

Analysis of Bitcoin Data: A Case Study on Modelling and Forecasting

Submitted by

Arpit Dubey

Indian Institute of Technology, Kharagpur

under the guidance of

R Team, IIT BOMBAY

July 2024

Contents

1	Abstract	2
2	Introduction	3
3	Methodology	4
3.1	Data Collection	4
3.1.1	Introduction	4
3.1.2	Data Sources	4
3.1.3	Data Description	4
3.2	Data Analysis	5
3.3	Modeling and Forecasting	8
3.3.1	ARIMA Model	8
3.3.2	Threshold Autoregressive (TAR) Model for 2 Regime	10
3.3.3	Threshold Autoregressive (TAR) Model for 3 Regimes	14
4	Results	18
4.1	Forecasting using ARIMA model:	18
4.2	Forecasting using Tar model for regime 3:	19
4.3	Forecasting using Tar model for regime 2:	20
4.4	Model Evaluation	22
5	Conclusion	24
	References	25

1 Abstract

The volatility of Bitcoin prices has garnered significant interest from both researchers and investors. This case study aims to analyse historical daily closing prices of Bitcoin, obtained from Coinbase [1] and the Federal Reserve Bank of St. Louis [3], using statistical tools to predict future prices. Predictions were made using both linear (ARIMA) and non-linear (TAR) time series models. Model parameters were optimized for accuracy, and performance was evaluated using metrics such as the Akaike Information Criterion (AIC) and Root Mean Square Error (RMSE). Results were compared to assess the suitability of each model for forecasting Bitcoin prices, providing insights into effective strategies for cryptocurrency market analysis and prediction.

2 Introduction

This report provides a detailed analysis of Bitcoin price dynamics from December 1, 2014, to June 23, 2024, using daily closing prices obtained from Coinbase [1] through the Federal Reserve Bank of St. Louis' FRED [3] platform. It explores Bitcoin's volatility and unpredictability through advanced time series models, including ARIMA and TAR, and assesses model performance using metrics such as RMSE and AIC. The study offers insights into the cryptocurrency market and its broader economic implications, aiming to provide a structured framework for analysing digital asset data and enhancing decision-making for investors, policymakers, and researchers.

3 Methodology

3.1 Data Collection

3.1.1 Introduction

The dataset contains daily closing prices of Bitcoin in U.S. Dollars from Coinbase, spanning December 1, 2014, to June 23, 2024, recorded at 5 PM PST. The documentation aims to provide comprehensive details about the dataset for use in financial analysis, econometrics, and cryptocurrency research. Primary objectives include time series analysis, statistical modelling, market behaviour studies, investment analysis, and educational research. This dataset is valuable for researchers, analysts, and enthusiasts studying Bitcoin market dynamics.

3.1.2 Data Sources

The following table shows the information about the dataset we used in our report (for reference to the data source and its legal use, you can refer to [1] and [2], respectively).:

Data Source	Year	Unit	Frequency
Coinbase Bitcoin (CBBT-CUSD) on FRED.	2014-2024	U.S. Dollars, Not Seasonally Adjusted	Daily, 7-Day

Table 1: List of data sources used to construct the dataset.

3.1.3 Data Description

The dataset is in CSV format. The description of the dataset's columns is given in the table below:

Attributes	Description	Data type
Date	The date of the recorded Bitcoin price	Character
CBBTCUSD	Closing price of Bitcoin in U.S. Dollars at 5 PM PST	Character

Table 2: Description of dataset.

3.2 Data Analysis

Preprocessing the dataset involved the following steps:

- Loading the Data:

```
1 path = "CBBTCUSD.csv"
2 df = read.csv(path)
3 head(df)
```

Description: df [6 x 2]

	DATE <chr>	CBBTCUSD <chr>
1	2014-12-01	370
2	2014-12-02	378
3	2014-12-03	378
4	2014-12-04	377.1
5	2014-12-05	.
6	2014-12-06	378

6 rows

Figure 1: First 6 values of the dataset

The first six values of the dataset. We can clearly see that there are some missing values.

- Inspecting the Data:

```
1 str(df)
```

```
'data.frame':  3483 obs. of  2 variables:
 $ DATE   : chr  "2014-12-01" "2014-12-02" "2014-12-03" "2014-12-04" ...
 $ CBBTCUSD: chr  "370" "378" "378" "377.1" ...
```

Figure 2: summary of the Data

- Plotting of the Data:

```
1 plot(as.Date(df$DATE), as.double(df$CBBTCUSD), type = "l", xlab = "Date",
      ylab = "Bitcoin Price (USD)", main = "Bitcoin Prices Over Time")
```

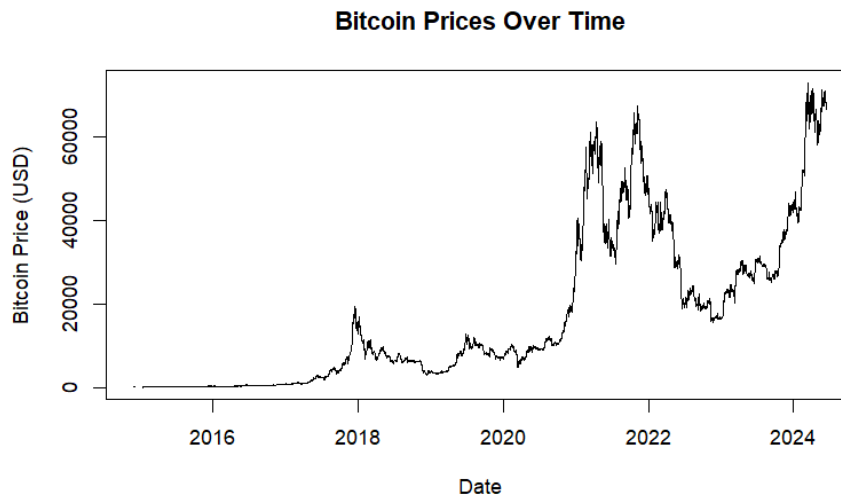


Figure 3: Time Series Plot of Bitcoin Closing Prices

- **Columns datatype, renaming, and handling missing values:**

```

1  # Assign column names
2  colnames(df)= c("Date","Bitcoin")
3  # Convert the datatype
4  df$Date=as.Date(as.character(df$Date))
5  df$Bitcoin=as.double(as.character(df$Bitcoin))
6  # number of missing entries
7  cat("total number of missing values in the entire Dataset is :",sum(is
    .na(df)))

```

In the above code, we assigned the column names "Date" and "Bitcoin" to the data frame df, converted the "Date" column to date format, and the "Bitcoin" column to numeric format to ensure proper data handling. We also counted the number of missing values (NA) in each dataset column.

