prediction_arima, Truth_Value - Prediction_arima))
    predicted_arima.append(Prediction_arima)
    Truth_vals = np.append(Truth_vals, Truth_Value)
arima_predictions = pd.DataFrame(predicted_arima)

## Moving Average Univariate Time Series Forecasting Model and plots
get_ipython().run_line_magic('matplotlib', 'inline')
priceDataframe = pd.DataFrame(test_exchange_value)
ma = priceDataframe.iloc[:,0].rolling(window=time_period).mean()
#Plotting all the predictions on a single graph for comparison
plt.rcParams["figure.figsize"] = (19,9)
plt.plot(test_exchange_value, color = 'black')
plt.plot(predicted_exchange_value, color = 'red')
plt.plot(predicted_exchange_value_GRU, color='gray', )
plt.plot(predicted_arima, color = 'lightgreen')
plt.plot(ma, color= 'blue')

```

```

plt.title('1 Day Time Series Forecasting Prediction for USD/CAD Exchange Rate Value',
fontSize=25)
plt.xlabel('Trading Days', fontSize=22)
plt.ylabel('USD/CAD Exchange Rate Value', fontSize=15)
plt.legend(['Actual Exchange Value', "LSTM", "GRU", "ARIMA", "Moving Average"],
loc='best', fontSize=25)
plt.show()
ma_np = np.array(ma[time_period:])
ma_np = ma_np.tolist()
print("ma np: ",len(ma_np) )
print(ma[time_period:].shape)
test_exchange_value[time_period:].shape

```

**# # COMPARATIVE ANALYSIS USING 3 DIFFERENT PERFORMANCE METRICS**

**# Defining mean absolute percentage error**

```

def mean_absolute_percentage_error(truth_val, predict_val):
    return np.mean(np.abs((truth_val - predict_val) / truth_val)) * 100
print(predicted_exchange_value, test_exchange_value)

```

**# LSTM Metrics**

```

lstm_MSE = mean_squared_error(test_exchange_value, predicted_exchange_value)
lstm_MSE="{:e}".format(lstm_MSE)
lstm_MAE = round(mean_absolute_error(test_exchange_value, predicted_exchange_value), 6)
lstm_MAPE = round(mean_absolute_percentage_error(test_exchange_value,
predicted_exchange_value), 6)

```

```

print("\nLSTM:::")
print("LSTM Mean_Squared_Error: ", lstm_MSE)
print("LSTM Mean Absolute Error: ", lstm_MAE)
print("LSTM Mean Absolute Percentage Error: ", lstm_MAPE)

```

```

# GRU Metrics
GRU_MSE = mean_squared_error(test_exchange_value, predicted_exchange_value_GRU)
GRU_MSE="{:e}".format(GRU_MSE)
GRU_MAE = round(mean_absolute_error(test_exchange_value,
predicted_exchange_value_GRU), 6)
GRU_MAPE = round(mean_absolute_percentage_error(test_exchange_value,
predicted_exchange_value_GRU), 6)

print("\nGRU:: ")
print("GRU Mean_Squared_Error: ", GRU_MSE)
print("GRU Mean Absolute Error: ", GRU_MAE)
print("GRU Mean Absolute Percentage Error: ", GRU_MAPE)

#ARIMA Metrics
arima_MSE = mean_squared_error(test_exchange_value, predicted_arima)
arima_MSE="{:e}".format(arima_MSE)
arima_MAE = round(mean_absolute_error(test_exchange_value, predicted_arima), 6)
arima_MAPE = round(mean_absolute_percentage_error(test_exchange_value, predicted_arima),
6)
print("\nARIMA")
print("ARIMA Mean_Squared_Error: ", arima_MSE)
print("ARIMA Mean Absolute Error: ", arima_MAE)
print("ARIMA Mean Absolute Percentage Error:: ", arima_MAPE)

