

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
TECHNICAL UNIVERSITY OF CRETE



DIPLOMA THESIS

NEXT-DAY BITCOIN PRICE FORECASTING
USING TIME SERIES MODELS

AUTHOR: CHRISTINA-DIONYSIA PANOU

THESIS COMMITTEE: PROF. DIONISSIOS HRISTOPULOS
PROF. MICHALIS ZERVAKIS
DR. SOPHIA TSAKIRIDOU

*A thesis submitted in fulfilment of the requirements
for the diploma of Electrical and Computer Engineering*

MARCH, 2024

Abstract

In an era of a thriving cryptocurrency market that disrupts the traditional economic system, new opportunities for capitalization emerge, accompanied by elevated risks. Managing these risks and optimizing investment decisions is imperative, underscoring the need for dependable tools in cryptocurrency market forecasting. This thesis explores the forecasting capabilities of both the Auto-Regressive Integrated Moving Average (ARIMA) and composite Auto-Regressive Integrated Moving Average-Generalized Auto-Regressive Conditional Heteroscedastic (ARIMA-GARCH) time series models for the daily closing prices of the Bitcoin cryptocurrency. The pronounced presence of heteroscedasticity in the Bitcoin time series data renders the ARIMA models unsuitable for accurate modeling and subsequent forecasting of the data. Conversely, the ARIMA-GARCH(0,1) models effectively deal with heteroscedasticity and demonstrate adequacy in capturing the patterns and structure of the time series. This study is conducted using three distinct test time periods and experiments with various training-test splits to evaluate several ARIMA-GARCH(0,1) models. Additionally, it compares their performance to that of some Recurrent Neural Network (RNN) models available in the literature. Mean Square Error (MSE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE) validation measures are employed for this purpose. Using prominent stock market indices as exogenous variables of ARIMA-GARCH(0,1) models leads to enhanced performance scores for many scenarios, suggesting a possible impact of the stock market on Bitcoin prices. The top-performing candidates among the proposed ARIMA-GARCH(0,1) and ARIMAX-GARCH(0,1) models exhibit similar, and in some cases, superior forecasting performance when compared to the Long Short-Term Memory (LSTM), Bidirectional-Long Short-Term Memory (Bi-LSTM), Gated Recurrent Unit (GRU), and Bidirectional-Gated Recurrent Unit (Bi-GRU) models employed in other research studies.

Περίληψη

Σε μια εποχή όπου η αγορά κρυπτονομισμάτων ακμάζει διαταράσσοντας το παραδοσιακό οικονομικό σύστημα, νέες ευκαιρίες κεφαλαιοποίησης δημιουργούνται, συνοδευόμενες, ωστόσο, από αυξημένους κινδύνους. Η διαχείριση αυτών των κινδύνων, αλλά και η βελτιστοποίηση των επενδυτικών αποφάσεων σε αυτό το νέο πεδίο είναι επιτακτική, υπογραμμίζοντας την ανάγκη για αξιόπιστα εργαλεία πρόβλεψης της αγοράς κρυπτονομισμάτων. Η παρούσα διπλωματική εργασία διερευνά τις δυνατότητες πρόβλεψης τόσο του ολοκληρωμένου αυτοπαλινδρούμενου μοντέλου κινούμενου μέσου (ARIMA) όσο και του σύνθετου (ARIMA-GARCH) ολοκληρωμένου αυτοπαλινδρούμενου μοντέλου κινούμενου μέσου σε συνδυασμό με μεταβλητότητα που ακολουθεί το μοντέλο γενικευμένης αυτοπαλινδρούμενης δεσμευμένης ετεροσκεδαστικότητας για τις ημερήσιες τιμές κλεισίματος του κρυπτονομίσματος Bitcoin. Η έντονη παρουσία ετεροσκεδαστικότητας στα δεδομένα της χρονοσειράς Bitcoin καθιστά τα μοντέλα ARIMA ακατάλληλα για ακριβή μοντελοποίηση και επακόλουθη πρόβλεψη των δεδομένων. Από την άλλη, τα σύνθετα μοντέλα ARIMA-GARCH(0,1) αντιμετωπίζουν αποτελεσματικά την ετεροσκεδαστικότητα και επιδεικνύουν επάρκεια στην αποτύπωση των μοτίβων και της δομής της χρονοσειράς. Η μελέτη διεξάγεται χρησιμοποιώντας τρεις διακριτές χρονικές περιόδους δοκιμών και πειραματίζεται με διάφορες αναλογίες δεδομένων εκπαίδευσης και δεδομένων επικύρωσης για την αξιολόγηση πολλών διαφορετικών μοντέλων ARIMA-GARCH(0,1). Επιπλέον, συγκρίνει την απόδοσή τους με εκείνη ορισμένων μοντέλων αναδρομικών νευρωνικών δικτύων (RNN) της βιβλιογραφίας. Για το σκοπό αυτό χρησιμοποιούνται οι μέθοδοι μέτρησης του μέσου τετραγωνικού σφάλματος (MSE), της ρίζας του μέσου τετραγώνου σφάλματος (RMSE) και του μέσου απόλυτου ποσοστού σφάλματος (MAPE). Η χρήση σημαντικών δεικτών της χρηματιστηριακής αγοράς ως εξωγενών μεταβλητών σε μοντέλα ARIMA-GARCH(0,1) βελτιώνει την απόδοση σε πολλά σενάρια, υποδηλώνοντας την πιθανή επίδραση της χρηματιστηριακής αγοράς στις τιμές του Bitcoin. Ανάμεσα στα υπό εξέταση ARIMA-GARCH(0,1) και ARIMAX-GARCH(0,1) μοντέλα, εκείνα με τις καλύτερες επιδόσεις παρουσιάζουν παρόμοια, και σε ορισμένες περιπτώσεις, ανώτερη απόδοση πρόβλεψης σε σύγκριση με τα μοντέλα μακράς βραχύχρονης μνήμης (LSTM), αμφίδρομης μακράς βραχύχρονης μνήμης (Bi-LSTM), ανατροφοδοτούμενης μονάδας με πύλες ελέγχου (GRU) και αμφίδρομης ανατροφοδοτούμενης μονάδας με πύλες ελέγχου (Bi-GRU) από άλλες ερευνητικές μελέτες.

Acknowledgements

I would like to express my sincere gratitude to all those who contributed to the successful completion of this thesis. First and foremost, I am deeply thankful to my thesis supervisor, Prof. Dionissios Hristopoulos, for his trust, continuous support, and expert guidance throughout the research and writing process. My thanks also go to Dr. Sophia Tsakiridou for her support and availability over the past months. Her contribution was substantial. Special thanks to Prof. Michalis Zervakis for being a member of the thesis committee and for assessing my work. Last but not least, I wish to thank my family for their unwavering support, understanding, and boundless love during the ups and downs of this academic journey. This thesis would not have been possible without the appreciation and collective support of all those mentioned above. I am thankful for everything.