Total number of missing values in the entire Dataset is : 35

Figure 4: Number of missing values

- **Handling Missing Values:** Missing values in the dataset were identified and replaced using linear interpolation to ensure a continuous time series without gaps. A total of 35 missing values were filled using this technique.

```

1  df$Bitcoin <- na.approx(df$Bitcoin, rule=2)

```

For linear interpolation, the na.approx function was used with rule=2.

- **Subsetting the Data:** The dataset was resized to focus on the period from 2021 to 2024. This was done to use newer, more variable data to make better predictions.

```
1 df <- df[df$Date >= as.Date("2021-01-01"), ]
```

- **Transformation:** We applied a log transformation to the Bitcoin column as it preserves the proportional relationships between data points.

```
1 df$Bitcoin=log(df$Bitcoin)
2 ggplot(dframe, aes(x = Date, y = Bitcoin)) +geom_line()+xlab("Date")+
  ylab("Bitcoin in USD")+labs(title = "Bitcoin Prices Over Time")
```

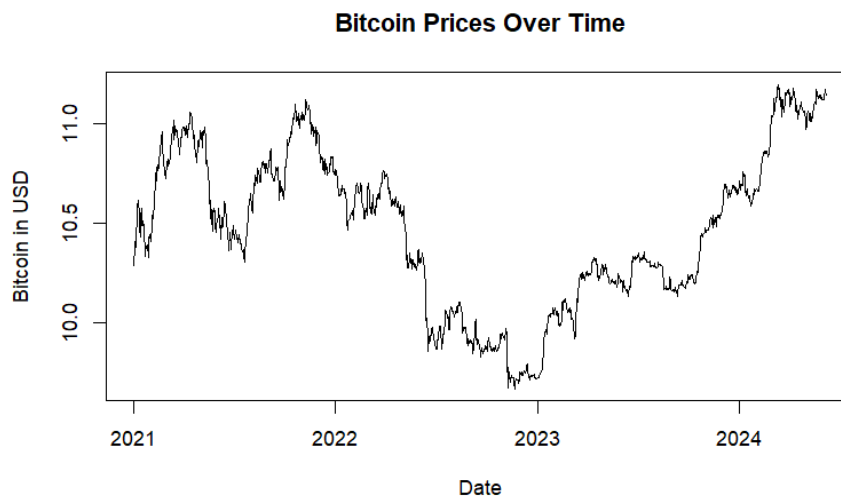


Figure 5: Plot of Bitcoin Closing Prices

- **Summary Statistics:** Descriptive statistics, such as mean, variance, and standard deviation, were computed.

The mean of the Bitcoin column of the Dataset is: 10.46837

Variance of the Bitcoin column of the Dataset is: 0.1590694

The standard Deviation of the Bitcoin column of the Dataset is: 0.3988351

```
1 cat("The Mean of the Bitcoin column of the Dataset is :",mean(df$
  Bitcoin))
2 cat("\n Variance of the Bitcoin column of the Dataset is :",var(df$
  Bitcoin))
3 cat("\n The Standard Deviation of the Bitcoin column of the Dataset is
  :",sd(df$Bitcoin))
```

- **Visualization:** Time series plots (See figure 5), along with Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots, have been generated to visualize the data and identify any temporal dependencies.

```

1 ggplot(dframe, aes(x = Date, y = Bitcoin)) +geom_line()+xlab("Date")+
  ylab("Bitcoin in USD")+labs(title = "Bitcoin Prices Over Time")
2 acf(dframe$Bitcoin,main="ACF plot for Bitcoin values")
3 pacf(dframe$Bitcoin,main="PACF plot for Bitcoin values")

```

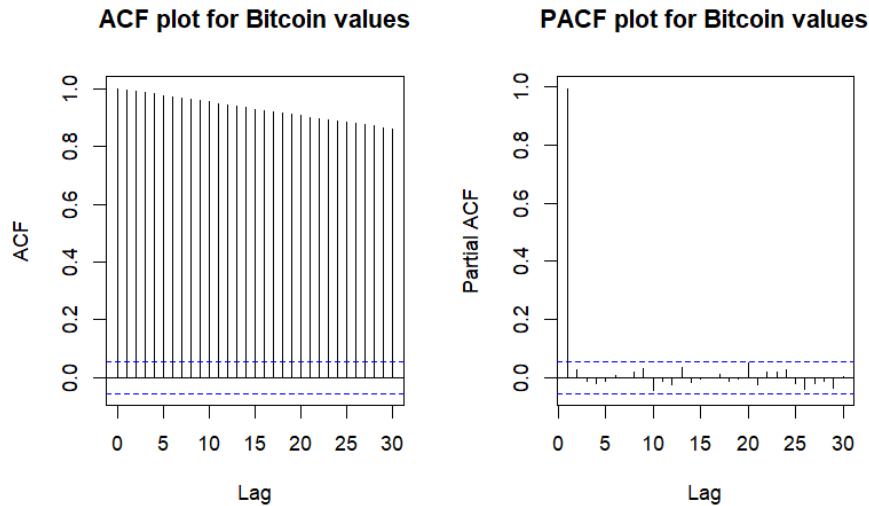


Figure 6: ACF and PACF Plots

1. Plot ACF and PACF for your data.
2. Identify significant lags in ACF/PACF exceeding confidence intervals.

3.3 Modeling and Forecasting

Three different models were considered for time series analysis and forecasting:

3.3.1 ARIMA Model

- **ARIMA Model Summary:** The ARIMA model with parameters AR(1), differencing of order 1, and MA(0) is represented by the following equation:

$$Y_t - Y_{t-1} = c + \phi_1(Y_{t-1} - Y_{t-2}) + \epsilon_t$$

where:

- Y_t is the value of the time series at time t ,
- ϕ_1 is the autoregressive parameter,
- c is the constant term (intercept),
- ϵ_t is the error term at time t , assumed to be independently and identically distributed (i.i.d.) with mean zero and constant variance.