# Moving Average Metrics
ma_MSE = mean_squared_error(test_exchange_value[time_period:], ma[time_period:])
ma_MSE="{:e}".format(ma_MSE)
ma_MAE = round(mean_absolute_error(test_exchange_value[time_period:], ma[time_period:]),
6)
ma_MAPE = round(mean_absolute_percentage_error(test_exchange_value[time_period:],
ma_np[time_period:]), 6)

```

```

print("\n Moving Average")
print("MA Mean_Squared_Error: ", ma_MSE)
print("MA Mean Absolute Error: ", ma_MAE)
print("MA Mean Absolute Percentage Error: ", ma_MAPE)

# # ERROR DISTRUBUTION FOR Moving Average
k_ma = np.abs((test_exchange_value - ma_np))
df_ma = pd.DataFrame(test_exchange_value, columns = ['actual'])
df_ma['predicts'] = ma
df_ma.head(10)
plt.figure(figsize=(12, 4))
plt.subplot(121)
df_ma['error'] = df_ma.actual - df_ma.predicts
df_ma['error_percent'] = (df_ma.actual - df_ma.predicts)*100/df_ma.actual
print("error percent mean value:", df_ma.error_percent.mean())
df_ma['adjusted_forecast'] = df_ma.predicts*((df_ma.error_percent.mean()/100)+1)
df_ma['error_adjusted_forecast'] = df_ma.actual - df_ma.adjusted_forecast
df_ma['error_adjusted_forecast_percent'] = 100*(df_ma.actual -
df_ma.adjusted_forecast)/df_ma.actual
print("errors description:", df_ma.error.describe())
print("error percent description:", df_ma.error_percent.describe())
print("error error_adjusted_forecast description:", df_ma.error_adjusted_forecast.describe())
print("error error_adjusted_forecast percent description:",
df_ma.error_adjusted_forecast_percent.describe())
df_ma.error.hist(bins=50).set_title('Error distribution for Moving Average')
plt.subplot(122)
df_ma.error.plot(kind='box', grid=True).set_title("Error for Moving Average")

# # ERROR DISTRUBUTION FOR ARIMA

```

```

k_arima = np.abs((test_exchange_value - arima_predictions))
df_arima = pd.DataFrame(test_exchange_value, columns = ['actual'])
df_arima['predicts'] = arima_predictions
plt.figure(figsize=(12, 4))
plt.subplot(121)
#df = pd.DataFrame({'actual': actual, 'predicts': predicts})
df_arima['error'] = df_arima.actual - df_arima.predicts
df_arima['error_percent'] = (df_arima.actual - df_arima.predicts)*100/df_arima.actual
df_arima['adjusted_forecast'] = df_arima.predicts*((df_arima.error_percent.mean()/100)+1)
df_arima['error_adjusted_forecast'] = df_arima.actual - df_arima.adjusted_forecast
df_arima['error_adjusted_forecast_percent'] = 100*(df_arima.actual -
df_arima.adjusted_forecast)/df_arima.actual
print("errors description:", df_arima.error.describe())
print("error percent description:", df_arima.error_percent.describe())
print("error error_adjusted_forecast description:", df_arima.error_adjusted_forecast.describe())
print("error error_adjusted_forecast percent description:",
df_arima.error_adjusted_forecast_percent.describe())
df_arima.error.hist(bins=50).set_title('Error distribution for ARIMA')
plt.subplot(122)
df_arima.error.plot(kind='box', grid=True).set_title("Error for ARIMA")

```

## ## ERROR DISTRUBUTION FOR LSTM

```

k = np.abs((test_exchange_value - predicted_exchange_value))
df_lstm = pd.DataFrame(test_exchange_value, columns = ['actual'])
df_lstm['predicts'] = predicted_exchange_value
plt.figure(figsize=(12, 4))
plt.subplot(121)
df_lstm['error'] = df_lstm.actual - df_lstm.predicts
df_lstm['error_percent'] = (df_lstm.actual - df_lstm.predicts)*100/df_lstm.actual
df_lstm['adjusted_forecast'] = df_lstm.predicts*((df_lstm.error_percent.mean()/100)+1)

```

