THE FORECAST PERFORMANCES OF CLASSICAL TIME SERIES MODELS AND MACHINE LEARNING ALGORITHMS ON BITCOIN SERIES USING EXOGENOUS VARIABLES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

SEVİLAY DOĞAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
STATISTICS

NOVEMBER 2022

Approval of the thesis:

THE FORECAST PERFORMANCES OF CLASSICAL TIME SERIES MODELS AND MACHINE LEARNING ALGORITHMS ON BITCOIN SERIES USING EXOGENOUS VARIABLES

submitted by **SEVİLAY DOĞAN** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Statistics, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** ________________

Prof. Dr. Özlem İlk Dağ
Head of the Department, **Statistics** ________________

Prof. Dr. Ceylan Yozgatlıgil
Supervisor, **Statistics, METU** ________________

Examining Committee Members:

Prof. Dr. Özlem İlk Dağ
Department of Statistics, METU ________________

Prof. Dr. Ceylan Yozgatlıgil
Department of Statistics, METU ________________

Assist. Prof. Dr. Kamil Demirberk Ünlü
Department of Industrial Eng., Atilim University ________________

Date: 30.11.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Sevilay Doğan

Signature :

ABSTRACT

THE FORECAST PERFORMANCES OF CLASSICAL TIME SERIES MODELS AND MACHINE LEARNING ALGORITHMS ON BITCOIN SERIES USING EXOGENOUS VARIABLES

Doğan, Sevilay
Master of Science, Statistics
Supervisor: Prof. Dr. Ceylan Yozgatlıgil

November 2022, 72 pages

Time series analysis importantly gives insight into what happens to a time series on any subject for days, weeks, months or years. Bitcoin is the most popular technology since it has exclusive attention in economics and finance. In this study, some of the approaches are investigated about forecasting and modeling the most popular cryptocurency bitcoin prices.The performance of classical time series methods and machine learning algorithms are compared in the study. As classical time series models, Autoregressive Integrated Moving Average (ARIMA) and Holt's Exponential Smoothing methods are used, and Prophet, Bayesian Neural Network, Feed Forward Neural Network and Long Short Term Memory and Random Forest are the methods used as machine learning algortihms. The study period is chosen from 2019-07-30 to 2021-10-19 as daily. The bitcoin prices are predicted with exogenous variables which are ethereum and tether two top cryptocurrencies, economic and technological variables via these models. According to forecast performances of the models, machine learning methods mostly outperform the classical time series methods. With Random Forest algorithm, a very good forecast perfromance is obtained with the exogenous variables on the bitcoin prices.

Keywords: Time series analysis, Forecast, Machine Learning, Cryptocurrency

ÖZ

KLASİK ZAMAN SERİSİ MODELLERİNİN VE MAKİNE ÖĞRENME ALGORİTMALARININ BİTCOİN SERİSİ ÜZERİNDE DIŞSAL DEĞİŞKENLER KULLANARAK ÖNGÖRÜ PERFORMANSLARI

Doğan, Sevilay
Yüksek Lisans, İstatistik
Tez Yöneticisi: Prof. Dr. Ceylan Yozgatlıgil

Kasım 2022, 72 sayfa

Zaman serisi analizi, günler, haftalar, aylar veya yıllar boyunca herhangi bir alandaki zaman serisine neler olduğuna dair önemli bilgiler verir. Bitcoin, ekonomi ve finansta özel ilgi gördüğü için en popüler teknolojidir. Bu çalışmada en popüler kripto para bitcoin fiyatları için zaman serisi öngörüleri ve modellemesi ile ilgili bazı yaklaşımlar incelenmiştir. Çalışmada klasik zaman serisi yöntemleri ile makine öğrenmesi algoritmaları karşılaştırılmıştır. Klasik zaman serisi modelleri olarak bütünleşik otoregresif hareketli ortalama (ARIMA) ve Holt'un Üstel Düzgünleştirme yöntemleri kullanılmış olup, makine öğrenmesi algoritmaları olarak Prophet, Bayes Sinir Ağı, İleri Beslemeli Sinir Ağı ve Uzun Kısa Süreli Bellek ve Rastgele Ormanlar yöntemleri kullanılmıştır. Çalışmada kullanılan seri günlük olarak 2019-07-30 ile 2021-10-19 arasında seçilmiştir. Bitcoin fiyatları, bu modeller aracılığıyla en iyi iki kripto para birimi ethereum ve tether olan dışsal değişkenler, ekonomik ve teknolojik değişkenler ile tahmin edilmektedir. Modelin tahmin performanslarına göre, makine öğrenmesi yöntemleri çoğunlukla klasik zaman serisi yöntemlerinden daha iyi performans göstermektedir. Bitcoin fiyatları üzerinde dışsal değişkenler eklenerek kurulan Rastgele Orman algoritması ile iyi bir öngörü performansı elde edilmiştir.

Kelimeler: Zaman Serisi Analizi, Tahmin etme, Makine Öğrenmesi, Kripto Para

ACKNOWLEDGMENTS

Firstly, I would like to express my deepest thankfulness to my supervisor Prof. Dr. Ceylan Yozgatlıgil for their guidance, encouragement, and insight throughout the research. This thesis would not be ended without her advice, encouraging behaviours and guidance.

I would like to thank the examining committee members; Prof. Özlem İlk Dağ and Assist. Prof. Dr. Kamil Demirberk Ünlü for their valuable advice and comments.

I am grateful to my friends Burcu Koca, Onur Çamlı and Özdemir Özkum for helping me whenever I need, supporting and motivational energy.

I want to present my thanks to my friends Ayşe Özgün, Ayşe Sena Akdeniz, Dilara Kartal who are always by my side

Finally, I would like to thank my family members, Hüseyin Doğan, Songül Doğan, Meryem Sıla Doğan and Elif Diyar Doğan, for always encouraging me.

To my family…

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

AIC	Akaike Information Criteria
ARIMA	Autoregressive Integrated Moving Average
BIC	Bayesian Information Criteria
BNN	Bayesian Neural Network
BRNN	Bayesian Regularized Neural Network
CNN	Convolutional Neural Network
FFNN	Feed Forward Neural Network
GARCH	Generalized AutoRegressive Conditional Heteroskedasticity
GB	Gradient Boosting
LSTM	Long Short Term Memory
MAPE	Mean Absolute Percentage Error
MASE	Mean Absolute Scaled Error
MLE	Maximum Likelihood Estimation
MSE	Mean Square Error
NNAR	Neural Network Autoregression
OOB	Out-Of-Bag
RMSE	Root Mean Square Errors
RNN	Recurrent Neural Network
SARIMA	Seasonal Autoregressive Integrated Moving Average
SVM	Support Vector Machine
XGBoost	Extreme Gradient Boosting

CHAPTER 1

INTRODUCTION

Bitcoin is the most popular cryptocurrency in the world, and it is also the first cryptocurrency. Consumers, investors, and businesses pay strangely attention to this (Frankenfield, What Is Bitcoin? How to Mine, Buy, and Use It, 2022). Bitcoin is not subject to regulation since its use anonymous. Furthermore, it enables easy and cheap international payments using a digital wallet. (Brito & Castillo, 2016). Bitcoin takes over the role of investment assets. Because of this popularity, the bitcoin price is forecasted by comparing classical time series and machine learning models.

Bitcoin series is a time series and investors are curious about its future values. To be ablcase to forecast the future bitcoin prices, time series analysis and forecasting methods are higly popular. Time series analysis includes methods for analyzing time series data so as to receive impressive statistics about data. Forecasting is a method in order to find the future values of the interesting things bottomed in the previous values. It includes eloborate analysis of present and past trends in order to predict the future values. For example, it provides better resource utilization, enhancing the quality of management and so on. Forecasting has captivated everyone for thousands of years since sometimes aforethought as divine inspiration, also sometimes considered as a blame activity. In ancient Babylon, forecasters could predict the future bottomed on circulation of maggots in a rotten sheep liver. At the same time, people asking predictions would go to Delphi in Greece to consult the Oracle who would ensure predictions while intoxicated via by ethylene vapours. In AD357, forecasters had hard time when the emperor Constantius since he had forbidden consulting to a forevaster, a mathematician or a soothsayer. Furthermore, then same restriction came out in England 1736 when charging money for predictions was blame (Athanasopoulos & Hyndman, 2021). As understood, forecasting has a big importance in life.

Time series forecasting having great importance on people's life can be done via various methods. For example, Autoregressive Integrated Moving Average (ARIMA), Seasonal Autoregressive Integrated Moving Average (SARIMA), Exponential Smoothing, Simple Forecasting Methods are classical time series analysis methods. Support Vector Machine (SVM), Feed Forward Neural Network (FFNN), Bayesian Regularized Neural Network (BRNN), Recurrent Neural Network (RNN), Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) are machine learning algorithms.

The bitcoin prices are predicted with some economic and technological variables in the previous studies (Kjærland et al., 2018). In some studies, only some of the leading cryptocurrencies are used to forecast the bitcoin prices as exogenous variables since it is considered that the cryptocurrencies can affect each other. The main contribution of the study is forecasting the up-to-date bitcoin prices by using classical time series and machine learning models and reaching the highest forecast performance by using the information coming from top cryptocurrencies, economic and technological variables via classical time series and the machine learning models. ARIMA and Holt's Exponential Smoothing methods are used as classical time series methods and Prophet, FFNN, BRNN, LSTM and Random Forest methods are used as machine learning algorithms in the study. In this thesis, RStudio is used in order to fit the models with version 2022.07.0.

This study is divided into five parts. The second chapter is the literature review, the related studies are explained. The used methods are explained in the third chapter. The fourth chapter shows results of the analysis and performance comparisons. In the last chapter, conclusion and recommendations take part.

CHAPTER 2

LITERATURE REVIEW

Since the emergence of bitcoin in 2008 (Nakamoto, 2008), many studies have arised. There are different approaches for the bitcoin price prediction. Intervals are categorized as daily, hourly, minutely, secondly, and weekly in bitcoin datasets and in some studies some features are added as explanatory variables for the prediction. According to interval of the datasets, added explanatory variables and used algorithms, accuracy results change. Investigators of the bitcoin look at it from different perspectives. There are generally two different approaches for the bitcoin price prediction. Firstly, some of the features like economical parameters such as crude oil price, gold price and technological parameters such as block size, hash rate are used as explanatory variables for the bitcoin price prediction. In the second approach, only bitcoin currency values are used by making time series analysis to predict the bitcoin price (Aygün & Kabakçı, 2021). In the first approach, most commonly used algorithms are machine learning algorithms such as LSTM and CNN. The classical time series models such as MA, SMA, ARIMA, Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) are used in the second approach(Munim et al., 2019; Bakar & Rosbi, 2017; Katsiampa, 2017; Bouabdallah, 2022). In most of the studies, the machine learning models, and the classical time series models are compared for the bitcoin price prediction.

In most of the studies, ARIMA model is used to forecast the Bitcoin price prediction (Munim et al., 2019; Ayaz et al., 2020; Bakar & Rosbi, 2017). In the study of Munim et al. (2019) the performance of two univariate methods which are ARIMA and Neural Network Autoregression (NNAR) are compared on daily prices, and it is concluded that ARIMA model performs better than NNAR in terms of forecast accuracy measures (Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE) and Mean Absolute Scaled Error (MASE)) when there is less volatility (Munim , Shakil, & Alon , 2019). In the other study, ARIMA model is also used. According to this study, although a dependable forecast model is founded out, high volatility causes larger error (Bakar & Rosbi, 2017). Moreover, in another study, a Bidirectional Auto-Regressive Transformers (BART) model which is the classic algorithm classification and regression tress are applied to forecast future values of the bitcoin. This proposed model was better than ARIMA or Autoregressive Fractionally Integrated Moving Average (ARFIMA) models both in the periods of change trend and in the periods of falling in forecasting cryptocurrencies (Derbentsev et al., 2020).

Apart from ARIMA, GARCH-type models are constructed to express the volatility of bitcoin price daily and compared their results. As a result, AR-CGARCH model is the best model about the goodness of fit data (Katsiampa, 2017). Furthermore, GARCH and Least Square Support Vector Machine (LS-SVM) models are compared. Without external variables, the volatility of series was used. The forecasts were obtained for 1 month, 2 months and 3 months. According to MAPE results, the GARCH model has better results for all these three periods (Abar, 2020). Apart from the parametric GARCH model, the non-parametric GARCH model was also used in order to forecast the bitcoin return volatility. According to this study, using non-parametric GARCH model suggests inviting and effective alternative (Mestiri, 2021). Moreover, by using Double Exponential Smoothing method, the bitcoin price prediction was also done. The MAPE values is utilized to discover the best alpha so as to forecast the bitcoin price. As a result, the best weighting (parameter) value was 0.9 which causes the smallest MAPE value of 2.89%. The error rate on the bitcoin price prediction 0.0373% (Liantoni & Agusti, 2020).

As machine learning algorithms, LSTM was mostly used for the bitcoin price prediction. In one study, both LSTM and ARIMA were used for the bitcoin daily price forecasting. The used bitcoin prices approximately include 2 years. It is resulted that LSTM forecasts of bitcoin prices develop on an average ARIMA forecasts according to MAPE and MAE (Mendes, 2019). In another study, machine learning methods and statistical methods were compared and both daily and minutely prices were used. As a result of this study, it is concluded that ARIMA provides the best among the statistical methods. ARIMA results stayed weak by the side of RNN, LSTM, SVM and Multilayer Perceptron (MLP). Therefore, it was proved that machine learning methods provide better results as against of the statistical methods. Furthermore, deep learning methods gives better results than traditional ML methods (Akay & Kervancı, 2020). ARIMA, Facebook Prophet and XG Boosting algorithms were also used and compared for the bitcoin price prediction on the bitcoin currency values. According to this study, ARIMAX algorithm which is extended version of ARIMA was the best in order to forecast in the bitcoin price about RMSE values. This study suggests that hyper tuning the parameters of time series analysis can improve the accuracy values (Iqbal et al., 2021).

Apart from these studies, features such as economical parameters and technological parameters are also used as explanatory variables for the bitcoin price prediction since it is thought that cryptocurrency market is affected by a range of factors different from traditional market. For example, DMCrypt which is a multimodal AdaBoost-LSTM ensemble learning approach was constructed by using social media sentiments, blockchain information, search volumes and trading data. It was resulted that this approach provides RMSE to decrease of $38 (19.29% improvement) (Bouabdallah, 2022).

As distinct from predicting the bitcoin price, the bitcoin price trends were predicted by classification in some studies. For instance, LSTM and Gradient Boosting (GB) algorithms were used in order to classify the bitcoin price rising or not. Hyper-parameters were tuned to get better accuracy results. It is concluded that LSTM has found more appropriate than GB model with high volatility situations (Kwon et al., 2019). On one study, both bitcoin price and trends were investigated. For the bitcoin price prediction, linear regression and SVM regression were compared by taking a naïve model forecasting method as a baseline. Thus, linear regression provides better results than SVM regression according to MSE values. In addition, the Bitcoin price trends were analyzed on logistic regression, SVM and neural network by accepting k-nearest neighbors as baseline. Neural network gives the best results among these algorithms for the classification of the bitcoin trends (Greaves & Au, 2015). Logistic regression, linear discriminant analysis and XGBoost algorithms were compared in order to classify the bitcoin price. The results show that logistic regression and linear discriminant analysis models provide better results with high-dimensional features for daily data. However, XGBoost performs the best in all other machine learning algorithms for high frequency data. This study suggests that such machine learning algorithms as RNN, LSTM can improve the results (Ranjan et al., 2022).

This studies' main contribution unlike other studies in the literature is forecasting the bitcoin prices by using classical and machine learning models with top cryptocurrencies, economic and technological variables and getting highest forecast performance.

CHAPTER 3

METHODOLOGY

Forecasting methods used in this study are discussed in this section. The statistical methods are ARIMA and Exponential Smoothing. The machine learning algorithms are Prophet, Feed Forward Neural Network, Bayesian Neural Network and Long Short Term Memory Neural Network, Random Forest.

3.1 Autoregressive Integrated Moving Average Model (ARIMA)

The Autoregressive Integrated Moving Average Model (ARIMA) is a statistical model using time series data in order to predict the future values or understand the historical behavior of the series.

The ARIMA model is defined by three components which are autoregression, integration and moving average parts, respectively.

- Autoregression (AR) shows a model which indicates changing variable regressing its own prior values alias lagged values.
- Integrated (I) refers to differencing of the observations in order to be stationary.
- Moving Average (MA) represent the dependency between a residual and an observation of a moving average model performed to lagged observations.
  The ARIMA model parameters are *p, d* and *q*.
- *p*: the lag order of AR
- *d*: the number of differences
- *q*: the order of the moving average model

If the future values are predicted from past values, the model is autoregressive. The goal of the model is predicting the future values of the series by investigating the differences of the values in the series in place of the actual values (Hayes, 2022). A moving average model uses previous forecast errors in a model as regression model.