Using Auto.arima, We get the values of the parameters of the ARIMA model i.e ARIMA(p,d,q) which best fits the data, Here:

- $p = 1$: Number of autoregressive (AR) terms, capturing the effect of one previous value on the current value.
- $d = 1$: Degree of differencing, transforming the time series into a stationary series.
- $q = 0$: Number of moving average (MA) terms, assuming no moving average component.

The ARIMA model integrates autoregressive (AR), differencing (I), and moving average (MA) components to model and forecast time series data, accommodating trends and autocorrelation structures effectively.

- **Model Fitting:** The ARIMA(1, 1, 0) model was fitted to the transformed Bitcoin data with first-order differencing.

```

1 arima_model=auto.arima(dframe$Bitcoin)
2 summary(arima_model)
3 arima_fitted_val= fitted(arima_model)
4 plot(dframe$Bitcoin, col = 'red', lwd = 1, main = "Fitted Vs Actual",
      ylab = "Bitcoin value ($)",xlab="Date", type = "l")
5 lines(arima_fitted_val, col = "blue", lwd = 1)
6 legend('bottomright', legend = c("Actual", "Fitted"), col = c("red", "
      blue"), lwd = 2)

```

```

Series: dframe$Bitcoin
ARIMA(1,1,0)

Coefficients:
      ar1
    -0.0364
s.e.    0.0283

sigma^2 = 0.001116: log likelihood = 2481.58
AIC=-4959.16   AICC=-4959.15   BIC=-4948.89

Training set error measures:
      ME      RMSE      MAE      MPE      MAPE      MASE      ACF1
Training set 0.0007174924 0.03337957 0.02287173 0.00617688 0.2179083 0.996358 0.0009469197

```

Figure 7: Arima Model Summary

`auto.arima()` simplifies model selection by automatically identifying and fitting the most suitable ARIMA model based on the characteristics of the data. After fitting the model, the forecasting equation obtained is

$$\hat{y}_t = -0.0364 \times y_t$$

where \hat{y}_t represents the predicted values based on the model and y_t denotes the transformed Bitcoin data after 1st order differencing. The linear equation indicates a negative relationship, where an increase in Bitcoin values leads to a predicted decrease in the outcome, and vice versa.

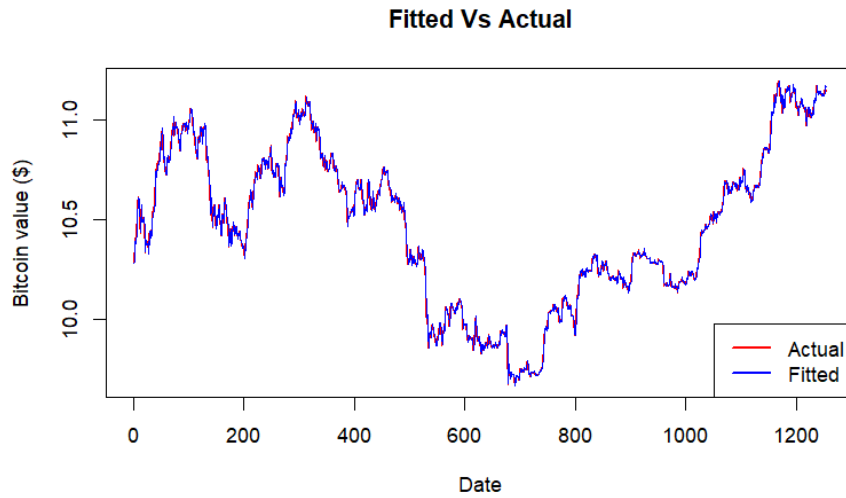


Figure 8: Model Fitting

- **Residual Analysis:** We plotted the residuals of the fitted model to assess its performance.

```
1 plot(dframe$Date, arima_model$residuals, main= "ARIMA Model's Residuals",
      ylab="Residuals", xlab="Date", type="l")
```

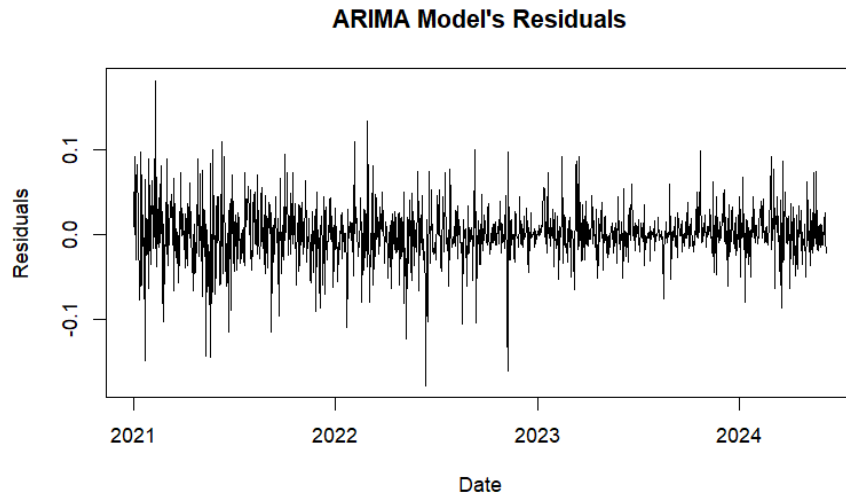


Figure 9: Residuals Plots of ARIMA model

The residuals are centred around zero with relatively constant variance over time, indicating no significant bias and consistent model performance.

3.3.2 Threshold Autoregressive (TAR) Model for 2 Regime

- **Model Specification:** The Threshold Autoregressive (TAR) model is a nonlinear time series model that captures regime shifts or nonlinear behaviours in the data. It is particu-

larly suited for situations where the data exhibit distinct periods of behaviour or different response patterns based on past observations.

- **Model Equation:**

The TAR model used in this analysis can be described as follows:

$$y_t = \begin{cases} \mu_1 + \phi_1 y_{t-1} + \epsilon_t & \text{if } y_{t-1} \leq \theta \\ \mu_2 + \phi_2 y_{t-1} + \epsilon_t & \text{if } y_{t-1} > \theta \end{cases}$$

Where:

- y_t represents the Bitcoin price at time t .
- μ_1 and μ_2 are intercepts for the lower and upper regimes, respectively.
- ϕ_1 and ϕ_2 denote autoregressive coefficients associated with the lower and upper regimes.
- θ denotes the threshold value that determines the shift between regimes based on the lagged value y_{t-1} .
- ϵ_t represents the error term assumed to be white noise.

