EXPLORING THE EFFECTIVENESS OF TRADITIONAL TIME SERIES MODELS AND MACHINE LEARNING ALGORITHMS FOR PRICE FORECASTING

By

Muhammad Naeem

Department of Statistics

Faculty of Natural Sciences

Quaid-i-Azam University, Islamabad

2023

بسم الله الرحمن الرحيم

In the Name of Allah The Most Merciful and The Most Beneficent

EXPLORING THE EFFECTIVENESS OF TRADITIONAL TIME SERIES MODELS AND MACHINE LEARNING ALGORITHMS FOR PRICE FORECASTING

By

Muhammad Naeem

A THESIS SUBMITTED IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF PHILOSOPHY IN STATISTICS

Supervised By

Dr. Ismail Shah

Department of Statistics

Faculty of Natural Sciences

Quaid-i-Azam University, Islamabad

2023

Declaration

I "Muhammad Naem" hereby solemnly declare that this thesis titled, "Exploring the effectiveness of traditional time series models and machine learning algorithms for price forecasting".

- This work was done wholly in candidature for a degree of M.Phil Statistics at this University.
- Where I got help from the published work of others, this is always clearly stated.
- Where I have quoted from the work of others, the source is always mentioned. Except of such quotations, this thesis is entirely my own research work.
- Where the thesis is based on work done by myself jointly with my supervisor, I have made clear exactly what was done by others and what I have suggested

Dated:____________________ Signature:____________________

CERTIFICATE

Exploring the effectiveness of traditional time series models and machine learning algorithms for price forecasting

by

Muhammad Naeem

(Reg.No. 02222113005)

A THESIS SUBMITTED IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF M.PHIL. IN

STATISTICS

We accept this thesis as conforming to the required standards

1. [illegible]
Dr. Ismail Shah
(Supervisor)

2. [illegible]
Prof. Dr. Tahir Mehmood
(External Examiner)

3. [illegible] 19/9/23
Prof. Dr. Ijaz Hussain
(Chairman)

DEPARTMENT OF STATISTICS
QUAID-I-AZAM UNIVERSITY
ISLAMABAD, PAKISTAN
2023

Dedication

I am feeling great honor and pleasure to dedicate this research work to

My Parents and Family

Whose endless affection, prayers and wishes have been a great source of comfort for me during my whole education period and my life

Acknowledgments

All praises to Almighty Allah (SWT), the light of heavens and earth, who has bestowed upon me the ability to embark on this research journey and guided me with His blessings throughout. Best of praises and peace be upon all the sacred messengers, especially the last of them, Hazrat Muhammad (SAWW), who illuminate the path of knowledge for all of humanity.

It fills me with great honor and gratitude to express my heartfelt appreciation to my supervisor, Dr. Ismail Shah, for his exceptional supervision and unwavering support. His invaluable guidance, constructive comments, and suggestions have played a vital role in the success of this research. I am truly grateful for his constant encouragement and belief in my capabilities.

I would also like to extend my sincere gratitude to the Chairman, Department of Statistics Prof. Dr. Ijaz Hussain, and the esteemed teachers Prof. Dr. Javed Shabbir, Dr. Abdul Haq, Dr. Sajid Ali, and Dr. Manzoor Khan for their enlightening teachings and unwavering support throughout my academic journey. their knowledge and mentorship have been instrumental in shaping my understanding and growth.

To my beloved father, Mr. Muhammad Ibrahim, I extend my deepest thanks. His invaluable advice, both in research and in my career, has been a guiding light. I am truly blessed to have your unwavering support and wisdom in every aspect of my life.

Furthermore, I would like to express my sincere appreciation to my dear friends Syed Ezaz Ali, Safoorah Sabir, Sughra, and Qasim Ali for their unwavering support during the challenging times of this research journey. Your friendship and encouragement have been a source of strength and motivation.

As I conclude, I extend my heartfelt wishes to all those who have contributed to my academic and research endeavors. May Almighty Allah bless you abundantly and grant you success in all your future endeavors. May He shower His mercy and blessings upon the entire ummah, and may we continue to strive for knowledge and positively impact the world.

Abstract

This work is a comprehensive analysis of time series forecasting of price variation of three important financial assets: Bitcoin (BTC), gold, and crude oil. In order to forecast future prices accurately, the study uses a combination of conventional time series models such as ARIMA and advanced time series machine learning algorithms. Analyses of historical price data are conducted in order to evaluate the model's capacity to identify underlying trends and patterns.

The study results indicate that the neural network model exhibits high accuracy in forecasting BTC and gold prices. This finding suggests that the neural network's ability to capture complex relationships and nonlinearities within the data is advantageous for these assets. On the other hand, the LSTM model demonstrates superior forecasting accuracy for crude oil prices. This indicates that LSTM's ability to handle time dependencies and long-term memory proves effective in capturing the dynamics of crude oil markets.

Overall, the research highlights the importance of considering different time series models when forecasting prices in financial markets. The outcomes underscore the significance of utilizing neural network models, particularly for BTC and gold, as well as the value of LSTM models for accurate predictions in crude oil markets. The results add to the body of knowledge already available in time series analysis and offer information to investors, analysts, and policymakers who want to choose wisely when it comes to various asset classes.

Contents

List of Tables

List of Figures

List of Abbrevations

BTC	Bitcoin
ML	Machine learning
RF	Random Forest
Bagging	Bootstrap aggregating
OOB	Out-of-bag error
SVM	Support vector machine
k-NN	k-Nearest neighbor
GRU	Gated recurrent unit
ANN	Artificial neural network
RNN	Recurrent neural network
LSTM	Long Short-Term Memory
ARIMA	Autoregressive integrated moving average
SARIMA	Seasonal ARIMA
MAPE	Mean absolute percentage error
MAE	Mean absolute error
MSE	Mean square error

Chapter 1

Introduction

Traditional time series models have been widely used for many years due to their simplicity and ease of use. Examples include exponential smoothing, seasonal ARIMA, and autoregressive integrated moving average (ARIMA). when a time series is anticipated via modeling. One of the conventional time series models that is most commonly used is the ARIMA model. The present value of a variable is regressed on both its prior values and the prior values of its errors as long as the series is stationary. SARIMA models, on the other hand, are seasonality-aware extensions of ARIMA models. To account for seasonal patterns, they extend the ARIMA model with seasonal differencing and seasonal autoregression. Another popular traditional time series model is the exponential smoothing model, sometimes referred to as the Holt-Winters model. They are predicated on the notion that a variable's future value is a consequence of its prior values, with more recent values being given more weight in the forecast.

Despite their popularity, traditional time series models have several limitations. Beginning with the premise that the underlying data is stable—that is, that its statistical properties do not change over time—they proceed from there. In reality, though, a lot of time series show non-stationary behavior, including trends, seasonality, and cycles. Second, conventional time series models need a lot of past information in order to predict future values with accuracy. This might not always be possible, especially with regard to recently discovered phenomena. Additionally, non-linear and dynamic interactions in the data may be difficult for classic time series models to describe accurately. Therefore, there is a need for more advanced and flexible time series models, such as machine learning models, to handle these challenges and improve forecasting accuracy. Therefore, we mainly focused on machine learning models to obtain accurate forecasting by studying complex patterns and long-term trends of data in depth.

Time series are collections of data points measured over time at regular intervals, and it poses significant challenges for prediction due to its inherent characteristics of trend, seasonality, and noise. Time series data can be subjected to machine learning algorithms to generate predictions or spot patterns and trends in the data. By using previous data, these techniques seek to create a model that can accurately forecast future values for the time series. Long Short-Term Memory (LSTM), Support Vector Machines (SVM), Random Forest

(RF), and Gated Recurrent Unit are machine learning approaches for time series forecasting (GRU).

Recurrent neural networks (RNNs) such as LSTM use memory cells to selectively discard irrelevant inputs while remembering previous inputs in order to capture long-term dependency between successive inputs. Due to its capability to manage intricate non-linear interactions and capture temporal dynamics, it is frequently employed in time series forecasting. SVM, on the other hand, uses a hyperplane to partition the input data into several groups. By utilizing a variety of kernel functions to identify the underlying patterns in the data, SVM may be employed in time series forecasting for both regression and classification applications. The ensemble learning method RF combines many decision trees to improve prediction accuracy and reduce overfitting. It is especially useful for time series forecasting since it can handle unpredictable data and spot non-linear relationships. Another RNN type, GRU, is comparable to LSTM but has less parameters, which makes it quicker and more effective. It has been demonstrated to be effective in tasks like speech recognition and language translation and is particularly beneficial for short-term predictions. The approach selection is influenced by the particular properties of the time series data and the forecasting goal. These algorithms all have advantages and drawbacks. To select the machine learning algorithm that gives the highest generalization performance and prediction accuracy in this circumstance, it is crucial to carefully compare the various algorithms.

A wide range of sectors, including medical, banking, and agriculture, have increased reliance on machine learning time series models. These models are able to evaluate vast volumes of complex data, spot patterns, and trends, and forecast future events with accuracy. Medical machine learning (ML) time series models are used to forecast disease outbreaks, track patient health, and create specialized treatment regimens. These algorithms, for instance, might examine patient information like vital signs, test findings, and medical history to spot trends that might point to a risk for a certain ailment. Medical practitioners can create more effective treatment regimens and make better judgments regarding patient care by looking at these trends. In finance and economics, ML is used to predict stock prices, identify market trends, and detect fraudulent activity. These models can analyze financial data such as stock prices, economic indicators, and market sentiment to make predictions about future market behavior. By accurately predicting market trends, financial institutions can make more informed investment decisions and minimize risk. Furthermore, the ML time series model is also vastly used in agriculture to forecast crop yields, monitor soil health, and optimize resource allocation. These models can analyze data such as weather patterns, soil quality, and crop growth rates to identify patterns and predict future outcomes. By accurately predicting crop yields and monitoring soil health, Farmers can increase total crop yields and make better resource allocation decisions. In all of these industries, machine learning time series models can improve decision-making and optimize outcomes by providing accurate predictions and insights based on large amounts of data. As these models continue to improve, they are likely to become even more important across many different businesses.

Since gold is a precious commodity that has been used as a store of value for millennia, we forecast and analyze the intricate structure of the values of gold, Bitcoin (BTC), and crude oil in this study owing to their significance for research in finance and economics. As a result, it is an important asset to study in financial research. The analysis of gold prices over time can provide insights into market trends, inflation, and monetary policy. For example, the study of gold prices during periods of economic instability can provide insights into how investors perceive risk and uncertainty. The popularity of BTC, another digital asset, has increased significantly in recent years. Its decentralized nature and limited supply have made it an attractive investment for some investors. The study of BTC prices over time can provide insights into market trends, investor sentiment, and the adoption of new technologies. For example, the study of BTC prices during periods of market volatility can provide insights into how investors perceive risk and uncertainty in the cryptocurrency market. Lastly, crude oil is a critical resource that is used in the production of a wide range of goods and services. The analysis of crude oil prices over time can provide insights into market trends, global economic activity, and geopolitical events. For example, the study of crude oil prices during periods of global conflict can provide insights into how geopolitical tensions impact commodity markets and global supply chains.

Examining these assets is the main goal of this research using machine learning time series analysis techniques, such as LSTM, SVM, RF, and GRU, and comparing the outcomes with those obtained using more conventional approaches, such as ARIMA and VAR models. It can offer insightful information on market behavior and the factors that influence price changes. Researchers can more accurately estimate future price movements and spot potential hazards and opportunities by evaluating historical price data and seeing patterns and trends. Investors, decision-makers, and professionals in the sector can utilize this information to better manage risks and make more educated decisions.

1.1 Research objectives

The goal of this study is to present a useful ML time series model for predicting the values of gold, crude oil, and bitcoin. The specific research objectives are as follows:

- To create and train machine learning time series models, such as LSTM networks, RF, and GRU, in order to estimate the values of gold, crude oil, and bitcoin.
- To assess the machine learning models' accuracy and contrast their performance with that of the classic ARIMA models.
- To identify the key drivers of price movements in gold, crude oil, and BTC markets and assess the extent to which the machine learning models capture these drivers.
- To make recommendations on how machine learning time series models may be used to anticipate the prices of gold, crude oil, and bitcoin.

The long-term goal of this research is to support the development of the price of gold, crude oil, and bitcoin forecasting models that are more precise and efficient, allowing investors, decision-makers, and business experts to make better decisions and manage risks. By contrasting the effectiveness of conventional ARIMA models with more sophisticated machine learning strategies, the study will also contribute to the larger area of time series analysis.

1.2 Contribution of the study

By comparing the predictive abilities of conventional time series models and machine learning algorithms for the prices of three important financial assets—crude oil, bitcoin, and gold—the study presented in this thesis seeks to significantly advance the field of time series analysis. The accurate prediction of these asset prices holds immense importance for investors, financial institutions, and policymakers who seek to make informed decisions based on future price trends. Traditional time series models like ARIMA, SARIMA, and exponential smoothing are used along with popular machine learning algorithms like the RF, SVM, GRU, LSTM, and KNN, this study seeks to assess the forecasting performance of both methodologies and provide insights into their respective strengths and weaknesses.

The contribution of this study lies in its comprehensive and comparative analysis of different time series forecasting approaches. While traditional time series models have long been utilized in financial forecasting, the emergence of machine learning algorithms has opened up new avenues for accurate predictions.

By exploring the performance of both methodologies, the study will highlight that which technique is most accurate in forecasting the the cost of commodities like gold, bitcoin, and crude oil. Furthermore, the research will explore how well machine learning algorithms work on time series data by how much it captures the complex and nonlinear patterns of series, leading to improved time series models. The study will not only contribute to existing knowledge of time series but also provide useful insight for financial practitioners. The study will provide guidelines for future research and applications in financial forecasting

1.3 Thesis structure

The composition of the thesis includes five chapters. In Chapter 1 we study briefly the limitation of traditional methods, the importance and introduction of ML time series models, and the significance of our research work. Chapter 2 comprises literature work related to our work, whereas the proposed work's technique is described in Chapter 3. Results, discussion, and work restrictions are all included in Chapter 4. Chapter 5 of the work has its conclusion.

Chapter 2

Literature Review

The literature places a great deal of emphasis on the forecasting of gold and crude oil prices using time series analysis. Time series forecasting of these two critical commodities has been extensively studied and researched due to their significant impact on various aspects of the global economy. In the financial literature, accurate forecasting of gold and crude oil prices helps traders, financial institutions, and investors to make well-informed choices on risk management, hedging tactics, and portfolio management. To comprehend the underlying patterns and causes impacting the price fluctuations of gold and crude oil, researchers have investigated a variety of time series models and econometric methodologies. This chapter discusses recent efforts to anticipate the prices of gold and crude oil using time series models.

Guha and Bandyopadhyay (2016) forecasts the price of gold in India based on past data using the ARIMA model. The Multi Commodities Exchange of India Ltd. (MCX) provided the secondary monthly gold price data that was used for their study, which was conducted between November 2003 and January 2014. To examine autocorrelation, the Durbin-Watson test was applied. The ARIMA(1,1,1) model, which was created after examining the behavior of the gold price during the course of the previous 10 years of MCX trading value, may be used to forecast the future values of gold. ARIMA(1,1,1) was chosen above the other six model parameters because it produces the best model and meets all statistical criteria for fit, whereas the other five, ARIMA(1,0,1), (1,0,2), (1,0,3), (1,1,2), and (1,1,3) do not.

The Seasonal Auto-Regressive Integrated Moving Average (SARIMA) models were used to model and forecast average monthly gold prices, according to citechakutime. Between January 2015 and December 2020, data on gold was acquired. The research used monthly adjusted closing prices. Gold price data demonstrate that the price of gold increased continuously (with an upward trend) during the studied period of 2015–2020. In comparison to all previous years within the study period, 2020 had the highest gold prices. The Augmented Dickey-Fuller (ADF) test determines if the gold price data is stationary given the initial difference. IMA is the model that fits the gold price data the best with the lowest AIC and BIC values (1,1). Forecasts for gold prices were obtained from January 2021 to December 2025. Over the course of the forecast years, the average monthly price of gold is shown to have a consistent upward and downward trend (2021–2025). For the months of January 2021 to June 2021, The anticipated values and the actual values were compared. The difference between actual and expected gold prices was insignificant (t = 2.02, P = 0.07191 <0.05).

Khan et al. (2021) investigated a mix of linear and non-linear models, including the auto-regressive integrated moving average (ARIMA) and artificial neural network, to address the shortcomings of a single model for forecasting the price of gold (ANN). The ARIMA-ANN model is referred to as "combined approach" (Hybrid model). They make use of information on gold prices in Pakistan from June 2003 to June 2021. First, they split the data into two parts; the training data (from July 2003 to July 2018) was used to estimate the models, and the testing data (from August 2018 to June 2021) was used to evaluate the models. Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) are two error metrics they use to evaluate models. They found that ANN beats the ARIMA model in terms of forecasting accuracy, as shown by RMSE and MAE. The ARIMA-ANN beat ARIMA and ANN in terms of accuracy when their forecasts were combined. They come to the conclusion that ARIMA-ANN offers forecasts that are more precise than ARIMA and ANN.

In the article Abdullah (2012), they made the case that time series forecasting is a vibrant area of study that has drawn a lot of interest due to the potential implications it may have in a variety of sectors. ARIMA models are not typically employed to anticipate gold prices, despite being widely utilized in financial market forecasting during the previous three decades. They make an effort to deal with the issue of calculating gold bullion coin sales prices. During their study, they developed the ARIMA model for selling prices of gold bullion coins. The model that best matches the selling prices of gold bullion coins has shown out to be the ARIMA(2,1,2).

In order to forecast monthly gold prices, Rady et al. (2021) compares Decision Tree (DT), Random Forest (RF), Gradient Boosted Trees (GBT), and ARIMA models using tree-based techniques for time series data. With a total of 362 observations, the time series data for the monthly gold prices covers the months of November 1989 through December 2019. The determine which model is better at predicting the monthly gold price, the models are compared. 90% of the data was utilized as the training set for the fitting of the ARIMA, DT, RF, and GBT models. Then, using the RMSE criterion, they compared the results of ARIMA, DT, RF, and GBT. According to the results of this study and the discussion that follows, RF results were more precise (with the lowest RMSE) a more effective forecasting technique for the monthly price of gold than DT, GBT, and ARIMA models. In the future, they suggest enhancing the results by combining ARIMA and tree-based methods to gain the advantages of both models.

Daily gold price employing RNNs with LSTM cells and an attention mechanism, Kourti (2021) use three alternative model architectures were used to assess the utilization of different covariate combinations. The following are the main deductions drawn from their findings: 1) Accurate daily gold price predictions may be made using LSTM architectures. The performance of the model was significantly enhanced by the addition of the attention mechanism. 2) The model's predicting skills were not improved by the linked variables that were selected using the gold price time series. They instead seem to have inserted more undesired signals than beneficial information. The Silver price time series was the lone exception. 3) The suggested architecture and configuration outperformed the benchmark and delivered respectable results on a test set that was even bigger than the one used in previous literature.

Primananda and Isa (2021) use cross-validated grid search, a multivariate prediction model based on LSTM and GRU models is developed, and the model is then tuned to discover the best hyperparameter. The suggested model has good prediction accuracy, as evidenced by the lowest Mean Absolute Percentage Error (MAPE) and Root Root Mean Square Error (RMSE) values derived from the empirical data. According to the study, LSTM has superior accuracy for time intervals longer than three years, whereas GRU is the model to use for forecasting gold prices for time periods shorter than three years.

Grid search can reduce the RMSE by an average of 68% and decrease the MAPE score by 6.72, significantly improving LSTM accuracy. The paper suggests using metaheuristic techniques in future research to increase the effectiveness of hyperparameter modification, such as Ant Colony Optimization, Genetic Algorithms, or Chaos Metaheuristics.

Gold price forecasting has been a subject of research due to its multi-factorial and non-linear nature. Between 2001 and 2021, they examined the effects of the gold price, crude oil price, consumer price index, exchange rate index, stock market index, and interest indicator. Models developed using the LSTM, Bi-LSTM, and GRU algorithms were evaluated using the criteria Lowest Mean Absolute Percent Error (MAPE), Mean Absolute Error (MAE), and RMSE. This study assessed the effectiveness of three multivariate models: LSTM, Bi-LSTM, and GRU, to estimate the price of gold using monthly time series data. With 3.48 MAPE, 61,728 RMSE, and 48.85 MAE values, the LSTM model beat the opposition. When the real gold price was compared to the values predicted by each model. The simulations also showed how economic indicators affect how much an ounce of gold costs.