The following expression is known as the autoregressive representation of the *p* order and it is displayed as *AR(p)*, y_t shows the observation at time *t* for $t = 1, 2, \ldots, p$

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_p y_{t-p} + e_t \ . \qquad (3.1)$$

The following expression is named as moving average representation with *q* order and displayed as *MA(q).*

$$y_t = c + e_t - \theta_1 e_{t-1} - \theta_2 e_{t-2} - \cdots - \theta_q e_{t-q} \ . \qquad (3.2)$$

Each value of the y_t can be considered as a weighted average of the past forecast errors.

The composition of autoregressive (AR) and moving average (MA) representations is named as *ARMA (p, q)*. The expression of the *ARMA (p, q)* model is as the following;

$$\dot{y}_t = \phi_1 \dot{y}_{t-1} + \phi_2 \dot{y}_{t-2} + \ldots + \phi_p \dot{y}_{t-p} + e_t - \theta_1 e_{t-1} - \theta_2 e_{t-2} - \ldots - \theta_q e_{t-q} \qquad (3.3)$$

where $\dot{y}_t = (y_t - \mu)$, e_t is white noise terms being identical and independently distributed (i.i.d) white noise terms and constant variance. In addition, ϕ and θ are parameters of autoregressive and moving average respectively.

In time series process since each time series is considered as a random variable, making inference about the data is not straightforward. In order to get rid of this problem, an assumption being stationary is necessary. The stationary time series has constant mean, variance, and correlation. If it is not stationary, the Autoregressive Integrated Moving Average Model with orders *p, d,* and *q* ARIMA (p, d, q) is used. *I* means integration which gives the number of unit roots in the process. The following equation is used for the model.

$$\phi(B)(1-B)^d\dot{Y}_t = \theta(B)\epsilon_t \ . \quad (3.4)$$

For the model, B stands for backshift operator and $B^d\dot{Y}_t = \dot{Y}_{t-d}$, d shows the number of differences to make the series stationary. ϵ_t is white noise term which is i.i.d and has zero mean and constant variance σ^2.

Model identification, parameter estimation and diagnostic checks are steps of this approach. Differencing and transformations are applied in order to make the series stationary and constant in variance. According to autocorrelation and oration autocorrelation structure of the series, the appropriate ARIMA model is identified, and the model constructed. Finally, the assumptions of the models are checked via appropriate graphical tools and statistical tests.

In order to design ARIMA model, auto.arima function can be used in RStudio . With auto.arima function, the model parameters are selected according to some criteria such as Akaike information criterion (AIC), Bayesian information criterion (BIC) or Corrected Akaike test (AICc) (Hayes, 2022). Even we can use this automated model fitting tool, diagnostics of the model must be satisfied. In addition, the models are formed via Arima() function on RStudio with version 2022.07.0 by considering the model identification tools available for ARIMA models such as autocorrelation function(ACF) and partial autocorrelation function (PACF) plots..

3.2 Exponential Smoothing Method (ETS)

Exponential smoothing method is forecasting method for univariate data, and it was proposed in the late 1950s by Brown (Brown, 1959), Holt (Holt, 2004) and Winter (Winters, 1960). It is similar to ARIMA method. Both ETS and ARIMA improve a model which uses the weighted sum of the observations and lags. However, the weights down exponentially as the observations get older. The more recent observations have the higher association. Since its simplicity and forecast performances, it is generally used for time series forecasting.

There are three type of exponential smoothing methods which are simple exponential smoothing, double exponential smoothing method and triple exponential smoothing method.

3.2.1 Simple Exponential Smoothing

Simple exponential smoothing method is a univariate forecasting method without trend or a seasonality. It has only one parameter which is alpha (smoothing coefficient). Alpha can take value in the range of 0 and 1.

The equation of the model is given below.

$$\hat{y}_{t+1} = \alpha y_t + (1 - \alpha)\hat{y}_t \tag{3.5}$$

where $\hat{y}_{t+1}$ shows forecast at time $t+1$, y_t describes the current value of the series and α is smoothing parameter. As can be seen from Equation 3.5, the forecast from simple exponential smoothing method is the weighted average of the previous smoothed value and the current value.

3.2.2 Double Exponential Smoothing

Double exponential smoothing method was proposed for forecasting the series which has trend and no seasonal pattern. This method is also called as Holt's Exponential Smoothing method. The method is supplement to simple exponential smoothing method, it contains trend. It has two parameters which are alpha and beta. The main idea of the method is taking into consideration of the possibility of the series having trend. The method bottomed on the following equations which are level and trend.

$$\ell_t = \alpha y_t + (1 - \alpha)(\ell_{t-1} + b_{t-1}) \tag{3.6}$$

$$b_t = \beta(\ell_t - \ell_{t-1}) + (1 - \beta) b_{t-1} \tag{3.7}$$

$$\hat{y}_{t+h} = \ell_t + hb_t \tag{3.8}$$

where $\hat{y}_{t+h}$ shows the h-step forecasts, ℓ_t is the estimated level term and b_t is the estimated trend term. In addition, α and β are the smoothing parameters which take values between 0 and 1.

3.2.3 Triple Exponential Smoothing

Triple Exponential Smoothing method is also called as a Holt Winters Exponential Smoothing method. Exponential smoothing is applied three times. It is used for forecasting the time series which have both seasonal pattern and linear trend. The method has two different types which are additive and multiplicative.

3.2.3.1 Holts Winter Additive Method

The method is used when the series show additive seasonal attitude. The model equation is like below.

$$\hat{y}_{t+h} = l_t + b_t h + s_{t-m+(k+1)} \quad (3.9)$$

$$l_t = \alpha\,(y_t - s_{t-m}) + (1-\alpha)\,(l_{t-1} + b_{t-1}) \quad (3.10)$$

$$b_t = \beta\,(l_t - l_{t-1}) + (1-\beta)b_{t-1} \quad (3.11)$$

$$s_t = \gamma\,(y_t - l_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m} \quad (3.12)$$

$\hat{y}_{t+h}$ shows the forecast, l_t is the estimated level term at t, b_t is the estimated trend at t, s_t seasonal ingredient and α, β and γ are the smoothing parameters which take values between 0 and 1.

3.2.3.2 Holts Winter Multiplicative Method

The method is used when the series show multiplicative seasonal attitude. The model equation is like *below*.

$$\hat{y}_{t+h} = (\ell_t + hb_t)s_{t+h-m(k+1)} \quad (3.13)$$

$$l_t = \alpha \frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \quad (3.14)$$

$$b_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)b_{t-1} \quad (3.15)$$

$$s_t = \gamma \frac{y_t}{(l_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m} \quad (3.17)$$

$\hat{y}_{t+h}$ shows the forecast, l_t is the estimated level term at t, b_t is the estimated trend at t, s_t seasonal ingredient and α, β and γ are the smoothing parameters which take values between 0 and 1 (Hyndman & Athanasopoulas, 2021). It can be seen that seasonal component is different from the one in the additive method.

3.3 Prophet

Prophet is a forecasting method for the time series data bottomed on additive model where there are daily, weekly, yearly and holiday effects. With strong seasonal effects in time series data, it works well. It handles outliers and is more robust to missing data. The model equation is as like below.

$$y_t = g(t) + s(t) + h(t) + \varepsilon_t, \quad (3.18)$$

where $g(t)$ shows growth function, $s(t)$ describes the various seasonal patterns represented as Fourier series, $h(t)$ takes the holiday effects and ε_t is a white noise error term.

The growth function has three options which are Linear Growth, Logistic Growth and Flat. Linear Growth is default for Prophet. It utilizes with different slopes between change points piecewise linear equations. When the series has a floor where the values are saturated or a cap, Logistic Growth is useful. Flat is used when there is not growth.

It can be applied via prophet function and available in R (Taylor & Letham, 2021). If the changepoints are not clearly specified for the piecewise-linear trend, they are automatically selected. The seasonal components include Fourier terms of the thematic terms. Holiday effects are joined like simple dummy variables. The model is predicted by using a Bayesian approach to let for automatic election of the changepoints and other model characteristics.

3.4 Artificial Neural Networks (ANN)

Artificial neural networks (ANNs) are computational networks biologically inspired. In 1958, the first artificial neural network was invented by psychologist Frank Rosenblatt (Loiseau, 2019). It simulates the cells training in the brain. ANNs utilizes learning algorithms which can independently make regularization as they take inputs (Brockett, 1994).

There are advantages of artificial neural networks. ANNs have capability to model non-linear relationships and complex nonlinear problems. Another advantage of it, after learning from first inputs, it is able to make inferences on unseen data. Furthermore, unlike many other modeling techniques, they do not require any assumption about the models (Mahanta, 2017)

The method composes relationship between inputs and outputs. It is essential that biological neural network structure is understood so as to imbibe ANNs.

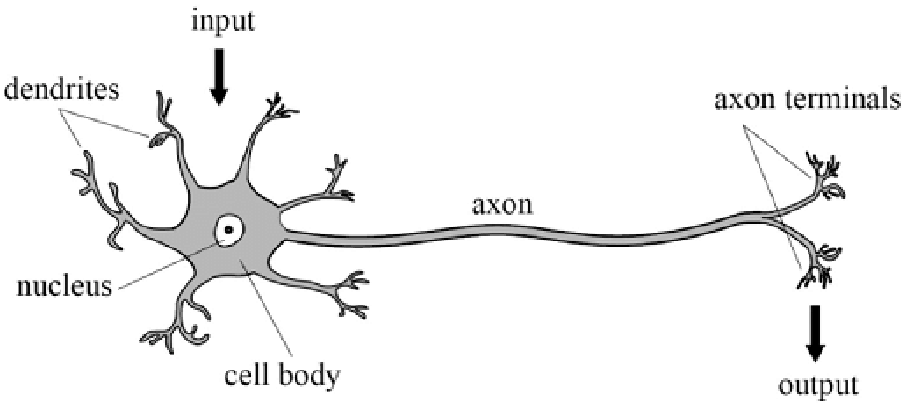

Figure 3.1: Structure of Biological Neural Network (Neves, Gonzalez, Leander, & Karoumi, 2017)

Figure 3.1 displays the biological neural network. Dendrites transfer the signals from neurons to nucleus. The nucleus collects all signals as a center. The nucleus transfers a gathered signal to axons. Axons transmit the information to synapses being located on the margin of axons. If the received signal is more than the threshold value, it is transferred to next neurons' dendrites. Thus, biological neural network is established in this way.

3.4.1 Artificial Neural Network Structure

The ANNs have pretty well similar components like the biological network.

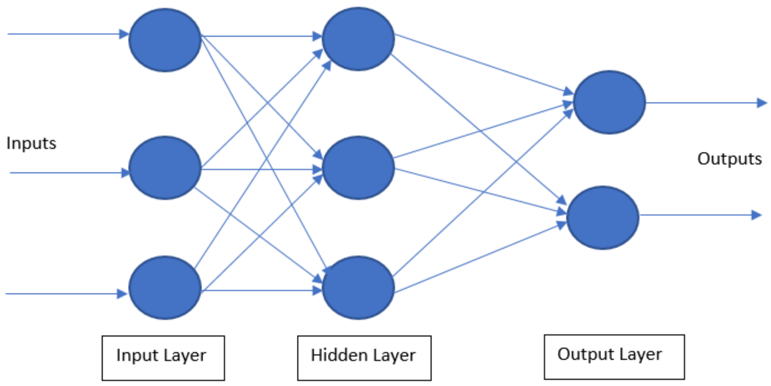

Figure 3.2: The General Structure of Artificial Neural Network (Khan et al., 2020)

ANNs includes three components which are input layer, hidden layer and output layer. Data is introduced to the network in the input layer. The information is proceeded in the hidden layer. The output layer shows the predicted values based on the inputs (Samsudin, Shabri, & Saad, 2010). The units named neuron in the layers are bonded to each other as illustrated in Figure 3.2. Weights which are linkage parameters are utilized to receive construction in the data.

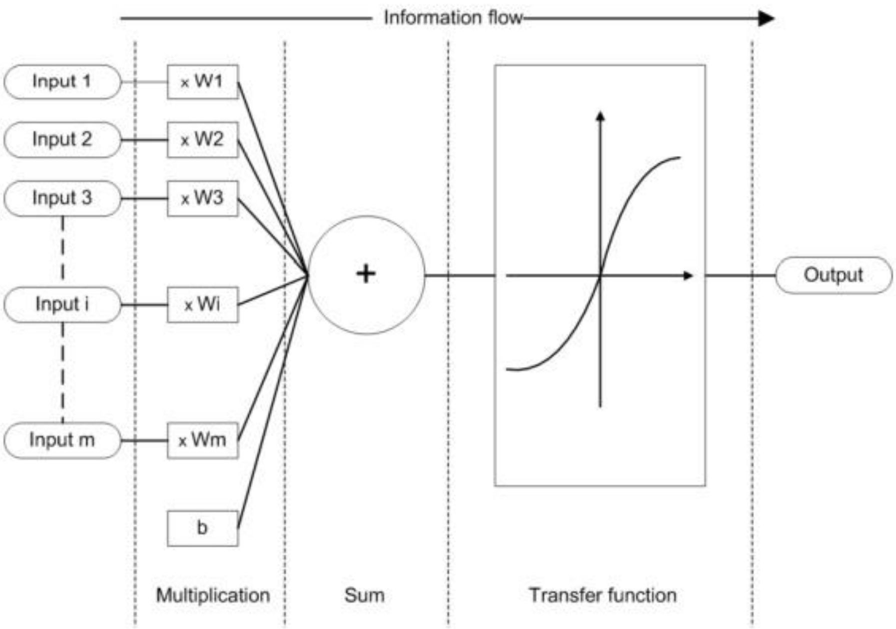

Figure 3.3: The Working Concept of Artificial Neural Network (Krenker et al., 2011)

ANN model contains inputs, bias and weights terms. As shown in Figure 3.3, multiplication, summation and activation are the steps in the working structure of the Artificial Neural Network. The first step is multiplying the weights by the given inputs. After this, the weighted inputs are summed and then a bias term is added to this in order to adjust the threshold of the activation function. Finally, the summation of the bias and weighted inputs is converted into the result by the activation function.

The mathematical representation the model is given below.

$$y(t) = F(\sum_{i=0}^{m} w_i(t).x_i(t) + b) \quad (3.19)$$

where $x_i(t)$ shows input at time t, $w_i(t)$ is the weight value at time t, b is bias, F is an activation function and $y(t)$ is the output at time t.

Activation function is essentially substantial for the ANN model since it makes possible to learn nonlinear complicated relationships between inputs and output. It describes the mathematical form of the ANN. If the summation of the bias and weighted sum is greater than the threshold value, it fires. The fired result is the output of the neurons. The type of activation function is subjected to the type of the problem. For example, sigmoid, hyperbolic and relu are some of it (Krenker et al., 2011).

In addition, training is important for the success of the model. Back-propagation is mostly used algorithm for the training for the process. The weights are updated by the Gradient Descent Algorithm based on the error values. The process is iterated till convergence is succeeded. In other words, the process continues until getting the weights which diminish the cost function such as RMSE.

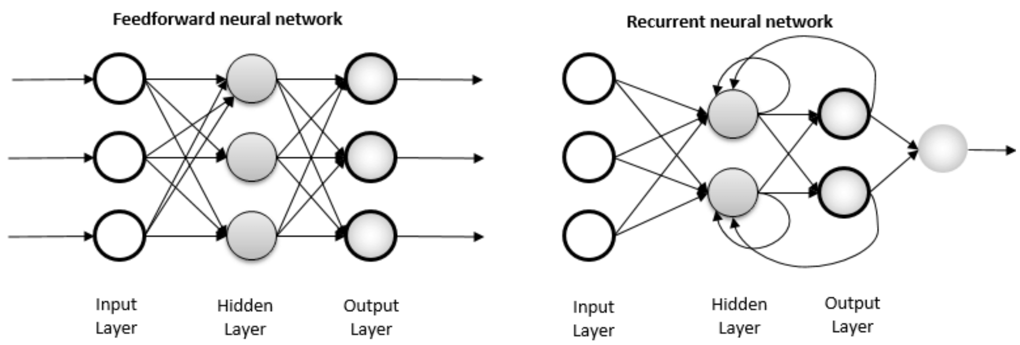

Figure 3.4: The flow of Feed Forward Neural Network and Recurrent Neural Network (Pekel & Kara, 2017).

If the output values are directly generated from the given input, the ANN is named as a static network like FFNN. If the output is generated from the previous inputs, outputs, hidden states and current value, it is named as dynamic.

3.4.2 Feed-Forward Neural Networks

Feed-forward neural network is an artificial neural network where connections do not constitute a cycle. A recurrent neural network is the opposite of feed-forward neural network. The feed-forward neural network is the basic form of the neural network since it processes the information in one direction. It generally seen as a single layer perceptron.

In single layer feed forward neural network, there are two layers which are input and output layers. Since the computation procedure does not occur in input layer, is not taken into consideration. The single layer feed forward neural network can figure out linear problems (Krenker et al., 2011). If there is a hidden layer in the feed-forward neural network, it can solve nonlinearity in the data, and it is named as multi-layer feed forward neural networks.