```

df_lstm['error_adjusted_forecast'] = df_lstm.actual - df_lstm.adjusted_forecast
df_lstm['error_adjusted_forecast_percent'] = 100*(df_lstm.actual -
df_lstm.adjusted_forecast)/df_lstm.actual
print("errors description:", df_lstm.error.describe())
print("error percent description:", df_lstm.error_percent.describe())
print("error error_adjusted_forecast description:", df_lstm.error_adjusted_forecast.describe())
print("error error_adjusted_forecast percent description:",
df_lstm.error_adjusted_forecast_percent.describe())
df_lstm.error.hist(bins=50).set_title('Error distribution for LSTM')
plt.subplot(122)
df_lstm.error.plot(kind='box', grid=True).set_title("Error for LSTM")

```

**## ERROR DISTRUBUTION FOR GRU**

```

k = np.abs((test_exchange_value - predicted_exchange_value_GRU))
df_gru = pd.DataFrame(test_exchange_value, columns = ['actual'])
df_gru['predicts'] = predicted_exchange_value_GRU
gru_with_date = quandl.get("SGE/CANCUR", authtoken="uTCxVQUPFy8T8xF-9BHA",
start_date="2000-12-31", end_date="2019-12-30")
gru_with_date = gru_with_date[-testing_size:]
actuals = df_gru.actual
predictions = df_gru.predicts
plt.figure(figsize=(12, 4))
plt.subplot(121)
df_gru['error'] = df_gru.actual - df_gru.predicts
df_gru['error_percent'] = (df_gru.actual - df_gru.predicts)*100/df_gru.actual
df_gru['adjusted_forecast'] = df_gru.predicts*((df_gru.error_percent.mean()/100)+1)
df_gru['error_adjusted_forecast'] = df_gru.actual - df_gru.adjusted_forecast
df_gru['error_adjusted_forecast_percent'] = 100*(df_gru.actual -
df_gru.adjusted_forecast)/df_gru.actual
print("errors description:", df_gru.error.describe())

```