- **Model Fitting for 2 regime Tar:**

- **The fitted model:** Now, We will fit The 2 regime TAR model on the Bitcoin data using the `setar` function from the `tsDyn` package in R:

```
1  tar_model <- setar(dframe$Bitcoin, m = 1, thDelay = 0, nthresh =
   1, model = "TAR")
2  summary(tar_model1)
3  # tar model fitting plot with two regime
4  const.L <- coef(tar_model)["const.L"]
5  phiL.1 <- coef(tar_model)["phiL.1"]
6  const.H <- coef(tar_model)["const.H"]
7  phiH.1 <- coef(tar_model)["phiH.1"]
8  threshold <- tar_model$coefficients["th"]
9  regime1 <- numeric(length(dframe$Bitcoin))
10 regime2 <- numeric(length(dframe$Bitcoin))
11 for (i in 2:length(dframe$Bitcoin)) {if (dframe$Bitcoin[i - 1] <=
   threshold) {regime1[i] <- const.L + phiL.1 * dframe$Bitcoin[i -
   1]} else {regime2[i] <- const.H + phiH.1 * dframe$Bitcoin[i -
   1] }}
12 plot(dframe$Date, dframe$Bitcoin, type = "l", col = "black", xlab =
   "Date", ylab = "log of Bitcoin Value in $", main = "Fitting of
   TAR Model and 2 Regime Separation Plot")
13 lines(dframe$Date, regime1, col = "yellow")
14 lines(dframe$Date, regime2, col = "green")
15 legend("top", legend = c("Bitcoin", "Regime 1", "Regime 2"), col
   = c("black", "yellow", "green"), lty = 1, cex = 0.8)
16 abline(h = threshold, col = "red", lty = 2)
17 axis.Date(1, at = seq(min(dframe$Date), max(dframe$Date), by = "
   year"))
```

```

Non linear autoregressive model

SETAR model ( 2 regimes)
Coefficients:
Low regime:
  const.L    phiL.1
0.02562248 0.99760588

High regime:
  const.H    phiH.1
0.5247176 0.9523690

Threshold:
-Variable: z(t) = + (1) x(t)
-Value: 10.84
Proportion of points in low regime: 80.69%      High regime: 19.31%

Residuals:
      Min       1Q   Median       3Q      Max
-0.1773253 -0.0143189 -0.0013176  0.0156176  0.1821717

Fit:
residuals variance = 0.001111,  AIC = -8521, MAPE = 0.2185%

Coefficient(s):

      Estimate Std. Error t value Pr(>|t|)
const.L 0.0256225  0.0340481   0.7525  0.45187
phiL.1  0.9976059  0.0032939 302.8638 < 2e-16 ***
const.H 0.5247176  0.2589615   2.0262  0.04295 *
phiH.1  0.9523690  0.0234991  40.5279 < 2e-16 ***
---
signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Threshold
Variable: z(t) = + (1) x(t)
value: 10.84

```

Figure 10: tar model summary 2 regime

The code fits a TAR model ($m = 2$) to Bitcoin prices in data frame `dframe`, setting `thDelay = 0` for immediate threshold application. The summary displays coefficients, thresholds, and diagnostics.

$$\hat{y}_t = \begin{cases} 0.0256225 + 0.9976059 \cdot y_t, & \text{if } y_{t-1} \leq 10.84 \\ 0.5247176 + 0.9523690 \cdot y_t, & \text{if } y_{t-1} > 10.84 \end{cases}$$

- **Model Fitting:** The TAR model with two regimes was fitted to the data with a threshold of 10.471078:

The following equations define the TAR model:

- * **Regime 1:** $\hat{y}_t = 0.0256225 + 0.9976059 \cdot y_t$
- * **Regime 2:** $\hat{y}_t = 0.5247176 + 0.9523690 \cdot y_t$

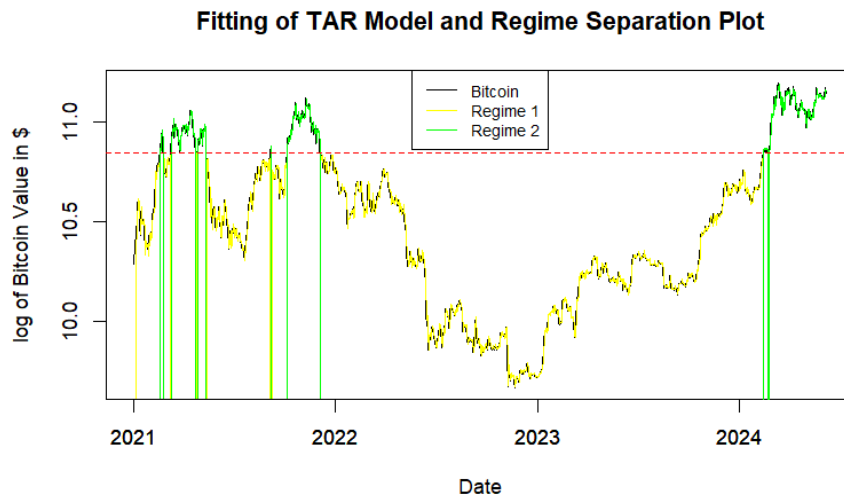


Figure 11: Model Fitting of TAR Model

- **Residual Analysis:** We will plot the residuals for the fitted Tar model of 2 regimes.

```
1 plot(residuals(tar_model1),ylab="Residuals",main="Residuals plot of
   Tar model for 3 regimes")
```

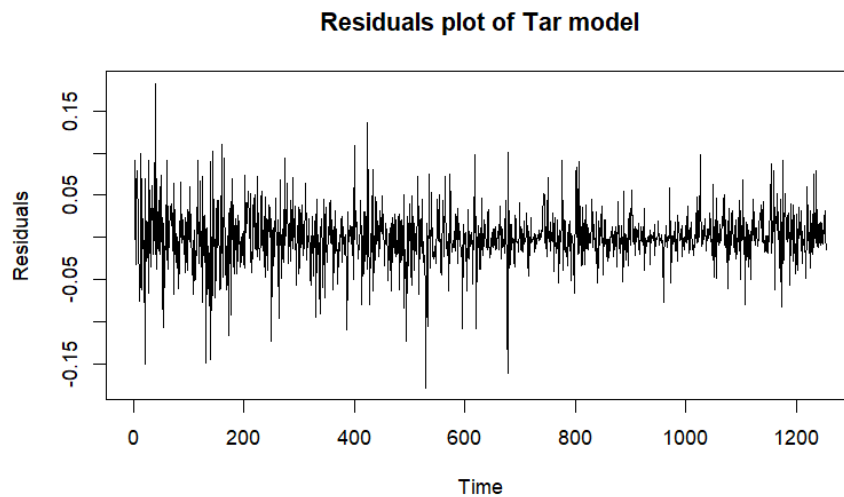


Figure 12: Residuals of TAR Model

The residuals are centred around zero, and the relatively constant variance over time indicates no significant bias and consistent model performance.