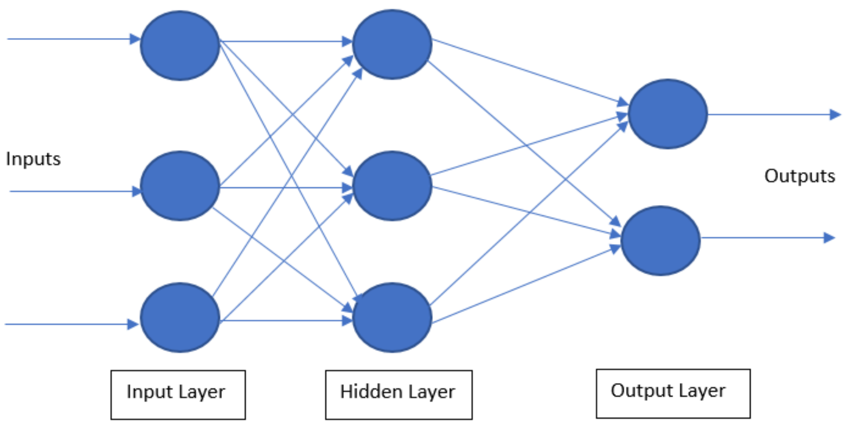

Figure 3.5: The General Structure of Multi-Layer Neural Network (Khan et al., 2020)

The structure of the multi-layer feed forward neural network as shown in Figure 3.5. The input layer transfers the data to hidden layer. After this, the hidden layer passes them the output layer. Both single layer neural network and multi-layer neural network use backpropagation in order to train the network.

The process is carried out by using nnetar () function in forecast package (Hyndman, et al., 2021).

3.4.3 Recurrent Neural Network

A Recurrent Neural Network (RNN) is an artificial neural network which is used for time series or sequential data. Feed forward neural network is worked for the data such as time series data whose values are independent from each other. If we have data in a queue which is bounded to former values, the neural network that count in the dependencies between these values. RNNs have the notion of memory which store the knowledge of previous values in order to compose the next outputs of the queue (Saeed, 2022). That is, it has a feedback loop. Each neuron nourishes its output and then its output return to the other neurons as the inputs.

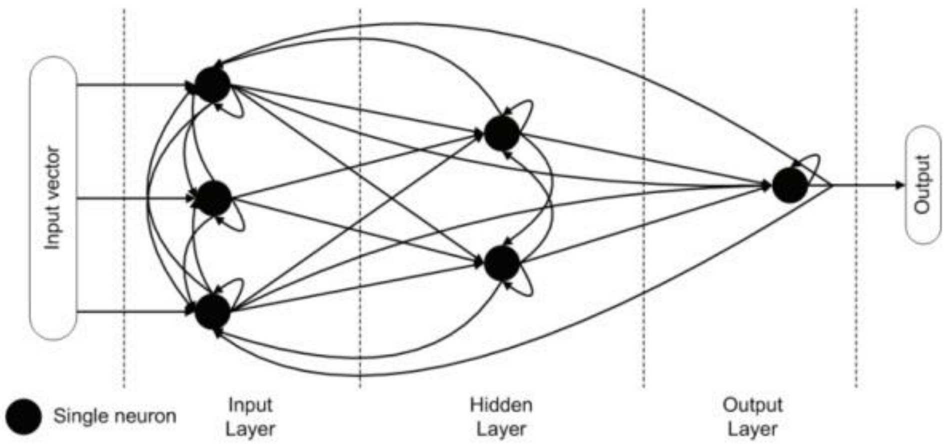

Figure 3.6: Recurrent Neural Network with Single Hidden Layer (Krenker et al., 2011)

As it can be seen from Figure 3.6, the nodes get feedback from the other nodes. The input signal is the mixture of the previous activation function via adding layer of a weight. Thus, the feedback connections are aroused thereafter updating the network.

The method is applied by using train () function in R from rnn package (Quast, 2016).

3.4.4 Long Short-Term Memory

A long short-term memory is a special form of recurrent neural network. When there are long term dependencies, RNNs fail to converge to the optimum minimum. It was proposed by Hochreiter and Schmidhuber in order to solve this problem (Hochreiter & Schmidhuber, 1997). After its first application, the method has been developed (Graves et al., 2013).

The RNNs have repeated simple hidden layers. However, LSTM has complex hidden layers. In the recurrent hidden layer, LSTM includes memory blocks which are special units. The blocks include memory cells with self-connection stocking the temporal state of the network and private gates which are multiplicative units in order to check the flow of information. Each memory block includes three gates which are input gate, output gate and forget gate (Salman et al., 2018). Moreover, memory blocks have self-connected memory cells. The input gate checks the entrance of the activations to the memory cell. That cell activations to filter and output are learned in the output gate. The forget gate enables the network to forget leave out the past input. Furthermore, in order to access and store the information for a long time. Thus, the structure can diminish the vanishing gradient difficulty and it makes LSTM practical for long term dependencies (Zheng et al., 2017).

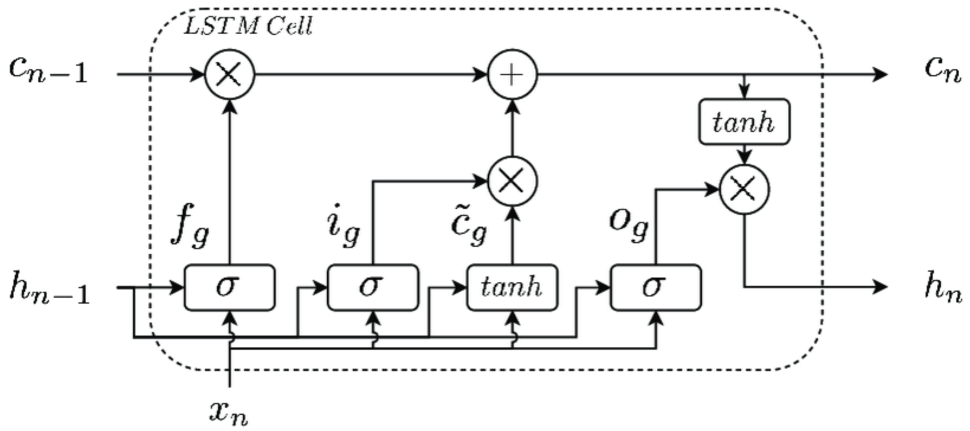

Figure 3.7: The Simple LSTM Cell Structure (Pisa et al., 2020)

3.4.5 Bayesian Regularized Neural Network

A Bayesian neural network (BNN) which is a type of neural network based on a Bayesian approach was proposed in 1992. It is a linear combination of artificial neural network and Bayesian approach. The network parameters are considered as random variables, and they are optimized in accordance with the Bayesian approach. That is, the prior distribution is posted to each weight (Ahmed et al., 2010). Thus, the model can produce straight fits. The posterior distributions of the weights are measures by presenting the data into the network, so accurate fits are produced.

The method is constructed in R by using brnn() function in brnn package (Rodriguez & Gianola, 2020).

3.4.6 Random Forest

Random Forest is a machine learning algorithm which is formed from decision tree algorithms. It is utilized to resolve classification and regression problems. It uses ensemble learning combining many classifiers to enable complex problems. It includes many decision trees. The forest is trained via bagging and bootstrap collection. The algorithms find the result based on the predictions of the decision trees. As the number of trees increases, the accuracy of the outcome increases. It reduces the overfitting problem and increases the precision of the outcome (Mbaabu, 2020). Some of the hyperparameters are directly specified in the function of the model. The hyperparameters are specified before the model construction in order to get more accurate results. The parameter showing the number of variables sampled as candidates is specified according to out-of-bag (OOB) errors. The ones which give the smallest out-of-bag (OOB) error is selected. As the number of trees is taken as 500 defaults, changing it can increase the accuracy of the model (Janitza & Hornung, 2018).

The algorithm is constructed on R in version 4.7-1.1 using randomForest () function from randomForest package (Liaw & Wiener, 2022).

3.5 Performance Measures

There are several measures to check the model forecasting performances.

3.5.1 Mean Square Error (MSE)

Mean Square Error evaluates dispersion of the errors. The smaller the MSE value is better. The MSE formula as shown below.

$$MSE = \frac{\sum_{t=1}^{n}(y_t - \hat{y_t})^2}{n} \tag{3.20}$$

where n shows the number of fitted values, y_t is the actual value and $\hat{y_t}$ is the forecast value.

3.5.2 Mean Absolute Error (MAE)

Mean Absolute Error is calculated as the average of the absolute errors. The MAE formula as shown below.

$$MAE = \sum_{t=1}^{n}\frac{|y_t - \hat{y_t}|^2}{n} \tag{3.21}$$

where n shows the number of fitted values, y_t is actual value and $\hat{y_t}$ is the forecast value.

3.5.3 Mean Absolute Percentage Error (MAPE)

Mean absolute percentage error evaluates the accuracy of the forecasting methods. It measures the accuracy as a percentage. It is the most commonly used performance measure. It is also named as the mean absolute percentage deviation. It works better when there are no zeros and excessive value. It is calculated as dividing the sum of absolute errors by the period. It is the mean of the percentage errors. The mean absolute percentage error formula as shown below.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t-\widehat{y_t}}{y_t}\right| \tag{3.22}$$

where *n* shows the number of fitted values, y_t is the actual value and $\widehat{y_t}$ is the forecast value.

3.5.4 Root Mean Squared Error (RMSE)

Root mean square error is a way in order to measure the accuracy of the predictions. It is computed by taking the square root of the mean squared root.

$$RMSE = \sqrt{\sum_{t=1}^{n}\frac{(y_t-\widehat{y_t})^2}{n}} \tag{3.23}$$

where n shows the number of fitted values, y_t is the actual value and $\widehat{y_t}$ is the predicted value. The lower values of it show the better fit.

CHAPTER 4

ANALYSIS

In this chapter, the Bitcoin price is investigated with classical time series and machine learning models. All the analysis are applied on RStudio. First, the datasets utilized will be presented. Second, the pre-processing will be showed. After these, the model implementations of the selected model are gien. Last, the accuracy of the forecast results will be evaluated and compared.

4.1 Data Description

The daily Bitcoin price data ranges from 2019-07-30 to 2021-10-19. There are 813 observations, approximately two years are investigated. The data consists of 5 features which are date, open, high, low, close values, and volume. The description of each column is shown in Table 4.1.

Table 4.1: The Description the Bitcoin Data Variables

Column Name	**Description**
Date	Time
Open	Open Price at that date
High	Highest Price at that date
Low	Lowest Price at that date
Close	Closing Price at that date
Volume	Amount of Bitcoin transected in time

The descriptive statistics of the daily Bitcoin series are shown below. The minimum value is 4,971 USD, and maximum value is 64,262 USD. For the Bitcoin price prediction, closing prices are used in the analysis. The last 14 days are used as test values for calculating the performances of the methods.

Table 4.2: The Descriptive Statistics of Bitcoin Price

Statistics	**Bitcoin Price**
Mean	22,977.00
Median	11,383.00
Maximum	64,262.00
Minimum	4,971.00
Std. Deviation	17,891.75
Observations	813

The mean values of the Bitcoin prices are far more than the median values of it, so it can be said that the Bitcoin prices are rightly skewed. According to Figure 4.1, there is an overall increasing trend.

Figure 4.1: The Bitcoin Daily Prices

In this period, the bitcoin Price received its highest value on 19th October 2021 and the lowest value at 12nd March 2020. Since Covid-19 pandemic causes investing in digital currencies, it can be said that the bitcoin prices greatly increased.

In addition to Bitcoin price data, other input variables are used. The lead cryptocurrencies which are ethereum and tether are investigated on Bitcoin prices. These two cryptocurrencies' closing values are selected. The graphical display of Tether prices is shown below.

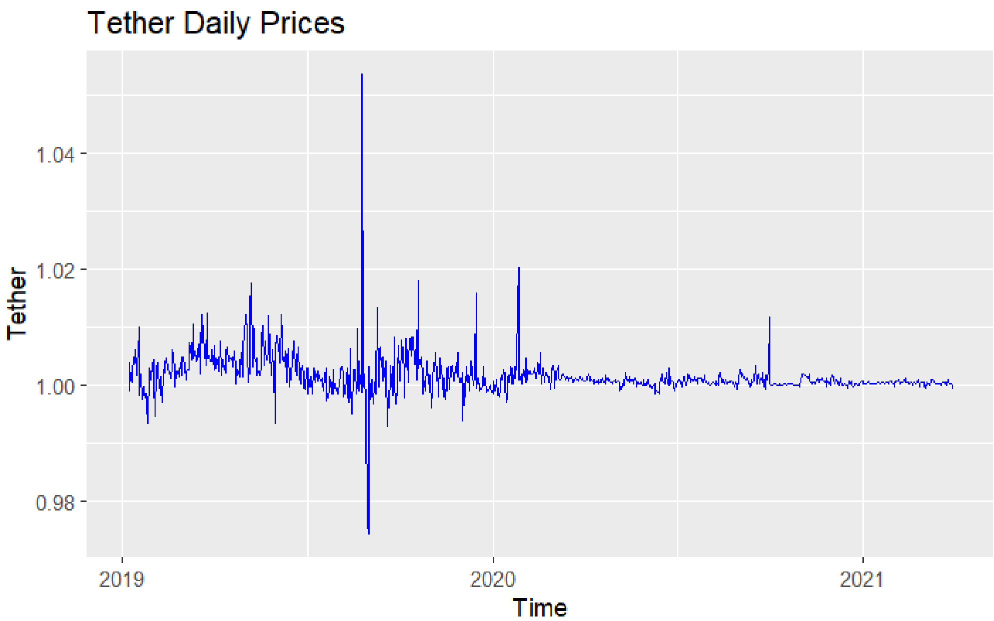

Figure 4.2: The Tether Daily Prices in the study period

Table 4.3: The Descriptive Statistics of Tether

Statistics	Tether Price
Mean	1.0016
Median	1.0008
Maximum	1.0536
Minimum	0.9720
Std. Deviation	0.0036
Observations	813

Since the mean and the median value of the Tether prices are very close to each other, it can be said that the series is approximately symmetric.

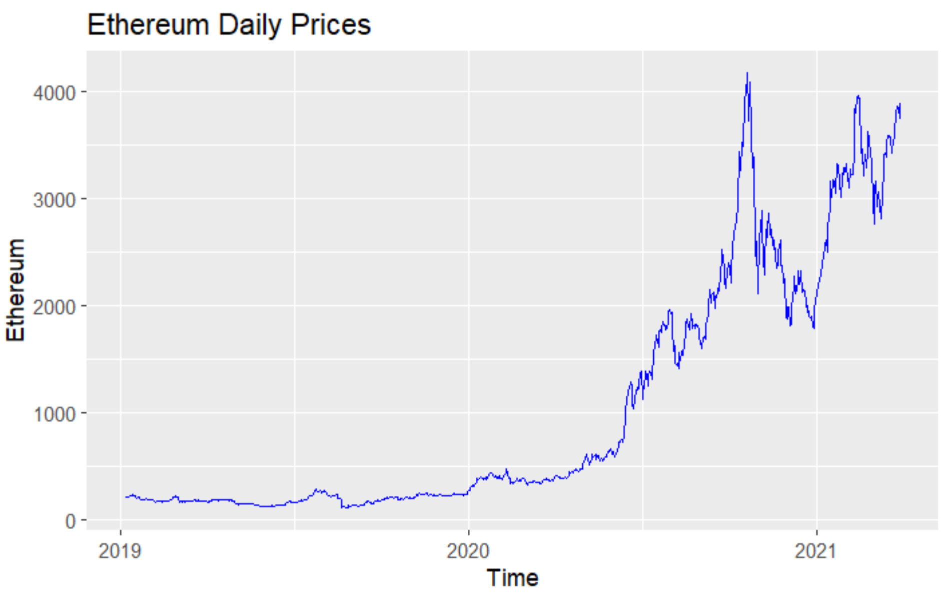

Figure 4.3: The Ethereum Daily Prices for the study period

Table 4.4: The Descriptive Statistics of Ethereum

Statistics	**Ethereum Price**
Mean	1,039.9
Median	1,875.0
Maximum	4,168.7
Minimum	110.6
Std. Deviation	0.0036
Observations	813

In addition to these leading cryptocurrencies, the other exogenous variables, which are technology variables and economic variables, are used.