Christina-Dionysia Panou

March, 2024

Contents

Abstract	I
Περίληψη	II
Acknowledgements	III
List of Figures	VII
List of Tables	IX
List of Acronyms	XI
1 Introduction	1
1.1 Motivation & objectives	2
1.2 Related work	3
1.3 Outline	4
2 Theoretical Background	7
2.1 Preliminary concepts	7
2.1.1 Time series & stochastic processes	7
2.1.2 Some characteristic time series	8
2.1.3 Decomposition of time series	8
2.2 Stationarity	11
2.2.1 Stationarity types	11
2.2.2 Augmented Dickey-Fuller test	12
2.2.3 Kwiatkowski-Phillips-Schmidt-Shin test	13
2.2.4 Phillips-Perron test	14
2.3 Transformations	15
2.3.1 Linear transformations	15
2.3.2 Non-linear transformations	16
3 Forecasting Methodology	19
3.1 Conditional mean models	19
3.1.1 Autoregressive model	19
3.1.2 Moving average model	20
3.1.3 Invertibility	21
3.1.4 Autoregressive moving average model	21
3.1.5 Autoregressive moving average with exogenous variables model	21

3.1.6	Autoregressive integrated moving average model	22
3.1.7	Homoscedasticity & heteroscedasticity	23
3.2	Conditional variance models	23
3.2.1	Autoregressive conditional heteroskedasticity model	24
3.2.2	Generalized autoregressive conditional heteroskedasticity model	24
3.2.3	Autoregressive conditional heteroscedastic test	25
3.3	Conditional mean & conditional variance model	25
3.4	Model selection criteria	26
3.4.1	Maximum likelihood estimation	26
3.4.2	Akaike information criterion	28
3.4.3	Bayesian information criterion	28
3.4.4	Anderson-Darling test	28
3.5	Time series forecasting	29
3.5.1	One-step ahead prediction	30
3.5.2	Prediction intervals	33
3.5.3	Evaluation metrics	34
4	Data Analysis & Results	35
4.1	Data collection & cleaning	35
4.2	Exploratory data analysis	38
4.3	Data splitting	42
4.4	Data transformation	42
4.5	Model estimation & selection	49
4.6	One-step ahead forecasting results	53
5	Conclusion & Future Work	59
	Bibliography	62

List of Figures

Figure 1.1	Bitcoin logo, made in 2010 by Satoshi Nakamoto, that portrays the cryptocurrency as a golden token (Wikipedia).	1
Figure 1.2	Total market capitalization of Bitcoin compared to the total market capitalization of all cryptocurrencies combined (CoinMarketCap).	1
Figure 4.1	Overview of the proposed forecasting methodology.	35
Figure 4.2	The Bitcoin dataset as obtained from Kaggle in CSV format.	36
Figure 4.3	The S&P500 index dataset as obtained from MarketWatch in CSV format.	37
Figure 4.4	The NASDAQ index dataset as obtained from Nasdaq in CSV format.	37
Figure 4.5	The DJIA index dataset as obtained from FRED in CSV format.	38
Figure 4.6	The Bitcoin time series.	39
Figure 4.7	Additive decomposition of Bitcoin time series.	40
Figure 4.8	ACF and PACF plots of Bitcoin time series.	40
Figure 4.9	The training dataset from 28 November 2014 to 26 June 2018.	44
Figure 4.10	Logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.	45
Figure 4.11	First differences of the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.	46
Figure 4.12	Box-Cox transformation of the training data.	47
Figure 4.13	First differences of the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.	48
Figure 4.14	ACF and PACF plots of the first differences of the Box-Cox transformation of the training data.	50
Figure 4.15	ACF and PACF plots of the residuals.	52
Figure 4.16	Histogram and fitted t-distribution of the residuals.	52
Figure 4.17	One-step ahead forecast of Bitcoin daily closing prices from 27 June 2018 to 27 June 2019 using the ARIMA(3,1,1)-ARCH(1) model with an 80:20 ratio of training to test data split.	55
Figure 4.18	One-step ahead forecast of Bitcoin daily closing prices from 26 February 2020 to 26 February 2021 using the ARIMA(1,1,7)-ARCH(1) model with an 80:20 ratio of training to test data split.	56
Figure 4.19	One-step ahead forecast of Bitcoin daily closing prices from 5 February 2018 to 5 June 2019 using the ARIMA(7,1,7)-ARCH(1) model with an 70:30 ratio of training to test data split.	57

List of Tables

2.1	Stationarity results from applying both ADF and KPSS tests in a time series.	14
2.2	Box-Cox transformation parameters λ and their associated transformations for a time series $\{x_t\}$	17
4.1	ADF test results for the Bitcoin time series.	41
4.2	KPSS test results for the Bitcoin time series.	41
4.3	PP test results for the Bitcoin time series.	41
4.4	ADF test results for the training data from 28 November 2014 to 26 June 2018.	44
4.5	KPSS test results for the training data from 28 November 2014 to 26 June 2018.	44
4.6	PP test results for the training data from 28 November 2014 to 26 June 2018.	44
4.7	ADF test results for the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.	45
4.8	KPSS test results for the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.	45
4.9	PP test results for the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.	45
4.10	ADF test results for the the first differences of the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.	46
4.11	KPSS test results for the the first differences of the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.	46
4.12	PP test results for the the first differences of the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.	46
4.13	ADF test results for the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.	47
4.14	KPSS test results for the the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.	47
4.15	PP test results for the the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.	47
4.16	ADF test results for the first differences of the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.	48
4.17	KPSS test results for the first differences of the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.	48
4.18	PP test results for the first differences of the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.	48

4.19 Characteristics for the autocorrelation functions. 49

4.20 MSE, RMSE, MAPE, and coverage probability results of one-step ahead
forecast of Bitcoin daily closing prices from 27 June 2018 to 27 June 2019. 55

4.21 MSE, RMSE, MAPE, and coverage probability results of one-step ahead
forecast of Bitcoin daily closing prices from 26 February 2020 to 26 Febru-
ary 2021. 56

4.22 MSE, RMSE, MAPE, and coverage probability results of one-step ahead
forecast of Bitcoin daily closing prices from 5 February 2018 to 5 June 2019. 57

List of Acronyms

ACF	Auto-Correlation Function
ADF	Augmented Dickey-Fuller
AIC	Akaike Information Criterion
AR	Auto-Regressive
ARCH	Auto-Regressive Conditional Heteroscedastic
ARIMA	Auto-Regressive Integrated Moving Average
ARMA	Auto-Regressive Moving Average
ARMAX	Auto-Regressive Moving Average with eXogenous covariates
BIC	Bayesian Information Criterion
BTC	Bitcoin
DL	Deep Learning
EDA	Exploratory Data Analysis
GARCH	Generalized Auto-Regressive Conditional Heteroscedastic
GRU	Gated Recurrent Unit
KPSS	Kwiatkowski-Phillips-Schmidt-Shin
LSTM	Long Short-Term Memory
MA	Moving Average
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MSE	Mean Square Error
PACF	Partial Auto-Correlation Function
PP	Phillips–Perron
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SSA	Singular Spectrum Analysis
SVD	Singular Value Decomposition

Introduction

While not the first cryptocurrency, Bitcoin (BTC) is the oldest surviving one. Invented in 2008 by an anonymous developer or group of developers using the pseudonym Satoshi Nakamoto, the currency was introduced to the public in 2009 when its implementation was released as open-source software. Cryptocurrencies were initially met with skepticism and disapproval, but they have experienced a widespread market adoption over the years. Today, the global cryptocurrency market capitalization is over \$1 trillion, while the number of different cryptocurrencies exceeds 20,000 according to the UK's Financial Conduct Authority. Bitcoin, with a dominance rate of about 50%, remains by far the largest, most influential, and best-known cryptocurrency.