- **Fitted Model observations:**
 - The value of Bitcoin exhibits significant volatility, frequently switching regimes during periods such as 2021 and early 2022, but showing greater stability in late 2022 and throughout 2023.

- The TAR model identifies different behaviours: in Regime 2 (above the threshold), the value tends to trend upward or remain stable, while in Regime 1 (below the threshold), it often trends downward or becomes more volatile.

3.3.3 Threshold Autoregressive (TAR) Model for 3 Regimes

- **Model Equation:** The TAR model used in this analysis can be described as follows:

$$y_t = \begin{cases} \mu_1 + \phi_1 y_{t-1} + \epsilon_t & \text{if } y_{t-1} \leq \theta_1 \\ \mu_2 + \phi_2 y_{t-1} + \epsilon_t & \text{if } \theta_1 < y_{t-1} \leq \theta_2 \\ \mu_3 + \phi_3 y_{t-1} + \epsilon_t & \text{if } y_{t-1} > \theta_2 \end{cases}$$

Where:

- y_t represents the Bitcoin price at time t .
- μ_1 , μ_2 , and μ_3 intercepts for the low, middle, and high regimes, respectively.
- ϕ_1 , ϕ_2 , and ϕ_3 denote autoregressive coefficients associated with the low, middle, and high regimes.
- θ_1 and θ_2 denote the threshold values that determine the shift between regimes based on the lagged value y_{t-1} .
- ϵ_t represents the error term assumed to be white noise.
- **Model Fitting for 3-regime TAR:**
 - **The fitted model:** The TAR model was fitted to the Bitcoin price data using the `setar` function from the `tsDyn` package in R:

```
1 library(tsDyn)
2 tar_model1=setar(dframe$Bitcoin, m = 1, thDelay = 0, nthresh = 2,
3   model = "TAR")
4 summary(tar_model)
```

```

Non linear autoregressive model

SETAR model ( 3 regimes)
Coefficients:
Low regime:
  const.L    phiL.1
0.05674433 0.99444240

Mid regime:
  const.M    phiM.1
0.5953270 0.9437478

High regime:
  const.H    phiH.1
0.1858152 0.9830002

Threshold:
-Variable: Z(t) = + (1) X(t)
-Value: 10.37 10.69
Proportion of points in low regime: 43.34%      Middle regime: 25.46%      High regime: 31.21%

Residuals:
      Min       1Q   Median       3Q      Max
-0.17621164 -0.01449618 -0.00058882  0.01559528  0.18166393

Fit:
residuals variance = 0.001106,  AIC = -8520, MAPE = 0.2182%

Coefficient(s):

      Estimate Std. Error t value Pr(>|t|)
const.L 0.0567443  0.0738657   0.7682  0.442508
phiL.1  0.9944424  0.0073275 135.7144 < 2.2e-16 ***
const.M 0.5953270  0.2254693   2.6404  0.008384 **
phiM.1  0.9437478  0.0213495  44.2047 < 2.2e-16 ***
const.H 0.1858152  0.1263567   1.4706  0.141662
phiH.1  0.9830002  0.0115677  84.9784 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Threshold
Variable: Z(t) = + (1) X(t)
value: 10.37 10.69

```

Figure 13:

The above code fits a Threshold Autoregressive (TAR) model with 3 regimes ($m = 3$) to the Bitcoin price series stored in the data frame `dframe`.

$$\hat{y}_t = \begin{cases} 0.0567443 + 0.9944424 \cdot y_{t-1}, & \text{if } y_{t-1} \leq 10.37 \\ 0.5953270 + 0.9437478 \cdot y_{t-1}, & \text{if } 10.37 < y_{t-1} \leq 10.69 \\ 0.1858152 + 0.9830002 \cdot y_{t-1}, & \text{if } y_{t-1} > 10.69 \end{cases}$$

- **Model Fitting:** The TAR model with three regimes was fitted to the data with thresholds of 10.37 and 10.69:

The following equations define the TAR model:

- * **Low regime:** $\hat{y}_t = 0.0567443 + 0.9944424 \cdot y_{t-1}$
- * **Mid regime:** $\hat{y}_t = 0.5953270 + 0.9437478 \cdot y_{t-1}$
- * **High regime:** $\hat{y}_t = 0.1858152 + 0.9830002 \cdot y_{t-1}$

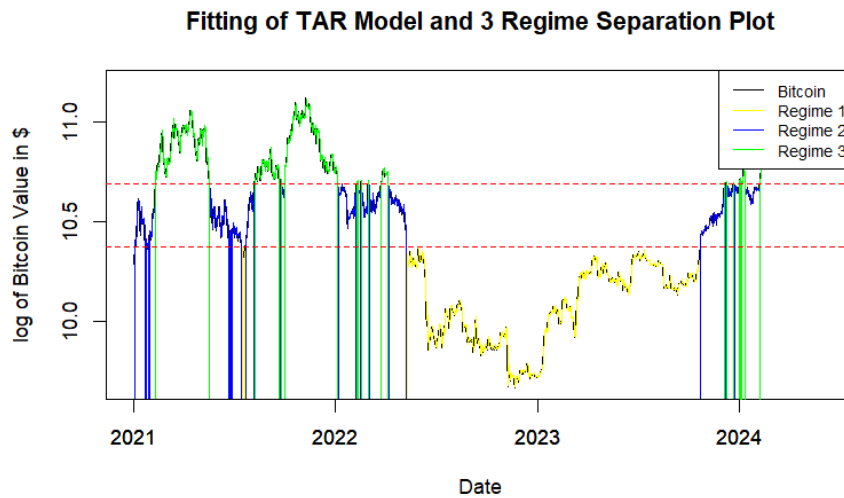


Figure 14: Model Fitting of 3-Regime TAR Model

- **Residual Analysis:** We will plot the residuals of the fitted model to check how well our model performs on the data.

```
1 plot(residuals(tar_model1),ylab="Residuals",main="Residuals plot of
   Tar model for 3 regimes")
```