Table 4.5: The Description of Technological Variables

Variable	**Description**
Average Block Size	The average block size over the past 24 hours in megabaytes.
Average Transaction per Block	Average Transaction per block over the past 24 hours
Hash Rate	A measure of the computational power on a blockchain network
Average Confirmation Time	The average time for a transaction with miner fees to be included in a mined block and added to the public ledger
Trade Volume	The total USD value of trading volume on major bitcoin exchanges
Cost per Transaction	Showing miners revenue divided by the number of transactions
Difficulty	A relative measure of how difficult is to mine a new block for the blockchain

Table 4.6: The Description of Economic Variables

Variable	Description
Gold	Gold value in USD
Crude Oil	Crude oil value in USD
CHF	Swiss Francs in USD
EUR	Euro value in USD
GBP	Pound sterling in USD
JPY	Japanese yen in USD
CNY	Renminbi in USD
VIX	Chicago Board Options Exchange Volatility Index
SP500	A stock market index in the U.S.
Nasdaq	A stock market index in the U.S.
DOW30	A stock market index in the U.S.
FTSE100	The Financial Times Stock Exchange 100 Index

The technological seven variables and economical twelve variables are used for forecasting the bitcoin prices. The last 14 values of Bitcoin prices used as test sets and all of the remaining values used for training the model. Apart from 14 values of the other variables, Ethereum prices and Tether prices are also used for training the model. The datasests were collected from Investing.com (Investing.com, 2022), Blockchain.com (Blockchain.com, 2022) and wsj.com (The Wall Street Journal, 2022).

4.2 Data Pre-processing

At the beginning of the study, the anomaly detection is applied in order to specify values deviating significantly from other values via *anomalize* package (Dancho & Vaughan, 2021). According to the result of this analysis, it is concluded that there is not any anomaly.

Some of the methods requires assumptions about the structure of the data. For instance, they may require scaled format, stationarity, time delay embedded format and so on. However, some of the methods do not need any assumption. The pre-processing can be applied via many ways such as Box-Cox transformation, detrending and differencing.

4.2.1 Variance Stabilization and Stationarity Condition

Before the application of ARIMA method, firstly, the variance stabilization assumption is checked. Since the variance is not constant, Box-Cox transformation is implemented. According to the result of BoxCox.ar (), log transformation is found suitable for the series since the ranges of lambda values consists of 0 for all the methods which are Maximum Likelihood Estimation (MLE), Yule-Walker and Ordinary Least Square. Therefore, the train set is transformed by log transformation and the variance stabilization assumption is satisfied. The process is also applied when the feed forward neural network is applied.

After checking the variance stabilization assumption, stationary assumption is checked for the application ARIMA. The series is not stationary if the series have a unit root. The stationarity is checked via some of the statistical tests which are Augmented Dickey Fuller test, KPSS test and Phillip-Perron test. According to all of these test results, the series is found non-stationary and there is stochastic trend. In order to get rid of non-stationary condition, the necessary steps are constructed. After taking one difference, stationarity is controlled again via the tests. Thus, it is resulted that the stationarity assumption is satisfied after differencing.

4.2.2 Standardization

Some of the methods are requires that the series are standardized. The standardization is formed by min-max scaling via the following formula.

$$y_{scaled} = \frac{y_t - y_{min}}{y_{max} - y_{min}} \tag{4.1}$$

where y_{scaled} shows scaled values, y_t is the actual observation at t, y_{min} and y_{max} are minimum and maximum value of the observations, respectively.

While implementing Long Short-Term Memory and Random Forest, the standardization is applied. The Bitcoin prices, all the exogenous variables and the other top cryptocurrencies are scaled between 0 and 1.

4.2.3 Time Delay Embedding

The time series have autocorrelation structure between the observations; that is, the observations depend on its past values. The statistical methods judge this characteristic. However, the machine learning algorithms mainly do not. In this instance, holding the past values in the model named as a Time Delay Embedding can come through this issue (Pan & Duraisamy, 2020). The optimal lag value is selected as 7 since the autocorrelation decreases quite linearly until lag 7. After this lag, all lags are in the White Noise bands. The Bitcoin series are transformed as embedded form by taking the lag 7. This process is applied while using the Bayesian

Regularized Neural Network, the Long Short-Term Memory and the Random Forest on the Bitcoin series.

In addition, the two top cryptocurrencies which are the ethereum and the tether are investigated by examining their cross correlations on the Bitcoin prices. The maximum correlation between the ethereum and the bitcoin prices is obtained at lag 2; thus, the ethereum is used by taking it at lag 2. Furthermore, the highest correlation between the tether and the bitcoin prices is gotten at lag4. Therefore, the tether is used at lag 4 form.

4.2.4 Feature Selection

In some cases, reducing the number of variables may be desirable to predict the future values of Bitcoin prices. Applying feature selection methods can decrease the computational cost and improve the model performance (Brownlee, 2019). Feature selection with Boruta () function which is a random forest-based algorithm to select important features in Boruta package is used on exogenous variables (Kursa & Rudnicki, 2010). According to the result of this function, some of the variables are found unimportant. The models are constructed with the other variables, but it is resulted that there is not significant difference on the computational cost and the model performance whether the unimportant variables are used. Accordingly, all the exogenous variables are used to predict the future values of the bitcoin prices.

4.3 Model Implementation

In this part, the classical time series and the machine learning algorithms are explained in Chapter 3 will be implemented to forecast the future value of the bitcoin price.

4.3.1 Statistical Models Using Only Bitcoin Prices

In this part, Autoregressive Integrated Moving Average (ARIMA) and Exponential Smoothing methods are fitted in order to forecast the bitcoin price.

4.3.1.1 ARIMA

ARIMA models are applied on the series via using the classical approach estimated by the *Arima* function and via *auto.arima* provided in *forecast* package (Hyndman & Khandakar). The function is capable of removing the stochastic trend when there exists, and then it enables the series suitable for the model suggestion. After this, it defines parameters according to the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC). The order of AR and MA are selected by minimizing these values. The mathematical forms of the AIC and BIC are below.

$$\mathrm{AIC} = -2\log(\hat{\mathrm{L}}) + 2\mathrm{k} \tag{4.2}$$

$$BIC = \log(\mathrm{n})\mathrm{k} - 2\log(\hat{\mathrm{L}}) \tag{4.3}$$

where $\hat{\mathrm{L}}$ shows the maximum value of the likelihood function, *n* is the length of the series and *k* is the number of parameters.

In order to apply ARIMA to the series, some of the assumptions are checked on the original bitcoin prices. Firstly, constant variance assumption is controlled, and BoxCox.ar function is used. According to this function results, log transformation is found suitable for the data since the ranges of lambda values consists of 0 for all the methods which are Maximum Likelihood Estimation (MLE), Yule-Walker and Ordinary Least Square. Therefore, the train set is transformed by log transformation.

Figure 4.4: The Bitcoin Prices (log transformed)

Autocorrelation and Partial Autocorrelation plots are shown in Figure 4.5. ACF plot indicates non-stationary since it slowly decays.

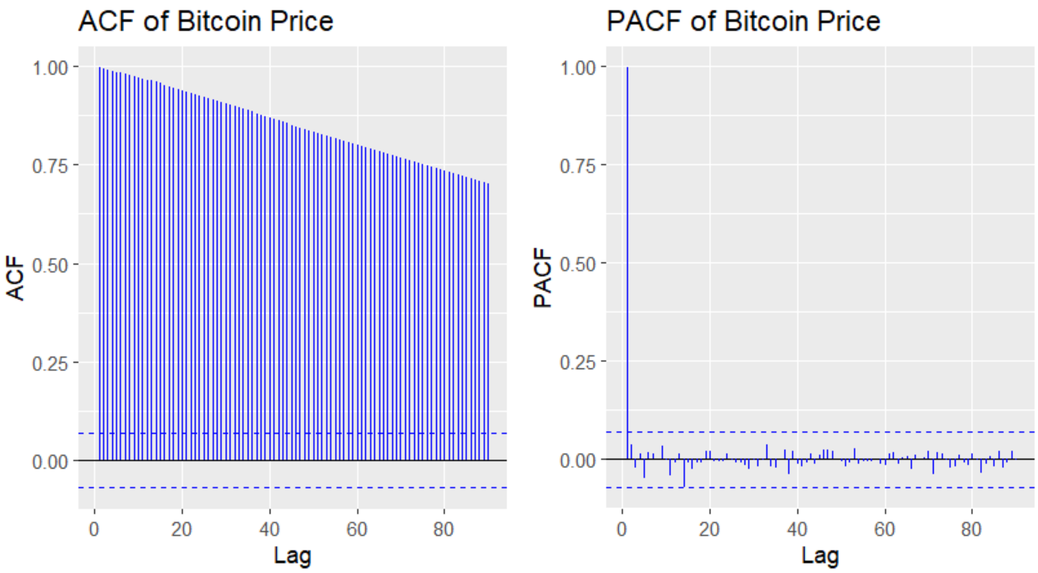

Figure 4.5:The ACF and PACF Plots of Bitcoin Price

In addition, stationarity is checked via some of the statistical tests, which are the Augmented Dickey Fuller test, the KPSS test and the Phillip-Perron test. According to all of these test results, the series is found non-stationary and there is a stochastic trend. To get rid of nonstationary, the necessary steps are constructed. After taking one difference, stationarity is controlled again via the tests. Thus, it is concluded that the stationarity assumption is satisfied. The last version of the series can be seen from the Figure 4.6. It seems stationary around 0 mean.

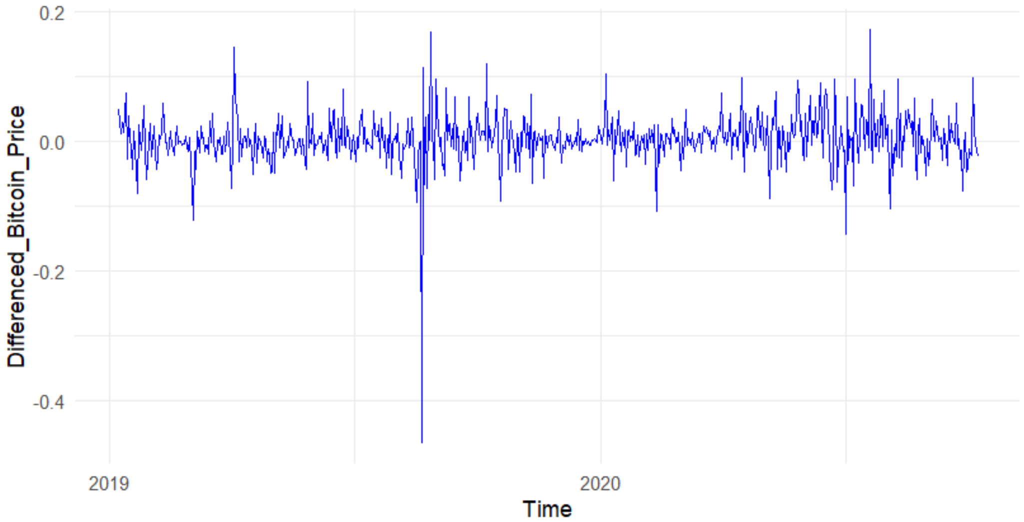

Figure 4.6:The Plot of Bitcoin Prices (log transformed and differenced)

The sample Autocorrelation and Partial Autocorrelation plots are shown in Figure 4.7. It seems that there is not any stationarity problem.

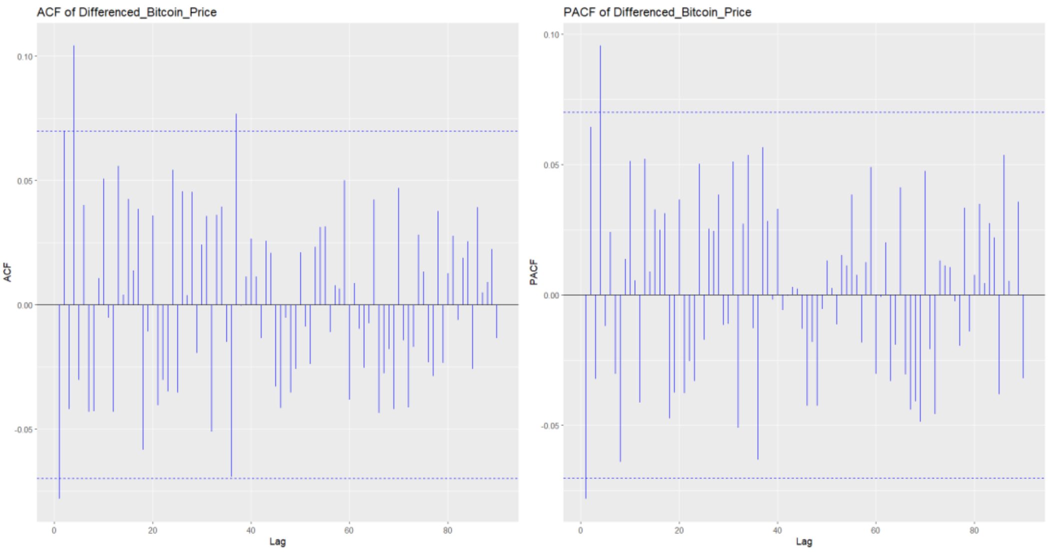

Figure 4.7:The ACF and PACF Plots of Differenced Bitcoin Price

ARIMA *(p, d, q)* describes the lag of the autoregressive part, the number of differences and size of the moving average, respectively. In order to find best parameters for p, d and q, AIC and BIC criteria are used. According to result of auto.arima () function, ARIMA (1,1,1) model is suggested. In addition to this, ARIMA (4,1,4) model is constructed according to ACF and PACF plots since lag 4 is significant in both ACF and PACF plot. Apart from these two models, the other models are tried on the bitcoin prices, ARIMA (5,1,3) model and ARIMA (7,1,7) are also selected as a candidate model since their components are significant.

Table 4.7: The AIC and BIC Values of Suggested Models

Criteria	ARIMA (1,1,1)	ARIMA (4,1,4)	ARIMA (5,1,3)	ARIMA (7,1,7)
AIC	-2,805.35	-2,819.74	-2,817.35	-2,824.81
BIC	-2,784.86	-2,777.77	-2,775.39	-2,791.36

The model which has the lowest AIC and BIC is better for forecasting. According to this, the best model is ARIMA (7,1,7). Table 4.8 shows the parameter estimates and the standard errors. AR7 and MA7 components are significant.

Table 4.8: The Estimates and Standard Errors of the ARIMA (7,1,7)

Component	AR1	AR2	AR3	AR4	AR5
Coefficient Standard Error	-0.4814 0.0592	-0.3953 0.0433	-0.3716 0.0407	0.4014 0.0281	0.2783 0.0407
Component	AR6	**AR7**	MA1	MA2	MA3
Coefficient Standard Error	0.5505 0.0348	0.8365 0.0741	0.4217 0.0619	0.4268 0.0421	0.3509 0.0462
Component	MA4	MA5	MA6	**MA7**	
Coefficient Standard Error	-0.3586 0.0420	-0.2496 0.0498	-0.4958 0.0471	-0.8300 0.0758	

The model parameters are estimated via MLE method. The coefficients given in Table 4.8 show the estimates of the parameters and their standard errors.

After construction of the model, model diagnostics are checked. The forecasting methods must satisfy some assumptions. The residuals must be uncorrelated, they must have zero mean, they must have constant variance, and they must be normally distributed.

According to the Breusch-Godfrey test result, there is no serial correlation which means the residuals are uncorrelated. The residuals are founded as homoscedastic by the Breusch-Pagan test. As a result of the Shapiro-Wilk test, the residuals are not normally distributed but, it can be said that they are approximately normal according to the QQ-plot of the residuals.

By using this model, the bitcoin prices for the 14 following days are forecasted, the forecasts are inversely transformed, and the following results are obtained. The predicted values and the actual values which are test set are compared.