By combining traditional indices, new indicators, commodities, and historical price time series of gold, Yang et al. (2022) build a time series model to anticipate gold prices. Also, they examine a number of machine learning models, including ANN, LSTM, and SVR. They employ three machine learning methods to create the models that predict the price of gold: Artificial Neural Networks (ANN), Long Short-Term Memory (LSTM), and Support Vector Regression (SVR). For this study, a sizable dataset of 1006 observations of daily close prices for 9 input parameters across 4 calendar years, from January 1st, 2017, to December 31st, 2020, is used. This dataset includes the S&P 500, DJI, Bitcoin, Ethereum, silver, crude oil, Dollar index, gold price, and volatility. on the basis of the two training. Model performance is evaluated using MAE, RMSE, and MAPE. Three models are subjected to comparative analysis in the first phase. The evaluation of how cryptocurrency affects the models is demonstrated in the second phase. The results demonstrate that SVR performs better than both other models and is the best model for predicting gold's price. The outcome demonstrates how bitcoin data may increase a model's accuracy, however the improvement differs between different kinds of models. This is accomplished by contrasting the results of two different model series.

Using a novel forecasting model dubbed CNN-LSTM in their study, Livieris et al. (2020) predicted the price and movement of gold. In two distinct iterations, each with two convolutional layers and a variable number of filters, the proposed CNN-LSTM model was compared to state-of-the-art deep learning and machine learning forecasting models. The first demonstrated the best forecasting performance for regression problems, reporting the lowest MAE and RMSE performance, while the second demonstrated the best performance for the classification problem of predicting the movement of gold, outperforming conventional time series models for price movements. They underline that while LSTM models are a popular and successful solution for time series of the gold price, their usage in conjunction with extra convolutional layers considerably increases prediction capacity. The suggested CNN-high LSTM's complexity and sensitivity to many hyper-parameters enable further enhanced configuration, namely feature engineering, to increase forecasting performance.

They claim that Madziwa et al. (2022) employed the ARDL approach to forecast gold prices over the period 2000–2016. Stochastic geometric Brownian motion and ARIMA were both used to compare the results of the predictions. Using Cointegration Tests, the ARDL method examined the long-term link between gold prices, gold demand, and interest rates between 2000 and 2016. Stationarity tests, cointegration tests, estimating the short- and long-run equations, and diagnostic checks are all steps in the ARDL modeling procedure. All of the tests supported the model's stated econometric features, showing that the functional form was acceptable and that the residuals were homoscedastic, normally distributed, and serially uncorrelated. The structural stability was assessed utilizing the CUSUM statistics in addition to the diagnostic tests. The computed coefficients were steady, as seen by the CUSUM statistic's plot, which consistently stayed within a 5% threshold of significance. There were thus no structural gaps. The calculated model is stable, it was found. As a consequence, the findings are trustworthy and legitimate from a technical perspective. RMSE, MAD, MAPE, Theil's coefficient, and RMSE were used to assess the results. The mean reversion O-U process (geometric Brownian motion), which came close behind with errors of 2.2127, 0.1422, and 0.173, and then ARIMA, emerged as the best approach based on all error assessments, with respective errors of 0.653, 0.981, and 0.06. With a MAD of 2.489 and an RMSE of 0.219, the ARIMA's measurements exhibited a substantial amount of inaccuracy. Mean reversion and the ARIMA technique were surpassed by ARDL. A TUC value of less than one, showing that it outperformed the naïve projection, further demonstrated its success.

For predicting the price of gold, a brand-new decomposition-ensemble model termed ICEEMDAN-LSTM-CNN-CBAM is put forward in Liang et al. (2022). The model sublayers the original gold prices into various frequency sublayers using the improved complete ensemble empirical mode decomposition (ICEEMDAN) and adaptive noise. The sublayers are then collectively predicted using the long short-term memory, convolutional neural networks, and convolutional block attention module (LSTM-CNN-CBAM). The model outperforms 22 other comparison models according to tests for RMSE, MAE, MAPE, SMAPE, R-square, and MCS. Moreover, it states that the ICEEMDAN decomposition strategy outperforms other decomposition techniques and that Compared to other prediction models, the LSTM-CNN-CBAM model is more accurate. The study finds promise for improvement of the ICEEMDAN-LSTM-CNN-CBAM model.

An overview of the gold market is provided in the article Shafiee and Topal (2010), which looks at the supply, demand, and price of gold. According to the study, jewelry makes up a sizable amount of the demand side of the gold market, and before it started to fluctuate, the price of gold was stable for more than a century. The paper also emphasizes the significance of variables like the price of oil and inflation on the gold market and discovers a strong correlation between gold and oil prices. The study, however, finds no conclusive link between inflation and the price of gold. The paper offers a novel model including components for long-term reversion, diffusion, and jump/dip diffusion that overcomes the non-stationarity problems in prior models. The study forecasts that the gold price will remain unusually high until the end of 2014, assuming that the current price rise that began in 2007 behaves similarly to the one that occurred in 1978. The model is verified using historical data on gold prices. The price would then return to the long-term trend until 2018. The essay gives a novel methodology for forecasting the price of gold and delivers insightful information on the dynamics of the gold market overall.

In order to analyze the volatility of the gold price, the article Wen et al. (2017) analyzes the effectiveness of the empirical mode decomposition (EMD) and full ensemble empirical mode decomposition (CEEMD) methods. The paper analyzes the features of gold price volatility and starting events by extracting market fluctuations, significant events, and long-term trends in the gold price from global gold price series data. The study employs artificial neural network (ANN) and support vector machine (SVM) models to forecast short-term volatility, shocks from major events, and long-term price patterns. The findings indicate that based on their investment needs, capital allocation techniques, and risk tolerance, investors should develop short-, mid-, and long-term investment plans. The technique suggested in the article has enhanced prediction accuracy and may be used in a variety of situations. Future studies should, however, take into account a number of factors that affect the price of gold.

The article, which was written by Makala and Li (2021), uses ARIMA and SVM models to estimate the price of gold between 1979 and 2019 using daily data from the World Gold Council. In terms of prediction accuracy, SVM surpasses ARIMA, with the SVM (Poly) model greatly outperforming the SVM (RBF) and ARIMA models. The paper promotes more research in this field to advance the predictive science paradigm and proposes that machine learning, in particular SVM, is a potential alternative for commodity price prediction.

In order to compare the effectiveness of the LSTM, random forest regression, and linear regression algorithms in forecasting Indian stock prices, the waghgold study is undertaken. These machine-learning techniques are employed to determine the current gold price. They conducted their investigation to learn more about how the price of gold and some of its influencing factors interact. The daily gold price figures are calculated for the years 2010 through 2022. The monthly gold price values are based on the period from 2015 to 2022. Values for the six-month gold price are generated using data from 2010 to 2022. Data for the annual gold price were gathered from 2010 to 2022. This data were examined using three machine learning algorithms: LSTM Model, Random Forest Regression, and Linear Regression. The LSTM Model provides better period-wide forecast accuracy when the two periods are examined independently. The study finds that machine learning techniques are quite useful in this situation, although their accuracy relies on the characteristics of the data. To learn more about how these tactics work, more study might be done utilizing this data and different methodologies.

According to the research, gold's complicated and nonstationary nature makes it difficult to anticipate gold prices with any degree of accuracy. CiteLI2021New suggests a brand-new hybrid forecasting method called VMD-ICSS-BiGRU that combines the ICSS algorithm, the BiGRU deep learning model, and the VMD signal decomposition tool. This method enables accurate price forecasts in the gold futures market by extracting internal factors and patterns inside market movements, dissecting their relationship with external markets, and spotting changes within market situations. Comparing the VMD-ICSS-BiGRU technique to earlier single and hybrid deep learning models, it shows improved prediction performance outcomes. It regularly generates favorable returns and reduces unfavorable consequences, pointing to its generalizability. Future study may take into account using a hybrid long and short strategy as the suggested approach is a long-only strategy.

The association between gold prices and different macroeconomic variables in India, including the consumer price index, currency rates, US bond rates, and stock market index, is highlighted by the author in the passage cited as Wen et al. (2017). Gold is considered a hedge against inflation and a useful portfolio hedge in India. The study suggests that policies to limit gold imports may not be effective as gold serves as an effective hedge for investors. Instead, policies targeting the root causes of inflation and offering alternative investment options may be more effective in reducing gold imports. They use worldwide gold price series data ranging from January 1968 to June 2016 to concentrate on the features of gold price volatility and the factors that set it off. To forecast the short- and long-term price patterns of gold, they suggest a model that makes use of the empirical mode decomposition (EMD) and complete ensemble empirical mode decomposition (CEEMD) algorithms. With SVMs performing better for short-term forecasts, the price of gold was predicted using ANNs and SVMs models. According to the paper's conclusion, investors should create short-, mid-, and long-term investment plans based on their capital allocation methods and risk tolerance. The study suggests that future research should consider multiple variables that influence the price of gold.

Every investor is particularly interested in understanding the direction of the gold price, whether it will rise or fall, according to the source Ali et al. (2016). The price of gold, which has been shifting significantly over the past few months, has recently piqued everyone's curiosity. They use the data set of US Dollars per ounce from January 2, 2014, to July 3, 2015, and give a time series model in their study to achieve the aforementioned objective. To forecast the daily price of gold, they employ the Box-Jenkins technique. They used the Line Diagram, Correlogram, and ADF Test to determine that our data is stationary at the first difference. After models have been estimated, the Box-Jenkins approach is used to pick the Autoregressive Integrated Moving Average (ARIMA) model, and the model selection criteria AIC and SBC show that ARIMA(1,1,0) and (0,1,1) are almost equivalent for forecasting the daily Gold price. They assess the projected values' accuracy using MAE, MAPE, and RMSE. The analysis cited above shows that ARIMA(0,1,1) is more efficient than ARIMA (1,1,0).

By integrating ARIMA and GARCH models, the study Yaziz et al. (2013) suggests a novel method for examining and forecasting the price of gold. The research demonstrates that when it comes to estimating and forecasting the accuracy of a 40-day gold price data series, the hybrid ARIMA-GARCH model surpasses 10 other forecasting strategies. The suggested hybrid model uses a two-phase procedure, with the first phase utilizing the ARIMA model to describe the linear time series data and the second phase using GARCH to explain the residual of this linear model. Data are normalized using the Box-Cox transformation, which also reduces heteroskedasticity and stabilizes variance. The empirical findings indicate that combining ARIMA with GARCH is a novel possible approach for examining and predicting daily gold prices that overcomes the limitations of

Dutta et al. (2020) to support the claim that the global price of crude oil significantly dropped as a result of the unique coronavirus outbreak COVID-19 that struck in December 2019. Investments in these markets carry the danger of suffering substantial losses because all of the main oil markets have since witnessed extraordinarily high volatility. The goal of this paper is to determine whether gold serves as a safe-haven asset for the global crude oil markets during the COVID-19 timeframe by conducting an experimental investigation of the time-varying relationships between gold and oil prices. The haven status of Bitcoin is also assessed for comparison's sake. They access the St. Louis FRED database for data on gold, bitcoin, and crude oil. The availability of Bitcoin data limits the sample period's beginning point, which runs from December 2014 to March 2020. The DCC-GARCH model's findings on time-varying correlations imply that gold may be a safe-haven asset for the world's crude oil markets. Bitcoin, on the other hand, only diversifies the crude oil market. The results also show that retaining assets in the oil and bitcoin markets decreases portfolio risk more than adding these assets to portfolios. Because global events like a financial slowdown, terrorist attacks, pandemics, and other analogous occurrences usually play a large part in portfolio risk assessments, their results may be intriguing to persons who invest in the oil, gold, and bitcoin markets.

The study by Gkillas et al. (2020) employs a quantile-regression heterogeneous autoregressive realized volatility model to examine the predictive value of geopolitical risks on realized gold returns volatility. They develop a daily measure of realized volatility using intraday data on gold futures traded on the NYMEX and discover that, after controlling for economic policy uncertainty, geopolitical risks are better able to estimate realized volatility at a longer forecast horizon. They specifically demonstrate that, when asymmetries in the loss function are taken into account, threats and actions have predictive value for realized volatility, primarily at a longer forecasting horizon. According to the report, investors may enhance their gold portfolios by adding knowledge of geopolitical risks to protect themselves from mostly long-term dangers. The authors emphasize the importance of considering potential asymmetries in the loss function used to assess forecasts, and their findings can have implications for risk management and portfolio optimization.

Chapter 3

Methodology

This chapter describes the methods in detail which are used in this work to forecast gold, crude oil, and BTC prices. This chapter also includes an explanation of the data in great depth. For time series forecasting, we applied machine learning models and contrasted them with traditional approaches. The three datasets gold, crude oil, and BTC are used as a univariate time series y_t and are forecasted through machine learning and conventional models.

3.1 Machine learning time series models

3.1.1 Random forest

Random forest (RF) is an ensemble learning method developed by Breiman (2001). It is working by using the concept of decision trees and combining many trees and taking the average response of all trees to predict the outcome of a continuous variable.

The key concept of the working method of random forest is given as

Decision trees: The RF collect a subset of values from the training set and learned the function and creates a single decision tree. It is a hierarchical structure of continuous splitting of data through the specific condition which gives a minimum error to make prediction.

The key component of decision trees is as follow:

Nodes: It is a decision point or question based on predictors. The starting point is known as root note and their further split is known as internal nodes. The tree becomes narrow at each split and reaches to final nodes called terminal nodes which is the final prediction of the unknown variables.

Splitting: The corresponding value that divides the dataset into each node based on specific criteria is called splitting. The criteria may be minimizing impurity or maximizing information gain.

Leaf nodes: It is the outcome of the splitting which gives the prediction based on lags or predictors. In regression problems, it is a continuous variable.

Pruning: It is a technique of removing unnecessary nodes and branches that cause overfitting in the model.

Predictions: The final step of the tree is a prediction in which a test value is passed to the tree and answers the conditions until it reaches a leaf node. The corresponding value associated with the leaf note is the final prediction by the decision tree.

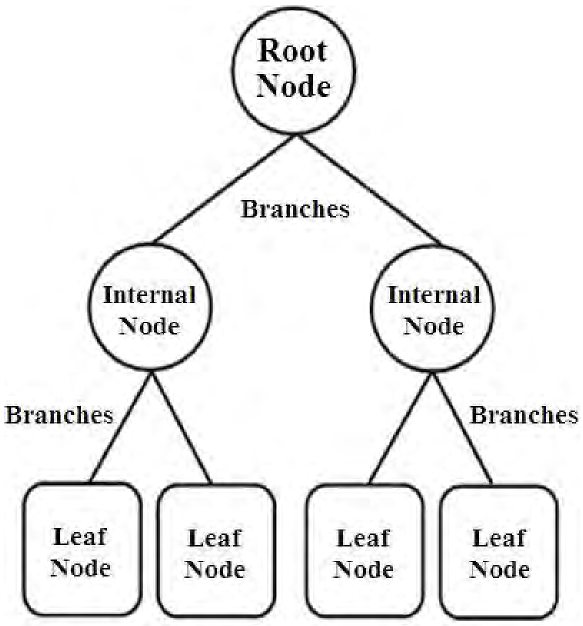


Figure 3.1: The general skeleton of decision tree. Source: (Rigatti, 2017)

Decision trees are easy to comprehend and use, and they can handle both category and numerical data. More advanced ensemble methods are required to improve forecast accuracy like random forests and gradient boosting, which integrate several decision trees, frequently use them as the building blocks.

The concept used by RF to split the trees is impurity reduction, usually measured by Gini impurity. The Gini impurity gini(p), is the probability of misclassifying or error (in case of regression problems) when the condition of splitting is chosen randomly. The Gini impurity have $(c_1, ..., c_k)$ classes with each node and their respective probabilities $(p_1, ..., p_k)$, is calculated as:

$$\text{Gini}(p) = 1 - (p_1^2 + p_2^2 + ... + p_k^2)$$

The algorithm repeatedly calculates the error rate by changing the threshold of splitting the nodes and computing that point which gives a minimum error or low Gini Index. The algorithm aims to find the split that creates the most homogeneous child nodes based on the target variable.

The drawing of the subset is repeated multiple times (usually 500 times) and each time generates a tree therefore combing all trees make a forest and take an average of all tree's response to return the outcome. The subset is drawn randomly so that no correlation between trees is occur and the forest become diversified.

Bagging: The overfitting of the random forest is reduced by bagging. The RF work is based on bosting in which the subset is drawn randomly with a replacement which means that from all available samples, RF draws a subset of observation with replacement so that each observation can come twice or more times in each sample and build a tree on each subset of observation.

Out-of-beg (OOB) error: The accuracy of the random forest model on unseen data is measured through OOB error. Since RF draws with replacement sampling to select a subset of the observation, therefore, some of the observations are left and not selected in the subset. Based on the observation select in a subset a tree is built and predicts those outcomes which are not included in the model and compared with the truth value. It is usually calculated as the mean square error (MSE) of all data points in the testing part.

The OOB error is a useful measure that evaluates the performance of random forests on unseen data. It provides testing accuracy without any further cross-validation or inputting of extra data. The overall estimate of the prediction accuracy of RF is calculated by averaging OOB error on all data points. Lower OOB error better the predictive power.

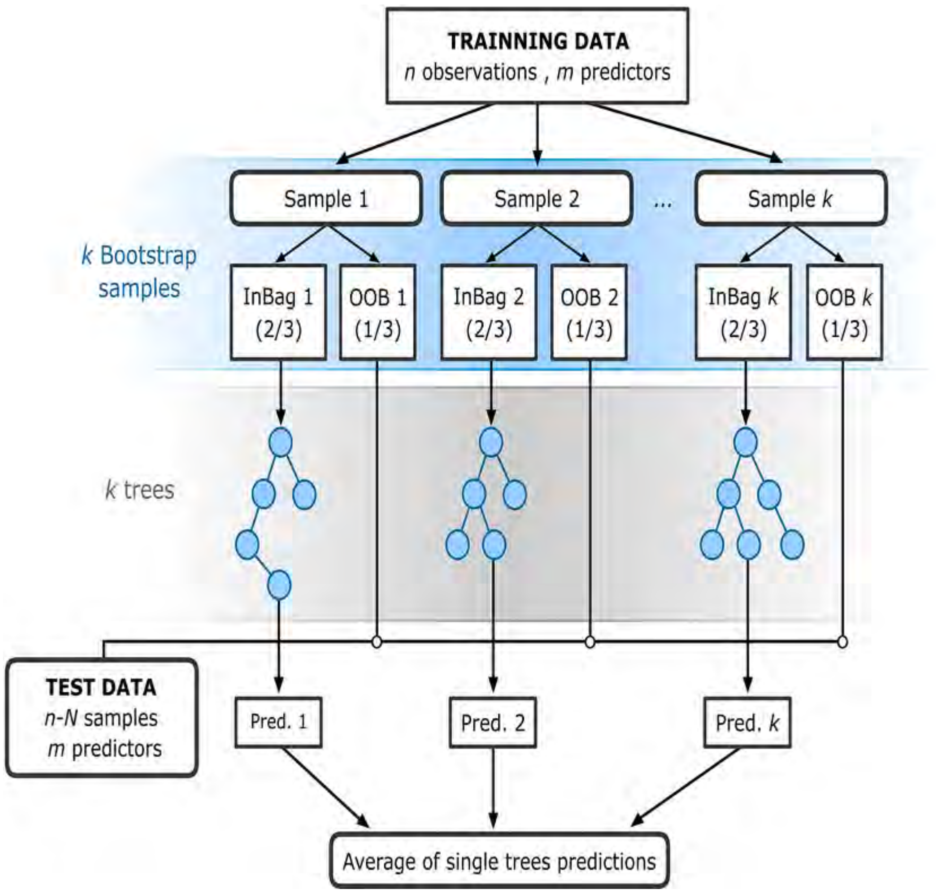


Figure 3.2: Flow chart of random forest for regression problems. Source: (Rodriguez-Galiano et al., 2015)

Figure 3.2. Shows the flow chart of RF in which random samples are drawn from the training part. The two bootstrapping of size (2/3) and (1/3) are performed on each set of samples to create a tree on each bootstrapping. The process is repeated on all k samples. The observation of testing data is predicted by averaging the prediction of each tree to give the final prediction.