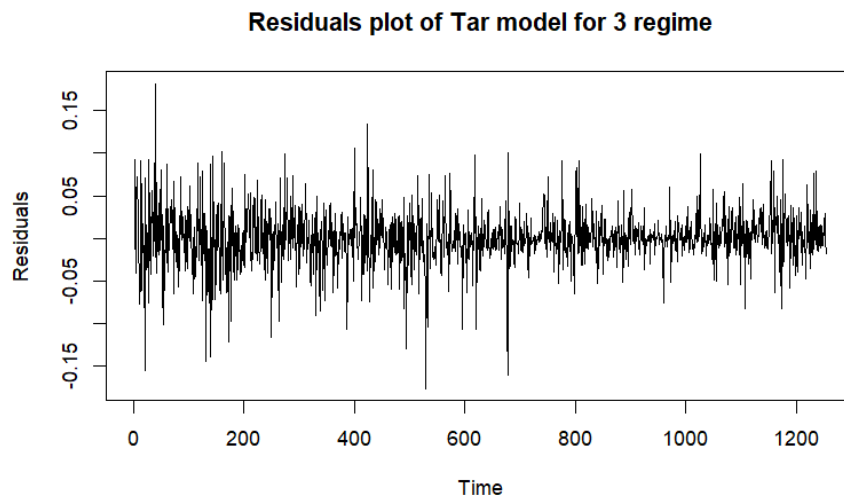


Figure 15: Residuals of 3-Regime TAR Model

The residuals are centred around zero with a relatively constant variance over time, indicating no significant bias and consistent model performance.

- **Fitted model observations:**
 - The graph (Figure 14) reveals significant volatility in Bitcoin prices, with frequent regime switching, particularly around threshold values.

- Bitcoin prices remained low and volatile in late 2022 and early 2023, showing high volatility across all regimes in 2024, with high regimes trending upwards, low regimes trending downwards or being volatile, and the mid regime serving as a transitional phase.

4 Results

4.1 Forecasting using ARIMA model:

We will now forecast the future value using the fitted ARIMA model.

```
1 library(tsDyn)
2 predictions <- forecast(arima_model, h = num)
3 point_forecast = predictions$mean
4 plot(dframe$Date,(dframe$Bitcoin), col = 'red', lwd = 2, main = "Actual Vs
   Forecast", ylab = "Bitcoin value ($)",xlab="Date", type = "l",ylim=c
   (8,12), xlim = as.Date(c("2021-01-11", "2024-06-23")))
5 lines(df[(nrow(dframe)+1):(nrow(dframe)+num)],]$Date,(point_forecast), col =
   "blue", lwd = 3)
6 lines(df[(nrow(dframe)+1):(nrow(dframe)+num)],]$Date,(df[(nrow(dframe)+1):(
   nrow(dframe)+num)],]$Bitcoin), col = "green", lwd = 3)
7 legend('bottomright', legend = c("Actual", "Forecasted","Rest Data"), col =
   c("red", "blue","green"), lwd = 2)
```

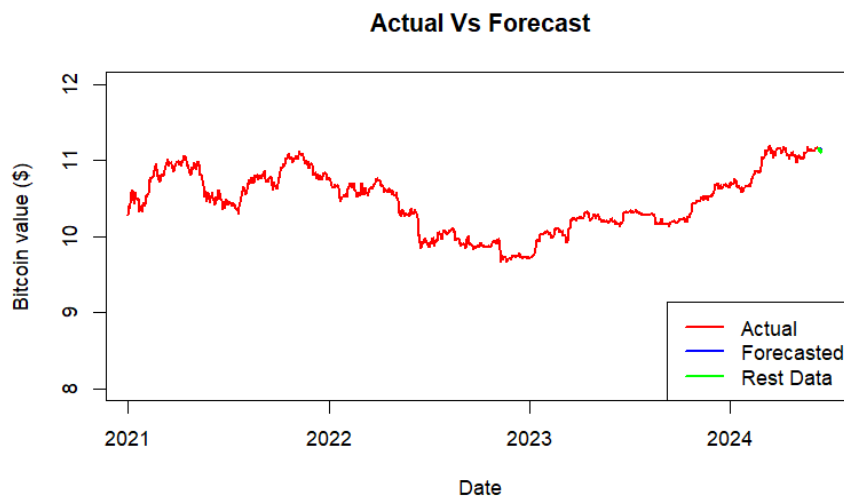


Figure 16: Forecasting Plot

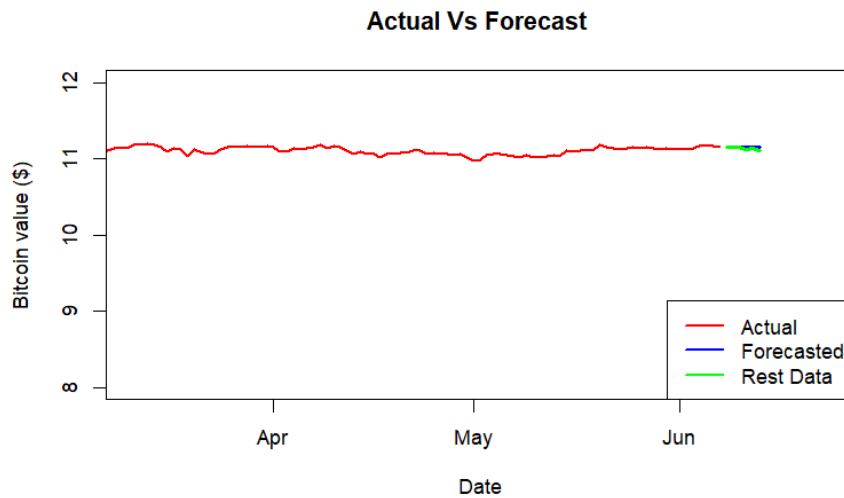


Figure 17: Zoomed-in Forecasting Plot

Observations from the Plots:

- The ARIMA(1, 1, 0) model effectively captures the overall trend and short-term fluctuations of Bitcoin values. For accuracy comparison, we can refer to a zoomed-in plot. However, there is a slight difference between the actual and forecast plots (Figure 17).