FIGURE 1.1: Bitcoin logo, made in 2010 by Satoshi Nakamoto, that portrays the cryptocurrency as a golden token ([Wikipedia](#)).

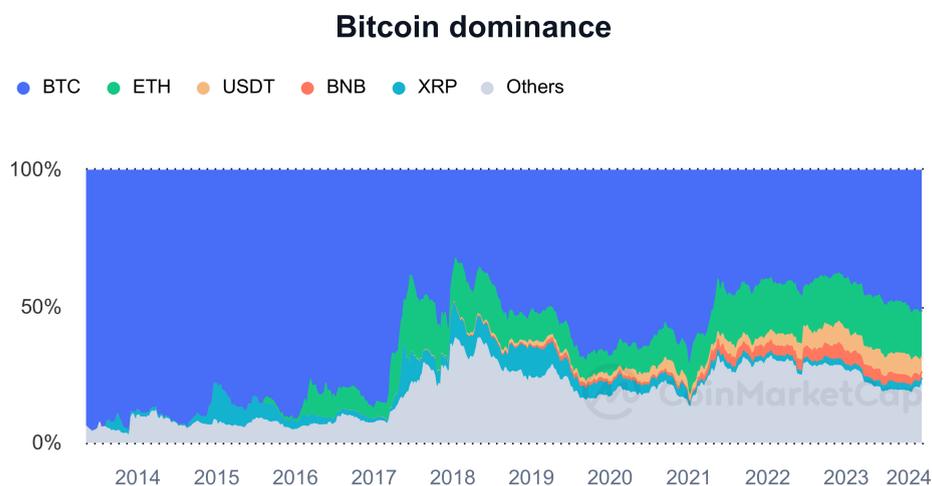


FIGURE 1.2: Total market capitalization of Bitcoin compared to the total market capitalization of all cryptocurrencies combined ([CoinMarketCap](#)).

Cryptocurrencies operate in a unique and decentralized ecosystem, which sets them apart from traditional forms of currency and financial systems. Due to the absence of a central governing authority, their global character, as well as the fact that cryptocurrency markets are open 24/7, cryptocurrencies have a highly volatile nature. Their price is influenced by a myriad of factors, including events and news from various time zones, market sentiment, regulatory developments, macroeconomic trends, and technological advancements.

With the cryptocurrency market skyrocketing in value and social media fueling its popularity, Bitcoin has evolved into a store of value and a mainstream investment asset. It attracts significant attention from a diverse range of stakeholders, including traders, institutional and individual investors, businesses that accept Bitcoin as a form of payment, financial analysts, policymakers and regulatory bodies, academic researchers, as well as individuals who perform Bitcoin transactions with the prospect of lucrative returns. Their shared aspiration to manage risk and optimize investment decisions to capitalize on emerging opportunities in the cryptocurrency market, with a specific emphasis on Bitcoin, drives their need for reliable tools.

1.1 Motivation & objectives

In an attempt to shed light on the challenge of navigating the dynamic and uncertain landscape of cryptocurrencies, this study focuses on predicting the daily closing price of Bitcoin using the composite ARIMA-GARCH time series model. The daily closing price in terms of Bitcoin cryptocurrency refers to the price at which Bitcoin closes at the end of a specific day. As a case study, we chose to explore the efficiency of the ARIMA-GARCH model in predicting the prices of the dominant currency, Bitcoin, in a way that can be generalized for the rest of cryptocurrencies. Despite being a traditional approach in financial forecasting, there have been limited contributions in the literature regarding the forecast of cryptocurrency prices using the ARIMA time series model, as well as the composite ARIMA-GARCH model. Therefore, this study aims to fill a gap in the existing scientific literature, enhancing the knowledge base on the potential capabilities and limitations of ARIMA and ARIMA-GARCH models in forecasting Bitcoin prices. Additionally, it investigates whether incorporating information from prominent financial market indices can enhance our ability to forecast cryptocurrency prices, examining the potential influence of the stock market on the cryptocurrency market. Finally, the study assesses the efficiency of these classical time series models by comparing their prediction performance with Recurrent Neural Network (RNN) models based on results available in the literature.

1.2 Related work

Numerous research works have been done on cryptocurrency prediction. Below there is a representative sample of the existing literature that helped us to define key elements of the direction of our research. The first study is towards the price prediction of five popular cryptocurrencies in an attempt of evaluating all the available categories of time series forecasting models. It gave us an overview of the cryptocurrencies prediction task using the one-step ahead forecasting method. The second one is concentrated on Bitcoin price prediction using the ARIMA model examining both one-step and multi-step ahead forecasting. Candidate ARIMA models performance was tested for short term predictions in a horizon from 1 to 7 days ahead. It encouraged us in setting our research target to be the next-day forecasting. The rest three papers were chosen indicatively as a benchmark for the effectiveness of our method as they include both statistical (ARIMA) and Deep Learning (DL) (Long Short-Term Memory (LSTM), Bidirectional-Long Short-Term Memory (Bi-LSTM), Gated Recurrent Unit (GRU), Bidirectional Gated Recurrent Unit (Bi-GRU)) methods for comparison.

Kate Murray et al. [1] provided a comparison of statistical, Machine Learning (ML), Deep Learning, hybrid, and ensemble models for forecasting the daily prices of five popular cryptocurrencies, Bitcoin, Ethereum (ETH), Litecoin (LTC), Monero (XMR) and Ripple (XRP). The data were gathered from the trading platforms of *Binance* and *Investing* in the time frame from 01-06-2017 to 31-05-2022. After the typical pre-processing step of data normalization, a temporal training-test split of 80:20 ratio was initially performed on each dataset. Then they further partitioned the test set into twelve non-overlapping monthly windows in order to implement an one-step ahead incremental monthly-based strategy spanning over one year of data. Using this as their evaluation methodology they compared the models in terms of accuracy and computational time. Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and R^2 were the metrics employed for the accuracy measurement. After the average performance of each model computation across all cryptocurrencies, DL approaches found to be the best. In particular, LSTM was the best-performing model, and its training was less expensive than the other DL models. K-nearest neighbor (KNN) and ARIMA showed a good trade-off between accuracy and computational expense. Finally, the individual LSTM approach outperformed all the ensemble algorithms.