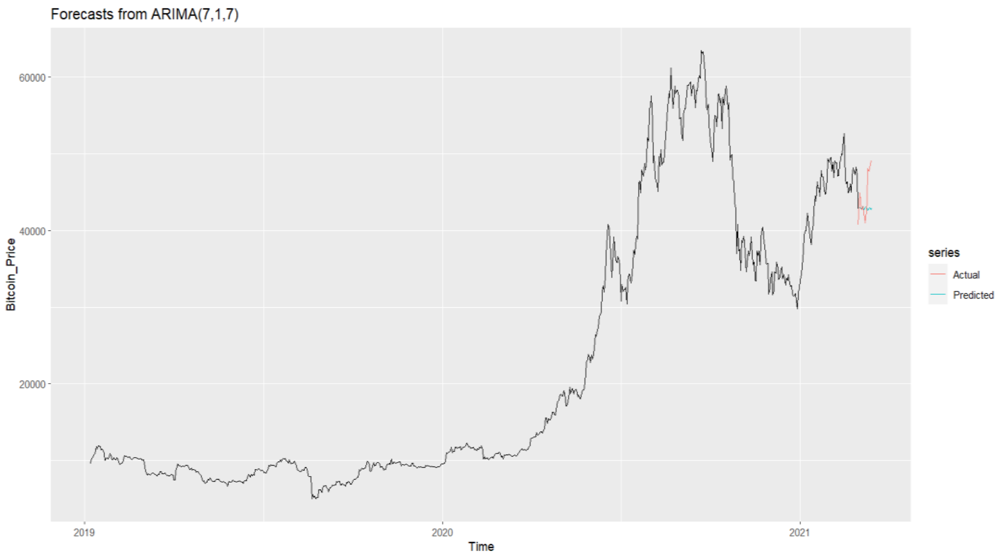

Figure 4.8: The Forecast Plot of ARIMA (7,1,7)

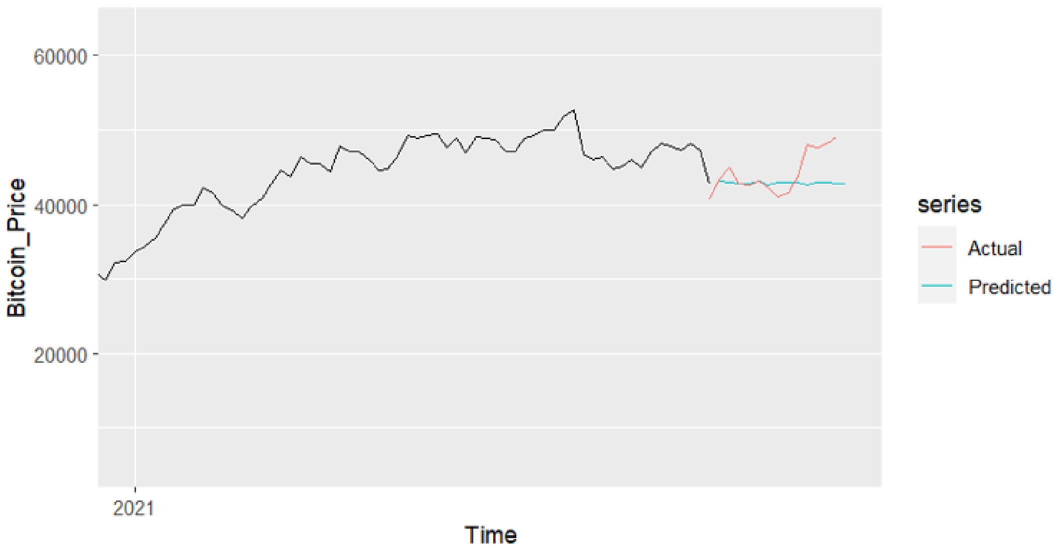

Figure 4.9: The zoomed version of the Forecast Plot of ARIMA (7,1,7)

According to Figure 4.8 & 4.9, the actual and the predicted bitcoin prices are not close to each other. The actual prices are mostly higher than predicted values on the average. Table 4.9 displays the forecast performance of the ARIMA model on the test set.

Table 4.9: The Forecast Performance of the ARIMA (7,1,7)

ME	RMSE	MAE	MPE	MAPE
738.25	3431.65	2863.32	0.99	**5.7**

Since the MAPE value which is 5.9% is less than 10%, the predictive performance of the ARIMA is good. From Table 4.10, the actual and the predicted bitcoin prices can be seen for 14 days.

Table 4.10: The Forecast results of ARIMA (7,1,7) model for 14 days

Time	Actual Prices	Forecasted Prices	Forecast Error
2021-09-21	40,693.68	43,186.14	-2,492.46
2021-09-22	43,574.51	42,993.22	581.29
2021-09-23	44,895.10	42,866.59	2,028.51
2021-09-24	42,839.75	42,862.27	-22.52
2021-09-25	42,716.59	43,146.08	-429.49
2021-09-26	43,208.54	42,675.06	-533.48
2021-09-27	42,235.73	42,961.00	-725.27
2021-09-28	41,034.54	43,047.13	-2,012.59
2021-09-29	41,564.36	42,949.43	-1,385.07
2021-09-30	43,790.89	42,638.23	5,478.71
2021-10-01	48,116.94	42,930.42	1,152.66
2021-10-02	47,711.49	43,040.95	4,670.54
2021-10-03	48,199.95	42,736.64	5,463.31
2021-10-04	49,112.90	42,864.98	6,247.92

The actual and predicted prices are not close to each other according to Table 4.10. The minimum forecast error is -22.42. The date when the actual and predicted values are closest is September 24, 2021

4.3.1.2 Exponential Smoothing Method

Exponential smoothing method which is a univariate forecasting method is constructed, and different alpha parameters are applied. Holt's Exponential Smoothing method is applied on the data. Firstly, by using *holt* function, the model is constructed with optimal parameters according to the minimum AIC and BIC criteria using the original bitcoin series. The model gives alpha parameter as 0.929 meaning fast learning in the day-to-day movements and beta parameter as 0.0133 which means slow learning for the trend. The following 14 days are forecasted with this model and compared with the actual observations.

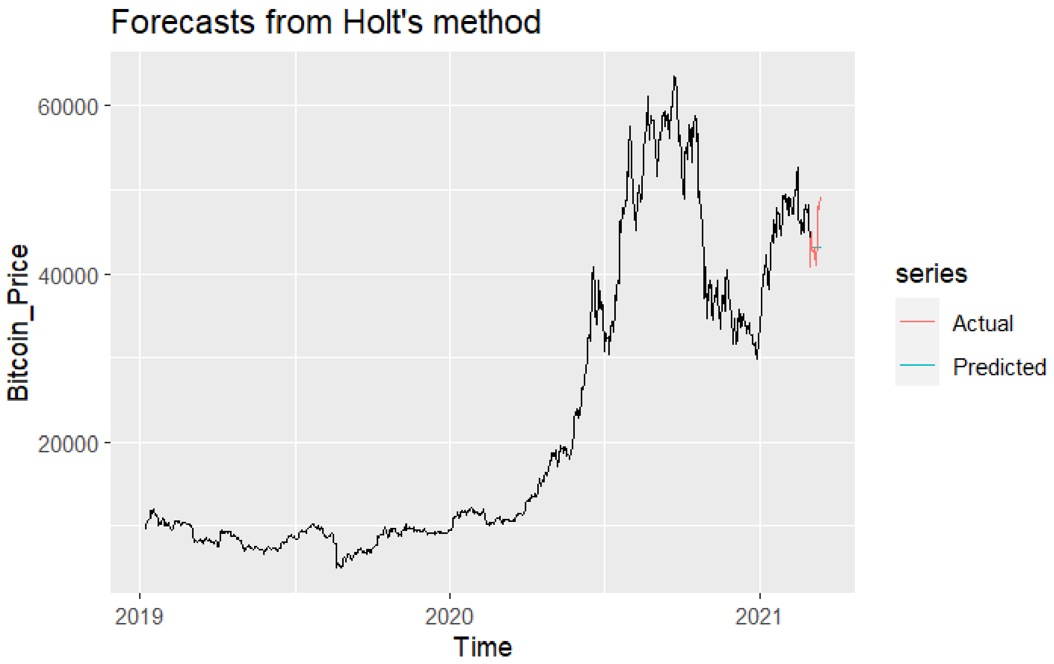

Figure 4.10: The Forecast Plot of Holt's Exponential Method

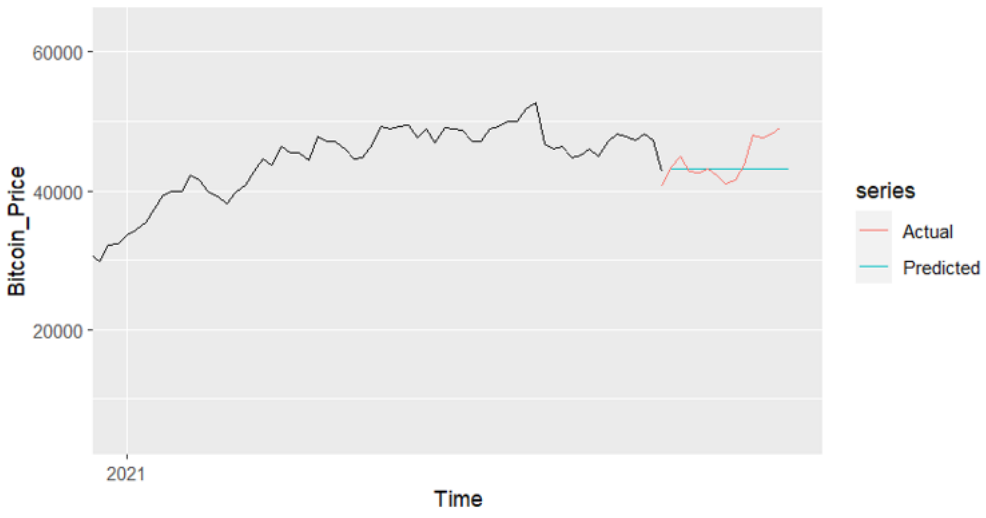

Figure 4.11: The zoomed version of the Forecast Plot of Holt's Exponential Method

According to Figure 4.10, the actual and the predicted bitcoin prices are more different from each other. The actual prices values are generally higher than predicted values on average.

Table 4.11: The Forecast Performance of Holt's Exponential Method

ME	RMSE	MAE	MPE	MAPE
1378.363	3013.847	2211.702	2.7527	**4.75**

According to the forecast performance of Holt's Exponential method, the MAPE values is less than 10% and it is a very good value. Thus, it can be said that Holt's Exponential method's predictive performance is good for the bitcoin prices. The better results are obtained in the Holt's Exponential method compared to ARIMA. Table 4.12 shows actual and predicted Bitcoin prices and forecast errors.

Table 4.12: The Forecast results of Holt's Exponential Method for 14 days

Time	Actual Prices	Forecasted Prices	Forecast Error
2021-09-21	40693.68	43165.93	-2472.25
2021-09-22	43574.51	43164.98	409.53
2021-09-23	44895.10	43164.02	1171.08
2021-09-24	42839.75	43163.07	-323.32
2021-09-25	42716.59	43162.11	-445.52
2021-09-26	43208.54	43161.15	47.39
2021-09-27	42235.73	43160.20	-924.47
2021-09-28	41034.54	43159.24	-2124.70
2021-09-29	41564.36	43158.29	-1593.93
2021-09-30	43790.89	43157.33	633.56
2021-10-01	48116.94	43156.38	4960.56
2021-10-02	47711.49	43155.42	4556.07
2021-10-03	48199.95	43154.47	5045.48
2021-10-04	49112.90	43153.51	5959.39

The date when the actual and the predicted value are closest is on September 26, 2021.

4.3.2 Machine Learning Models Using Only Bitcoin Prices

In this part, Prophet, Feed Forward Neural Network, Bayesian Regularized Neural Network, Long Short-Term Memory and Random Forest are fitted for forecasting the bitcoin price.

4.3.2.1 Prophet

Prophet is a method for the time series data based on additive model where there are daily, weekly, yearly and holiday effects. It is applied on prophet package and available in R. In order to forecast the bitcoin price, by using prophet function the model is constructed. The future values are forecasted. To be able to increase the performance of the model, hyperparameter tuning is done on the change point inclusion.

Prophet model is constructed with the following hyperparameters which give the best performance by doing random search on the original bitcoin series given in Table 4.13.

Table 4.13: The Prophet Model Parameters

Parameter of Prophet Model	**Parameter Value**
Changepoint Range	0.9
Changepoint Prior Scale	0.05
Seasonality Prior Scale	10

After model building, the forecast plot of the Prophet model for the following 14 days is shown in Figure 4.12. Table 4.14 represents the performance of the Prophet model.

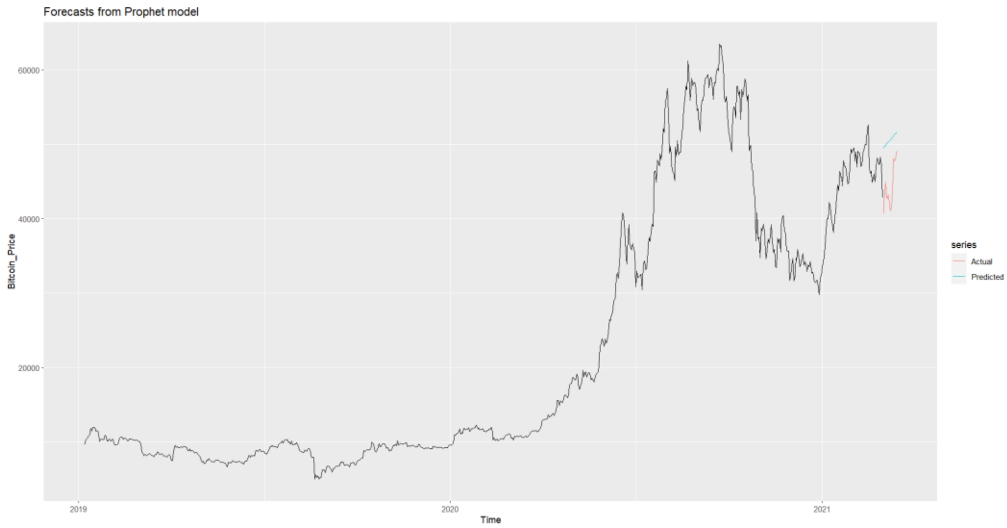

Figure 4.12: The Forecast plot of Prophet Model

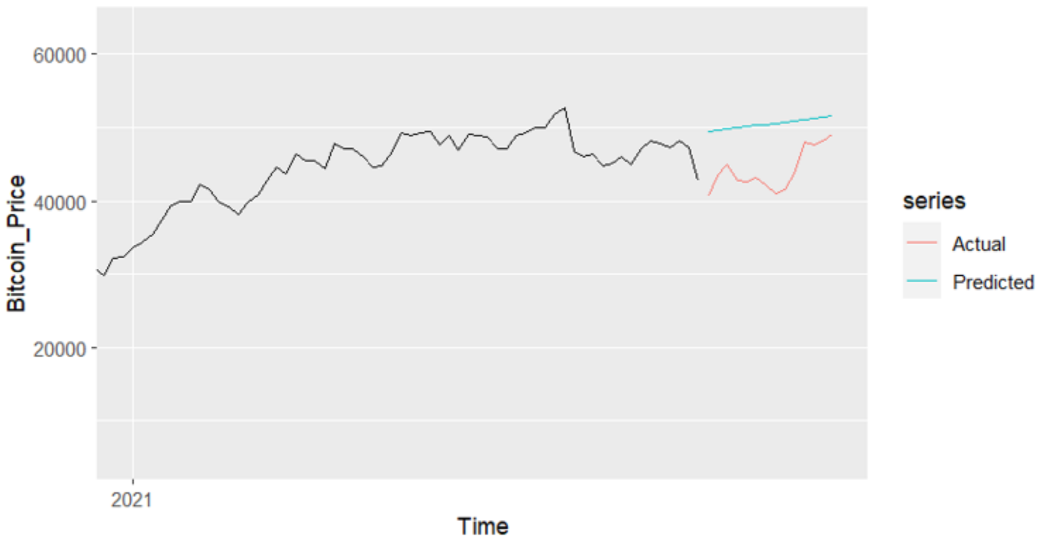

Figure 4.13: The zoomed version of the Forecast Plot of Prophet Model

As it can be seen from Figure 4.12 & 4.13, the predicted values are different than the actual ones.

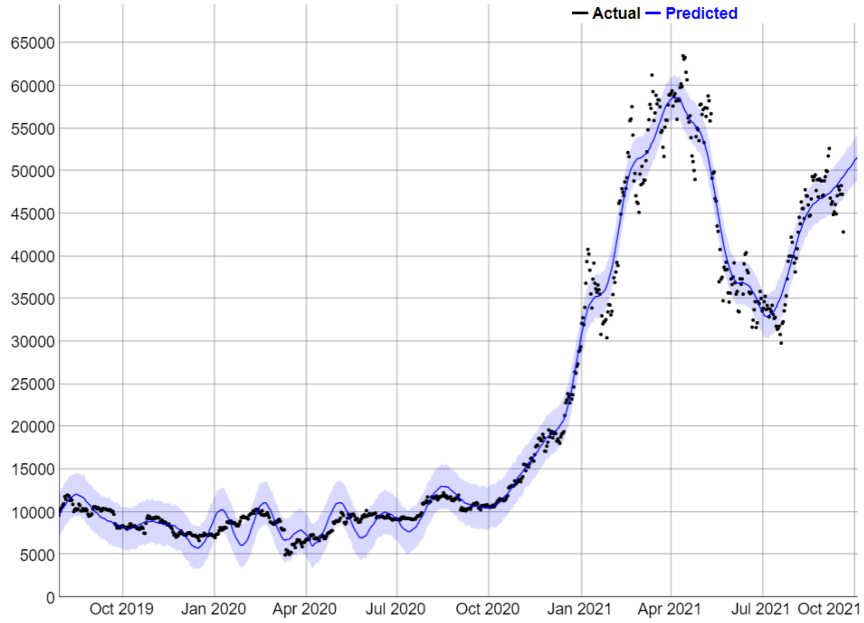

Figure 4.14: The Forecast plot of Prophet Model for all the Bitcoin Series

According to Figure 4.14, most of the actual bitcoin prices are in the prediction interval of the Prophet model.

Table 4.14:The Forecast Performance of Prophet Model

ME	RMSE	MAE	MPE	MAPE
-6301.80	6717.349	6301.801	-14.606	**14.61**

The MAPE value is 14.61% and it is moderately good since it is in the 10-20% range. The predictive performance of the Prophet model is quite lower than the ARIMA model and the Holt's Exponential model.