3.1.2 Random forest for time series

To apply the random forest on time series data it will need to account for additional techniques to understand the temporal nature of the data. The random forest forecast the time series values through the following approach

Lagged values: Since the time series variable is univariate and has no features, therefore, the model will create a lag value which is just the difference between present and past values. It can be a difference between any values but at the same duration. The lagged value y_{t-1} is placed against the current value y_t and hence the feature is made and now these lag values will capture a trend or pattern over a specific window of time.

Training and testing sets: To maintain the temporal nature of data and calculating accuracy it divides the data into two parts sequentially rather than randomly as in the regression case.

Ensemble of trees: Same as in the regression case the RF builds multiple decision trees on a different subset from data drawn with replacement

Prediction: The values are forecasted by each tree by inputting the lagged value and averaging all tree's outcomes to give the final response.

Evaluation: By calculating the difference between the actual value and the value predicted, Random Forest's efficacy is evaluated. Mean absolute error (MAE), mean square error (MSE), and mean absolute percentage error are the three metrics that are most frequently utilized (MAPE).

3.1.3 k-nearest neighbor (k-NN)

A non-parametric machine learning algorithm called K-NN is utilized for regression and classification. It measures the k^{th} nearest point from training data through a distance matrix usually Euclidean distance from new data point and computes their median value or labels in case of classification to predict the value of unknown data point.

The working algorithm of k-NN is described below.

- The user will provide a k value which means that how many nearest points will be computed from an unknown data point.
- In addition to storing the k observations and their values that are close to the unknown data point, the algorithm will calculate the separation between each data point in the training set and the new data point.
- The algorithm will compute the weighted average value of all k^{th} observations by using distance as a weight matrix to predict the unknown data point

Figure 3.3 shows the graphical methodology of k-NN by predicting the unknown data point X_u. The algorithm computes distance with each point and identifies the five nearest points with X_u. Four out of five observations belong to the W_1 class and one to the W_3 class. In the case of regression, the algorithm will calculate the weighted mean of all five observations and assign the value to an unknown data point.

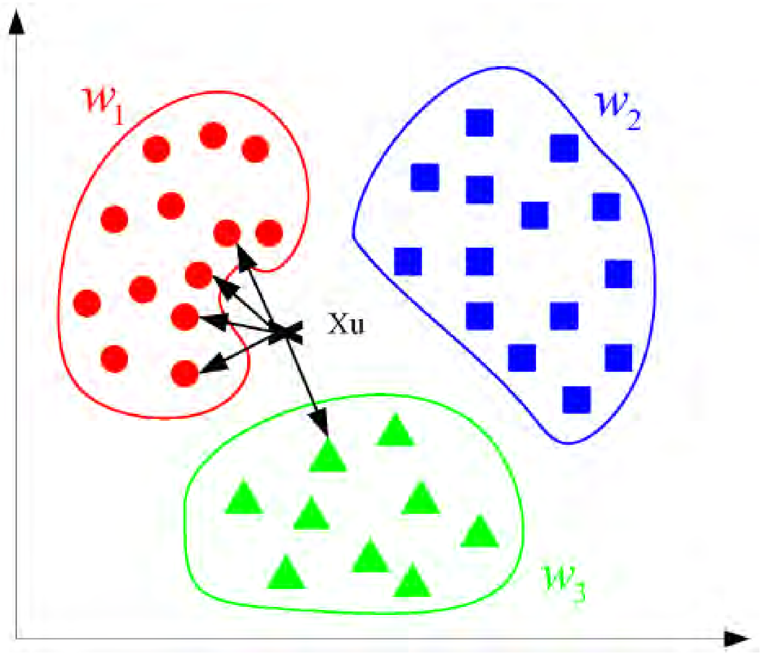


Figure 3.3: Flow chart of K-NN. Source: (Zhang et al., 2017).

Euclidean distance: The Euclidean distance is named after the Greek mathematician Euclid who calculates the straight distance between two locations in multidimensional space. It is commonly used in mathematicians and statistics to measure similarity.

In two-dimensional space the Euclidean distance between two points, $P(x_2, x_1)$ and $Q(y_2, y_1)$, can be calculated using the Pythagorean theorem:

$$d(P, Q) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

It will return a length of a straight line between these two points. The distance formula satisfies the properties of the matrix such as non-negativity, identity of indiscernible, symmetry, and triangle inequality.

Selecting the value of parameter k: Generally, the optimum value of k is calculated through cross-validation in which a grid of k vector is provided, and the model is trained on each value of k and predicts the unknown data. The value of k is plot then against their corresponding error measure such as mean square error (MSE) and choose the value which gives the lowest error.

The smaller value of k makes the model more sensitive to noise and local variation causing overfitting in the model. The larger the k value make smooths out decision boundaries but may introduce bias. Therefore, choosing an optimum value is an important step in developing the kNN model.

The k-NN algorithm relies on the stored training datasets for prediction and does not involve explicit model training. For high dimensional data, the algorithm can be computationally expensive as it calculates the distance between each point.

Overall, the k-NN algorithm is simple and easy to understand as it relies only on nearest neighbors and distance matrix.

3.1.4 k-NN for time series data

Dynamic Time Warping: Comparing the resemblance between two time series with differing lengths, beginning locations, and rates of change is done using a technique called dynamic time warping (DTW). By choosing the best alignment for the two time series, DTW calculates their separation. The two-time series' time axes may be stretched and compressed to get the optimal alignment.

Two time series X and Y are separated by the following DTW distance:

$$DTW(X,Y) = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} w(i,j)(X_i - Y_j)^2} \tag{3.1}$$

The values of the time series X and Y are X_i and Y_j, respectively, whereas the lengths of the time series X and Y are m and n, respectively. The similarity between the i and j points of the time series X and Y is represented by the weight $w(i,j)$. There are numerous ways to determine the weight function $w(i,j)$, including using a Gaussian function or a straightforward linear function.

KNN for Time Series Forecasting The k-NN method may also be used to forecast time series data based on how similar historical time series data are. The pre-processing of the data, including the removal of NA, normalization, and smoothing or filtering, is where the k-NN method begins.

Using Dynamic Time Warping (DTW) as a distance matrix, k-NN locates the k nearest training time series and uses their values at the following time step to predict the value of the new time series. The weighted average of the values of the k nearest neighbors is the anticipated value. At each time step in the prediction horizon, this procedure is repeated.

The mathematical formula for k-NN for time series forecasting is as follows:

Assume that T is a collection of n time series, each with m observations.

We are attempting to forecast the value of a brand new time series (X) at time $(t{+}1)$. DTW is used to determine the k closest neighbors of X in T, represented as $N_k(X) = T_{i1}, T_{i1}, ..., T_{i1}$. So, the following is the expected value of X at time $t+1$:

$$\hat{X}t + 1 = \frac{1}{k}\sum i = 1^k T_{i,t+1} \tag{3.2}$$

where $T_{i,t+1}$ is the value of the i-th neighbor at time $t+1$.

Prepare a labeled dataset consisting of historical time series data with their corresponding labels or classes. The optimum k value is selected through cross-validation. To forecast the new time series data, point the model will compute the distance between the lag value of the unknown time series point and all observations in the training sample by using the DTW distance matrix. The k^{th} nearest point is obtained and compute the weighted average of the training points is to forecast the unknown value. This process is repeated until all the required horizons are forecast.

3.1.5 Support Vector Machines (SVM)

For classification and regression issues, a Support Vector Machine (SVM) method known as supervised machine learning is used. It effectively handles complicated datasets with a clear margin of difference between classes.

The SVM algorithm represents the data as a vector in multi-dimensional space. The goal of the procedure is to locate a hyperplane that divides the classes by the greatest possible distance. The support vector is the data point that is most near the hyperplane. It plays an important role in defining the hyperplane position as shown in Fig. 3.4. Support Vector Regression (SVR), an adaptation of the SVM for regression applications, aims to predict continuous numerical values rather than discrete class labels.

SVM has several advantages, including handling high-dimensional spaces, robustness against overfitting, and effectiveness in handling small-to-medium-sized datasets. However, due to its computational complexity, SVM performance may degrade with large datasets.

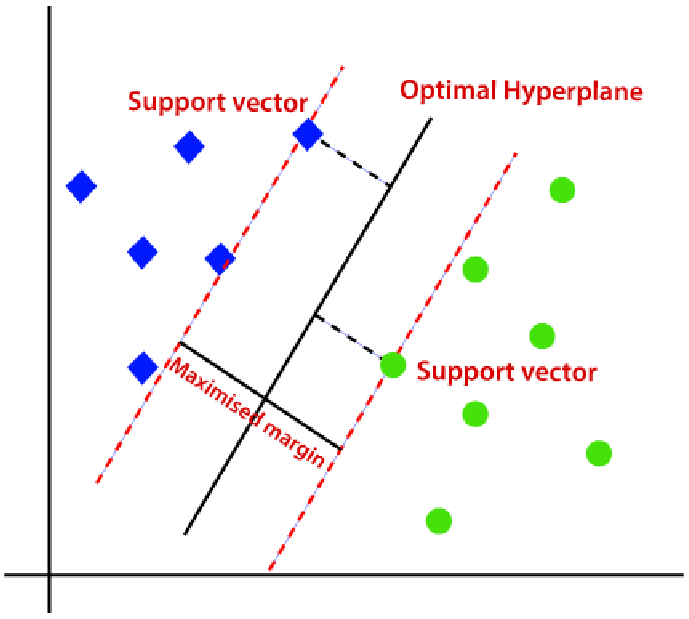


Figure 3.4: Support vector machine. Source: (Vishwanathan and Murty, 2002)

SVM Algorithm The algorithm of SVM to perform regression tasks is defined below.

1. Gathering and preparing the data for regression should come first. Missing values are eliminated, the features are normalized, and the data is divided into training and testing sets.

2. Setting up the model: the SVM regression model should be launched. As a kernel function in SVM, you can use a linear, polynomial, or radial basis function (RBF).

Choose the right kernel function based on the kind of data.

3. Model training: Create the SVM regression model and train it using the practice data. The goal of SVM regression is to find a hyperplane that maximizes the margin around the support vectors while lowering the error on the training set. To do this, an optimization issue is solved using techniques like the Sequential Minimum Optimization (SMO) algorithm.

4. Model Evaluation: Finding a suitable error matrix, such as mean squared error (MSE), root mean square error (RMSE), and mean absolute percentage error, allows the model to be tested on training data (MAPE).

5. Hyperparameter Tuning: Adjust the hyperparameters to improve the SVM regression model's performance. Hyperparameters include, for example, the regularization parameter (C), the kernel parameter (e.g., degree for polynomial kernel or gamma for RBF kernel), and the epsilon parameter (*epsilon*) for the epsilon-insensitive loss function.

6. Model Optimization: If needed, perform additional optimizations to improve the model's performance, such as feature selection or dimensionality reduction techniques. You can also explore different kernel functions or try ensemble methods to boost the regression performance.

7. Predictions: After you are satisfied with the model's performance, you may use it to make predictions about future events based on brand-new, unanticipated data. Apply the learnt model to new input instances to predict the related target values.

A set of weights, w, and a bias term, b, define the hyperplane. According to a constraint on the margin, where margin is the separation between the hyperplane and support vector, the SVM sets the values of w and b to minimize the residual on training data.

The mathematical formula for the SVM algorithm is as follows:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i \tag{3.3}$$

subject to:

$$y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i \tag{3.4}$$

$$\xi_i \geq 0 \tag{3.5}$$

where x_i are slack variables that allow for misclassifications, C is a parameter that regulates the trade-off between maximizing the margin and reducing the classification error, and $\phi(x)$ is a function that transforms the input x into a higher-dimensional space.

Kernels in Support Vector Machines (SVM) are operations that enable the input data to be transformed into a higher-dimensional feature space. Kernels play a crucial role in

SVM by enabling nonlinear decision boundaries in classification tasks and capturing complex relationships in regression tasks.

When it comes to SVM for regression tasks, there are several commonly used kernels:

Linear Kernel: The linear kernel performs a regression task by computing the dot products between the input samples by transforming data linearly. It is suitable for linear regression. The linear kernel is also known as the dot product kernel and is the most simple kernel. The kernel is defined as

$$\mathrm{K}(x, x^t) = x \cdot x^t$$

Where x and x^t are input feature vectors.

Polynomial Kernel: The polynomial kernel is introducing polynomial terms of the original feature to build a non-linear relationship. To capture the complex structure of data it maps the data into a high-dimensional space. The polynomial kernel is defined as:

$$\mathrm{K}(x, x') = (\gamma(x \cdot x^t) + r)^d$$

, where γ is a scaling factor, r is an offset, and d is the degree of the polynomial.

Radial Basis Function (RBF) Kernel: Due to its ability to capture more complex nonlinear interactions, the RBF kernel is a common option for SVM regression. Since the decision boundary isn't anticipated to be linear, it turns the data into an infinitely dimensional space and is very helpful. The source of the RBF kernel is:

$$\mathrm{K}(x, x^t) = \exp\left(-\gamma \cdot \parallel x - x^t \parallel^2\right)$$

Where γ controls the shape of the decision boundary.

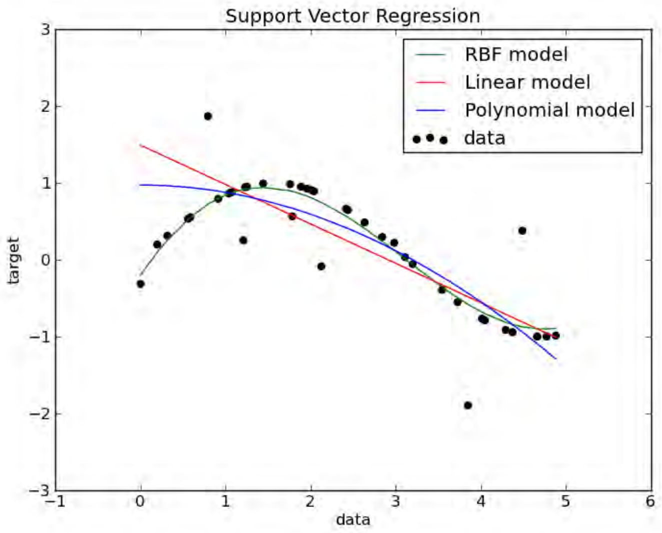


Figure 3.5: Support vector regression by linear and non-linear kernals. Source: (Vishwanathan and Murty, 2002).

SVM for Time Series Forecasting A well-liked machine learning approach for time series analysis and prediction is called Support Vector Machines (SVM). The core of SVMs is the discovery of a hyperplane in a high-dimensional space that splits the data into discrete classes. The mathematical formula for SVM for time series will be explained in this section.

Similar to the classification problem of SVM, the SVM for time series forecasting works by classifying each step as rising or decreasing based on the preceding step. This is similar to the sliding window approach in which a future value is forecast by the window of previous observations.

The mathematical formula for SVM for time series forecasting is as follows:

Let $X = x_1, x_2, ..., x_n$ be a series of n time steps, where x_t is the value at time t. Let $Y = y_1, y_2, ..., y_n$ be a sequence of labels, where y_t is the label for time step t, where $y_t = 1$ if $x_t > x_{t-1}$ and $y_t = -1$ if $x_t \leq x_{t-1}$. The hyperplane which separates the data into two-time series class can be obtained by the SVM algorithm to the series X and label Y

The forecasting is made by applying a sliding window approach in which the window of k time series step is used to label the next time step. The size of the window and lag are generally identified by cross-validation. When the label is predicted by the algorithm for future time steps then the time series value is forecast by adding or subtracting the average absolute difference between consecutive time steps.

3.1.6 Artificial Neural Networks (ANN)

An approach for machine learning called Artificial Neural Networks (ANN) was developed as a result of studying brain neuron networks. It is made up of layers of interconnected artificial neurons known as nodes or units. ANN is used to model the learning and decision-making processes by drawing inspiration from the information that the human brain processes.

Before reaching the output layer, data must travel from the input layer through one or more hidden levels. Input signals from other neurons are received by each neuron in the network, processes those signals, and then produces an output signal. Weights that specify the signal's significance or strength define the connections between neurons.

An ANN can be trained by changing the link weights to enhance the network's ability to solve a particular problem. Backpropagation is a technique that is frequently used for this procedure. The network's outputs are compared to what was anticipated, and any discrepancies are subsequently sent backward through the network to change the weights.

In pattern identification, picture and speech recognition, natural language processing, recommendation systems, and financial forecasting, ANNs have a wide range of applications. Both classification and regression tasks employ it.

The ability to learn from data, generalize from patterns, and make predictions or judgments based on intricate relationships within the input data makes artificial neural networks potent tools in machine learning overall.

3.1.7 Algorithm of ANN

Layers of interconnected nodes, commonly referred to as artificial neurons or units, make up an ANN. The most common kind of ANN is a feedforward neural network, in which data travels in a single direction from the input layer to the output layer.

The following gives a step-by-step explanation of the ANN algorithm.

Architecture: An ANN is made up of several layers, including an input layer, one or more hidden layers, and an output layer. Each layer contains a specific number of neurons or units. Fig. 3.6 depicts the ANN's architecture.

Information moves through the network in a forward manner through a process called forward propagation. The input layer receives the input data, and the successive layers compute each neuron's activations until they reach the output layer.

Weighted Connections: Weights are used to describe the connections between neurons. The strength or significance of the signal sent between neurons is determined by these weights. The weights are modified throughout training to enhance the network's functionality.

Training: When an ANN is trained, the weights are changed to reduce the discrepancy between the expected and desired results. Backpropagation is a common method for accomplishing this, which updates the weights by calculating the error at the output layer and propagating it backward through the network.

Loss Function and Optimization: A loss function is used to calculate the discrepancy between the output that was expected and the output that was actually obtained. During training, the objective is to reduce this loss function. The weights are adjusted using optimization procedures, such as gradient descent, in a way that minimizes the loss.

Prediction: The ANN can be utilized for classification or prediction tasks once it has been trained. The network receives fresh input data, and the forward propagation mechanism uses the learnt weights to compute the output.

During training, ANNs can learn complicated patterns and correlations in the data by iteratively modifying the weights and biases. They have the ability to extrapolate from specific examples and make predictions based on unobserved data.

It's crucial to remember that the discussion above only gives a high-level understanding of how ANNs function. To increase the effectiveness and performance of ANNs in various tasks, a variety of designs, activation functions, training algorithms, and optimization strategies can be used.

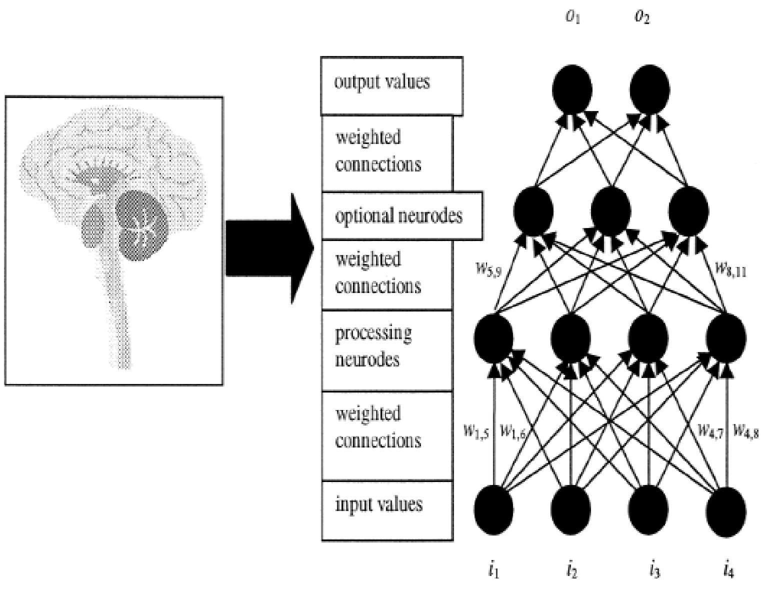


Figure 3.6: Sample artificial neural network architecture (not all weights are shown). Source: (Agatonovic-Kustrin and Beresford, 2000).