I. M. Wirawan et al. [2] deployed the ARIMA method for one to seven days ahead Bitcoin price prediction. The dataset was obtained from *Coingecko* containing a history of Bitcoin prices from 01-05-2013 to 07-06-2019. The p , d and q parameters of the ARIMA model candidates were determined using a correlogram method analyzed through the plot Auto-Correlation Function (ACF) and Partial Auto-Correlation Function (PACF). Testing was conducted in three data scenarios for the final sixteen ARIMA model candidates comparison. Scenario 1: training data from 01-05-2013 to 31-03-2019 and testing

data from 01-04-2019 to 07-04-2019. Scenario 2: training data from 01-05-2013 to 30-04-2019 and testing data from 01-05-2019 to 07-05-2019. Scenario 3: training data from 01-05-2013 to 31-05-2019 and testing data from 01-06-2019 to 07-06-2019. After the calculation of MAPE for the next until seventh day prediction for each scenario, the results were averaged. The ARIMA (4,1,4) model predicted the price of Bitcoin with the best level of accuracy in the specific scenarios according to the MAPE metric that was 0.87 for the next day prediction and 5.98 for the seventh day prediction on average. The conclusion was that the ARIMA method can perform Bitcoin price prediction for one to seven days ahead with good results. The longer the prediction period, the lower the level of accuracy.

Ferdiansyah et al. [3] used LSTM networks for providing daily predictions of Bitcoin closing prices. Data for a 5-year period from 27-06-2014 to 27-06-2019 were collected from *Yahoo Finance* and divided into training and test sets at a ratio of 80:20. Combining the epoch (10, 100, 1000, 200, 400, 800, 2000, and 5000) with the model dropout (0, 0,1 and 0,5) to minimise the RMSE metric, they found 288.59866 to be the best score.

Ashish Singh et al. [4] implemented four different deep learning approaches, LSTM, Bidirectional LSTM (Bi-LSTM), GRU, and Bidirectional GRU (Bi-GRU) to predict the fluctuating closing price of Bitcoin. The dataset was also taken from *Yahoo Finance* for the period from 01-03-2016 to 26-02-2021. Dataset samples were normalized using a function that maps each feature into the range of 0 to 1 and then split into the ratio of 80:20 for training and testing of the proposed system, respectively. The normalized prediction values of Bitcoin closing price were transformed back to the original scale. The performance of the models was then measured in terms of RMSE, MAE, and R^2 score. The results showed that overall Bi-GRU outperforms the other three models with RMSE equal to 981.18, MAE equal to 540.81 and R^2 score equal to 0.9926.

Peter T. Yamak et al. [5] compared ARIMA, LSTM, and GRU in predicting Bitcoin's closing price. They used daily data from 28-11-2014 to 5-06-2019 gathered from *CryptoDataDownload* and split into training and test sets, with a ratio of 70:30. They transformed the data to fit within the range of 0 to 1 and then proceeded with applying the three different time series forecasting methods. With forecast values inverted back to their original scale, the ARIMA model gave the best accuracy in terms of the RMSE and MAPE metrics with values 302.53 and 2.76 respectively. Also, ARIMA appeared to be the fastest method.

1.3 Outline

The present thesis is structured as follows:

- **Chapter 2 - Theoretical Background** introduces and explains fundamental theoretical concepts of time series analysis.

-
- **Chapter 3 - Forecasting Methodology** provides the theoretical framework behind the time series one-step ahead forecasting methodology applied in this work.
 - **Chapter 4 - Data Analysis & Results** presents the application of the proposed time series forecasting methodology in three different back-testing time-period scenarios. The obtained performance results are then compared with those of other time series forecasting techniques found in the existing literature.
 - **Chapter 5 - Conclusion & Future Work** gives concluding remarks based on the results, along with suggestions for future research directions.

Theoretical Background

2.1 Preliminary concepts

2.1.1 Time series & stochastic processes

A set of observations made sequentially through time is called a *time series* [6]. To represent a time series, one proceeds as follows. The set of time points at which measurements are made is called T . The observation made at time t is indicated by $x(t)$. The set of the observations $\{x(t), t \in T\}$ is called a time series. In regard to the index set T , one may be observing (i) a *discrete* time series $x(t)$ or x_t , in which case T is a finite set of points written $T = \{1, 2, \dots, N\}$, or (ii) a *continuous* time series $x(t)$, in which case T is a finite interval written $T = \{t : 0 \leq t \leq L\}$. Time series can be further categorized into *univariate* and *multivariate*. The former contains records of a single variable, while the latter considers records of more than one variable. Also, a time series is characterized as either *linear* or *non-linear* based on whether the current value of the series is a linear or non-linear function of previous observations.

The basic idea of the statistical theory of analysis of a time series $\{x(t), t \in T\}$ is to interpret the time series as an observation made on a family of random variables $\{X(t), t \in T\}$; that is, for each t in T , $x(t)$ is an observed value of a random variable. A family of random variables $\{X(t), t \in T\}$ is called a *stochastic process*. An observed time series $\{x(t), t \in T\}$ is thus regarded as an observation (or, in different terminology, a *realization*) of a stochastic process $\{X(t), t \in T\}$.

There are two broad categories of statistical problems: problems of stochastic model construction for natural phenomena and problems related to statistical decision-making. These two categories of problems are well illustrated in the analysis of economic time series; some study time series in order to understand the mechanism of the economic system while others study time series with the aim of being able to forecast, for example, stock market prices. In general, it may be said that the objectives of time series analysis are

1. to understand the mechanism generating the time series,
2. to forecast how the time series will behave in the future [7].

For the purposes of this work when we say a time series we will assume a univariate, linear, discrete time series $\{x_t, t \in \mathbb{Z}^+\}$. Similarly, for a stochastic process.

2.1.2 Some characteristic time series

• White noise

A time series $\{w_t, t \in \mathbb{Z}^+\}$ is said to be *white noise* with mean 0 and variance σ_w^2 , written

$$w_t \sim WN(0, \sigma_w^2), \quad (2.1)$$

if w_t are independent and identically distributed with $E[w_t] = 0$ and $Var(w_t) = \sigma_w^2$.

• Random walk

Suppose that $\{w_t, t \in \mathbb{Z}^+\}$ is zero mean white noise with $Var(w_t) = \sigma_w^2$. Define

$$\begin{aligned} x_1 &= w_1 \\ x_2 &= w_1 + w_2 \\ &\vdots \\ x_n &= w_1 + w_2 + \cdots + w_n. \end{aligned}$$

By this definition, note that we can write, for $t > 1$,

$$x_t = x_{t-1} + w_t, \quad (2.2)$$

where $E[w_t] = 0$ and $Var(w_t) = \sigma_w^2$. The time series x_t is called a *random walk*. The mean of x_t is

$$\begin{aligned} \mu_t &= E[x_t] \\ &= E[w_1 + w_2 + \cdots + w_t] \\ &= E[w_1] + E[w_2] + \cdots + E[w_t] = 0 \end{aligned} \quad (2.3)$$

That is, x_t is a zero mean time series.

The variance of x_t is

$$\begin{aligned} Var(x_t) &= Var(w_1 + w_2 + \cdots + w_t) \\ &= Var(w_1) + Var(w_2) + \dots + Var(w_t) = t\sigma_w^2, \end{aligned} \quad (2.4)$$

because $Var(w_1) = Var(w_2) = \cdots = Var(w_t) = \sigma_w^2$ and $Cov(w_t, w_s) = 0$ for all $t \neq s$.