4.3.2.2 Feed Forward Neural Network

Feed-forward neural network model is constructed using nnetar () function in forecast package in R version 8.16. The function forms a feed-forward neural network with a single hidden layer. The forecast package is used for this model building in RStudio in order to predict the future values of the bitcoin prices. It models time series by taking the lagged values of the time series as inputs. If lambda value is selected as "auto", Box-Cox transformation is selected and if it is not selected, transformation is ignored for making the stationary in variance. Therefore, lambda value is specified within nnetar () function, or the data is transformed before constructing the model.

The *nnetar* () function includes *P*, *p* and size. *P* shows the number of seasonal lags. The default value of it is the optimal number of lags according to AIC. *p* is the number of non-seasonal lags, and the size shows the number of nodes in the hidden layer. The default value of the size is half of the number of the input nodes plus1. There is only one hidden layer in the *nnetar* function that cannot be changed.

Because there n no seasonal behavior in the series, the seasonal parameter is not included. After applying nnetar () function on log transformed form of the bitcoin series for the model, it suggests NNAR (1,1). The number of non-seasonal lags taken as inputs is 1 and the number of the nodes in the hidden layer is 1. However, the other models are generated and NNAR (30,16) gives the best result. The structure of the model is 30-16-1. 30 shows non-seasonal lags used as inputs and 16 shows the number of the nodes in the hidden layer, and 1 shows the output. The different forecast results are obtained since the weights are randomly specified in the nnetar function that can be manipulated. After repeating the model function with 30 non-seasonal lags and 16 nodes many times, the best model is chosen according to MAPE and RMSE results. Then, the following 14 days are forecasted, and the forecasts are inversely transformed into their original scale. The following results are obtained from the selected model.

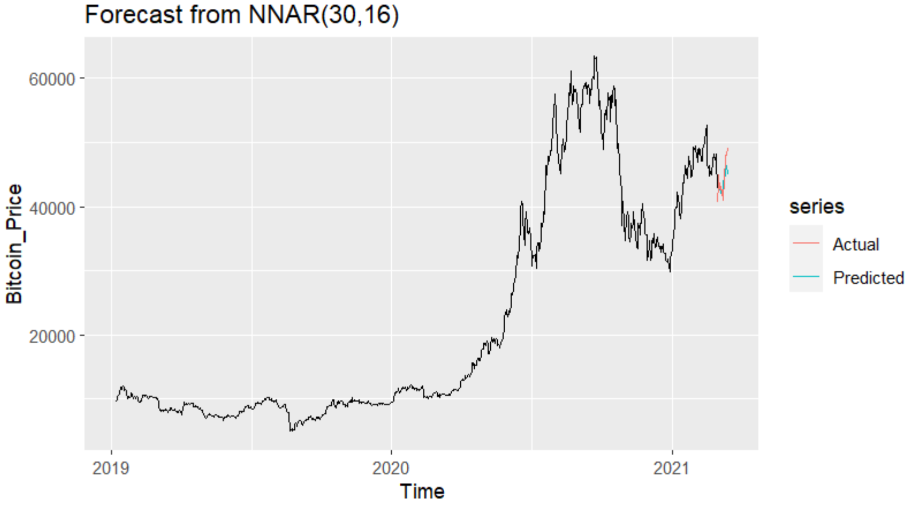

Figure 4.15: The Forecast Plot of NNAR(30,16) Model

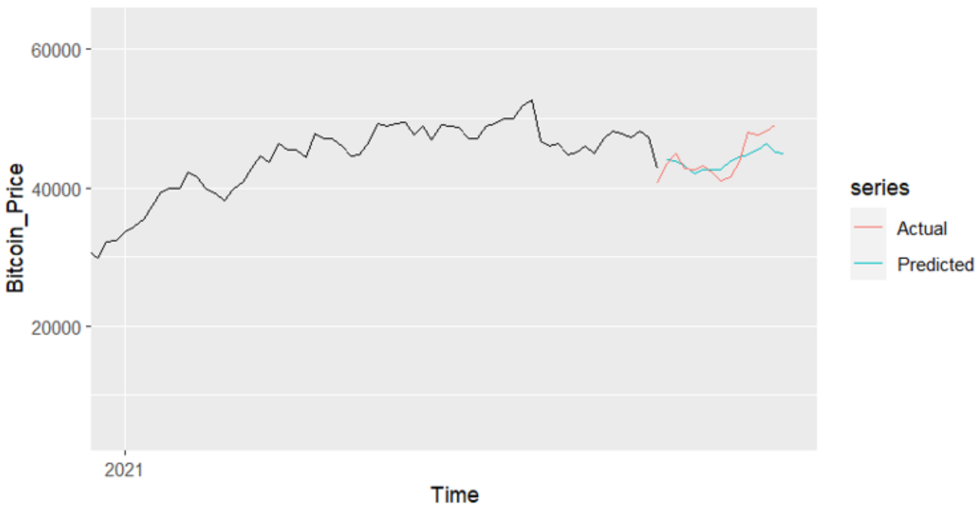

Figure 4.16: The zoomed version of the Forecast Plot of NNAR(30,16) Model

According to Figure 4.15 & 4.16, actual and predicted values are closer to each other compared to previous models. Table 4.15 shows the forecast performance of the model. In addition, MAPE values is less than 10%. This shows that the predictive performance of the NNAR model is good for forecasting the bitcoin prices, and it is better than the ARIMA, the Holt's Exponential model and the Prophet model.

Table 4.15: The Forecast Performance of the NNAR (30,16)

ME	RMSE	MAE	MPE	MAPE
939.889	2175.03	1589.237	1.8717	**3.42**

4.3.2.3 Bayesian Regularized Neural Network

BNN is obtained by brnn () function in brnn package in RStudio (Rodriguez & Gianola, 2020).The function constructs a two-layer neural network. While constructing the model, it scales the input which are historical observations and output. The network is trained by using the Back Propagation and 1000 epoch are used to train as a default. After constructing the model, the forecast values are obtained by the predict () function. Then, RMSE and MAPE values are calculated.

By selecting "normalize" choice as TRUE, normalizing inputs and outputs are provided. Since the time series have autocorrelation structure within the observations, the time delay embedding form is constructed. The optimal lag value is selected as 7 since the autocorrelation decreases quite linearly until lag 7, the time delay embedding form is obtained. The data in this form is divided into two parts as train and test set as in the previous methods. The test values sizes are taken as 14 days. By doing random research, epoch number is selected as 10000, the model is formed, and the following 14 days are forecasted. The predicted values and the actual values are compared, and the following results are obtained.

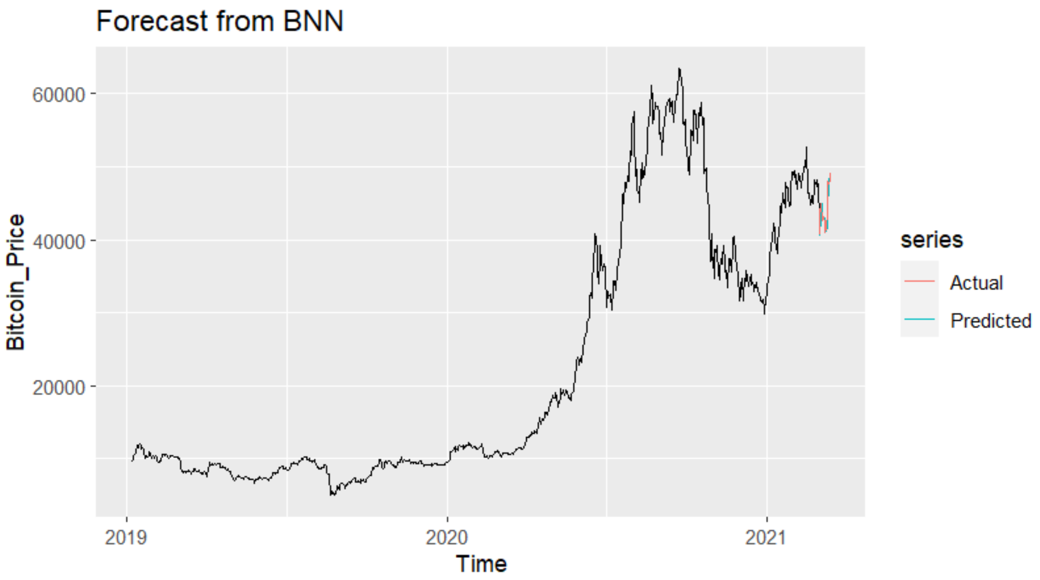

Figure 4.17: The Forecast Plot of BNN Model

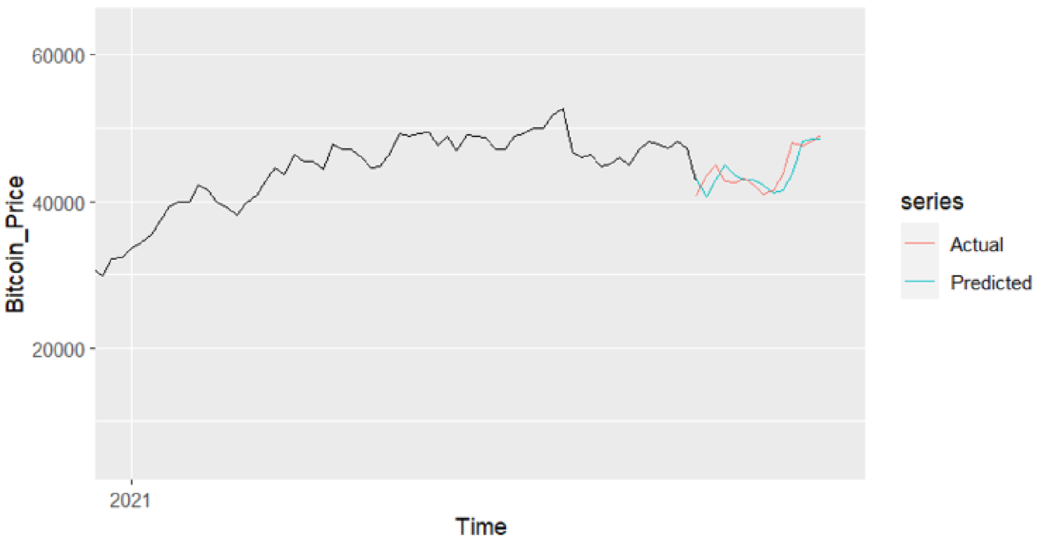

Figure 4.18: The zoomed version of the Forecast Plot of BNN Model

The predicted values and actual ones are closer to each other compared to previous models according to Figure 4.17 & 4.18. Furthermore, the performance results of

BNN model shows that the BNN model is good for predicting bitcoin prices. The BNN model shows quite high performance than the previous models.

Table 4.16: The Forecast Performance of the BNN Model

ME	RMSE	MAE	MPE	MAPE
646.6371	1805.123	1403.089	1.25515	**3.06**

4.3.2.4 Long Short-Term Memory

Long Short-Term Memory (LSTM) network is capable of long-term dependencies. LSTM is able to maintain long term and short-term states. In this study, Keras which is the most popular deep learning library is used, and it runs on the top of the Tensor Flow. In addition, Tensor Flow is used in this study.

Firstly, the train set is scaled between 0 and 1 by min-max scaling. LSTM predicts based on the lagged values; therefore, the lagged form of the train set is formed by taking the lag 7. Keras LSTM needs tensor format of a shape 3D array format [samples, timesteps, features] for predictors and target values. The samples describe the number of observations being processed in batches. Timesteps show the lags and features show the number of predictors. After scaling, the train data are formed as 3D array format. The same procedure is applied on the test data.

LSTM model is built using keras_model_sequential function and adding to layers. Two hidden layers are used, the size of the layer is selected as 50, the learning rate is selected as 0.001 and batch size is taken as 14, the activation function is selected as hyperbolic tangent by doing random search and loss function is selected as the mean squared error. The number of epochs is specified as 1000 by controlling the loss function graphs. The LSTM model provided by R Studio is very limited to tune the hyperparameters. The batch size, dropout, initializer or optimizer cannot be changed. The LSTM model is constructed with these hyperparameters, and the following 14 days are forecasted. The forecasts are transformed by taking the inverse of min-max scaling. The actual and predicted prices are compared.

Figure 4.19 & 4.20 shows the actual and the predicted values. As it can be understood, the forecasts are far from the actual prices.

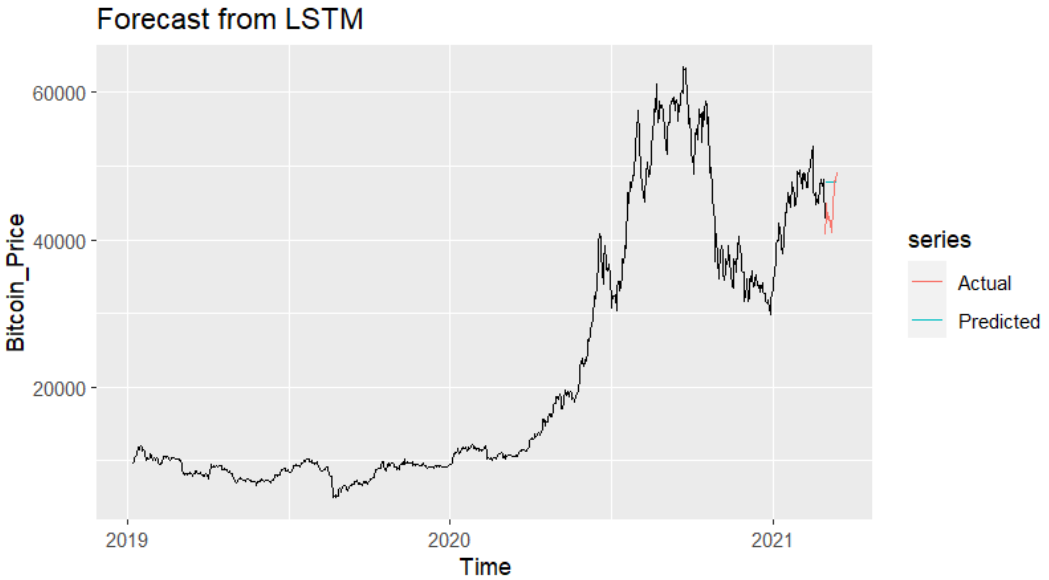

Figure 4.19: The Forecast Plot of LSTM Model

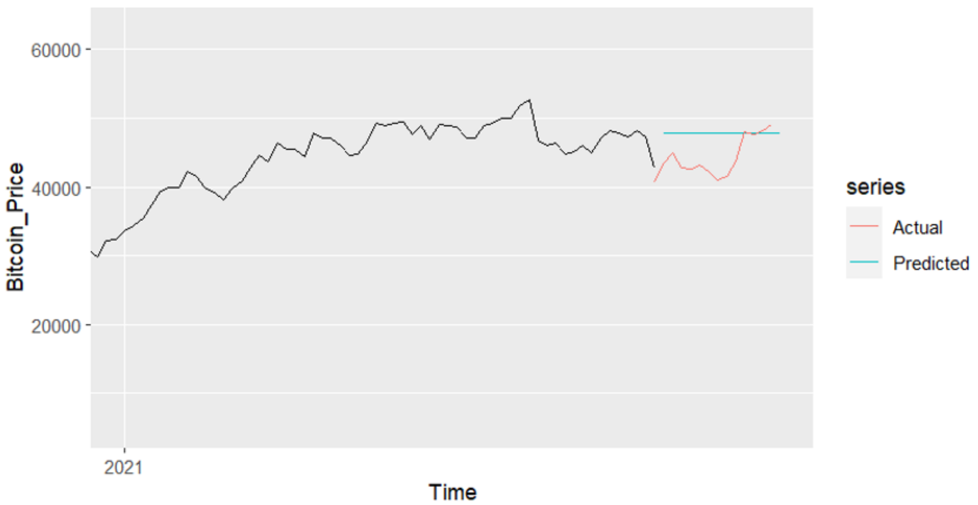

Figure 4.20: A zoomed-in version of the Forecast Plot of LSTM Model

According to Figure 4.19 & 4.20, the actual and the predicted values are far from each other. While there is mostly increasing acceleration in the actual values, the predicted values mostly stay steady.

Table 4.17: The Forecast Performance of the LSTM Model

ME	RMSE	MAE	MPE	MAPE
1860.299	2258.87	2119.05	3.7901	**4.37**

When we look the forecast performance of the LSTM model, the MAPE value is 4.37. Since the MAPE value is less than 10%, it seems good. The predictive performance of the LSTM model is better than the ARIMA, Holt's Exponential Smoothing method and the Prophet model. However, the NNAR and the BNN models represents better performance than the LSTM model. Table 4.18 shows the forecasts of the LSTM model.