4.2 Forecasting using Tar model for regime 3:

Having examined the forecasting plot using the ARIMA model, we will now perform forecasting with the TAR model using three regimes.

```

1 tar_forecast1 <- predict(tar_model1, n.ahead=num)
2 plot(dframe$Date, dframe$Bitcoin, type="l", ylab="Bitcoin in USD", xlab="Date"
3 )
4 lines(df[(nrow(dframe)+1):(nrow(dframe)+num)], $Date, tar_forecast1, col="red",
5       , lwd=2)
6 lines(df[(nrow(dframe)+1):(nrow(dframe)+num)], $Date, (df[(nrow(dframe)+1):(
7       nrow(dframe)+num)], $Bitcoin), col = "green", lwd = 1)

```

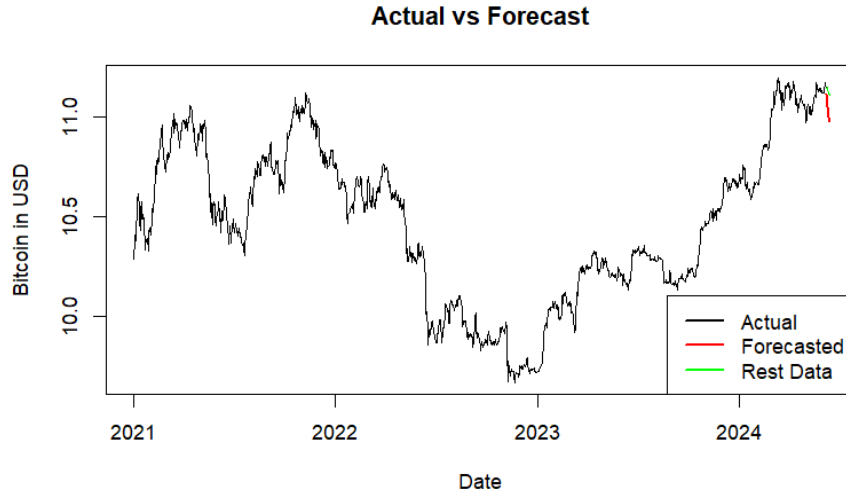


Figure 18: Forecasting Plot

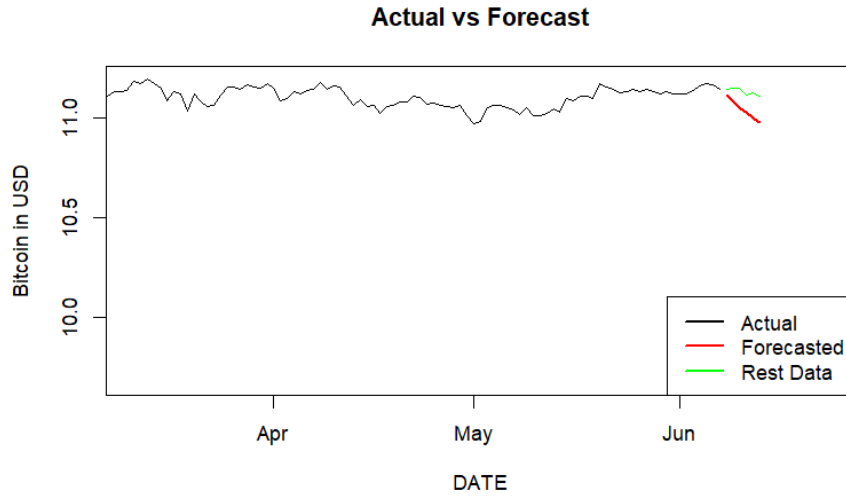


Figure 19: Zoomed-in Forecasting Plot

Observations from the Plots:

- The TAR model with 3 regimes struggles to capture the overall trend and short-term fluctuations of Bitcoin values, as shown by its lower accuracy in the zoomed-in plot (Figure 19).

4.3 Forecasting using Tar model for regime 2:

We have examined forecasts from both the ARIMA model and the TAR model with three regimes. To improve accuracy, we will now forecast using the TAR model with two regimes.

```

1 tar_forecast <- predict(tar_model, n.ahead=num)
2 plot(dframe$Date, dframe$Bitcoin, type="l", ylab="Bitcoin in USD", xlab="Date"
3 )
4 lines(df[(nrow(dframe)+1):(nrow(dframe)+num)], $Date, tar_forecast, col="red",
5       lwd=2)
6 lines(df[(nrow(dframe)+1):(nrow(dframe)+num)], $Date, (df[(nrow(dframe)+1):(
7       nrow(dframe)+num)], $Bitcoin), col = "green", lwd = 1)