2.1.3 Decomposition of time series

Decomposition in time series analysis provides a systematic approach of understanding the structure and dynamics of the data over time. It is valuable tool for modeling the underlying components of a time series, which typically include:

1. Trend component, that represents the long-term tendency of the time series to increase, decrease, or remain relatively stable over time. It does not have to be linear.

2. Seasonality component, i.e., the predictable patterns or variations that occur at fixed and known intervals, such as days, weeks, or seasons.
3. Cyclical component, which describes the oscillations that are not tied to fixed calendar intervals. Their magnitudes tend to be more variable than those of seasonal patterns.
4. Remainder component, that is the residuals of the time series after all the other components have been removed. It captures the random elements of the data.

Assuming an *additive decomposition*, the time series $\{x_t\}$ can be expressed as

$$x_t = T_t + S_t + C_t + R_t, \quad (2.5)$$

where T_t is the trend component, S_t is the seasonal component, C_t is the cyclical component, and R_t is the remainder component, at time t . As another option, using the *multiplicative decomposition* we have

$$x_t = T_t \times S_t \times C_t \times R_t. \quad (2.6)$$

An alternative to multiplicative decomposition is to initially transform the data until the variation in the time series appears consistent over time, and then employ an additive decomposition [8]. Using a log transformation is essentially equivalent to using multiplicative decomposition. Generally, an additive model is preferred when the seasonal variation is relatively constant throughout the time series, whereas, if the seasonal variation is proportional to the level of the time series, a multiplicative model is considered more suitable.

There are several approaches to time series decomposition. Below we detail the decomposition method applied in this study, specifically, at the exploratory stage of the examined time series.

• Singular spectrum analysis

The Singular Spectrum Analysis (SSA) algorithm according to Golyandina et al. [9] consists of four steps. The first step, called the *embedding step*, transfers the one-dimensional time series $F = (f_0, \dots, f_{N-1})$ of length N into a sequence of L -dimensional vectors $\mathbf{X}_i = (f_{i-1}, \dots, f_{i+L-2})^T$, ($i = 1, \dots, K = N - L + 1$). The single parameter of this delay procedure is the *window length* L , ($1 < L < N$). The K vectors \mathbf{X}_i will form

the columns of the $(L \times K)$ trajectory matrix

$$\mathbf{X} = \begin{bmatrix} f_0 & f_1 & f_2 & \cdots & f_{K-1} \\ f_1 & f_2 & f_3 & \cdots & f_K \\ f_2 & f_3 & f_4 & \cdots & f_{K+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{L-1} & f_L & f_{L+1} & \cdots & f_{N-1} \end{bmatrix}. \quad (2.7)$$

The trajectory matrix \mathbf{X} is a *Hankel matrix* with equal elements on the diagonals $i + j = \text{constant}$.

The second step is the Singular Value Decomposition (SVD) of the trajectory matrix into a sum of rank-one bi-orthogonal element matrices

$$\mathbf{X} = \mathbf{X}_1 + \cdots + \mathbf{X}_L. \quad (2.8)$$

The elementary matrices \mathbf{X}_i are given by $\mathbf{X}_i = s_i \mathbf{U}_i \mathbf{V}_i^T$, where s_i is the i th singular value of \mathbf{X} (equivalent to the square root of the i th eigenvalue of matrix $\mathbf{X} \mathbf{X}^T$), \mathbf{U}_i is the i th left singular vector of \mathbf{X} (equivalent to the i th eigenvector of $\mathbf{X}^T \mathbf{X}$), and \mathbf{V}_i is the i th right singular vector of \mathbf{X} . The collection $(s_i, \mathbf{U}_i, \mathbf{V}_i)$ is called the i th *eigen-triple* of the SVD. These first two steps make up the *decomposition stage* of SSA and the next two steps the *reconstruction stage*.

In the third step, the *grouping step*, the index set $\{1, \dots, L\}$ is partitioned into m disjoint subsets I_1, \dots, I_m , corresponding to splitting the elementary matrices into m groups and summing the matrices within each group. Let $I = \{i_1, \dots, i_p\}$, then the *resultant matrix* \mathbf{X}_I is defined as $\mathbf{X}_I = \mathbf{X}_{i_1} + \cdots + \mathbf{X}_{i_p}$. The resultant matrices are computed for $I = I_1, \dots, I_m$ and by substituting in (2.8) one obtains the new expansion

$$\mathbf{X} = \mathbf{X}_{I_1} + \cdots + \mathbf{X}_{I_m}, \quad (2.9)$$

where the trajectory matrix is represented as a sum of m resultant matrices. The procedure of choosing sets I_1, \dots, I_m is called *eigen-triple grouping*.

The last step transforms each resultant matrix of the *grouped decomposition* (2.9) into a new one-dimensional series of length N , and is called *diagonal averaging*. Let \mathbf{Y} be a $(L \times K)$ matrix with elements y_{ij} , $1 \leq i \leq L$, $1 \leq j \leq K$. Make $L^* = \min(L, K)$, $K^* = \max(L, K)$ and $N = L + K - 1$. Let $y_{ij}^* = y_{ij}$ if $L < K$ and $y_{ij}^* = y_{ji}$ otherwise. Diagonal

averaging transfers matrix \mathbf{Y} to a series g_0, \dots, g_{N-1} by the formula:

$$g_k = \begin{cases} \frac{1}{k+1} \sum_{m=1}^{k+1} y_{m,k-m+2}^*, & 0 \leq k < L^* - 1 \\ \frac{1}{L^*} \sum_{m=1}^{L^*} y_{m,k-m+2}^*, & L^* - 1 \leq k < K^* \\ \frac{1}{N-k} \sum_{m=k-K^*+2}^{N-k+1} y_{m,k-m+2}^*, & K^* \leq k < N. \end{cases} \quad (2.10)$$

Expression (2.10) corresponds to averaging the elements along diagonals $i + j = k + 2$. This diagonal averaging, applied to a resultant matrix \mathbf{X}_{I_k} , produces a N -length time series F_k , and thus the initial series F is decomposed into the sum of m series:

$$F = F_1 + \dots + F_m. \quad (2.11)$$

For proper choices of L and of sets I_1, \dots, I_m , the components F_k can be associated with the trend, oscillation or noise of the original time series F .

2.2 Stationarity

Stationarity is as a crucial concept in the analysis of time series, as the majority of time series models demand the data to be stationary to make accurate and precise predictions. Loosely speaking, a time series is stationary, if its statistical properties, i.e., mean, variance and autocovariance are time-shift invariant.

2.2.1 Stationarity types

The time series $\{x_t, t \in \mathbb{Z}^+\}$ is said to be *strictly stationary* if the joint distribution of

$$x_{t_1}, x_{t_2}, \dots, x_{t_k}$$

is the same as that of

$$x_{t_1-h}, x_{t_2-h}, \dots, x_{t_k-h}$$

for all time points t_1, t_2, \dots, t_k and for all time lags h . In other words, shifting the time origin by an amount h has no effect on the joint distributions. The joint distributions of all orders depend only on the intervals between time instants t_1, t_2, \dots, t_k . If a process is strictly stationary,

1. The mean function μ_t is constant throughout time; i.e., μ_t is free of t .
2. The covariance between any two observations is determined solely by the time lag between them; i.e., the autocovariance function $\gamma_{t,t-h}$ depends only on h and not on t .