ANN Architecture The input layer, one or more hidden layers, and the output layer make up the ANN architecture. There are many nodes, also known as neurons, in each layer that are linked to one another by weights. While the output layer creates the output, the input layer receives the input data. The incoming data is transformed nonlinearly by the hidden layers.

The mathematical formula for the ANN architecture is as follows:

The weight between the i^{th} neuron in the $(k-1)^{th}$ layer and the j^{th} neuron in the k^{th} layer is w_{ij}^k. Let x_t be the input at time step t, y_t be the output at time step t, and w_{ij}^k be the weight. The bias term for the j^{th} neuron in the k^{th} layer is b_j^k. The activation function is f.

The output of the j^{th} neuron in the k^{th} layer is calculated as follows:

$$a_j^{(k)} = \sum_{i=1}^{n^{(k-1)}} w_{ij}^{(k)} y_i^{(k-1)} + b_j^{(k)} \tag{3.6}$$

where $n^{(k-1)}$ is the number of neurons in the $(k-1)^{th}$ layer.

The output of the j^{th} neuron in the k^{th} layer after the activation function is applied is:

$$y_j^{(k)} = f(a_j^{(k)}) \tag{3.7}$$

The output of the output layer is the final prediction.

ANN for Time Series Forecasting By considering the time series as a regression issue, where the objective is to predict the value at the next time step based on the values of the past time steps, ANN can be utilized for time series forecasting. This can be accomplished by employing a sliding window technique, in which the subsequent time step is predicted using a window of earlier time steps.

The following is the mathematical formula for ANN for time series forecasting:

A sequence of n time steps, where x_t represents the value at time t, is defined as $X = x_1, x_2, ..., x_n$. A sequence of target values, $y = y_1, y_2, ..., y_n$, where y_t is the value we wish to anticipate at time t. Then, we can divide the sequence X and targets y into training and testing sets using a sliding window technique. Using cross-validation, the window size and the lag can be adjusted.

An ANN model that has been trained on the training data may be used to predict the value at the next time step based on the values from the previous time steps. The model may be trained using a variety of optimization techniques, including stochastic gradient descent (SGD).

3.1.8 Recurrent neural network (RNN)

The ANN takes a fixed-length input and returns fixed lengths out however, RNN is a modified form of ANN used for time series analysis and forecasting capable of taking variable-length input and returning variable-length outputs. This feature make it suitable for time series analysis which can have a different number of observations over time.

The key characteristic of ANN is that it functions like a memory-type algorithm, recalling previous observations and patterns to predict future values. The network maintains a concealed state that is updated at each time step through recurrent connections. The network can store temporal relationships in the data using the hidden state as a kind of memory.

An RNN's mathematical formula is represented as follows:

$$\begin{aligned} h_t &= f(Wx_t + Uh_{t-1} + b) \\ y_t &= g(Vh_t + c) \end{aligned} \tag{3.8}$$

Where W, U, and V are weight matrices, b and c are bias vectors, f and g are activation functions, and the superscript T stands for a matrix's transpose. The hidden state at time t is represented as ht, the input is represented as xt, and the output is represented as yt.

The hidden state at time t is computed as a function of the input at time t (xt), the hidden state at the preceding time step ($ht - 1$), and biases (b). After that, the hidden state (ht) and biases (c) at time t are determined as a function of the output. The activation functions f and g are often chosen from nonlinear functions like the sigmoid or hyperbolic tangent function.

To lessen the gap between the predicted output and the actual output for a particular input sequence, the weights and biases of the RNN are altered during the training phase. This frequently involves the use of backpropagation through time (BPTT), a development of the backpropagation technique employed in traditional feedforward neural networks.

3.1.9 Long Short-Term Memory (LSTM)

Recurrent neural networks (RNNs) of the Long Short-Term Memory (LSTM) variety can be applied to time series analysis and prediction. It is made to get over the issue of vanishing gradients that regular RNNs have, which prevents them from capturing long-term relationships in sequential data. We will describe LSTM for time series and its mathematical formula in this part.

LSTM Architecture The input, output, and forget gates, along with a memory cell, make up the LSTM architecture. The gates control how information enters and exits the memory cell, which stores data over time. The input gate chooses how much new data should be fed into the cell, while the forget gate chooses how much old data should be forgotten. How much information should be produced from the cell is decided by the output gate.

The following are the mathematical formulas for the LSTM architecture:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

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

$$\tilde{c}t = tanh(W_c x_t + U_c ht - 1 + b_c)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

$$h_t = o_t \odot tanh(c_t)$$

where x_t is the input at time t, h_{t-1} is the output of the previous time step, i_t, f_t, and o_t are the input, forget, and output gates at time t, respectively, $\tilde{c}_t$ is the candidate cell state at time t, c_t is the cell state at time t, h_t is the output at time t, and $\odot$ denotes element-wise multiplication. W_i, W_f, W_o, W_c, U_i, U_f, U_o, and U_c are weight matrices and b_i, b_f, b_o, and b_c are bias vectors.

3.1.10 Working procedure of ANN

Recurrent neural network (RNN) architectures with Long Short-Term Memory (LSTM) are made to manage long-term dependencies and record sequential information in data. LSTMs, as opposed to conventional feedforward neural networks, feature memory cells that can store and update data over time.

The working method of LST is described below

- Memory Cell: The memory cell, which is a crucial component of an LSTM, is made up of the three multiplicative gates input gate, forget gate, and output gate.

- Gates: An LSTM's gates control the flow of information by allowing only specific information to pass. A sigmoid activation function on each gate produces output values between 0 and 1.
- Input Gate: The input gate determines how much new data should be added to the cell state. To decide which values should be modified, it applies a sigmoid activation function to the current input and the previous hidden state.
- Forget Gate: The forget gate decides what data from the previous cell state should be erased. To decide which data should be discarded, it applies a sigmoid activation function to the current input and the previous hidden state.
- Cell State: The cell state is a representation of the LSTM's long-term memory. It is updated based on the forget gate and input gate. The input gate regulates how much new data is fed into the cell state, while the forget gate regulates how much previous data is retained.
- Output Gate: How much information from the cell state should be made available as the LSTM's output is controlled by the output gate. By using a sigmoid activation function on the input from the current input and the previous hidden state, the values that should be produced are determined.
- Hidden State: The LSTM's output is the hidden state. It is a filtered representation of the cell state that the output gate controls. The pertinent data gleaned from the input sequence is stored in the concealed state.

The gradients flow backward through the time steps when the LSTM is trained via backpropagation across time. In order to recognize relationships and patterns in sequential input, the LSTM may then learn to update the gates and the cell state.

LSTMs have shown to be successful in a variety of applications involving sequential data, including time series analysis, machine translation, speech recognition, and natural language processing. They are ideally suited for activities where context and memory are critical because of their capacity to capture long-term dependencies.

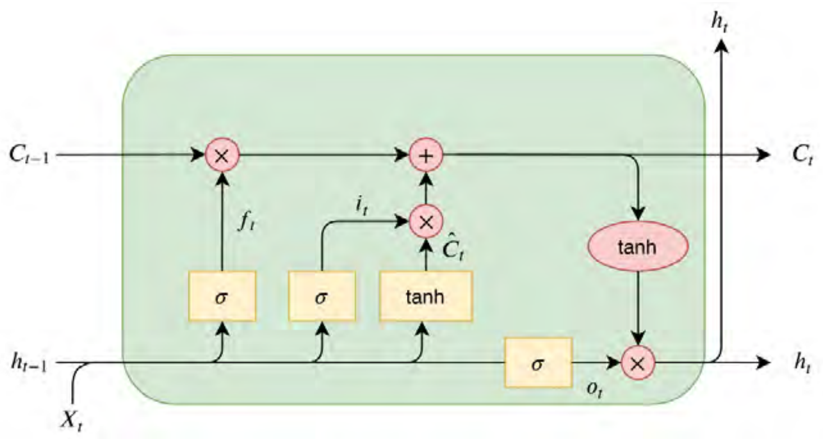


Figure 3.7: Architecture of LSTM. Source: (Nowak et al., 2017)

LSTM for Time Series Forecasting By using historical time series data to forecast future values, LSTM may be utilized for time series forecasting. A series of time steps serve as the LSTM network's input, and its output is the projected value for the subsequent time step. Backpropagation through time (BPTT), a backpropagation variant that takes into consideration the sequential nature of the data, is used to train the network.

The mathematical formula for LSTM for time series forecasting is as follows:

Let $X = x_1, x_2, ..., x_n$ be a sequence of n time steps, where x_t is the value

3.1.11 Gated Recurrent Unit (GRU)

Recurrent neural networks (RNNs) with gated recurrent units (GRUs) are frequently employed for time series analysis and prediction. The long short-term memory (LSTM) model and GRU are comparable. But it has fewer parameters and is easier to train. In this section, we will explain GRU for time series and its mathematical formula.

GRU Architecture The update gate and reset gate make up the GRU architecture, together with a hidden state. The amount of the prior concealed state to maintain and the amount of the new candidate state to add are decided by the update gate. How much of the prior concealed state should be forgotten is decided by the reset gate.

The mathematical formula for the GRU architecture is as follows:

Assume that h_t is the hidden state at time step $t-1$, h_t is the hidden state at time step t, and x_t is the input at time step t. Assume that the update gate is z_t and the reset gate is r_t. Let b be the bias vector and W be the weight matrix.

The definitions of the update gate and reset gate are as follows:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{3.9}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{3.10}$$

where σ is the sigmoid function.

The candidate hidden state $\tilde{h_t}$ is defined as follows:

$$\tilde{h_t} = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{3.11}$$

where $\odot$ is the element-wise product.

The hidden state h_t is then updated as follows:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h_t} \tag{3.12}$$

3.1.12 Algorithm of GRU

Here is an overview of the algorithm of GRU:

- Initialize the GRU: Set the initial hidden state vector, h, to zeros or to some predefined values. The knowledge gained from the earlier processes is stored in the hidden state.
- Update Gate: Calculate the update gate, z. This gate decides how much of the previous concealed state should be preserved and how much should be updated with fresh information. A sigmoid activation function is applied to the update gate's inputs, the current input (x) and the prior hidden state (h). It outputs values between 0 and 1.
- Reset Gate: Do the reset gate's r calculation. The amount of the prior concealed state that has to be forgotten or reset is determined by this gate. The prior hidden state, h, and the current input, x, are sent into the reset gate by a sigmoid activation function. It produces values ranging from 0 to 1.
- Current Memory: Compute the current memory content, c. At this phase, the reset gate, r, and the current input, x, are combined. The current input offers fresh information, and the reset gate chooses which portions of the prior concealed state to erase. The result is processed via a non-linear activation function, such as the hyperbolic tangent ($tanh$), before being multiplied element-wise with the current input and reset gate.
- Current Hidden State: Determine the hidden state's current value, $tildeh$. The update gate, Z, together with the prior hidden state, H, and the current memory content, $tildec$, are combined in this phase. The amount of the prior concealed state to retain is decided by the update gate, while fresh information is provided by the contents of the present memory. A non-linear activation function, such as the hyperbolic tangent ($tanh$), is used to combine the update gate, the prior hidden state, and the current memory content.
- Hidden State Update: Please update the hidden state. In this phase, the update gate, z, and the current hidden state, $barh$, are combined to create the updated hidden state, h. The contribution of the prior hidden state and the present hidden state is determined by the update gate. The product of $(1 - z)$ and the previous hidden state is added to the product of the update gate and the current hidden state.
- Repeat: Repeat steps 2 to 6 for each input in the sequence.

The GRU technique enables the network to capture long-term dependencies and handle sequential input efficiently by allowing it to selectively update and forget information from the prior hidden state. GRUs have been effectively used for a variety of applications, including time series analysis, machine translation, speech recognition, and natural language processing.

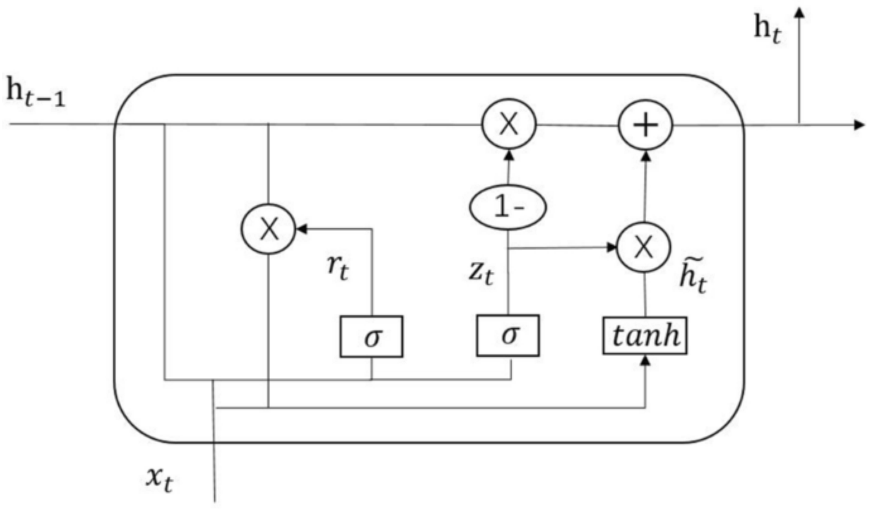


Figure 3.8: Architecture of GRU. Source: (Fu et al., 2016)

GRU for Time Series Forecasting The purpose of using GRU for time series forecasting is to predict the value at the next time step based on the values of the past time steps. This is done by considering the time series as a sequence-to-sequence issue. This can be done by using a sliding window approach, where a window of past time steps is used to predict the next time step.

The mathematical formula for GRU for time series forecasting is as follows:

Let $X = x_1, x_2, ..., x_n$ be a sequence of n time steps, where x_t is the value at time t. Let $y = y_1, y_2, ..., y_n$ be a sequence of target values, where y_t is the value we want to predict at time t. We can then use a sliding window approach to split the sequence X and targets y into training and testing sets. The window size and the lag can be tuned using cross-validation.

To make predictions, we can train a GRU model on the training data and use it to predict the value at the next time step based on the values of the previous time

3.2 Conventional Time series models

3.2.1 Auto Regressive Integrated Moving Average (ARIMA).

For the analysis and forecasting of time-dependent data, the popular time series forecasting model ARIMA (Autoregressive Integrated Moving Average) is frequently utilized. It combines moving average, differencing, and autoregressive (AR) techniques (MA). In order to capture the patterns and properties of the time series data, each component has a defined function.

1. Autoregressive (AR) Component: According to the autoregressive component, the present observation is dependent on earlier observations. It simulates the link between a linear combination of an observation's lag values and the observation itself. When "p" stands for the order of autoregression, the AR component of ARIMA is indicated as AR(p). The AR component's mathematical formula is denoted by the following symbols:

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ... + \phi_p Y_{t-p} + \varepsilon_t$$

Here, Y_t represents the time series at time "t", ϕ_1 to ϕ_p are the autoregressive coefficients, Y_{t-1} to Y_{t-p} represent the lagged values, and ε is the error term.

2. Differencing (I) Component: Non-stationarity of the time series is accounted for by the differencing component of ARIMA. It entails differencing the time series data to remove trends and seasonality and make it stationary. I(d), where "d" stands for the order of differencing, is used to denote the differencing operation. Subtracting the prior observation from the present observation results in the differenced time series, which may then be repeated d times until stationarity is reached.

3. Moving Average (MA) Component: The moving average component takes into account how much the current observation depends on the residuals or mistakes from earlier observations. The weighted sum of the earlier error factors is modelled. The order of the moving average is represented by "q," which is how the MA component of ARIMA is denoted. The MA component's mathematical formula has the following form:

$$Y_t = \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_p \varepsilon_{t-q} + \varepsilon_t$$

Here, θ_1 to θ_p are the moving average coefficients, ε_1 to ε_q represent the previous error terms, and ε is the current error term.

The ARIMA model, which is denoted by the notation ARIMA(p, d, q), is created by combining the AR, I, and MA components. Based on the properties of the time series data and statistical methods like autocorrelation and partial autocorrelation analysis, the parameters "p", "d", and "q" need to be calculated.

In the ARIMA model, the coefficients $(phiand$theta$)areestimated, andthebestvaluesof$(p), ($d$), and ($q$) are chosen using methods such maximum likelihood estimation or least squares. Future values of the time series may be predicted using the model after it has been fitted.

The mathematical form of the ARIMA model can be represented as follows:

$$Y_t = \mu + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ..._+ \phi_p Y_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q} + \epsilon_t \tag{3.13}$$

where

The value of the time series at time t is Y_t.

The ARIMA model's constant or intercept is μ.

The autoregressive (AR) coefficients $\phi_1, \phi_2, ..., phi_p$ show how previous values of the time series have an impact on the current value.

Moving average (MA) coefficients $theta_1, theta_2, ..., \theta_q$ indicate the residual effects of previous mistakes on the current value. And

The error term or residual at time t is *epsilont*.

The mathematical calculations involved in ARIMA modeling include parameter estimation, residual analysis, model diagnostics, and forecasting. These calculations are performed using statistical programming languages r, where specialized libraries like `tseries` or `forecast` are available to facilitate ARIMA modeling.

3.3 The Data

The data used in this study comes from Yahoo! Finance (2022) and includes daily USD prices for gold, bitcoin, and crude oil from October 2016 through October 2022.

Table 3.1: Descriptive statistics of gold, BTC, and crude oil daily prices

Statistic	Minimum	Mean	Maximum	Median	SD
Gold	1128	1530	2052	1490	265.2326
Bitcoin	619	16969	67567	9249	17039.96
crude oil	-37.63	61.28	123.70	58.02	18.8686

The Tables 3.1 provide descriptive statistics for summarizing the distribution's characteristics, including the minimum, mean, maximum, median, and standard deviation (SD). Training and testing sets of the data were created. The training set of three data sets is taken from October 2016 to October 2021 while the testing part is taken from October 2021 to October 2022, ensuring a large amount of data for training and covering a wide range of short and long-term trends. The testing set covers daily prices from October 2021 to October 2022 (1 year), ensuring that the forecasting models are evaluated using a significant amount of previously unreported "out-of-sample" data.

Chapter 4

Results and Analysis

The outcomes of traditional time series models and machine learning algorithms used to estimate the values of gold, bitcoin, and crude oil are displayed and thoroughly explained in this chapter. The models are asses by various error measurements such as RMSE, MAE, and MAPE. The efficient model has a more accurate price prediction and low error measurements.

4.1 BTC Price Data

The BTC price data has been forecast through machine learning and conventional time series models and assessed the performance matrix through MSE, MAE, and MAPE as how in Table 4.10.

The BTC price shown in Fig 4.1 has a positive linear trend until 10 October 2021. The prices show a downward trend after 10 October 2021. The prices are recorded daily and partitioned the whole data into testing and training parts. The daily data from 10 October 2016 to 10 October 2021 is taken as the training part while the one-year data from 11 October 2021 to 10 October 2022 is taken testing part which is forecasted through one step ahead forecasting by the following machine learning and conventional time series models.

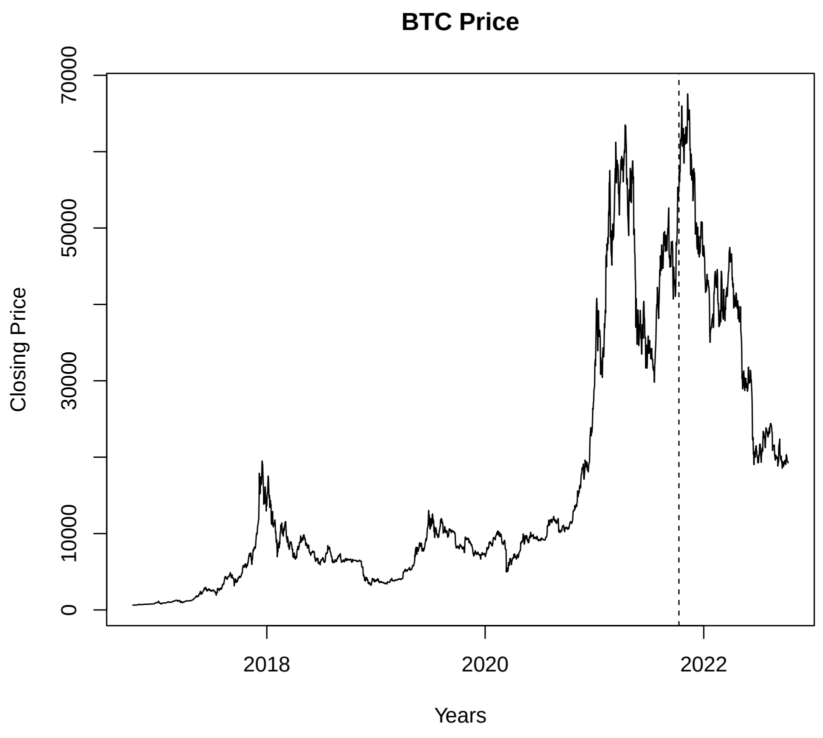


Figure 4.1: BTC daily price data from 10 October 2016 till 10 October 2022. The dashed line shows the testing and training boundary on 10 October 2021

Random Forest

The random forest model is developed on the training set to forecast the training set. The lags of one step previous are calculated to make an independent variable. The parameters of random forests such as mtry, number of trees, and pruning are performed through cross-validation and build a final model on optimum parameters that provide the lowest error rate.