Table 4.18: The Forecasts of LSTM Model for 14 Days

Time	Actual Prices	Forecasted Prices	Forecast Error
2021-09-21	40693.68	46274.88	-5581.208
2021-09-22	43574.51	46333.98	-2759.476
2021-09-23	44895.10	46331.03	-1435.936
2021-09-24	42839.75	46284.42	-3444.669
2021-09-25	42716.59	46268.05	-3551.459
2021-09-26	43208.54	46321.38	-3112.839
2021-09-27	42235.73	46338.29	-4102.563
2021-09-28	41034.54	46301.30	-5266.762
2021-09-29	41564.36	46302.84	-4738.475
2021-09-30	43790.89	46260.34	-2469.450
2021-10-01	48116.94	46332.36	1784.586
2021-10-02	47711.49	46315.16	1396.332

2021-10-03	48199.95	46321.07	1878.883
2021-10-04	49112.90	46269.27	2843.631

According to Table 4.18, the actual and predicted values are very different from each other.

4.3.2.5 Random Forest

Random forest model is formed via randomForest function in randomForest package. The Random Forest model is fitted to predict the future values of the bitcoin prices. Since the series have autocorrelation between the observations, time delay embedding form must be constructed. The embedded version is obtained by taking the lag 7 as with other models. The hyperparameter mtry showing the number of variables sampled as candidates at each split. By applying tuneRf () function to the embedded form the mtry value is specified as 2, it gives the smallest out-of-bag (OOB) error. The parameter ntree showing the number of trees to grow taken as 500 by doing the random search. The model is constructed via randomForest function with these hyperparameters on the embedded form of the data and following 14 days predicted and the performance measures are calculated.

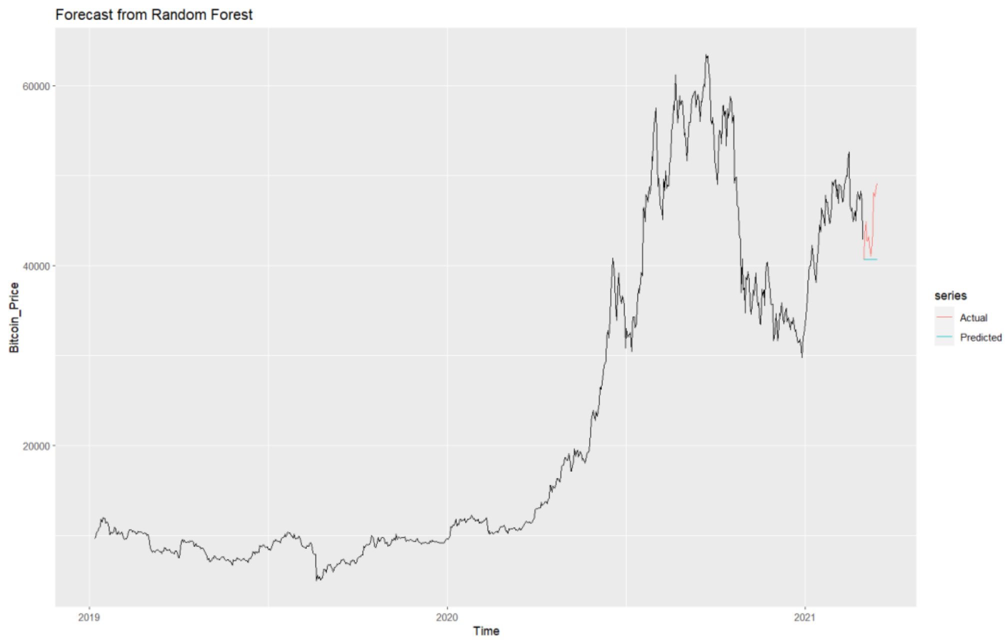

Figure 4.21: Forecast of Random Forest Model

The predicted values obtained from the Random Forest model are very similar at the beginning of the forecasts but then forecasts diverge from the actual ones as seen in Figure 4.15. Table 4.19 shows the forecast performance of the Random Forest model, and Table 4.20 shows the forecasts of the Random Forest model.

Table 4.19: The Forecast Performance of the Random Forest Model

ME	RMSE	MAE	MPE	MAPE
-3563.619	4508.026	3563.619	8.755	**8.75**

When we look the MAPE values, it is 8.75. The MAPE value seems good although the actual and the predicted values look very different from each other since it is less than 10%. The predictive performance of the Random Forest model is better than only the Prophet model. The other previous models' predictive performances are quite higher than the Random Forest model.

4.3.3 Machine Learning Models with Exogenous Variables

In this section machine learning models are used on the bitcoin series with the exogenous variables such as economic, technological variables and two top cryptocurrencies in order to predict the future value of the bitcoin series. The used machine learning models are Feed Forward Neural Network, Long Short-Term Memory and Random Forest. Before applying these models, Boruta () function is applied in Rstudio and some of the variables are founded unimportant on the bitcoin prices. The models are constructed both using all the variables and also with important variables found by the Boruta algorithm. The forecast performances are compared, and big differences do not emerge. Thus, the analysis continues with all of the variables. In order to forecast the bitcoin prices, the following 14 values of the economic and the technological variables are predicted by using *modeltime* package in RStudio. The ethereum, the tether, and the predicted economic and technological variables are used for predicting.

4.3.3.1 Feed Forward Neural Network

In the forecasting procedure of the bitcoin prices, Feed Forward Neural Network model is formed with the exogenous variable on the bitcoin series. Firstly, the time delayed form of the bitcoin series is constructed. After this, the cross correlation of the bitcoin and the two top cryptocurrencies are analyzed. The maximum correlation is obtained at lag 2 for the ethereum and the maximum correlation is gotten at lag 4 for the tether. The two delayed version of the ethereum series, the four delayed form of the tether series, technological variables and economic variables are combined with the time delayed form of the bitcoin series. Since the nnetar () function can normalize the variables, it is not necessary to scale the series. The model is constructed on the bitcoin series and xreg is taken as the combined variables.

After applying nnetar() function with exogenous variables for the model, it suggests NNAR(3,14). The number of non-seasonal lags taken as inputs is 3 and the number of the nodes in the hidden layer is 14. The other models are generated and NNAR (4,16) gives the best result. The structure of the model is “32-16-1”. 32 shows non-seasonal lags used as inputs,16 shows the number of the nodes in the hidden layer and 1 shows the output. The following 14 days are predicted with on the exogenous variables’ test values, and the predicted values are compared with the actual ones.

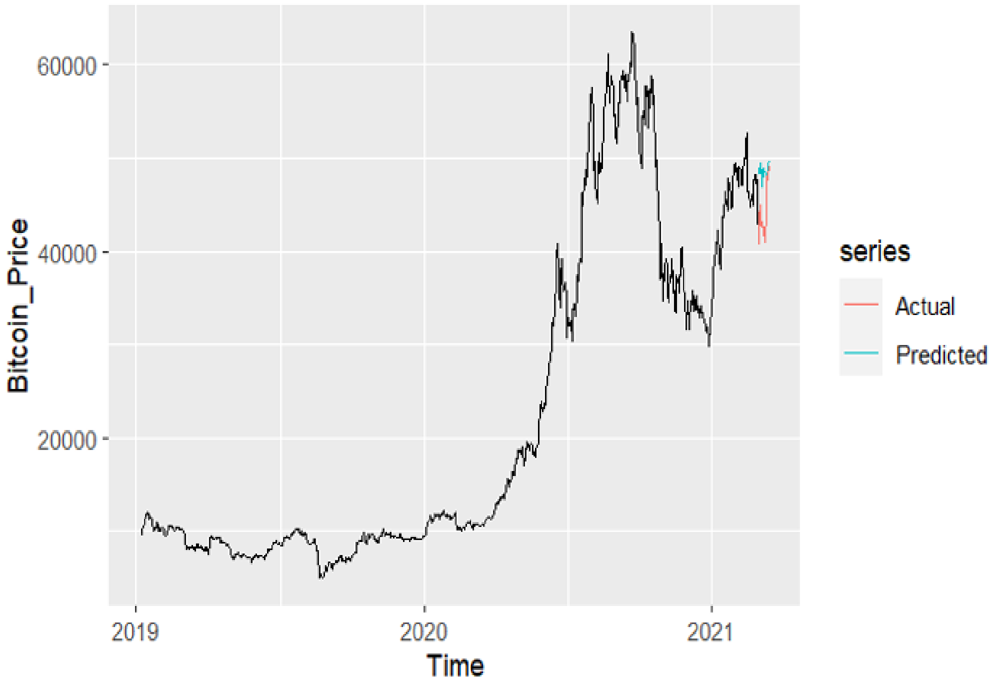

Figure 4.22: The Forecast of Feed Forward Neural Network Model (NNAR(4,16))

According to the Figure 4.22, the predicted values are mostly far from than the actual ones. When we look Table 4.20, the MAPE value is 9.18, which is good. However, the predictive performance of the NNAR model is quite lower than the previous models except from the Prophet model. The predictive performance of the NNAR model with exogenous variables do not increase as expected.

Table 4.20: The Forecast Performance of the NNAR Model

ME	RMSE	MAE	MPE	MAPE
4266.265	5146.174	4462.337	8.764595	**9.18**

4.3.3.2 Long Short-Term Memory

LSTM model is built using keras_model_sequential function and adding to layers. Firstly, all of the datasets are scaled between 0 and 1. After this, the time delay embedding form of the bitcoin series is formed by taking the lag at 7. The two delayed form of the ethereum and the four delayed form of the tether series are combined with the scaled form of the economic and the technological variables. The two hidden layers are used, the size of the layer is selected as 50. As in the previous LSTM model used with only the bitcoin prices, the learning rate is selected as 0.01 and batch size is taken as 14, the activation function is selected as hyperbolic tangent by doing a random search. In addition, the loss function is selected as the mean squared error. The number of epochs is specified as 500 by controlling the loss function graphs. The following 14 days are forecasted with the time delayed form of the bitcoin prices and the predicted exogenous variables. The actual and the predicted values are compared.

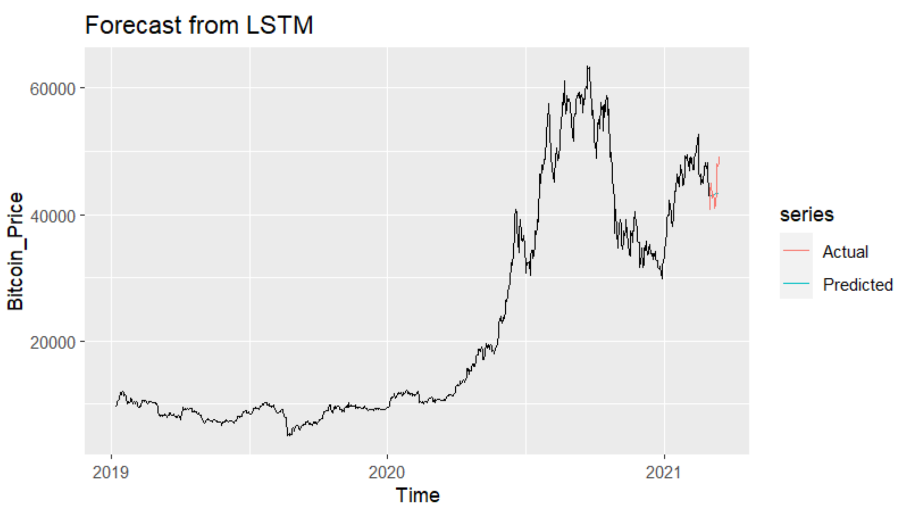

Figure 4.23: The Forecast Plot of LSTM Model

According to Figure 4.23, the actual and predicted values of the bitcoin prices are far from each other. When we look Table 4.21, the MAPE value is 8.37%. Since the MAPE value is less than 10%, it can be said that this model is good for forecasting Bitcoin prices.

Table 4.21: The Forecast Performance of the LSTM Model

ME	RMSE	MAE	MPE	MAPE
2840.118	3258.87	2719.05	5.8901	**8.37**

The predictive performance of the LSTM model with exogenous variables is worse than the LSTM used with only the bitcoin prices since its MAPE values is higher than the previous LSTM model. The predictive performance of the LSTM model does not increase as expected with using the exogenous variables.

4.3.3.3 Random Forest

Random forest model is constructed via randomForest function in randomForest package. All the datasets are transformed by min-max scaling between 0 and 1. The model is formed with the time delayed form of the bitcoin series at lag 7, the two delayed form of the ethereum series, the four delayed form of the tether series, the technological and the economic variables. These all of the series are combined. The two important hyperparameters which are mtry and ntree are specified. By applying tuneRF () function on the time delayed form of the bitcoin series with the combined form of the exogenous variables. According to this function result, the mtry value is defined as 8 since it gives the smallest out-of-bag (OOB) errors. In addition, ntree value is defined as 500 by doing random search and the model is constructed with these hyperparameters on the bitcoin series. The following 14 days are forecasted with the predicted values of the exogenous variables. The forecasts are inversely transformed by min-max scaling and compared with the actual values of the bitcoin series.

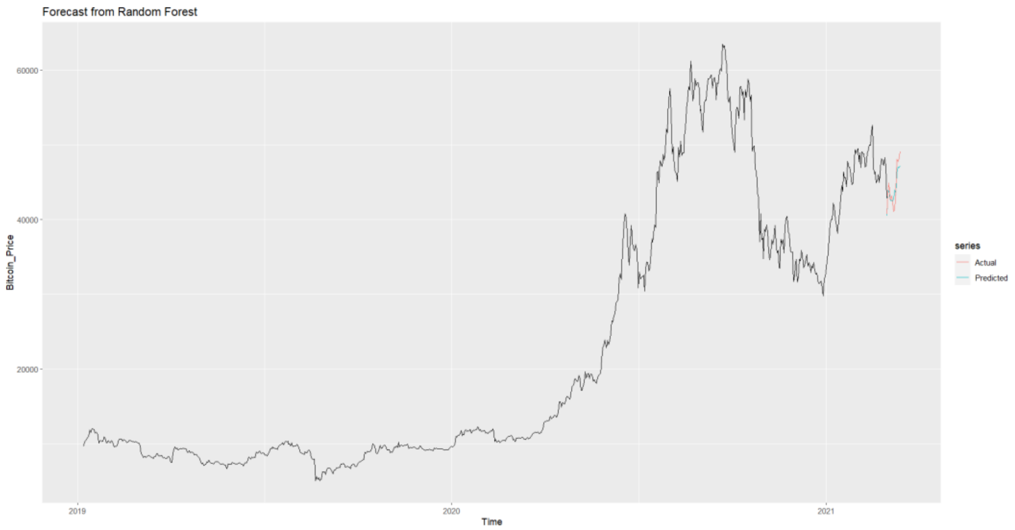

Figure 4.24: The Forecast Plot of Random Forest Model

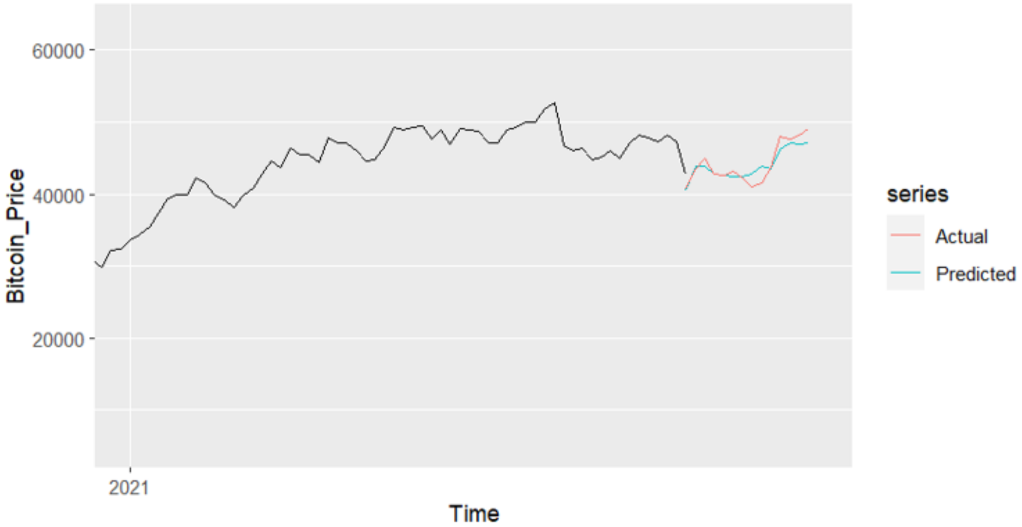

Figure 4.25: The zoomed version of the Forecast Plot of Random Forest Model

According to Figure 4.25 & 4.26, the actual and the predicted values of the bitcoin prices do not seem far from each other. Table 4.22 shows the forecast performance of the Random Forest model. The MAPE value is 1.71%, it is a very good value since it is less than 5%. The predictive performance of the Random Forest model is the best in all of the previous models. The best predictive performance for the bitcoin series is obtained in the Random Forest model with exogenous variables. The main purpose of the study is achieved via the Random Forest model.

Table 4.22: The Forecast Performance of the Random Forest Model

ME	RMSE	MAE	MPE	MAPE
-268.5075	1087.237	821.4592	-0.5646	**1.71**

4.3.4 Assessments of Forecast Performance of All Models

The following 14 days forecasts' performance of the classical time series models on the bitcoin daily prices are showed in Table 4.23. By using only the bitcoin prices, the highest predictive performance is obtained from the Holt's Exponential Smoothing method from among the classical time series methods although the RMSE and the MAPE values are great.