In most applications, strict stationarity is a condition that is much too restrictive. Moreover, it is difficult to assess the validity of this assumption in practice. As a result, we employ a less stringent form of stationarity that specifically addresses only the first two moments.

The time series $\{x_t, t \in \mathbb{Z}^+\}$ is said to be *weakly stationary* (or second-order stationary) if

1. The mean function μ_t is constant throughout time; i.e., μ_t is free of t .
2. The covariance between any two observations is defined only by the time lag between them; i.e., the autocovariance function $\gamma_{t,t-h}$ depends only on h and not on t .

It is clear that strict stationarity implies weak stationarity. The converse to statement is not true, in general. However, if we append the additional assumption of univariate normality then the two definitions do coincide.

$$\text{weak stationarity} + \text{univariate normality} \Rightarrow \text{strict stationarity}$$

In the course of this work, the term stationarity will specifically refer to weak stationarity, facilitating a practical and widely applicable framework for time series analysis.

2.2.2 Augmented Dickey-Fuller test

The Augmented Dickey-Fuller (ADF) test is a unit root test that aims to determine whether a time series is stationary or non-stationary [10], [11]. It involves estimating a regression equation with specific hypotheses about the coefficients. The general form of the ADF regression equation is:

$$x_t = c + \delta t + \varphi x_{t-1} + \sum_{j=1}^p \beta_j \Delta x_{t-j} + \varepsilon_t, \quad (2.12)$$

where x_t is the time series variable, p is the number of lagged difference terms so that the error term ε_t is serially uncorrelated, c is the drift coefficient, and δ is the deterministic trend coefficient. Also, the error term is assumed to be homoskedastic (see section Homoscedasticity & heteroscedasticity). The null and alternative hypotheses for the ADF test are as stated below:

- Null hypothesis (H_0): the time series has a unit root, indicating non-stationarity. Mathematically, this is expressed as $H_0 : \varphi = 1$.
- Alternative hypothesis (H_1): the time series does not have a unit root, indicating stationarity. Mathematically, this is expressed as $H_1 : \varphi < 1$.

The *ADF t-statistic* for the φ coefficient is the key factor in the ADF test. It is based on the least squares estimates of (2.12) and is given by

$$ADF_t = t_{\varphi=1} = \frac{\hat{\varphi} - 1}{SE(\hat{\varphi})}, \quad (2.13)$$

where $SE(\hat{\varphi})$ is the standard error of the estimated coefficient. The t-statistic calculates the number of standard deviations by which the estimated coefficient differs from one. If the absolute value of the t-statistic is sufficiently large, greater than a critical value, it suggests that the null hypothesis should be rejected supporting the conclusion that the time series is stationary.

An alternative formulation of the ADF test regression is

$$\Delta x_t = c + \delta t + \pi x_{t-1} + \sum_{j=1}^p \beta_j \Delta x_{t-j} + \varepsilon_t, \quad (2.14)$$

where $\pi = \varphi - 1$. Under the null hypothesis, Δx_t is non-stationary, which implies that $\pi = 0$. The ADF t-statistic is then the usual t-statistic for testing $\pi = 0$. The test regression (2.14) is often used in practice because the ADF t-statistic is the usual t-statistic reported for testing the significance of the coefficient of x_{t-1} .

2.2.3 Kwiatkowski-Phillips-Schmidt-Shin test

The Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test constitutes another statistical test utilized to assess the stationarity of a time series [12]. Unlike the ADF test, which tests for the presence of a unit root, the KPSS test focuses on detecting stationarity around a deterministic trend. The KPSS test uses the structural model

$$\begin{aligned} x_t &= c_t + \delta t + u_t \\ c_t &= c_{t-1} + \varepsilon_t, \end{aligned} \quad (2.15)$$

where x_t is the time series variable, δ is a the trend coefficient, u_t is a stationary error term that may be heteroskedastic and $\varepsilon_t \sim WN(0, \sigma_w^2)$. Notice that μ_t is a pure random walk with innovation variance σ_w^2 . The null and alternative hypotheses for the KPSS test are as stated below:

- Null hypothesis (H_0): $\sigma_w^2 = 0$, which implies that c_t is a constant, suggesting that the time series is trend-stationary.
- Alternative hypothesis (H_1): $\sigma_w^2 > 0$, which introduces the unit root in the random walk c_t , suggesting that the time series is not stationary around a deterministic trend.

The KPSS test is based on the calculation of the *KPSS test statistic*, which serves as the Lagrange multiplier for testing $\sigma_w^2 = 0$ against the alternative that $\sigma_w^2 > 0$ and is defined

as:

$$KPSS = \frac{\sum_{t=1}^T \hat{S}_t^2}{T^2 \hat{\lambda}^2}, \quad (2.16)$$

where T is the length of the time series, $\hat{S}_t = \sum_{j=1}^t \hat{u}_j$, \hat{u}_t is the residual of the regression of x_t on the estimated trend and $\hat{\lambda}^2$ is a consistent estimate of the long-run variance of u_t using \hat{u}_t . If the absolute value of the test statistic is significantly greater than a critical value, the null hypothesis of stationarity around a deterministic trend is rejected.

The KPSS test is often applied in conjunction with the ADF test, as together they ensure that the time series is truly stationary. Table 2.1 below shows the possible outcomes of using the combination of ADF and KPSS tests. Note that for a trend-stationary

ADF test	KPSS test	Tests combined
Non-stationary	Non-stationary	Time series is non-stationary
Non-stationary	Stationary	Time series is trend-stationary
Stationary	Non-stationary	Time series is difference stationary
Stationary	Stationary	Time series is stationary

TABLE 2.1: Stationarity results from applying both ADF and KPSS tests in a time series.

time series, it is necessary to eliminate the trend to achieve stationarity. Then, the detrended series needs to undergo a stationarity check. In the case of a difference stationary time series, differencing is required for the time series data to be rendered stationary. Subsequently, the differenced series should be subjected to a stationarity check.