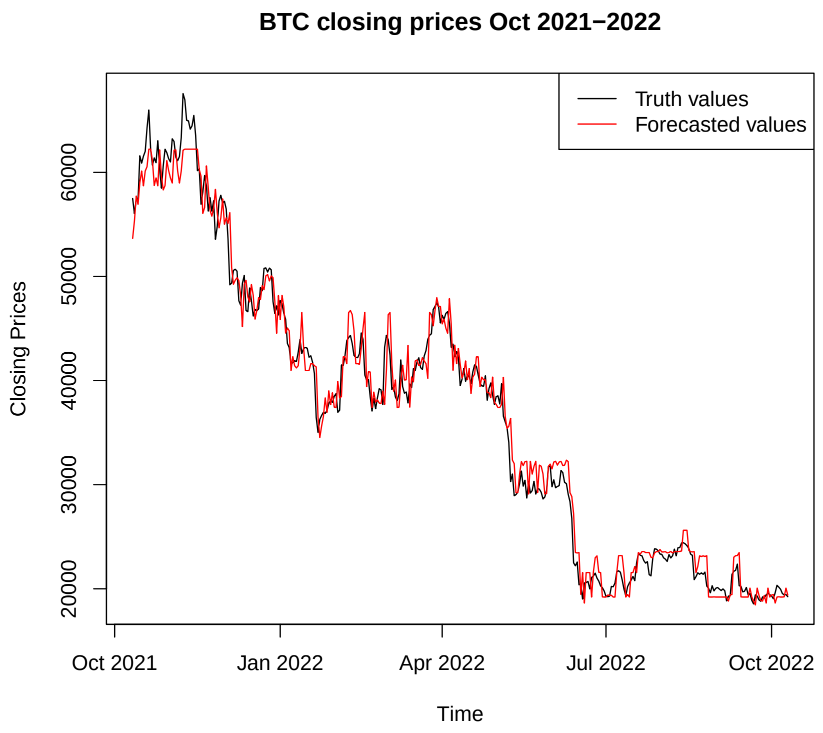


Figure 4.2: The plot shows observed and forecasted value of BTC closing prices through the random forest by one step ahead forecasting methods

Table 4.1: Error measure of Random forest

Method	MAE	RMSE	MAPE
RF	1379.25	1803.47	3.908

k-nearest neighbor (k-NN)

The k-NN algorithm is built on a training dataset by choosing the optimum k value through cross-validation. The year 2022 price data of BTC is forecast through a trained model and computed the MSE, MAE, and MAPE. The following Fig. 4.3 shows the observed (black) and forecasted value (red)

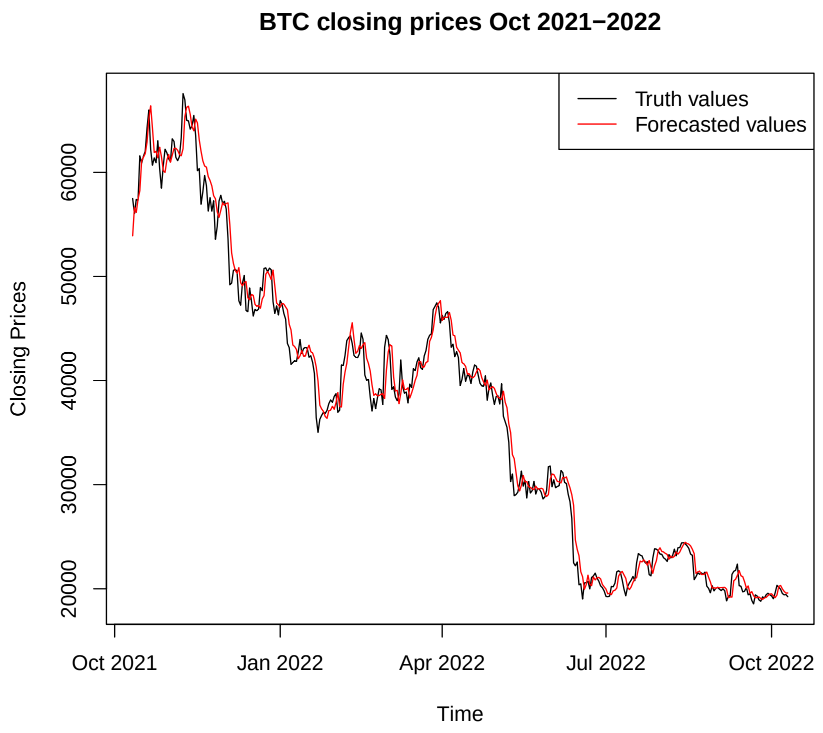


Figure 4.3: The plot shows the observed and forecasted value of BTC closing prices through k-NN by one step ahead of forecasting methods.

Table 4.2: Error measure of k-NN

Method	MAE	RMSE	MAPE
k-NN	1106.34	1547.08	3.071

Support Vector Machines (SVM)

The SVM linear model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of BTC is forecast through a trained model and computed the MSE, MAE, and MAPE. The following Figure 4.4 shows the observed (black) and forecasted value (red) through SVM

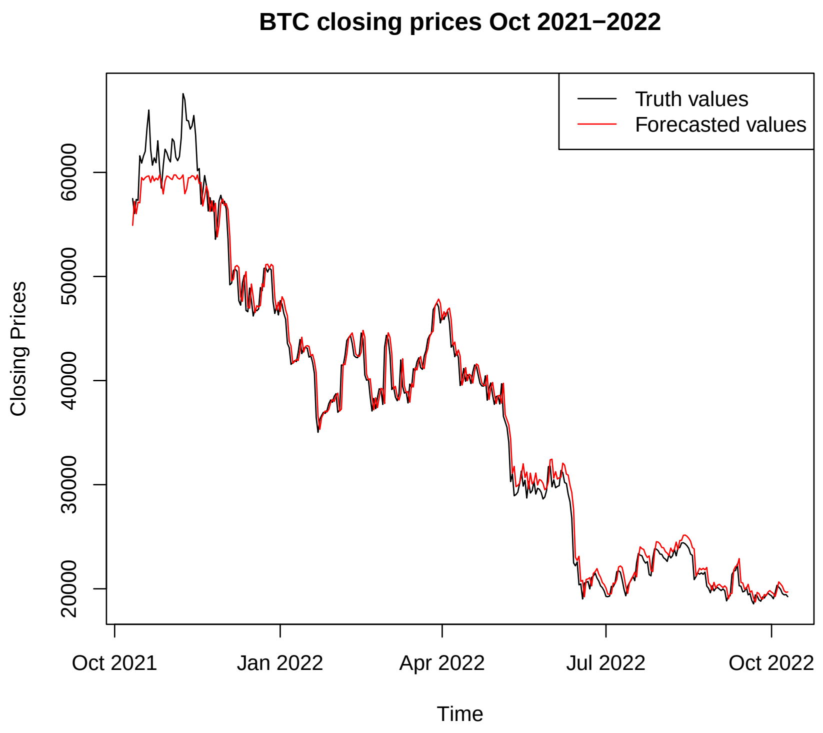


Figure 4.4: The plot shows the observed and forecasted value of BTC closing prices through SVM by one step ahead of forecasting methods.

Table 4.3: Error measure of SVM

Method	MAE	RMSE	MAPE
SVM	1162.83	1229.64	3.112

Artificial Neural Networks (ANN)

The ANN model is built on a training dataset by choosing the optimum parameters through cross-validation. The data is standardized through minmax scaling. The one-step previous lags are computed and used as predictors, removing the first observation of the training and testing set as the ANN cannot handle missing observations. The 2022 price data of BTC is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through ANN

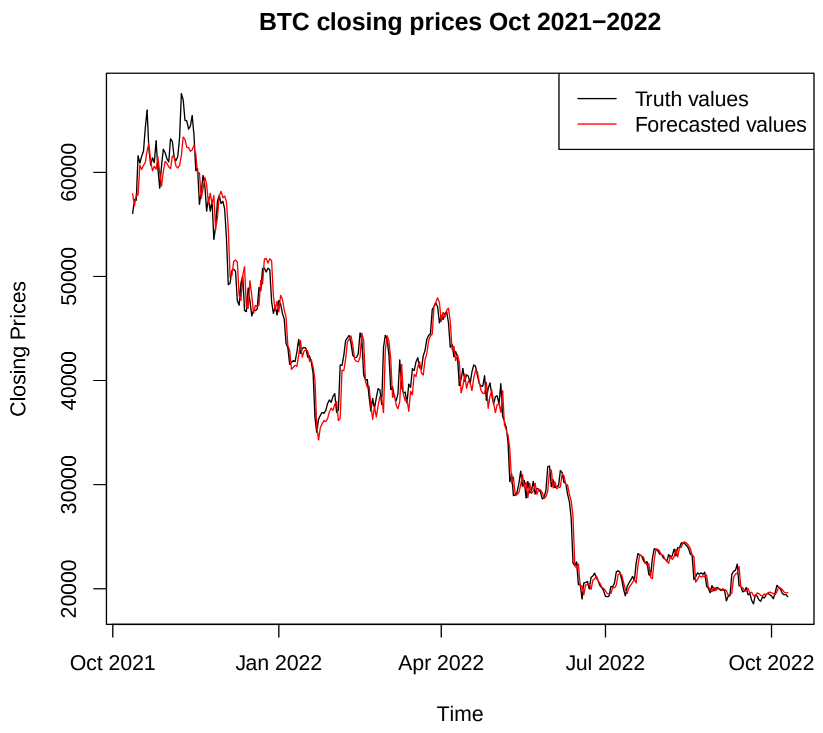


Figure 4.5: The plot shows the observed and forecasted value of BTC closing prices through ANN by one step ahead of forecasting methods.

Table 4.4: Error measure of ANN

Method	MAE	RMSE	MAPE
ANN	895.35	1299.12	2.451

Recurrent Neural Networks (RNN)

The RNN model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of BTC is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through ANN

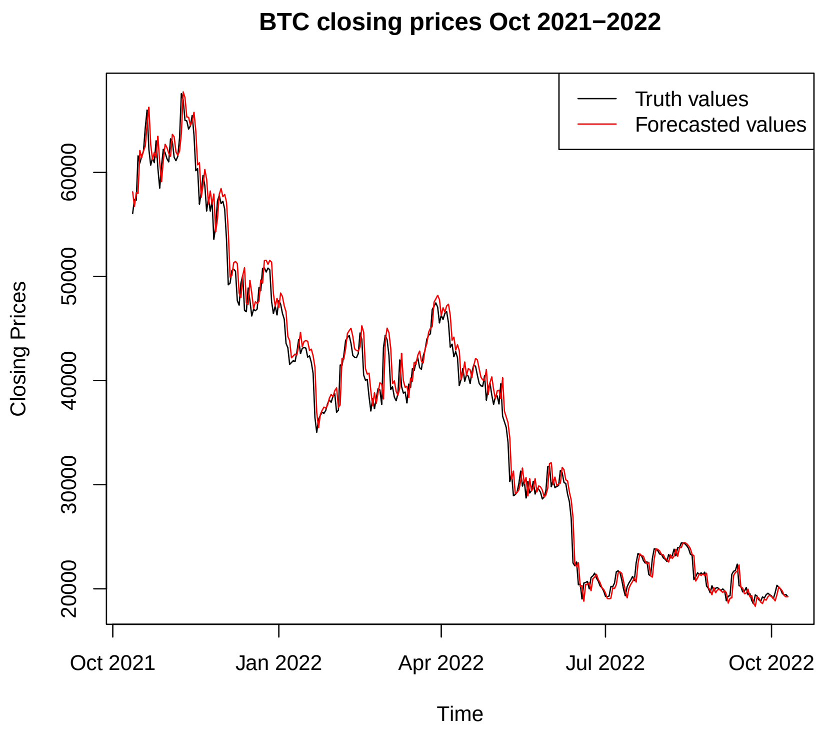


Figure 4.6: The plot shows the observed and forecasted value of BTC closing prices through RNN by one step ahead of forecasting methods.

Table 4.5: Error measure of RNN

Method	MAE	RMSE	MAPE
RNN	893.89	1298.89	2.445

Long Short-Term Memory (LSTM)

The LSTM model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of BTC is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through LSTM

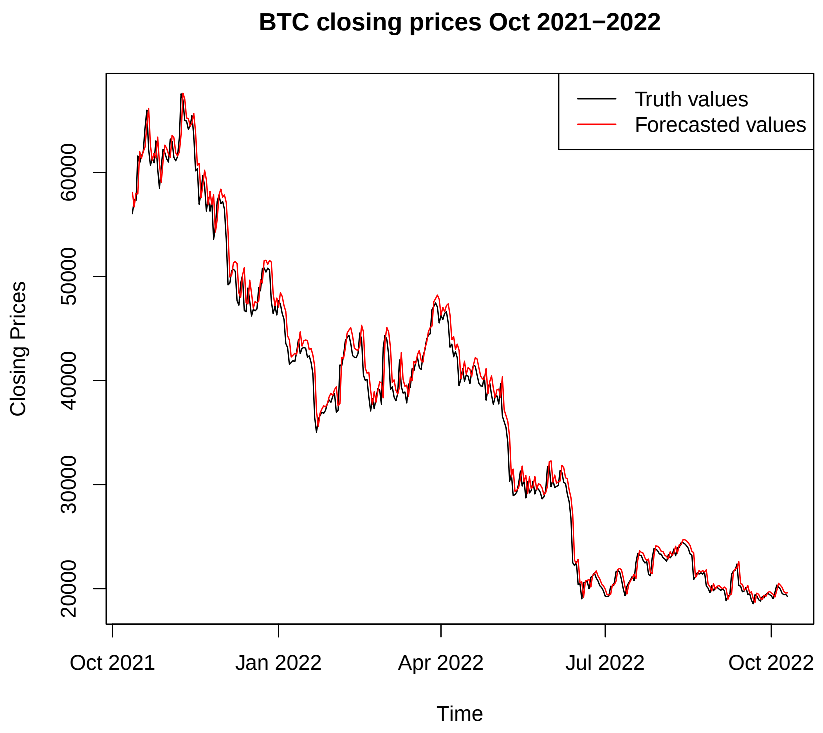


Figure 4.7: The plot shows the observed and forecasted value of BTC closing prices through LSTM by one step ahead of forecasting methods.

Table 4.6: Error measure of LSTM

Method	MAE	RMSE	MAPE
RF	929.14	1307.03	2.612

Gated Recurrent Unit (GRU)

The GRU model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of BTC is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through GRU

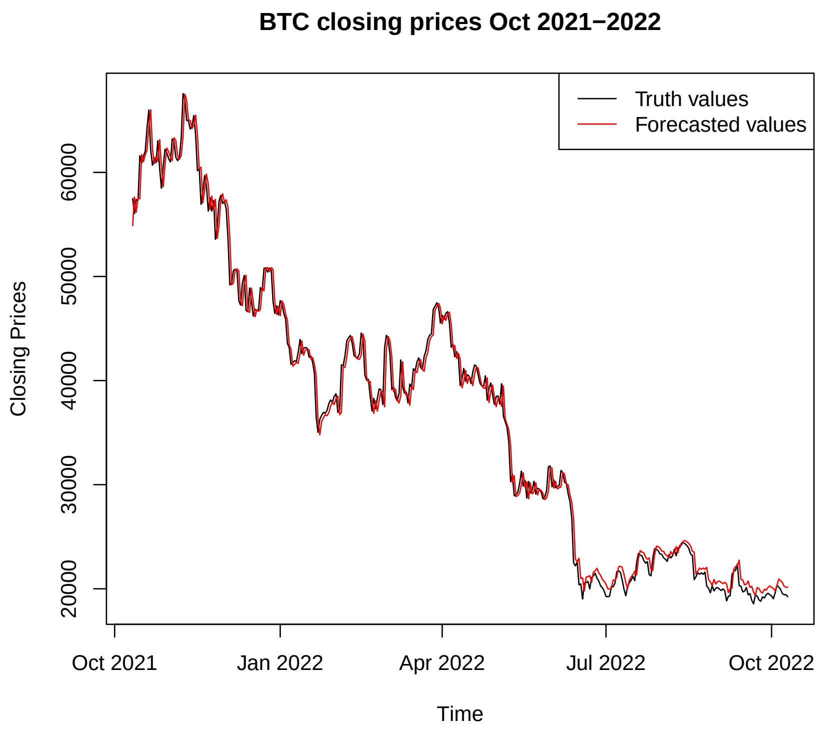


Figure 4.8: The plot shows the observed and forecasted value of BTC closing prices through GRU by one step ahead of forecasting methods.

Table 4.7: Error measure of GRU

Method	MAE	RMSE	MAPE
RF	976.03	1344.12	2.809

Auto Regressive Integrated Moving Average (ARIMA)

The ARIMA model is developed by computing the Autoregressive (p) through AR plots in which the order is taken in which the current observation has an insignificant relationship with the corresponding lag value. Similarly, the moving average (q) is calculated through PACF plots. The integrating term (d) is taken as 1 since the time series has a linear trend. The optimum parameters of ARIMA are found through trial and error methods. The one-step ahead forecasting is performed on the ARIMA model to forecast the training set based on the lag value which trained the ARIMA model on training data. The optimum parameters of ARIMA is computed through a trial and error method by AIC criteria. The parameters are selected whose AIC is measured low. The results of each model show that the ARIMA model of order p=0, d=1, and q=2 provides the lowest AIC.

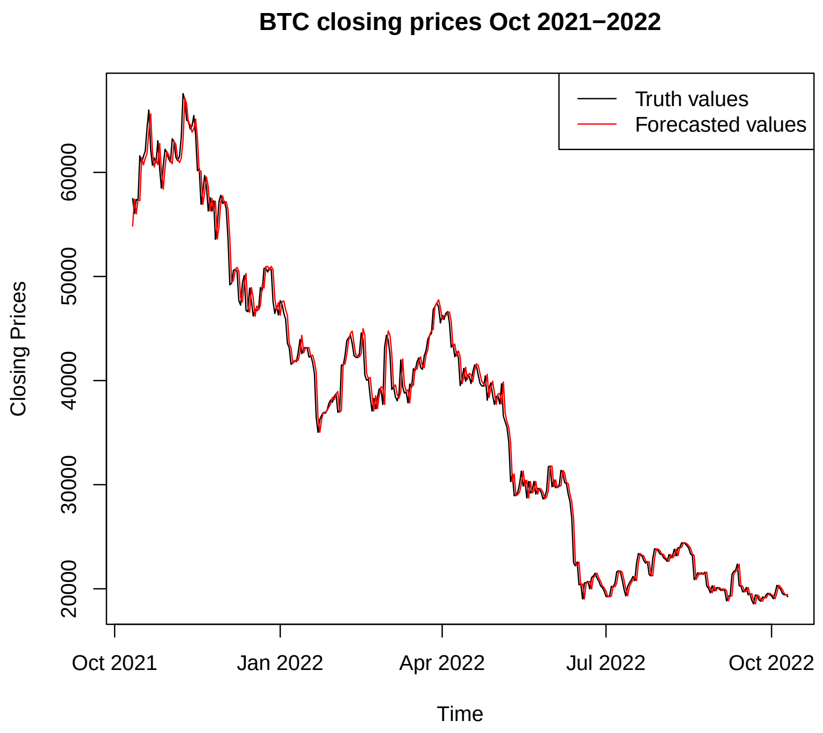


Figure 4.9: The plot shows the observed and forecasted value of BTC closing prices through ARIMA(0,1,2) by one step ahead of forecasting methods.