Table 4.23: The Forecast Performance of Classical Time Series Models Using Bitcoin Prices

Model	**RMSE**	**MAPE (%)**
ARIMA (7,1,7)	3431.65	**5.70**
Holt's Exponential Smoothing	1378.363	**4.75**

Table 4.24 shows the forecast performance of the machine learning models using only the bitcoin prices. The lowest performance is obtained with the Prophet model on the bitcoin prices according to Table 4.24. The Bayesian Regularized Neural Network and the Feed Forward Neural Network shows the similar performance, their MAPE values are close to each other. The Bayesian Regularized Neural Network shows better performance among others. Although the best performance among them was achieved with the Bayesian Regularized Neural Network, a good result could not be obtained as the studies in the literature.

Table 4.24: The Forecast Performance of Machine Learning Models Using the Bitcoin Prices

Model	**RMSE**	**MAPE**
Prophet	6717.349	**14.61**
Feed Forward Neural Network	2175.03	**3.42**
Bayesian Regularized Neural Network	1805.123	**3.06**
Long Short-Term Memory	2258.87	**4.37**
Random Forest	4508.026	**8.76**

Table 4.25 shows the forecast performance of Machine Learning models using exogenous variables. The Feed Forward Neural Network and the Long Short Term Memory with exogenous variables shows quite lower performance than the most of the models used with only the bitcoin prices. The highest performance is obtained with the Random Forest model with exogenous variables. Its MAPE value is 1.71%. Thus, the best performance is obtained with the Random Forest model using with the top cryptocurrencies, the economic and the technological variables.

Table 4.25: The Forecast Performance of Machine Learning Models Using Exogenous Variables

Model	**RMSE**	**MAPE**
Feed Forward Neural Network	5146.174	**9.18**
Long Short-Term Memory	3258.870	**8.37**
Random Forest	1087.237	**1.71**

CHAPTER 5

CONCLUSION

Forecasting is a method to find the future value of the things being wondered from people. Many techniques have been used to predict the future value of the interested things. For instance,the future could be predicted based on cycle of maggots in a rotten sheep live. People consulted the Oracle who made predictions while drunk. Forecasting has always been of great importance in people's life. Forecasting with high accuracy has became the main area of the interest. Cryptocurrencies being digital or virtual currency secured by cryptography not only serve alternative investment but also have possibility to switch society (Frankenfield, 2022). Bitcoin is the most popular cryptocurrency, so predicting the future value of the bitcoin prices has become foremost area. In the study, forecasting the bitcoin prices using other cryptocurrencies, and economic and technological variables higher forecast performance is aimed via classical time series and machine learning models.

First, the various data pre-processing methods are applied to the daily bitcoin series. The classical time series forecasting methods are constructed using only the bitcoin prices. ARIMA model is applied with the best suitable parameters on the bitcoin series and the following 14 days are predicted. In addition, Holt’s Exponential Smoothing is utilized with the best parameters and the future value of the bitcoin is forecasted. When these forecasts are compared with the actual value of the bitcoin prices, it is resulted that the predictive performance of Holt’s Exponential Smoothing method is higher than the ARIMA model.

Second, the various machine learning models which are LSTM, Feed Forward Neural Network, Bayesian Regularized Neural Network, and Random Forest are also applied using only the bitcoin series. The various pre-processing methods are applied before constructing the models. These machine learning models are formed with appropriate hyperparameters, the following 14 days are forecasted and compared with the actual values. The forecast performance of the models shows that the highest predictive performance is obtained via Bayesian Regularized Neural Network. It has resulted that though some of the machine learning models shows poor performance than the classical time series models, the best results are obtained from the machine learning models. The machine learning models have outstripped the classical time series models on predicting bitcoin prices.

Finally, the machine learning models are formed on the bitcoin prices with exogeneous variables which are two top cryptocurrencies, economic and technological variables. At first, the cross correlation between the bitcoin prices and two cryptocurrencies which are ethereum and tether’s price are investigated. According to the result of the cross correlations, the maximum correlation is obtained at lag 2 for the ethereum and at lag 4 for the tether. Two and four delayed form the ethereum and the tether values are taken respectively. For the economic and the technologic variables, feature selection method is applied and some of the variables are founded unimportant. The delayed forms of the top cryptocurrencies and the important variables are combined, some of the models are applied on the bitcoin prices. Furthermore, the machine models are constructed including all variables and the forecast performances are compared. The analysis continues with all of the economic and technological variables since there is not much effect of the feature selection method. The optimal hyperparameters are selected by doing random search, the machine learning models which are The Feed Forward Neural Network, Long Short-Term Memory and Random Forest are constructed on the bitcoin prices with all of the exogenous variables. The future 14 days of the bitcoin prices are forecasted and compared with the actual values. The Feed Forward Neural Network and the Long Short-Term Memory shows lower predictive performance than the Random Forest model.

The main contribution of the study is forecasting the up-to-date bitcoin prices with the top cryptocurrencies, the economic and the technological variables via the classical time series and the machine learning models and getting better performance results. When all forecast performances of the models are compared, the Random Forest model with exogenous variables shows the best performance on forecasting the bitcoin prices. In addition, the forecast performance reached by the help of this study gave better results than many previous studies in the literature.

For future studies, sentiment analysis can increase the performance of the models. Google trends, Twitter hashtags can be investigated on the high frequency bitcoin prices. The better results can be obtained for the bitcoin price prediction by combining used exogenous variables and these features.

REFERENCES

Abar, H. (2020). Cryptocurrency Price Forecasting with GARCH Model and LS - SVM Regression: An Application on Bitcoin Price. *Turkish Studies-Economics,Finance,Politics*, 705-725.

Ahmed, N. K., Atiya, A. F., Gayar, N. E., & El-Shishiny, H. (2010, September 15). An Empirical Comparison of Machine Learning Models for Tme Series Forecasting. *Econometric Reviews, 29*(5-6), 594-621.

Akay, M., & Kervancı, I. (2020, December 30). Review on Bitcoin Price Prediction Using Machine Learning and Statistical Methods. *Sakarya University Journal of Computer and Information Sciences, 33*, 272-282.

Athanasopoulos, G., & Hyndman, B. (2021). Getting Started. In R. J. Hyndman, & G. Athanasopoulos, *Forecasting: Principles and Practice* (Vol. 3). OTexts.

Ayaz, Z., Fiaidhi, J., Sabah, A., & Ansari, M. (2020, April 27). Bitcoin Price Prediction using ARIMA Model. *International Journal of Internet Technology and Secured Transactions, 10*(4), 396-406.

Aygün, B., & Kabakçı, E. (2021, January). Comparison of Statistical and Machine Learning Algorithms for Forecasting Daily Bitcoin Returns. *European Journal of Science and Technology*, pp. 444-454.

Bakar, N. A., & Rosbi, S. (2017). Autoregressive Integrated Moving Average (ARIMA) Model for Forecasting Cryptocurrency Exchange Rate in High Votality Environment: A New Insight of Bitcoin Transaction. *International Journal of Advanced Engineering Research and Science (IJAERS)*.

Benzekri, M., & Özütler, H. Ş. (2021). On the Predictability of Bitcoin Price Movements: A Short-term Price Prediction with ARIMA. *Journal of Economic Policy Researches*.

Blockchain. (2022). https://www.blockchain.com/

Bouabdallah, A. (2022). *Multimodal Approach for Cryptocurrency.* Universitat Koblenz Landau, Computer Science.

Brito, J., & Castillo, A. (2016). *Bitcoin: A Primer for Policymakers* (Vol. 2). USA: Mercatus Center at George Mason University.

Brockett, P. L., Cooper, W. W., & Pitaktong, U. (1994, September). A neural network method for obtaining an early warning of insurer insolvency. *The Journal of Risk and Insurance, 61*(3), 402–424.

Brown, R. G. (1959). *Statistical forecasting for inventory control.* McGraw-Hill Book.

Brownlee, J. (2019, November 27). *How to Choose a Feature Selection Method For Machine Learning.* Machine Learning Mastery. https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/

Dancho, M., & Vaughan, D. (2021). *anomalize: Tidy Anomaly Detection. R package version 0.2.2. https://github.com/business-science/anomalize*

Derbentsev, V., Datsenko, N., Stepanenko, O., & Bezkorovainyi, V. (2020). Forecasting Cryptocurrency Prices Time Series Using Machine Learning. *2020 IEEE International Conference on Problems of Infocommunications. Science and Technology (PIC S&T)* (pp. 707-712). Kyiv: IEEE.

Frankenfield, J. (2022). *Cryptocurrency Explained With Pros and Cons for Investment.* Investopedia. https://www.investopedia.com/terms/c/cryptocurrency.asp

Frankenfield, J. (2022,). *What Is Bitcoin? How to Mine, Buy, and Use It.* Investopedia. https://www.investopedia.com/terms/b/bitcoin.asp

Graves, A., Jaitly, N., & Mohammed, A.-r. (2013). Hybrid speech recognition with deep bidirectional lstm. *2013 IEEE Workshop on Automatic Speech Recognition and Understanding* (pp. 273-278). Olomouc: IEEE.

Greaves, A., & Au, B. (2015). *Using the Bitcoin Transaction Graph to Predict the Price of Bitcoin.* Stanford University.

Hayes, A. (2022, December 18). *Autoregressive Integrated Moving Average (ARIMA).* Investopedia. https://l24.im/VhYglKf

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation, 9*, 1735-1780.

Holt, C. C. (2004, March). Forecasting Seasonals and Trends by Exponential Weighted Moving Averages. *International Journal of Forecasting, 20*(1), 5-10.

Hyndman, R. J., & Athanasopoulas, G. (2021). Exponential Smoothing. In *Forecasting: Principles and Practice* (Vol. 3). OTexts.

Hyndman, R. J., & Khandakar, Y. (n.d.). Automatic time series forecasting: the forecast. *Journal of Statistical Software, 26*(3), pp. 1-22.

Hyndman, R., Athanasopoulos, C., Bergmeir, C., Caceres, G., Chhay, G., O'Hara-Wild, M., . . . Yasmeen, F. (2021). _ _forecast: Forecasting functions for time series and linear models_. R package version 8.15, <URL:https://pkg.robjhyndman.com/forecast/>.

Investing.com. (2022). https://www.investing.com/crypto/

Iqbal, M. S., Iqbal, M., Jaskani, F. H., Iqbal, K., & Hassan, A. (2021, July 7). Time-Series Prediction of Cryptocurrency Market using Machine Learning Techniques. *European Union Digital Library*, *21*(8).

Janitza, S., & Hornung, R. (2018, August 6). On the overestimation of random forest's out-of-bag error. *Plos One, 13*(8). https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0201904

Katsiampa, P. (2017, September). Volatility estimation for Bitcoin: A comparison of GARCH models. *ScienceDirect, 158,* 3-6.

Khan, M. A., Mukhtar, N., & Jan, A. U. (2020, June). Opinion Mining and Summarization: A Comprehensive Review. *Information Communication Technologies and Robotic Applications, 11*(1), 76-96.

Kjærland, F., Khazal, A., Krogstad, E. A., Nordstrøm, F. G., & Oust, A. (2018, October 15). An Analysis of Bitcoin's Price Dynamics. *Journal Risk and Financial Management, 11*(4), 63.

Krenker, A., Kos, A., & Janez, B. (2011). Introduction to the Artificial Neural Networks. In K. Suzuk (Ed.), *Artificial Neural Networks - Methodological Advances and Biomedical Applications.* InTechOpen. http://www.intechopen.com/books/artificial-neural-networksmethodological-advances-and-biomedical-applications/introduction-to-the-artificial-neural-networks

Krenker, A., Kos, A., & Janez, B. (2011, April 11). Introduction to the Artificial Neural Networks. In *Artificial Neural Networks-Methodological Advances and Biomedical Applications.* IntechOpen.

Kursa, M. B., & Rudnicki, W. R. (2010). Feature Selection with the Boruta Package. Journal of Statistical Software, 36(11), 1-13. http://www.jstatsoft.org/v36/i11/.

Kwon, D.-H., Kim, J.-B., Heo, J.-S., Kim, C.-M., & Han, Y.-H. (2019, June). Time Series Classification of Cryptocurrency Price Trend Based on a Recuurent Neural Network. *Journal of Information Processing Systems, 15*(3), 694-706.

Liantoni, F., & Agusti, A. (2020, May 26). Forecasting Bitcoin using Double Exponential Smoothing Method Based on Mean Absolute Percentage Error. *International Journal on Informatics Visualization, 4*(2).

Liaw, A., & Wiener, M. (2022). Classification and Regression by randomForest. R News

2(3), 18--22.

Livera, A. D., Hyndman , R. J., & Synder, R. D. (2011). Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association*.

Loiseau, J.-C. B. (2019, March 11). *Rosenblatt's perceptron, the first modern neural network.* Towards Data Science. https://towardsdatascience.com/rosenblatts-perceptron-the-very-first-neural-network-37a3ec09038a

Mahanta, J. (2017, July 10). *Introduction to Neural Networks, Advantages and Applications.* Towards Data Science. https://towardsdatascience.com/introduction-to-neural-networks-advantages-and-applications-96851bd1a207

Makinde, F., & Orodu, O. (n.d.). Prediction of crude oil viscosity using feed-forward back-propagation neural network.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018, March 27). Statistical and Machine Learning forecasting methods: Concerns and ways forward. *Plos One*.

Mbbabu, O. (2020, December 11). *Introduction to Random Forest in Machine Learning.* Section. https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/

Mendes, J. F. (2019, September). *Forecasting Bitcoin Prices: ARIMA vs LSTM.* ISCTE-IUL.

Mestiri, S. (2021, December 2019). *Modelling the volatility of Bitcoin returns using Nonparametric GARCH models.* Tunisia: Munich Personal RePEc Archive.

Munim, Z., Shakil, M., & Alon, I. (2019, June 20). Next-Day Bitcoin Price Forecast. *Journal of Risk and Financial Management, 12*(2), 103.

Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System.* Bitcoin.org. https://bitcoin.org/bitcoin.pdf

Neves, A. C., Gonzalez, I., Leander, J., & Karoumi, R. (2017). A New Approach to Damage Detection in Bridges Using Machine Learning. *nternational Conference on Experimental Vibration Analysis for Civil Engineering Structures.*

Pan, S., & Duraisamy, K. (2020). On the Structure of Time-delay Embedding in Linear Models of Non-linear Dynamical Systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science, 30*(7).

Pekel, E., & Kara, S. (2017, March). A Comprehensive Review for Artificial Neural Network Application to Public Transportation. *Sigma Journal of Engineering and Natural Sciences*, 157-179.

Pisa, V., Morell, A., Vicario, L. J., & Vilanova, R. (2020). Denoising Autoencoders and LSTM-Based Artificial Neural Networks Data Processing for Its Application to Internal Model Control in Industrial Environments—The Wastewater Treatment Plant Control Case. *Sensor, 20*(13).

Quast, B. (2016). (2016). rnn: a Recurrent Neural Network in R

Ranjan, S., Kayal, P., & Saraf, M. (2022, April 22). Bitcoin Price Prediction: A Machine Learning Sample Dimension Approach. *Computational Economics*. https://doi.org/10.1007/s10614-022-10262-6

Rodriguez, P. P., & Gianola, D. (2020). . *brnn: Bayesian Regularization for Feed-Forward Neural Networks.* R package version 0.8. https://CRAN.R-project.org/package=brnn.

RStudio Team (2022). RStudio: Integrated Development for R. RStudio, PBC, Boston, MA URL http://www.rstudio.com/.

Saeed, M. (2022, September 9). *An Introduction to Recurrent Neural Networks and the Math That Powers Them.* Retrieved from machinelearningmastery: https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them/.

Salman, A., Heryadi, Y., & Abdurahman , E. (2018). Single Layer & Multi-layer Long Short-Term Memory (LSTM) Model with Intermediate Variables for Weather Forecasting. *The 3rd International Conference on Computer Science and Computational Intelligence (ICCSCI 2018):Empowering Smart Technology in Digital Era for a Better Life. 135*, pp. 89-98. Procedia Computer Science.

Samsudin, R., Shabri, A., & Saad, P. (2010, January). A comparison of time series forecasting using support vector machine and artificial neural network model. *Journal of Applied Sciences*, pp. 950-958.

Taylor, S., & Letham, B. (2021). *prophet: Automatic Forecasting Procedure. R package version 1.0.* https://CRAN.R-project.org/package=prophet.

The Wall Street of Journal. (2022). https://www.wsj.com/market-data.

Winters, P. R. (1960). Forecasting Sales by Exponentially Weighted Moving Averages. *Management Science* , *6*(3), 324-342.

Winters, P. R. (n.d.). Forecasting sales by exponentially weighted moving averages. *Management Science*.

Zheng, J., Xu, C., Zihang, Z., & Li, X. (2017). Electric Load Forecasting in Smart Grids Using Long-Short-Term-Memory based Recurrent Neural Network.

2017 51st Annual Conference on Information Sciences and Systems (CISS) (pp. 1-6). Baltimore: IEEE.