2.2.4 Phillips–Perron test

The Phillips–Perron (PP) test is another unit root test [13] that differs from the ADF test, primarily in its treatment of serial correlation and heteroskedasticity in the errors. Particularly, while the ADF test utilizes a parametric autoregression to approximate the structure of the errors in the test regression, the PP test disregards any serial correlation in the test regression. The regression equation for the PP test is

$$x_t = c + \delta t + \varphi x_{t-1} + \varepsilon_t, \quad (2.17)$$

where x_t is the time series variable, c is the drift coefficient, δ is the deterministic trend coefficient, and ε_t is a stationary error term that may be heteroskedastic. The null and alternative hypotheses for the PP test are the same as those for the ADF test. The PP test corrects for any serial correlation and heteroskedasticity in the errors ε_t of the test regression by directly modifying the test statistic $t_{\pi=0}$. This modified statistic, denoted

Z_t , is given by

$$Z_t = \left(\frac{\hat{\sigma}^2}{\hat{\lambda}^2} \right)^{1/2} \cdot t_{\pi=0} - \frac{1}{2} \left(\frac{\hat{\lambda}^2 - \hat{\sigma}^2}{\hat{\lambda}^2} \right) \cdot \left(\frac{T \cdot SE(\hat{\pi})}{\hat{\sigma}^2} \right). \quad (2.18)$$

The terms $\hat{\sigma}^2$ and $\hat{\lambda}^2$ are consistent estimates of parameters

$$\sigma^2 = \lim_{T \rightarrow \infty} T^{-1} \sum_{t=1}^T E[\varepsilon_t^2], \quad (2.19)$$

$$\lambda^2 = \lim_{T \rightarrow \infty} \sum_{t=1}^T E[T^{-1} S_T^2], \quad (2.20)$$

where $S_T = \sum_{t=1}^T \varepsilon_t$.

2.3 Transformations

Before modeling the time series data, it is often necessary to preprocess them through various transformations to ensure their suitability for the specific time series modeling. The choice of transformations depends on the characteristics of the data and the requirements of the chosen modeling technique. Visual inspection of the data and statistical tests guide the selection of appropriate transformations. Below, we present the linear and non-linear transformations that we applied in combination on the Bitcoin time series data.

2.3.1 Linear transformations

- **Differencing**

Differencing can be an effective technique for stabilizing the mean of a non-stationary time series, facilitating its transformation into a stationary one.

Consider time series $\{x_t, t \in \mathbb{Z}^+\}$. The *first-order difference* is a time series $\{\Delta x_t\}$ given by

$$\Delta x_t = x_t - x_{t-1}, \quad (2.21)$$

for $t = 1, 2, \dots, n$. In many situations, a nonstationary time series $\{x_t\}$ can be "transformed" into a stationary one by taking first differences. For example, the random walk $x_t = x_{t-1} + w_t$, where $w_t \sim WN(0, \sigma_w^2)$, is not stationary. However, the first-order difference $\Delta x_t = x_t - x_{t-1} = w_t$ is zero mean white noise, which is stationary.

In some cases, first differences might not be sufficient, and additional differencing may be required. The *second-order difference* is a time series $\{\Delta^2 x_t\}$ given by

$$\begin{aligned}\Delta^2 x_t &= \Delta(\Delta x_t) = \Delta x_t - \Delta x_{t-1} \\ &= (x_t - x_{t-1}) - (x_{t-1} - x_{t-2}) \\ &= x_t - 2x_{t-1} + x_{t-2},\end{aligned}\tag{2.22}$$

for $t = 1, 2, \dots, n$. In general, the *dth difference* process $\{\Delta^d x_t\}$ is defined as

$$\Delta^d x_t = \Delta(\Delta^{d-1} x_t) = \Delta^{d-1} x_t - \Delta^{d-1} x_{t-1},\tag{2.23}$$

for $d = 1, 2, \dots$. We take $\Delta^0 x_t = x_t$ by convention.

We should note that while differencing can be a powerful tool, it should be applied judiciously, and the choice of the differencing order depends on the specific characteristics of the data. Excessive differencing can introduce noise and lead to overfitting.

2.3.2 Non-linear transformations

- **Logarithmic transformation**

The *logarithmic transformation* is often employed as an effective way of stabilizing the variance of a time series. Consider the time series $\{x_t\}$ and its logarithmic transformation $x'_t = \log(x_t)$ with variances $Var(x_t) = E[(x_t - \mu)^2]$ and $Var(x'_t) = E[(\log(x_t) - \mu')^2]$ respectively, where μ and μ' are the means of x_t and x'_t . If x_t has a large value, $\log(x_t)$ will be less affected by changes in x_t compared to when x_t has a small value. This compression of larger values reduces the variability in the transformed series. Thus, the variance of the log-transformed series tends to be more stable and less influenced by extreme values compared to the variance of the original scale series.

- **Box-Cox transformation**

The *Box-Cox transformation* is a family of power transformations, including the logarithmic transformation, that is primarily used for stabilizing the variance. It can also have the side effect of bringing the distribution of the data closer to normality. Given a time series $\{x_t\}$, the transformation is defined by

$$x_t^{(\lambda)} = \begin{cases} \frac{x_t^\lambda - 1}{\lambda}, & \lambda \neq 0 \\ \ln(x_t), & \lambda = 0, \end{cases}\tag{2.24}$$

where λ is called the *transformation parameter*. Some common values for λ , and their implied transformations are included in table 2.2. The determination of the value of λ occurs through the Maximum Likelihood Estimation (MLE) method, which aims to find

the λ that maximizes the log-likelihood function for the transformed data or, equivalently, stabilizes the variance as much as possible.

λ	$x_t^{(\lambda)}$	Description
-2.0	$1/x_t^2$	Inverse square
-1.0	$1/x_t$	Reciprocal
-0.5	$1/\sqrt{x_t}$	Inverse square root
0.0	$\ln(x_t)$	Logarithm
0.5	$\sqrt{x_t}$	Square root
1.0	x_t	Identity (no transformation)
2.0	x_t^2	Square

TABLE 2.2: Box-Cox transformation parameters λ and their associated transformations for a time series $\{x_t\}$.

Forecasting Methodology

3.1 Conditional mean models

The *conditional mean* of a time series x_t is the expected value of x_t given a conditioning set, Ω_t . In the context of time series modeling, a *conditional mean model* specifies a functional form for $E[x_t|\Omega_t]$. A conditional mean model of a time series is characterized as either *static* or *dynamic* based on how the expected value of the dependent variable is modeled with respect to conditioning set. For a static conditional mean model, the conditioning set is measured contemporaneously with the dependent variable x_t . On the other hand, a dynamic conditional mean model specifies the expected value of x_t as a function of historical information. Let H_{t-1} denote the history of the time series available at time t that includes

- Past observations, x_1, x_2, \dots, x_{t-1} .
- Vectors of past exogenous variables, z_1, z_2, \dots, z_{t-1} .
- Past innovations, $\varepsilon_1, \varepsilon_2, \dots, \varepsilon_{t-1}$.

A dynamic conditional mean model specifies the evolution of the conditional mean, $E[x_t|H_{t-1}]$. Specifically, according to Wold's decomposition [14], we can write the conditional mean of any stationary time series x_t as

$$E[x_t|H_{t-1}] = \mu + \sum_{i=1}^{\infty} \psi_i \varepsilon_{t-i}, \quad (3.1)$$

where ε_{t-1} are past observations of an uncorrelated innovation series with mean zero, and coefficients ψ_i are absolutely summable. $E[x_t] = \mu$ is the constant unconditional mean of the stationary time series. Any model of the general linear form given by equation (3.1) is a valid specification for the dynamic behavior of a stationary time series. Some special cases of stationary time series models, namely, the AR, MA, ARMA, ARIMA, and ARIMAX models, are presented in detail in the following sections.