The following table shows that ARIMA outperforms many popular machine learning algorithms. This is because ARIMA is a linear time series model and if the data fulfill the assumptions of linearity and normality, then it outperforms complex nonlinear machine learning algorithms. In this data, we transform the data by making it stationary through a first-order lag difference and making it stationary and linear; therefore, it outperforms the complex nonlinear machine learning algorithms such as SVM, GRU, and LSTM.

Table 4.8: Error measure of ARIMA

Method	MAE	RMSE	MAPE
ARIMA	904.33	1311.91	2.468

The following Table 4.9 shows the error rate of all machine learning time series models and conventional ARIMA. The error is measured by MAE, RMSE, and MAPE. It can be seen from the table that the RNN has a narrow low error in comparison to other networks such as ANN, LSTM, and GRU. The ARIMA almost performs accurately however, Random forest, k-NN, and SVM outperform on forecasting the daily price of BTC.

Table 4.9: Error metrics of different time series methods on BTC prices

Methods	MAE	RMSE	MAPE
Random Forest	1379.25	1803.47	3.908
kNN	1106.34	1547.08	3.071
SVM	1162.83	1729.64	3.112
Neural Network	895.35	1299.12	2.451
RNN	**893.89**	1298.29	**2.445**
LSTM	929.14	1307.03	2.612
GRU	976.03	1344.12	2.809
ARIMA	904.33	1311.91	2.468

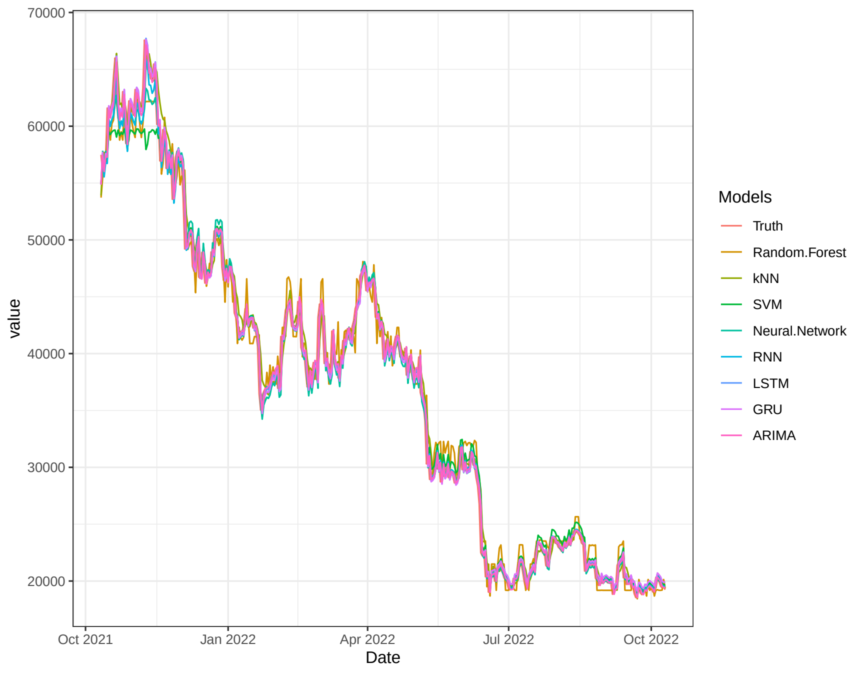


Figure 4.10: The plot shows the observed and forecasted value of BTC closing prices through all models by one step ahead of forecasting methods.

4.2 Gold Prices Data.

The gold data are taken from Yahoo! Finance (2022) from 10 October 2016 to 10 October 2022 on a daily basis excluding weekends. The data is divided into training and testing parts. The training parts contain 1251 observations from 10 October 2016 to 30 September 2021. The testing parts has 263 observation from 1 October 2021 to 10 October 2022 which is forecasted by the following ML and time series models through the one-step ahead forecasting method. The performance of each algorithm and model is shown in Table 4.20.

The Gold price has an overall positive linear trend until 30 September 2021 as shown in Fig. 4.11. The prices have been observed with a downward trend after 10 October 2021.

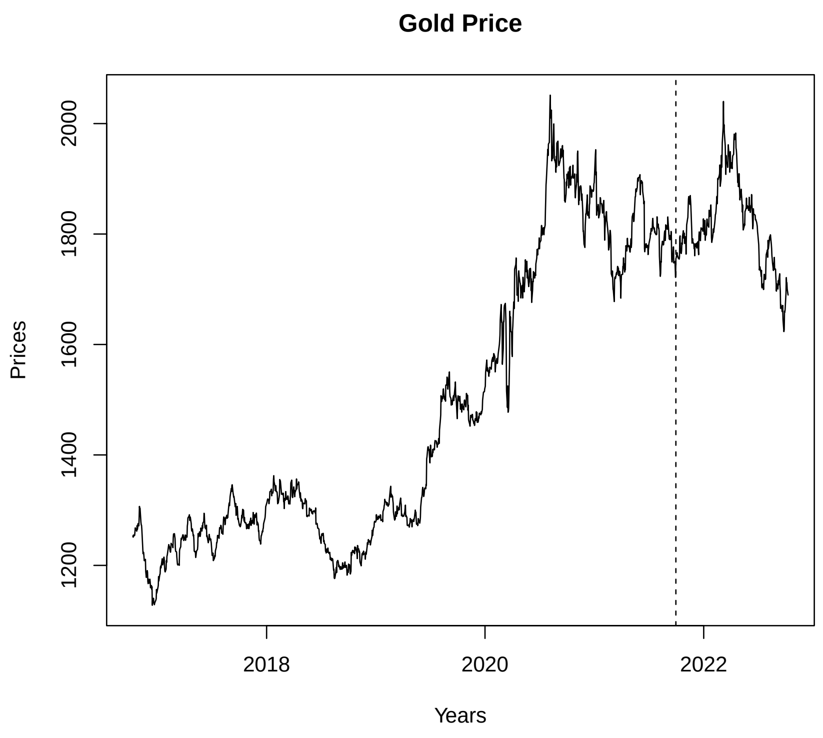


Figure 4.11: Gold daily price data of working days from 10 October 2016 till 10 October 2022. The dashed line shows the testing and training boundary on 30 September 2021

Random Forest

The random forest model is developed on the training set to forecast the training set. The lags of one step previous are calculated to make an independent variable. The parameters of random forests such as mtry, number of trees, and pruning are performed through cross-validation and build a final model on optimum parameters that provide the lowest error rate.

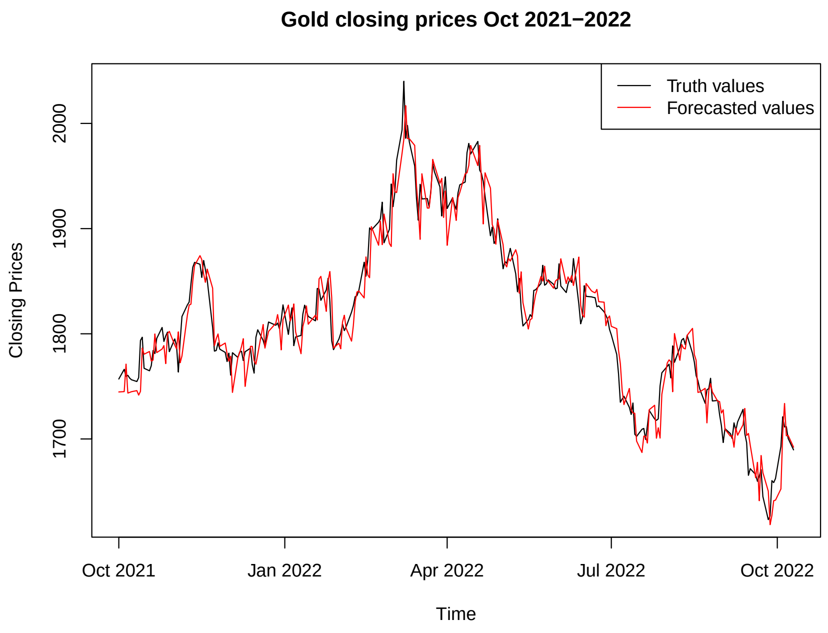


Figure 4.12: The plot shows observed and forecasted value of gold closing prices through the random forest by one step ahead forecasting methods

Table 4.10: Error measure of Random forest

Method	MAE	RMSE	MAPE
RF	14.605	19.022	0.805

k-nearest neighbor (k-NN)

The k-NN algorithm is built on a training dataset by choosing the optimum k value through cross-validation. The year 2022 price data of gold is forecast through a trained model and computed the MSE, MAE, and MAPE. The following Fig. 4.13 shows the observed (black) and forecasted value (red)

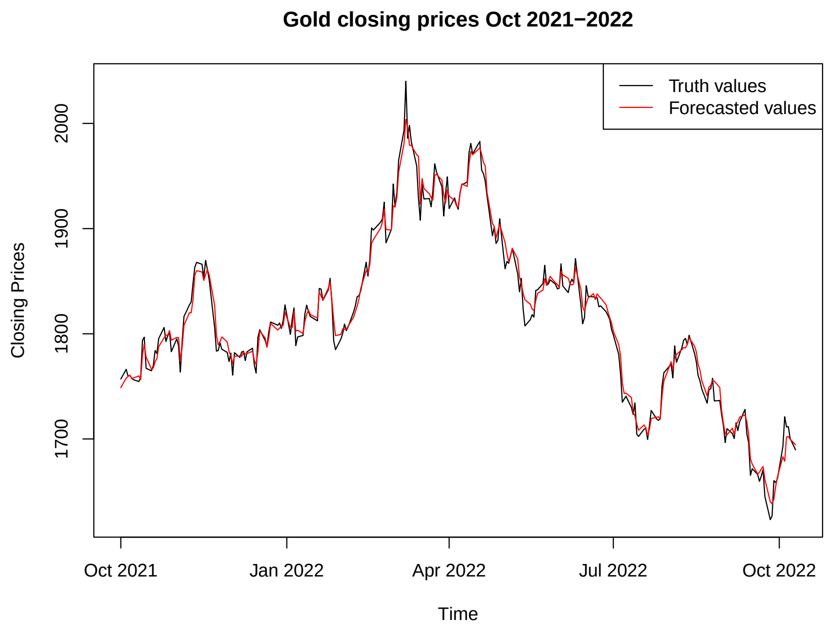


Figure 4.13: The plot shows the observed and forecasted value of gold closing prices through k-NN by one step ahead of forecasting methods.

Table 4.11: Error measure of k-NN

Method	MAE	RMSE	MAPE
k-NN	7.129	9.435	0.393

Support Vector Machines (SVM)

The SVM linear model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of BTC is forecast through a trained model and computed the MSE, MAE, and MAPE. The following Figure 4.14 shows the observed (black) and forecasted value (red) through SVM

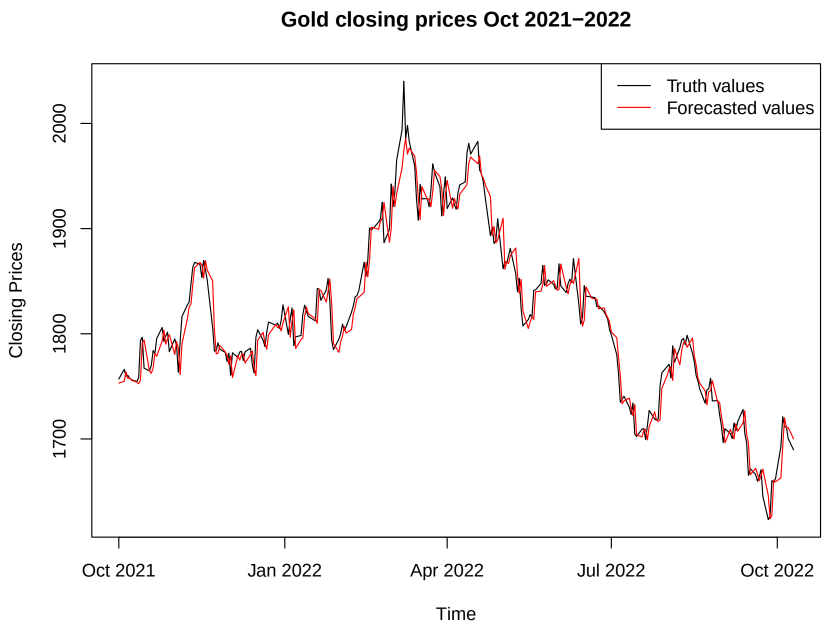


Figure 4.14: The plot shows the observed and forecasted value of gold closing prices through SVM by one step ahead of forecasting methods.

Table 4.12: Error measure of SVM

Method	MAE	RMSE	MAPE
SVM	12.506	16.428	0.686

Artificial Neural Networks (ANN)

The ANN model is built on a training dataset by choosing the optimum parameters through cross-validation. The data is standardized through minmax scaling. The one-step previous lags are computed and used as predictors, removing the first observation of the training and testing set as the ANN cannot handle missing observations. The 2022 price data of gold is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through ANN

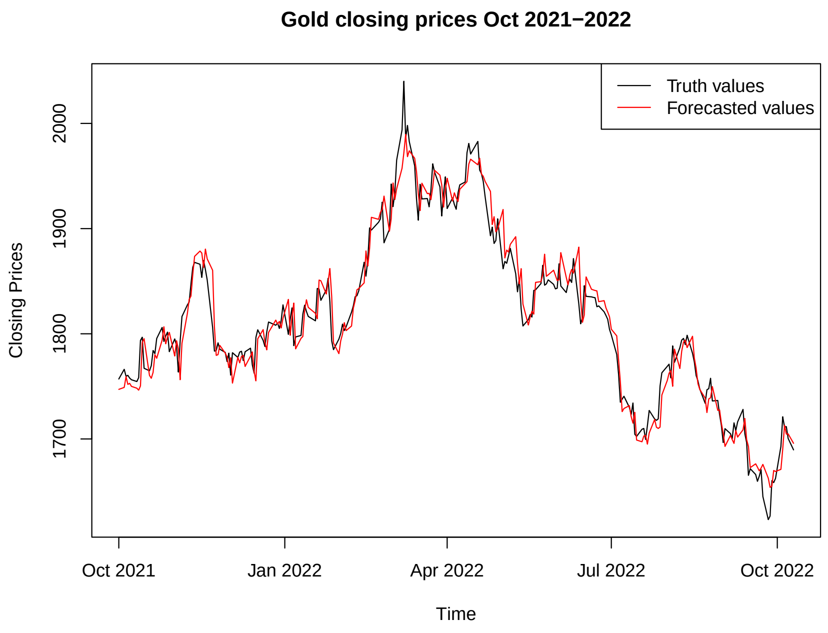


Figure 4.15: The plot shows the observed and forecasted value of gold closing prices through ANN by one step ahead of forecasting methods.

Table 4.13: Error measure of Random forest

Method	MAE	RMSE	MAPE
RF	12.504	16.428	0.686

Recurrent Neural Networks (RNN)

The RNN model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of gold is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through ANN

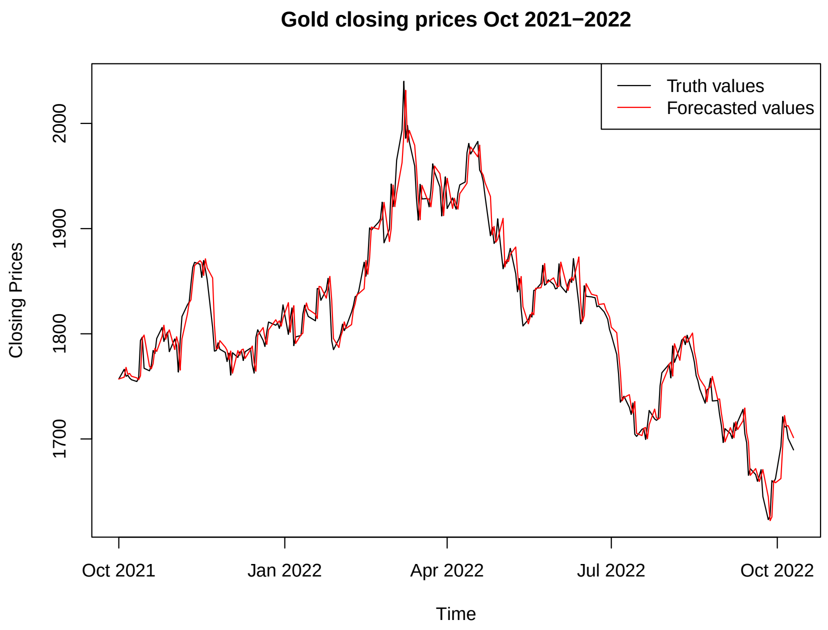


Figure 4.16: The plot shows the observed and forecasted value of gold closing prices through RNN by one step ahead of forecasting methods.

Table 4.14: Error measure of RNN

Method	MAE	RMSE	MAPE
RNN	12.757	16.682	0.701

Long Short-Term Memory (LSTM)

The LSTM model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of gold is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through LSTM

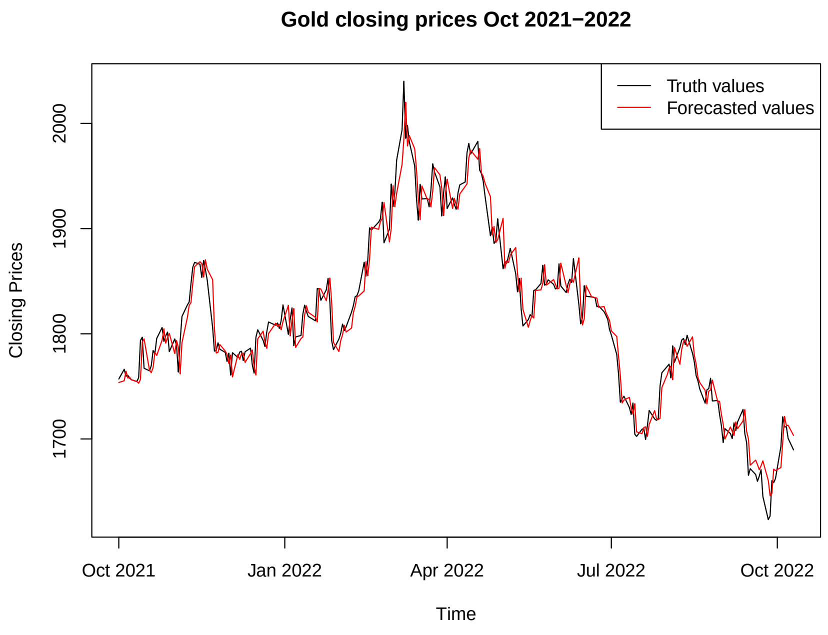


Figure 4.17: The plot shows the observed and forecasted value of gold closing prices through LSTM by one step ahead of forecasting methods.

Table 4.15: Error measure of LSTM

Method	MAE	RMSE	MAPE
LSTM	12.699	16.501	0.698

Gated Recurrent Unit (GRU)

The GRU model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of gold is forecast through trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through GRU

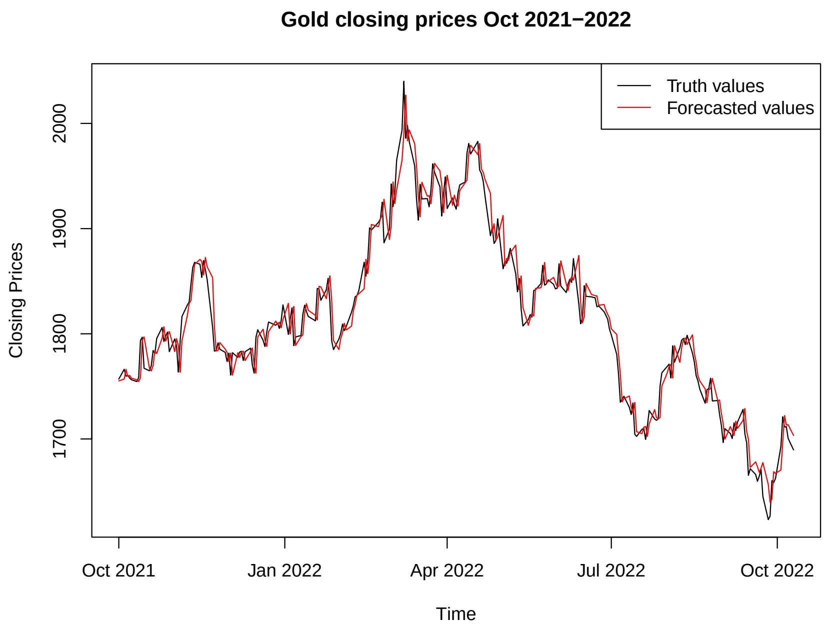


Figure 4.18: The plot shows the observed and forecasted value of gold closing prices through GRU by one step ahead of forecasting methods.