```
print("error percent description:", df_gru.error_percent.describe())
print("error error_adjusted_forecast description:", df_gru.error_adjusted_forecast.describe())
print("error error_adjusted_forecast percent description:",
df_gru.error_adjusted_forecast_percent.describe())
df_gru.error.hist(bins=50).set_title('Error distribution for GRU')
plt.subplot(122)
df_gru.error.plot(kind='box', grid=True).set_title("Error for GRU")
```



## REFERENCES

- [1] J. Chen, "Learn about trading FX with this beginner's Guide to Forex Trading," *Investopedia*, 27-Nov-2020. [Online]. Available: <https://www.investopedia.com/articles/forex/11/why-trade-forex.asp>. [Accessed: 01-Dec-2021].
- [2] T. L. Caraway, S. J. Rickard, and M. S. Anner, "International negotiations and Domestic Politics: The case of IMF labor market conditionality: International organization," Cambridge Core, 30-Jan-2012. [Online]. Available: <https://www.cambridge.org/core/journals/international-organization/article/international-negotiations-and-domestic-politics-the-case-of-imf-labor-market-conditionality/309E9AC686DE77C9E6C95EEB7C2213E7>. [Accessed: 20-Dec-2020].
- [3] J. Fernando, "Understanding the law of Supply and Demand," *Investopedia*, 20-Nov-2021. [Online]. Available: <https://www.investopedia.com/terms/l/law-of-supply-demand.asp>. [Accessed: 20-Dec-2021].
- [4] Z. Zeng and M. Khushi, "Wavelet denoising and attention-based RNN- Arima model to predict forex price," *IEEE Xplore*, 28-Sep-2020. [Online]. Available: <https://ieeexplore.ieee.org/document/9206832>. [Accessed: 01-Dec-2021].
- [5] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the Arima model," *IEEE Xplore*, 23-Feb-2015. [Online]. Available: <https://ieeexplore.ieee.org/document/7046047>. [Accessed: 01-Dec-2021].
- [6] O. Chantarakasemchit, S. Nuchitprasitchai, and Y. Nilsiam, "Forex rates prediction on EUR/USD with simple moving average technique and financial factors," *IEEE Xplore*, 04-Aug-2020. [Online]. Available: <https://ieeexplore.ieee.org/document/9157907>. [Accessed: 01-Dec-2021].
- [7] S. Nootyaskool and W. Choengtong, "Hidden markov models predict foreign exchange rate," *IEEE Xplore*, 19-Jan-2015. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/7011878>. [Accessed: 01-Dec-2021].
- [8] D. Aguilar, I. Batyrshin, and O. Pogrebnyak, "A survey on computer science techniques in the forex market: Models and applications: Semantic scholar," *Semantic Scholar*, 01-Dec-2017. [Online]. Available: <https://www.semanticscholar.org/paper/A-Survey-on-Computer-Science-Techniques-in-the-and-Aguilar-Batyrshin/b6d86b3722b2026feed5f8ecad7a1eafa1893ddd>. [Accessed: 01-Dec-2021].
- [9] P. Oncharoen and P. Vateekul, "Deep learning for stock market prediction using event embedding and technical indicators," *IEEE Xplore*, 22-Nov-2018. [Online]. Available: <https://ieeexplore.ieee.org/document/8541310>. [Accessed: 01-Dec-2021].

- [10] D. D. C. KADILAR, P. D. M. ŞİMŞEK, and A. G. Ç. H. ALADAĞ, “Forecasting the exchange rate series with ann: The case of Turkey,” *Istanbul University Econometrics and Statistics e-Journal*, 05-Sep-2011. [Online]. Available: <https://dergipark.org.tr/en/pub/iuekois/issue/8991/112073>. [Accessed: 15-Nov-2021].
- [11] J. Kamruzzaman and R. A. Sarker, “Forecasting of currency exchange rates using ANN: A case study,” *IEEE Xplore*, 14-Nov-2010. [Online]. Available: <https://ieeexplore.ieee.org/document/1279395/>. [Accessed: 29-Nov-2021].
- [12] M. P. Naeini, H. Taremian, and H. B. Hashemi, “Stock market value prediction using neural networks,” *IEEE Xplore*, 22-Nov-2010. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/5643675>. [Accessed: 29-Nov-2021].
- [13] N. Patil, S. Masih, J. Rumao, and V. Gaurea, “Predict Foreign Currency Exchange Rates Using Machine Learning,” *SpringerLink*, 19-Oct-2021. [Online]. Available: [https://link.springer.com/chapter/10.1007/978-981-16-4641-6\\_19](https://link.springer.com/chapter/10.1007/978-981-16-4641-6_19). [Accessed: 29-Nov-2021].
- [14] A. Dingli and K. S. Fournier, “Financial Time Series Forecasting – A Deep Learning Approach,” *International Journal of Machine Learning and Computing*, 05-Oct-2017. [Online]. Available: <http://www.ijmlc.org/vol7/632-P17.pdf>. [Accessed: 29-Nov-2021].