```

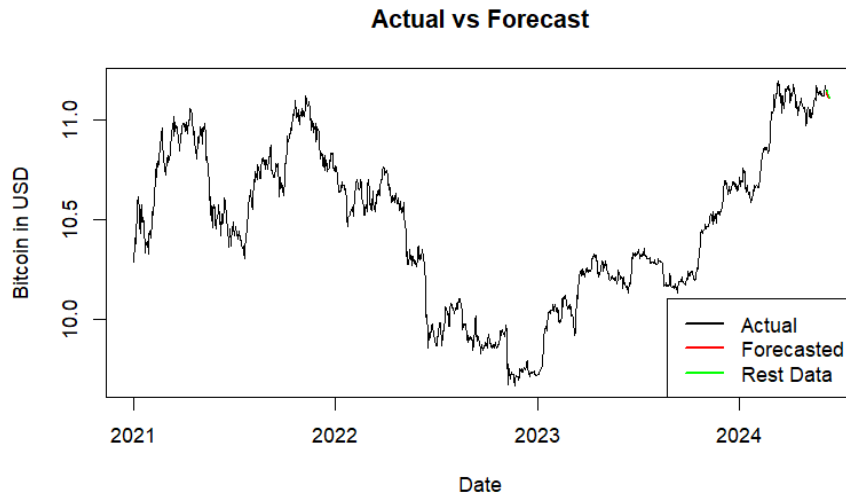


Figure 20: Forecasting Plot

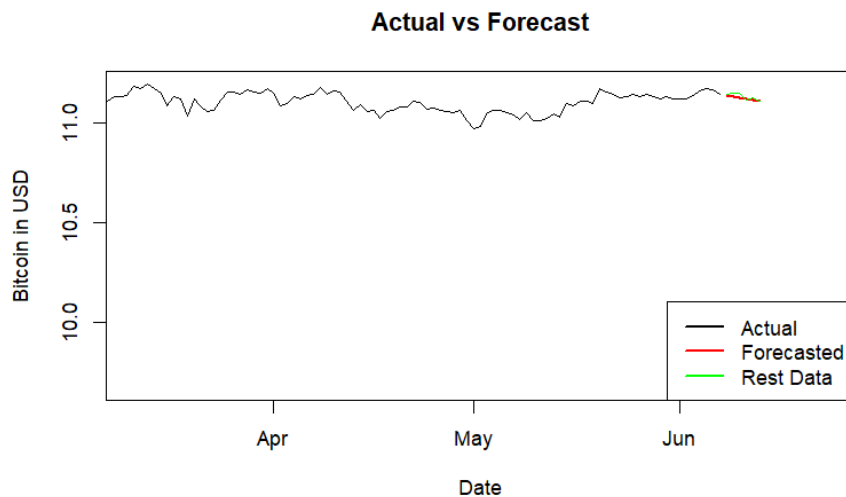


Figure 21: Zoomed-in Forecasting Plot

Observations from the Plots:

- The TAR model with 2 regimes effectively captures the overall trend and short-term fluctuations of Bitcoin values, as shown by the improved accuracy in the zoomed-in plot (Figure 21).

4.4 Model Evaluation

The models were evaluated based on the following metrics:

- **RMSE Calculation:** The RMSE of the models was calculated as follows:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2}$$

where:

- y_i is the observed value at time t ,
- \hat{y}_i is the forecasted value at time t ,
- N is the number of observations.

The RMSE quantifies the average magnitude of forecast errors, with a lower RMSE indicating better model performance.

- **AIC Calculation:** The AIC of the models was calculated using the formula:

$$\text{AIC} = -2 \log L + 2k$$

where:

- L is the maximum likelihood of the model,
- k is the number of estimated parameters.

The AIC balances model fit and complexity, with a lower AIC indicating a better model fit with fewer parameters.

- **Root Mean Square Error (RMSE):**

```
1 cat("ARIMA model :",rmse(arima_model))
2 cat("TAR model for 3 regimes :",rmse(tar_model1))
3 cat("TAR model for 2 regimes :",rmse(tar_model))
```

- ARIMA model : 0.0207
- TAR model for 3 regimes : 0.09506
- TAR model for 2 regimes : 0.0129

- **Akaike Information Criterion (AIC):**

```
1 cat("ARIMA model :",aic(arima_model))
2 cat("TAR model for 3 regimes :",aic(tar_model1))
3 cat("TAR model for 2 regimes :",aic(tar_model))
```

- ARIMA model : -4959.16
- TAR model for 3 regimes : -8519.58
- TAR model for 2 regimes : -8520.53

- **Model Performance:**

Model	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
ARIMA Forecast	11.14674	11.14671	11.14671	11.14671	11.14671	11.14671
TAR (2 Regime) Forecast	11.13979	11.13391	11.12830	11.12297	11.11789	11.11305
TAR (3 Regime) Forecast	11.11431	11.08443	11.05623	11.02962	11.00451	10.98081
Actual Values	11.14629	11.15131	11.14895	11.11716	11.13124	11.10892

Table 3: Forecasted and Actual Values for the Next 6 Days

5 Conclusion

This case study focused on analysing and forecasting the daily closing prices of Bitcoin in U.S. Dollars from December 1, 2014, to June 23, 2024, using various time series models, including ARIMA and Threshold Autoregressive (TAR) models with 2 and 3 regimes.

Key findings from the analysis include:

- **Model Performance:** The 2-regime TAR model exhibited the lowest RMSE (0.0129) and best AIC (-8520.53), outperforming the ARIMA model and the 3-regime TAR model in predictive accuracy and model parsimony.
- **Forecasting Accuracy:** The 2-regime TAR model provided the most robust short-term forecasts, closely matching the actual data for the next 6 days, while the ARIMA model's forecasts remained largely unchanged and less dynamic.
- **Regime Analysis:** The 3-regime TAR model identified high, mid, and low regimes, capturing Bitcoin's nonlinear and volatile behaviour, but the 2-regime TAR model proved more accurate and parsimonious.
- **Nonlinear Dynamics and Volatility:** TAR models effectively captured the nonlinear dynamics and high volatility of Bitcoin prices, which linear models like ARIMA failed to do.
- **Model Selection and Complexity:** The study highlighted the importance of selecting appropriate models for financial time series, noting that simpler models like the 2-regime TAR often provide more accurate and reliable forecasts than more complex models.

In conclusion, the 2-regime TAR model emerged as the most effective for forecasting Bitcoin prices, balancing accuracy and simplicity, and underscoring the importance of nonlinear modelling techniques for financial time series data.

References

- [1] Coinbase. Coinbase, 2024. <https://www.coinbase.com/>.
- [2] Coinbase. User agreement - united states, 2024. Accessed: 2024-07-22.
- [3] Federal Reserve Bank of St. Louis. Bitcoin data series, 2024. <https://fred.stlouisfed.org/series/CBBTCUSD>.
- [4] R.J. Hyndman and Y. Khandakar. Automatic time series for forecasting: The forecast package for r. Technical report, Monash University, Department of Econometrics and Business Statistics, 2007.
- [5] Robert H. Shumway and David S. Stoffer. *Time Series Analysis and Its Applications with R Examples*. Springer, 2020.
- [6] Ruey S. Tsay. *Analysis of Financial Time Series*. Wiley, 2010.
- [7] Ruey S. Tsay and Rong Chen. *Nonlinear Time Series Analysis*. Wiley, 2019.
- [8] I. M. Wirawan, T. Widiyaningtyas, and M. M. Hasan. Short term prediction on bitcoin price using arima method. In *Proceedings of the 2019 International Seminar on Application for Technology of Information and Communication (iSemantic)*, pages 260–265. IEEE, 2019.