Table 4.16: Error measure of GRU

Method	MAE	RMSE	MAPE
GRU	12.639	16.385	0.694

Auto Regressive Integrated Moving Average (ARIMA)

The ARIMA model is developed by computing the Autoregressive (p) through AR plots in which the order is taken in which the current observation has an insignificant relationship with the corresponding lag value. Similarly, the moving average (q) is calculated through PACF plots. The integrating term (d) is taken as 1 since the time series has a linear trend. The optimum parameters of ARIMA are found through trial and error methods. The one-step ahead forecasting is performed on the ARIMA model to forecast the training set based on the lag value which trained the ARIMA model on training data. The optimum parameters of ARIMA is computed through a trial and error method by AIC criteria. The parameters are selected whose AIC is measured low. The results of each model show that the ARIMA model of order p=0, d=1, and q=0 provides the lowest AIC.

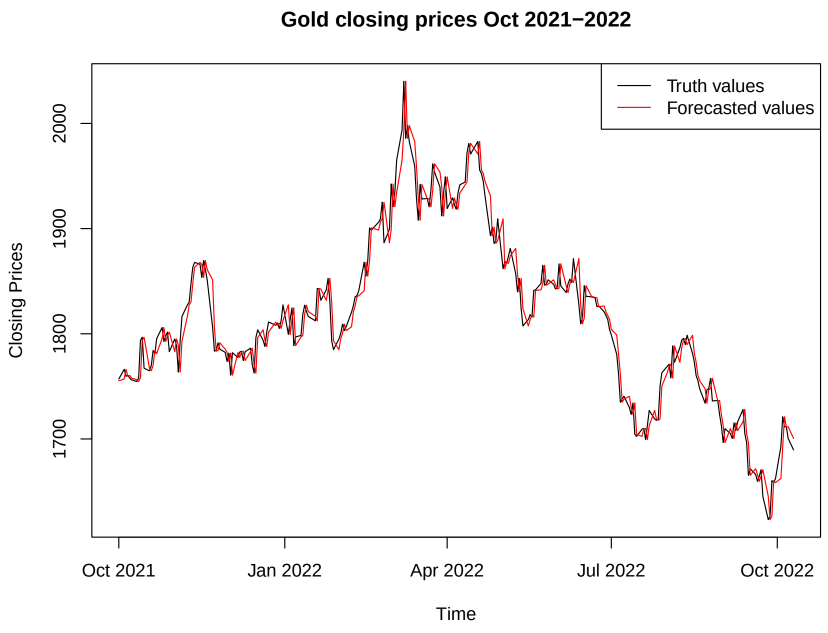


Figure 4.19: The plot shows the observed and forecasted value of BTC closing prices through ARIMA(0,1,0) by one step ahead of forecasting methods.

Table 4.17: Error measure of ARIMA

Method	MAE	RMSE	MAPE
ARIMA	12.636	16.480	0.693

The following Table 4.18 shows the error rate of all machine learning time series models and conventional ARIMA. The error is measured by MAE, RMSE, and MAPE. It can be seen from the table that the k-NN has a narrow low error in comparison to other networks such as SVM, and GRU. The ARIMA almost performs accurately however, Random Forest, RNN, and LSTM outperform as compared to other models.

Table 4.18: Error metrics of different time series methods on gold prices

Methods	MAE	RMSE	MAPE
Random Forest	14.605	19.022	0.805
k−NN	**7.129**	**9.435**	**0.393**
SVM	12.506	16.428	0.686
Neural Network	12.504	16.255	0.686
RNN	12.757	16.682	0.701
LSTM	12.699	16.501	0.698
GRU	12.639	16.385	0.694
ARIMA	12.636	16.480	0.693

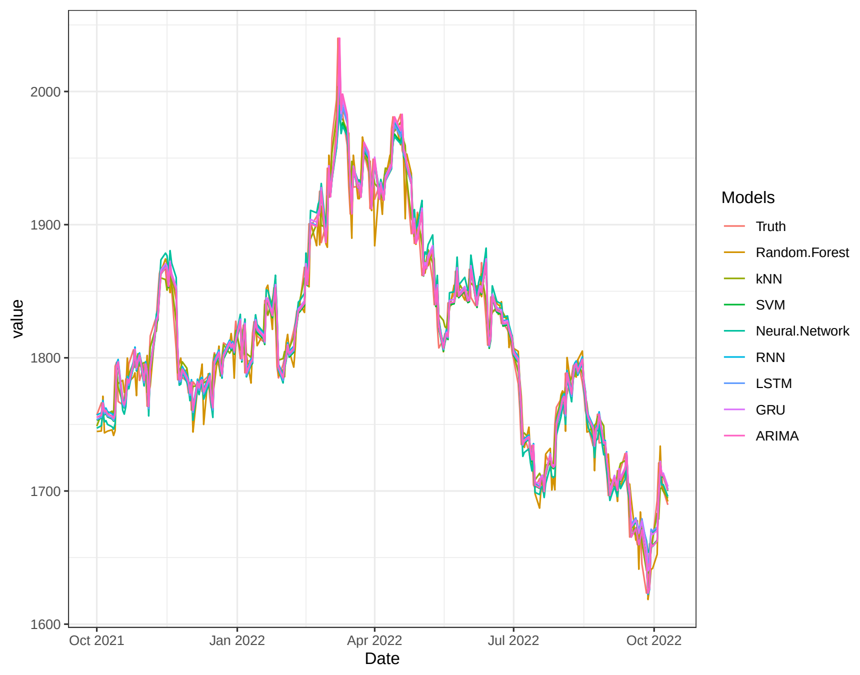


Figure 4.20: The plot shows the observed and forecasted value of gold closing prices through all models by one step ahead of forecasting methods.

4.3 Crude Oil Prices Data

The crude oil data are taken from Yahoo! Finance (2022) from 11 October 2016 to 7 October 2022 on a daily basis excluding weekends. The data is divided into training and testing parts. The training parts contain 1251 observations from 7 October 2016 to 30 September 2021. The testing parts has 262 observation from 1 October 2021 to 7 October 2022 which is forecasted by the following ML and time series models through the one-step ahead forecasting method. The performance of each algorithm and model is shown in Table 4.30.

The crude oil price data is plotted against time as shown in Figure 4.21. The crude oil price has been stationary until mid of 2020 when the prices record a sudden downward spike in March 2020 due to the COVID-19 pandemic. The crude oil price has recorded a positive linear trend from July 2022 and again decreased the prices of crude oil from March 2022. The crude oil data show frequently up and down in prices due to various economic and political situations of the world.

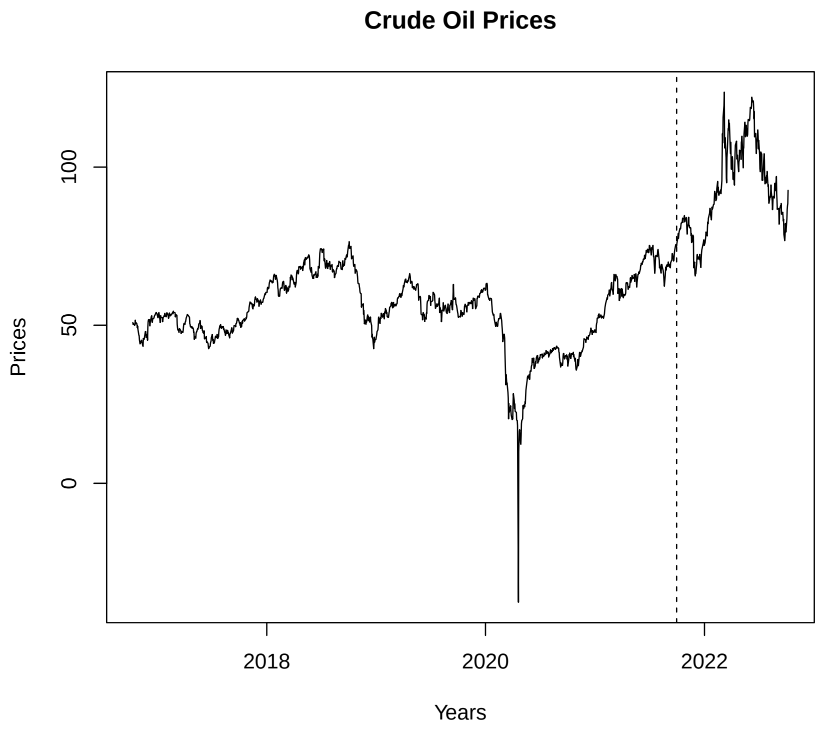


Figure 4.21: Crude oil daily price data of working days from 11 October 2016 till 7 October 2022. The dashed line shows the testing and training boundary on 30 September 2021

Random Forest

The random forest model is developed on the training set to forecast the training set. The lags of one step previous are calculated to make an independent variable. The parameters of random forests such as mtry, number of trees, and pruning are performed through cross-validation and build a final model on optimum parameters that provide the lowest error rate. The data has significant outliers therefore the data is transformed through the log and remove the trend through one step lag difference

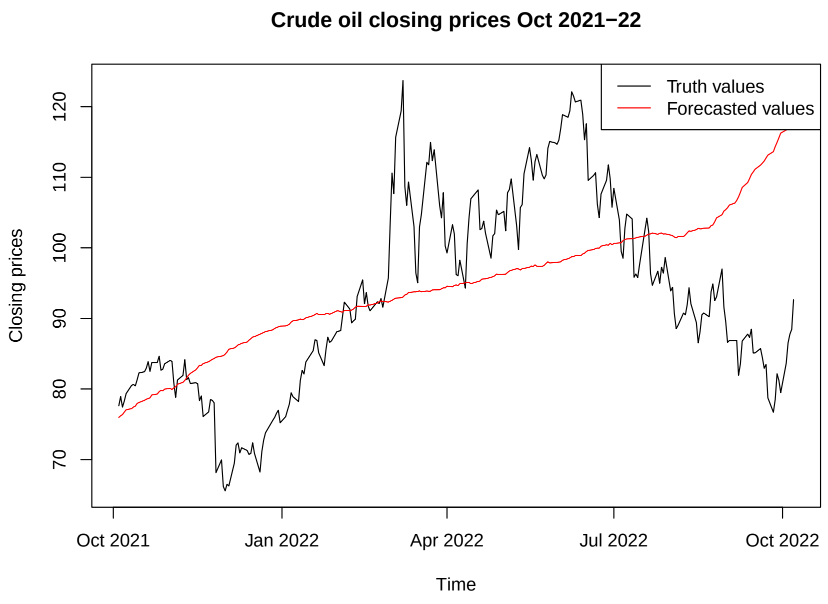


Figure 4.22: The plot shows observed and forecasted value of crude oil closing prices through random forest by one step ahead forecasting methods

Table 4.19: Error measure of Random forest

Method	MAE	RMSE	MAPE
RF	10.862	13.710	11.961

k-nearest neighbor (k-NN)

The k-NN algorithm is built on a training dataset by choosing the optimum k value through cross-validation. The year 2022 price data of crude oil is forecast through a trained model and computed the MSE, MAE, and MAPE. The following Fig. 4.23 shows the observed (black) and forecasted value (red)

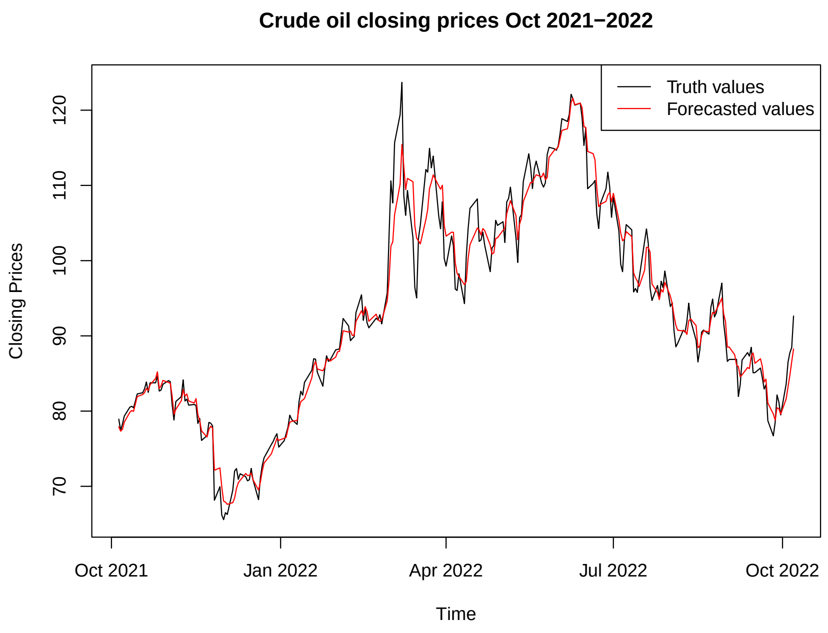


Figure 4.23: The plot shows the observed and forecasted value of crude oil closing prices through k-NN by one step ahead of forecasting methods.

Table 4.20: Error measure of k-NN

Method	MAE	RMSE	MAPE
k-NN	1.672	2.398	1.750

Support Vector Machines (SVM)

The SVM linear model is built on a training dataset by choosing the optimum parameters through cross-validation. The radial kernel is applied to train the model. The data is pre-processed before applying SVM by log transformation and removing the trend through first-order lag difference. The 2022 price data of crude oil is forecast through a trained model and computed the MSE, MAE and MAPE. The following Figure 4.24 shows the observed (black) and forecasted value (red) through SVM

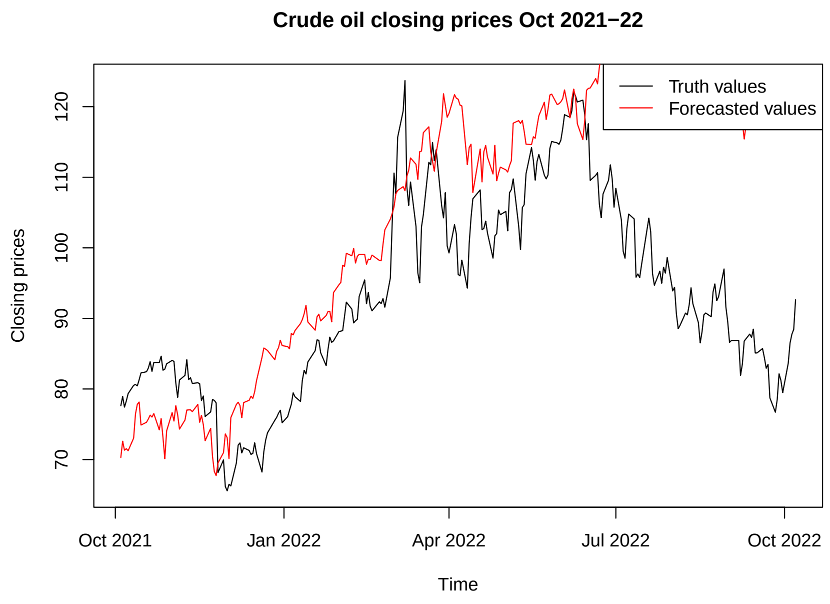


Figure 4.24: The plot shows the observed and forecasted value of crude oil closing prices through SVM by one step ahead of forecasting methods.

Table 4.21: Error measure of SVM

Method	MAE	RMSE	MAPE
SVM	16.489	22.006	18.200

Artificial Neural Networks (ANN)

The ANN model is built on a training dataset by choosing the optimum parameters through cross-validation. The data is standardized through minmax scaling. The one-step previous lags are computed and used as predictors, removing the first observation of the training and testing set as the ANN cannot handle missing observations. The 2022 price data of crude oil is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through ANN

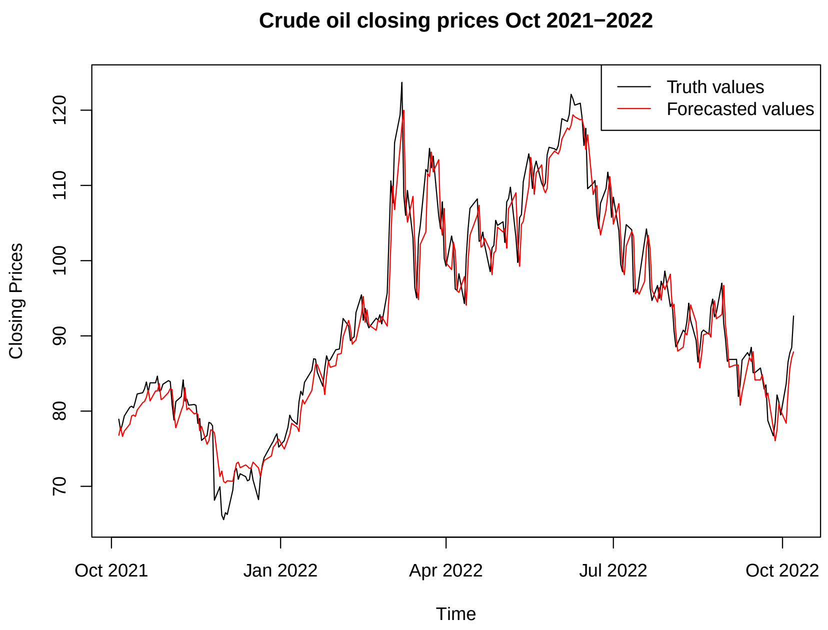


Figure 4.25: The plot shows the observed and forecasted value of crude oil closing prices through ANN by one step ahead of forecasting methods.

Table 4.22: Error measure of ANN

Method	MAE	RMSE	MAPE
ANN	2.228	2.974	2.385

Recurrent Neural Networks (RNN)

The RNN model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of crude oil is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through ANN

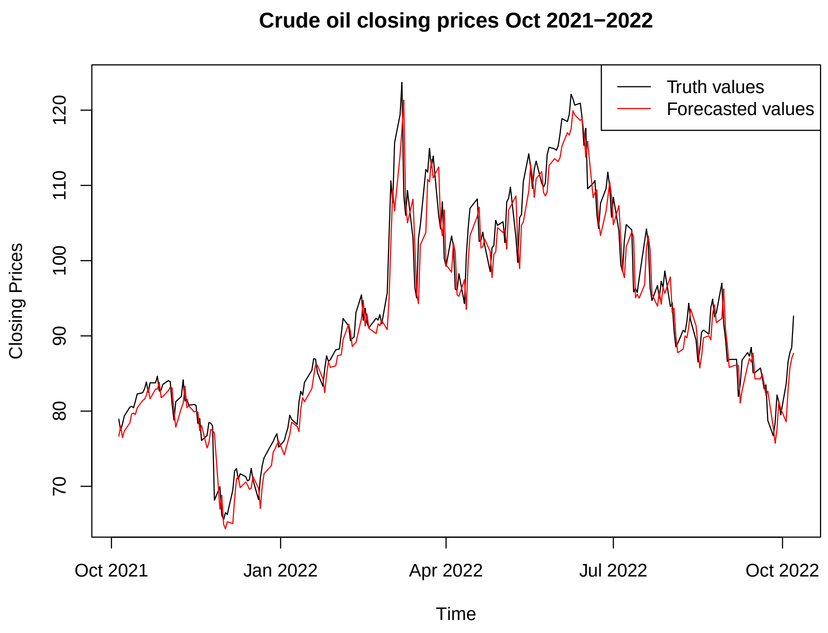


Figure 4.26: The plot shows the observed and forecasted value of crude oil closing prices through RNN by one step ahead of forecasting methods.