- [15] N. Ahmed, A. Atiya, N. El Gayar, and H. El-Shishiny, “An empirical comparison of machine learning models for time series forecasting,” *Research Gate*, Aug-2010. [Online]. Available: [https://www.researchgate.net/publication/227612766\\_An\\_Empirical\\_Comparison\\_of\\_Machine\\_Learning\\_Models\\_for\\_Time\\_Series\\_Forecasting](https://www.researchgate.net/publication/227612766_An_Empirical_Comparison_of_Machine_Learning_Models_for_Time_Series_Forecasting). [Accessed: 29-Nov-2021].
- [16] G. Papacharalampous, H. Tyralis, and D. Koutsoyiannis, “Univariate Time Series Forecasting of Temperature and Precipitation with a Focus on Machine Learning Algorithms: a Multiple-Case Study from Greece,” *Research Gate*, 01-Dec-2018. [Online]. Available: [https://www.researchgate.net/publication/329279811\\_Univariate\\_Time\\_Series\\_Forecasting\\_of\\_Temperature\\_and\\_Precipitation\\_with\\_a\\_Focus\\_on\\_Machine\\_Learning\\_Algorithms\\_a\\_Multiple-Case\\_Study\\_from\\_Greece](https://www.researchgate.net/publication/329279811_Univariate_Time_Series_Forecasting_of_Temperature_and_Precipitation_with_a_Focus_on_Machine_Learning_Algorithms_a_Multiple-Case_Study_from_Greece). [Accessed: 29-Nov-2021].
- [17] Sudimanto, Y. Heryadi, Lukas, and A. Wibowo, “Foreign Exchange Prediction Using Machine Learning Approach: A pilot study,” *IEEE Xplore*, 20-Oct-2021. [Online]. Available: <https://ieeexplore.ieee.org/document/9563998>. [Accessed: 29-Nov-2021].
- [18] I. E. Livieris, E. Pintelas, and P. Pintelas, “A CNN–LSTM model for gold price time-series forecasting,” *Neural Computing and Applications*, 13-Apr-2020. [Online]. Available: <https://link.springer.com/article/10.1007/s00521-020-04867-x>. [Accessed: 29-Nov-2021].

- [19] A. Twin, “6 factors that influence exchange rates,” *Investopedia*, 28-Jul-2021. [Online]. Available: <https://www.investopedia.com/trading/factors-influence-exchange-rates/>. [Accessed: 20-Dec-2020].
- [20] A. Mittal, “Understanding RNN and LSTM,” Medium, 12-Oct-2019. [Online]. Available: <https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e>. [Accessed: 01-Dec-2021].
- [21] M. Phi, “Illustrated guide to LSTM's and GRU's: A step by step explanation,” Medium, 28-Jun-2020. [Online]. Available: <https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>. [Accessed: 29-Nov-2021].
- [22] S. Saxena, “Gated recurrent unit: Introduction to Gated Recurrent Unit(GRU),” Analytics Vidhya, 18-Mar-2021. [Online]. Available: <https://www.analyticsvidhya.com/blog/2021/03/introduction-to-gated-recurrent-unit-gru/>. [Accessed: 29-Nov-2021].
- [23] S. Prabhakaran, “Arima model - complete guide to time series forecasting in python: ML+,” Machine Learning Plus, 22-Nov-2021. [Online]. Available: <https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/>. [Accessed: 29-Nov-2021].
- [24] C. Mitchell, “How to use a moving average to buy stocks,” Investopedia, 28-Apr-2021. [Online]. Available: <https://www.investopedia.com/articles/active-trading/052014/how-use-moving-average-buy-stocks.asp>. [Accessed: 29-Nov-2021].
- [25] M. Aien, A. Hajebrahimi, and M. Fotuhi-Firuzabad, “A comprehensive review on uncertainty modeling techniques in power system studies,” *Renewable and Sustainable Energy Reviews*, 08-Jan-2016. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1364032115014537>. [Accessed: 29-Nov-2021].
- [26] S. L. Savage, “Chancification,” *Probability Management*, 14-May-2021. [Online]. Available: <https://www.probabilitymanagement.org/blog/2021/5/14/chancification>. [Accessed: 29-Nov-2021].
- [27] A. Thomas, “API for Economic Data,” Nasdaq Data Link Blog, 30-Aug-2020. [Online]. Available: <https://blog.data.nasdaq.com/api-for-economic-data>. [Accessed: 29-Nov-2021].
- [28] K. Eckhardt, “Choosing the right hyperparameters for a simple LSTM using keras,” Medium, 29-Nov-2018. [Online]. Available: <https://towardsdatascience.com/choosing-the-right-hyperparameters-for-a-simple-lstm-using-keras-f8e9ed76f046>. [Accessed: 29-Nov-2021].