Table 4.23: Error measure of Random forest

Method	MAE	RMSE	MAPE
RF	2.318	3.024	2.477

Long Short-Term Memory (LSTM)

The LSTM model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of crude oil is forecast through a trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through LSTM

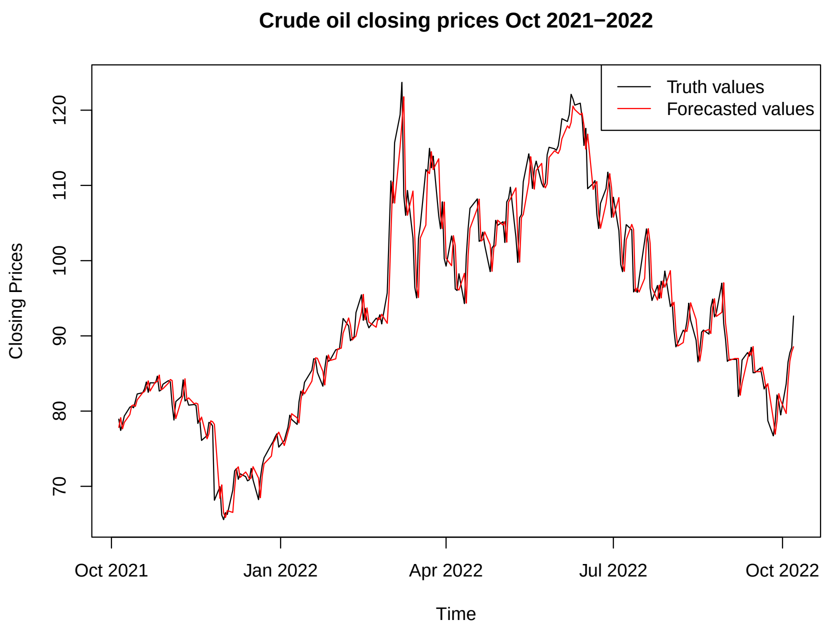


Figure 4.27: The plot shows the observed and forecasted value of crude oil closing prices through LSTM by one step ahead of forecasting methods.

Table 4.24: Error measure of LSTM

Method	MAE	RMSE	MAPE
LSTM	2.156	2.926	2.279

Gated Recurrent Unit (GRU)

The GRU model is built on a training dataset by choosing the optimum parameters through cross-validation. The 2022 price data of crude oil is forecast through trained model and computed the MSE, MAE, and MAPE. The following plot shows the observed (black) and forecasted value (red) through GRU

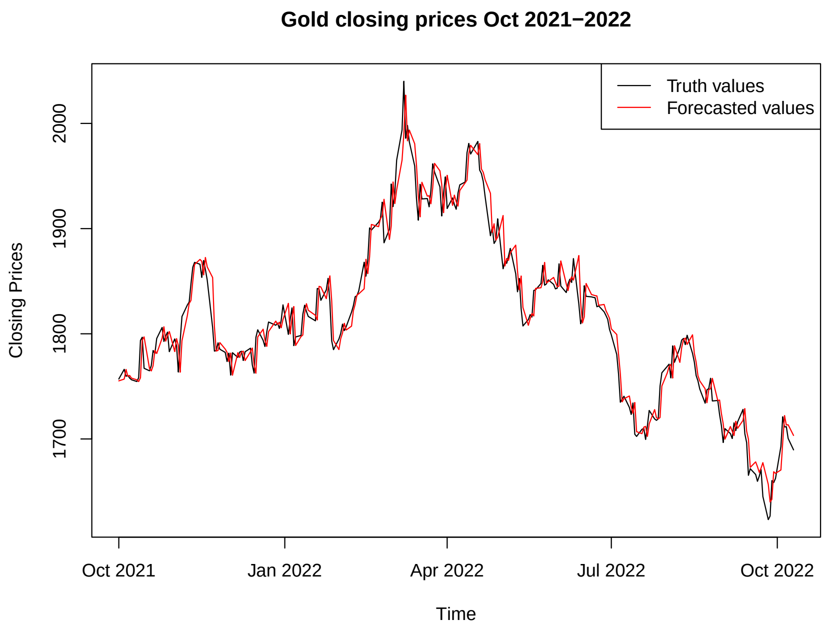


Figure 4.28: The plot shows the observed and forecasted value of crude oil prices through GRU by one step ahead of forecasting methods.

Table 4.25: Error measure of GRU

Method	MAE	RMSE	MAPE
GRU	2.151	2.911	2.268

Auto Regressive Integrated Moving Average (ARIMA)

The ARIMA model is developed by computing the Autoregressive (p) through AR plots in which the order is taken in which the current observation has an insignificant relationship with the corresponding lag value. Similarly, the moving average (q) is calculated through PACF plots. The integrating term (d) is taken as 1 since the time series has a linear trend. The optimum parameters of ARIMA are found through trial and error methods. The one-step ahead forecasting is performed on the ARIMA model to forecast the training set based on the lag value which trained the ARIMA model on training data. The optimum parameters of ARIMA is computed through a trial and error method by AIC criteria. The parameters are selected whose AIC is measured low. The results of each model show that the ARIMA model of order p=0, d=1, and q=1 provides the lowest AIC.

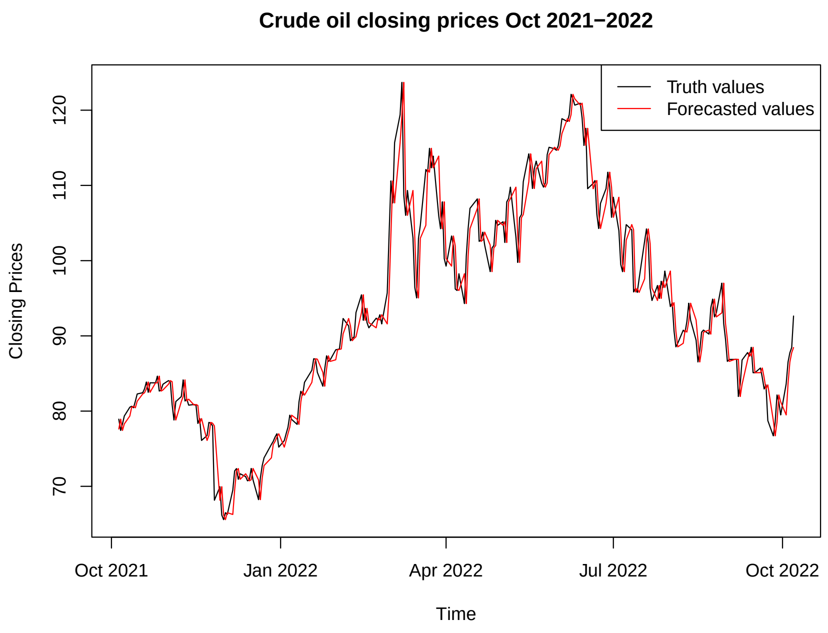


Figure 4.29: The plot shows the observed and forecasted value of crude oil closing prices through ARIMA(0,1,0) by one step ahead of forecasting methods.

Table 4.26: Error measure of ARIMA

Method	MAE	RMSE	MAPE
ARIMA	2.145	2.959	2.271

The following Table 4.27 shows the error rate of all machine learning time series models and conventional ARIMA. The error is measured by MAE, RMSE, and MAPE. It can be seen from the table that the k-NN performs highly accurately than other machine learning algorithms and ARIMA. The ANN, RNN, LSTM, GRU, and ARIMA has non-significant difference from each other in terms of accuracy however, SVM and RF perform very badly on price forecasting of all three commodities.

The following plot Figure 4.30 shows the forecasted and truth line of crude oil price in 2022 by all ML and ARIMA models.

Table 4.27: Error metrics of different time series methods on crude oil prices

Methods	MAE	RMSE	MAPE
Random Forest	10.862	13.710	11.961
***k*-NN**	**1.672**	**2.398**	**1.750**
SVM	16.489	22.006	18.200
Neural Network	2.228	2.974	2.385
RNN	2.318	3.024	2.477
LSTM	2.156	2.926	2.279
GRU	2.151	2.911	2.268
ARIMA	2.145	2.959	2.271

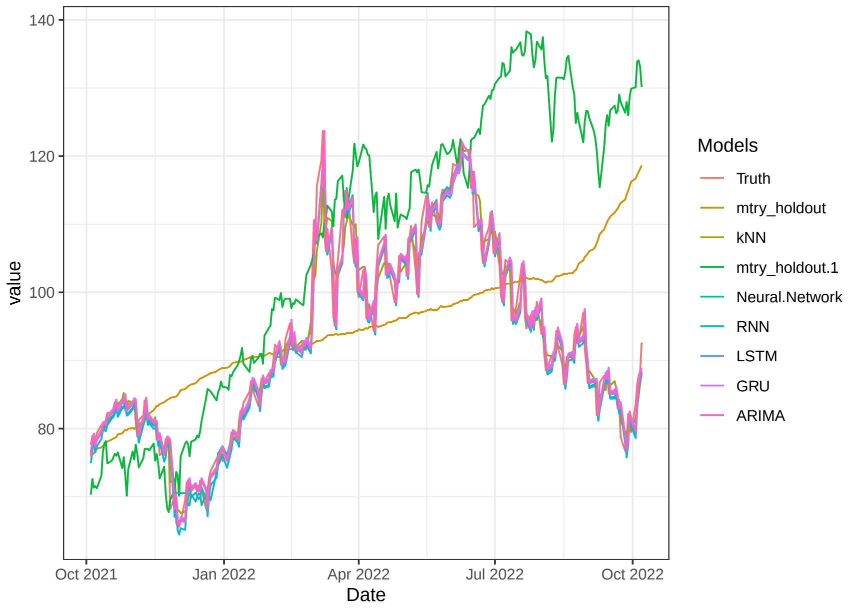


Figure 4.30: The plot shows the observed and forecasted value of gold closing prices through all models by one step ahead of forecasting methods.

Chapter 5

Conclusion

In this research work, we perform one day ahead forecasting of prices of BTC, gold, and crude oil. By using expending window techinque . Most algorithms assume that the data is stationary and should be standardized therefore the data is detrended through a one-step lag difference for ARIMA . The data is standardized through min-max scaling for neural networks and through z-score for LSTM, GRU, and RNN. The accuracy of the models is compared through MAPE, MAE, and MSE.

The Bitcoin data have a positive linear trend in the testing part while in the training portion, the prices are in decreasing pattern. The 2022 data is forecasted from previously observed data by one step ahead method and it is shown that the RNN has better performance than ARIMA, ANN, LSTM, and GRU. However, Random forest, SVM, and k-NN poorly perform on the training set. The parameters of these algorithms are selected through cross-validation and trial-and-error method.

The gold data have also a similar pattern as BTC and has a positive trend in the testing part while in the training portion, the prices are in decreasing pattern. One step ahead method is used to forecast the 2022 data based on the previously observed data, and it is shown that the k-NN has a better performance than other methods. The error measurement of ARIMA, ANN, RNN, LSTM, and GRU is almost similar while random forest has low accuracy.

The crude oil data have also a similar pattern as BTC and has a positive trend in the testing part while in the training portion, the prices are in decreasing pattern. One step ahead method is used to forecast the 2022 data from previously observed data, and it is shown that the k-NN has a better performance than other methods. The error measurement of ARIMA, ANN, RNN, LSTM, and GRU is almost similar while random forest has low accuracy.

It is concluded from the results that neural network and k-NN perform accurately in forecasting the prices of commodities and conventional time series models such as ARIMA also perform better however, the neural network has the ability to handle non-linear trends and learn the complex structure of data, therefore, it is recommended to forecast price prediction through a neural network and k-NN. Furthermore, on all three data sets, the random forest and SVM, which are primarily developed for in-sample prediction and classification problems, have performed poorly on time series trends.

References

Abdullah, L. (2012). Arima model for gold bullion coin selling prices forecasting. *International Journal of Advances in Applied Sciences*, 1(4):153–158.

Agatonovic-Kustrin, S. and Beresford, R. (2000). Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis*, 22(5):717–727.

Ali, A., Ch, M. I., Qamar, S., Akhtar, N., Mahmood, T., Hyder, M., and Jamshed, M. T. (2016). Forecasting of daily gold price by using box-jenkins methodology. *International Journal of Asian Social Science*, 6(11):614–624.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Dutta, A., Das, D., Jana, R., and Vo, X. V. (2020). Covid-19 and oil market crash: Revisiting the safe haven property of gold and bitcoin. *Resources Policy*, 69:101816.

Fu, R., Zhang, Z., and Li, L. (2016). Using lstm and gru neural network methods for traffic flow prediction. In *31st Youth academic annual conference of Chinese association of automation (YAC)*. IEEE.

Gkillas, K., Gupta, R., and Pierdzioch, C. (2020). Forecasting realized gold volatility: Is there a role of geopolitical risks? *Finance Research Letters*, 35:101280.

Guha, B. and Bandyopadhyay, G. (2016). Gold price forecasting using arima model. *Journal of Advanced Management Science*, 4(2).

Khan, F., Urooj, A., and Muhammadullah, S. (2021). An arima-ann hybrid model for monthly gold price forecasting: Empirical evidence from pakistan. *Pakistan Economic Review*, 4(1):61–75.

Kourti, K. A. L. (2021). Daily gold price forecasting using lstm cells and attention mechanism. *CS230*.

Liang, Y., Lin, Y., and Lu, Q. (2022). Forecasting gold price using a novel hybrid model with iceemdan and lstm-cnn-cbam. *Expert Systems with Applications*, 206:117847.

Livieris, I. E., Pintelas, E., and Pintelas, P. (2020). A cnn–lstm model for gold price time-series forecasting. *Neural computing and applications*, 32(23):17351–17360.

Madziwa, L., Pillalamarry, M., and Chatterjee, S. (2022). Gold price forecasting using multivariate stochastic model. *Resources Policy*, 76:102544.

Makala, D. and Li, Z. (2021). Prediction of gold price with arima and svm. In *Journal of Physics: Conference Series*. IOP Publishing.

Nowak, J., Taspinar, A., and Scherer, R. (2017). Lstm recurrent neural networks for short text and sentiment classification. In *International Conference on Artificial Intelligence and Soft Computing*. Springer.

Primananda, S. B. and Isa, S. M. (2021). Forecasting gold price in rupiah using multivariate analysis with lstm and gru neural networks. *Advances in Science, Technology and Engineering Systems Journal*.

Rady, E., Fawzy, H., and Fattah, A. M. A. (2021). Time series forecasting using tree based methods. *J. Stat. Appl. Probab*, 10(1):229–244.

Rigatti, S. J. (2017). Random forest. *Journal of Insurance Medicine*, 47(1):31–39.

Rodriguez-Galiano, V., Sanchez-Castillo, M., Chica-Olmo, M., and Chica-Rivas, M. (2015). Machine learning predictive models for mineral prospectivity: An evaluation of neural networks, random forest, regression trees and support vector machines. *Ore Geology Reviews*, 71:804–818.

Shafiee, S. and Topal, E. (2010). An overview of global gold market and gold price forecasting. *Resources policy*, 35(3):178–189.

Vishwanathan, S. and Murty, M. N. (2002). Ssvm: a simple svm algorithm. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*. IEEE.

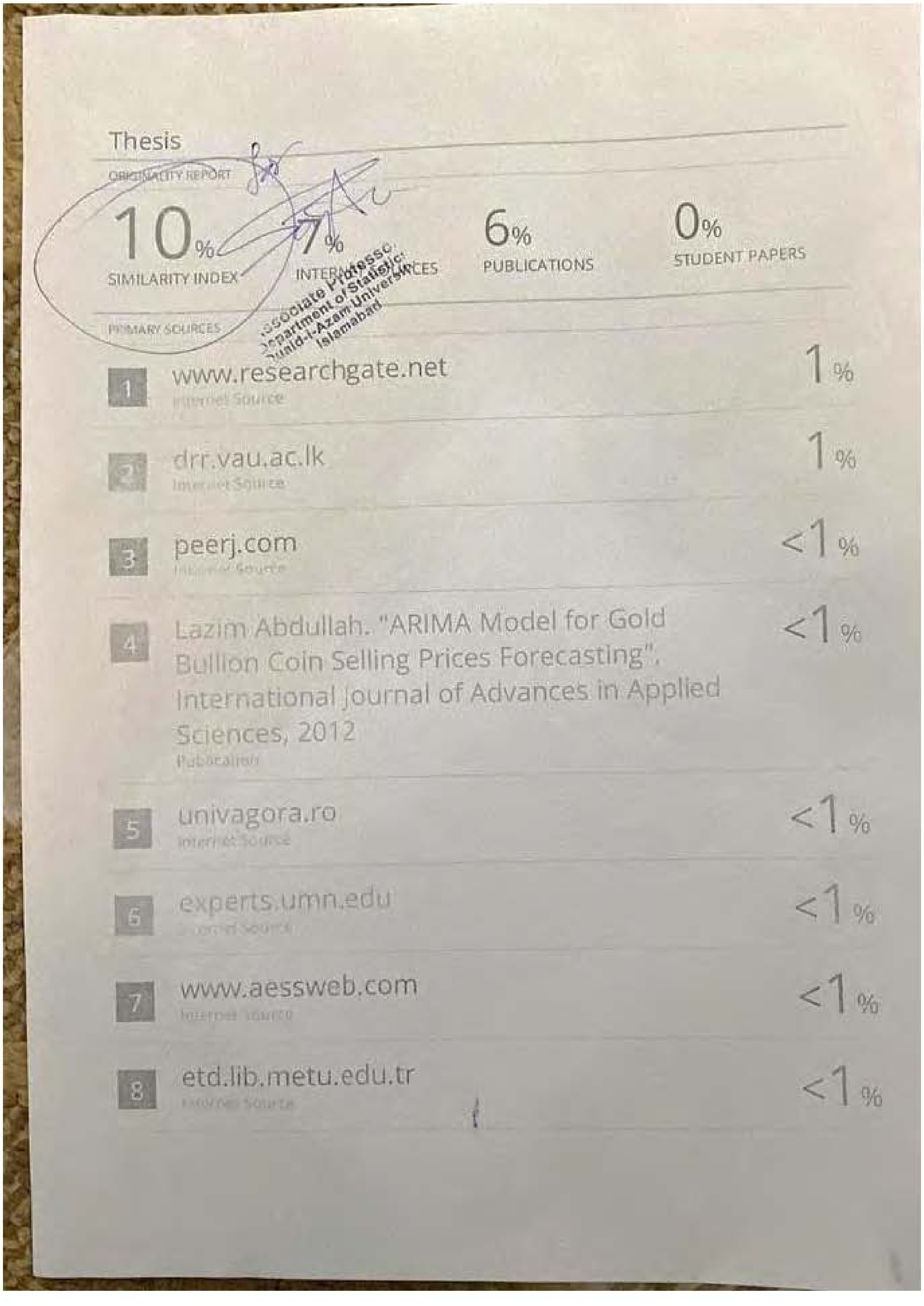
Wen, F., Yang, X., Gong, X., and Lai, K. K. (2017). Multi-scale volatility feature analysis and prediction of gold price. *International Journal of Information Technology & Decision Making*, 16(01):205–223.

Yahoo! Finance (0n 10 October 2022). https://finance.yahoo.com/.

Yang, J., De Montigny, D., and Treleaven, P. (2022). Ann, lstm, and svr for gold price forecasting. In *2022 IEEE Symposium on Computational Intelligence for Financial Engineering and Economics (CIFEr)*. IEEE.

Yaziz, S., Azizan, N., Zakaria, R., and Ahmad, M. (2013). The performance of hybrid arima-garch modeling in forecasting gold price. In *20th international congress on modelling and simulation, adelaide*.

Zhang, S., Li, X., Zong, M., Zhu, X., and Cheng, D. (2017). Learning k for knn classification. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(3):1–19.



Thesis

ORIGINALITY REPORT

10%	7%	6%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

[illegible]

Associate Professor
Department of Statistics
Quaid-i-Azam University
Islamabad

PRIMARY SOURCES

1	www.researchgate.net Internet Source	1%
2	drr.vau.ac.lk Internet Source	1%
3	peerj.com Internet Source	<1%
4	Lazim Abdullah. "ARIMA Model for Gold Bullion Coin Selling Prices Forecasting", International Journal of Advances in Applied Sciences, 2012 Publication	<1%
5	univagora.ro Internet Source	<1%
6	experts.umn.edu Internet Source	<1%
7	www.aessweb.com Internet Source	<1%
8	etd.lib.metu.edu.tr Internet Source	<1%