3.1.1 Autoregressive model

Auto-Regressive (AR) models are based on the idea that the current value of the series, x_t can be expressed as a linear combination of p previous values, $x_{t-1}, x_{t-2}, \dots, x_{t-p}$, together with a random error in the same series, i.e., x_t is obtained by a linear regression

from $x_{t-1}, x_{t-2}, \dots, x_{t-p}$, hence the name *autoregressive*. More specifically, an autoregressive model of order p , denoted as $AR(p)$, is of the form

$$x_t = \varphi_1 x_{t-1} + \varphi_2 x_{t-2} + \dots + \varphi_p x_{t-p} + \varepsilon_t = \sum_{i=1}^p \varphi_i x_{t-i} + \varepsilon_t, \quad (3.2)$$

where $\varphi_1, \varphi_2, \dots, \varphi_p$ ($\varphi_p \neq 0$) are the parameters of the model, and $\varepsilon_t \sim WN(0, \sigma_w^2)$ are the *innovations* or *error terms* or *random shocks*. The hyperparameter p indicates the length of the "direct look back" in the series.

We introduce the *backshift operator* B , so that $Bx_t = x_{t-1}$, $B^2 x_t = x_{t-2}$ and so on. Then the $AR(p)$ model can be equivalently rewritten using the backshift (or *lag*) operator B as

$$x_t = \sum_{i=1}^p \varphi_i B^i x_t + \varepsilon_t \quad (3.3)$$

so that, shifting the summation term to the left side and utilizing polynomial notation polynomial notation, we have

$$\varphi(B)x_t = \varepsilon_t, \quad (3.4)$$

where $\varphi(B) = 1 - \sum_{j=1}^p \varphi_j B^j$ is the *autoregressive operator*.

The stationarity condition for an $AR(p)$ model is that all the roots of the characteristic polynomial $\varphi(B)$ must lie outside the unit circle. Stationarity ensures that the autocorrelation structure of the time series remains consistent over time, so that there is a unique set of parameters $\{\varphi_1, \varphi_2, \dots, \varphi_p\}$ that characterizes the autocorrelation structure [15].

3.1.2 Moving average model

A Moving Average (MA) model of order q , or $MA(q)$, is defined to be

$$x_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \dots + \theta_q \varepsilon_{t-q} = \sum_{j=0}^q \theta_j \varepsilon_{t-j}, \quad (3.5)$$

where $\theta_0, \theta_1, \theta_2, \dots, \theta_q$ ($\theta_q \neq 0$) are the parameters of the model, we assume $\theta_0 = 1$, and $\varepsilon_t \sim WN(0, \sigma_w^2)$, as in the AR model, represent the innovations or error terms or random shocks. Similarly to the autoregressive operator, we define the *moving average operator* as $\theta(B) = 1 + \sum_{j=1}^q \theta_j B^j$ and therefore the moving average model of order q , abbreviated $MA(q)$, can be rewritten as

$$x_t = \theta(B)\varepsilon_t. \quad (3.6)$$

Regarding stationarity, MA models are inherently stationary as a consequence of being a linear combination of white noise error terms. However, they may not be unique. For a model to be unique for a particular moving average structure an invertibility condition must be imposed [15].

3.1.3 Invertibility

We define a time series $\{x_t\}$ to be *invertible* if it can be expressed as a "mathematically meaningful" autoregressive time series (possibly of infinite order). Stationary autoregressive models are invertible by definition, whereas the MA(q) process

$$x_t = \theta(B)\varepsilon_t = (1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q)\varepsilon_t \quad (3.7)$$

is invertible if and only if the roots of the MA(q) characteristic polynomial $\theta(z) = 1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q$ all fall outside the unit circle.

3.1.4 Autoregressive moving average model

The Auto-Regressive Moving Average (ARMA) model is the natural generalization of the AR and MA models defined as

$$x_t = \sum_{i=1}^p \varphi_i x_{t-i} + \sum_{j=0}^q \theta_j \varepsilon_{t-j}, \quad (3.8)$$

where $\varepsilon_t \sim WN(0, \sigma_w^2)$ and assuming $\theta_0 = 1$. Alternatively, the autoregressive moving average process of orders p and q , written ARMA(p, q), can be expressed using the backshift notation as

$$\varphi(B)x_t = \theta(B)\varepsilon_t, \quad (3.9)$$

where

$$\varphi(B) = 1 - \varphi_1 B - \varphi_2 B^2 - \cdots - \varphi_p B^p, \quad (3.10)$$

$$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q. \quad (3.11)$$

- For the ARMA(p, q) model to be stationary we need the roots of the AR characteristic polynomial $\varphi(B)$ to all lie outside the unit circle.
- For the ARMA(p, q) model to be invertible we need the roots of the MA characteristic polynomial $\theta(B)$ to all lie outside the unit circle [15].

3.1.5 Autoregressive moving average with exogenous variables model

When there are external variables that are believed to impact the time series of interest, an Auto-Regressive Moving Average with exogenous covariates (ARMAX) model can account for these influences. ARMAX models are a class of time series models that extend the ARMA models by incorporating exogenous variables. The general form of the ARMAX(p, q) model that includes r covariates $z_{1,t}, \dots, z_{r,t}$ is:

$$x_t = \sum_{i=1}^p \varphi_i x_{t-i} + \sum_{k=1}^r \beta_k z_{k,t} + \sum_{j=0}^q \theta_j \varepsilon_{t-j}, \quad (3.12)$$

where $\varepsilon_t \sim WN(0, \sigma_w^2)$, $\theta_0 = 1$ and $\beta_1, \beta_2, \dots, \beta_r$ ($\beta_r \neq 0$) are the coefficients of the covariates. The condensed form of the model in lag operator notation is:

$$\varphi(B)x_t = \beta z'_t + \theta(B)\varepsilon_t, \quad (3.13)$$

where vector z'_t holds the values of the r exogenous covariates at time t and β each of their coefficients.

The requirements for stationarity and invertibility in ARMAX models are the same as those for ARMA models. Note that before entering the model, the exogenous covariates must be assessed in terms of stationarity. In case they are not stationary, we have to apply the appropriate transformations.

3.1.6 Autoregressive integrated moving average model

In order to handle processes which are non-stationary we extend the class of ARMA models to include differencing. Doing so, we introduce a far wider class of models, the Auto-Regressive Integrated Moving Average (ARIMA) class.

A time series $\{x_t\}$ is said to follow an ARIMA model if the d th differences $y_t = \Delta^d x_t$ follow a stationary ARMA model. An ARIMA model is characterized by

- p , the order of the autoregressive component,
- d , the number of differences needed to arrive at a stationary ARMA(p, q) model, and,
- q , the order of the moving average component.

Specifically, we have the general relationship

$$x_t \text{ is ARIMA}(p, d, q) \Rightarrow y_t = \Delta^d x_t \text{ is ARMA}(p, q).$$

We have seen that a stationary ARMA(p, q) can be represented as $\varphi(B)x_t = \theta(B)\varepsilon_t$ where $\varepsilon_t \sim WN(0, \sigma_w^2)$. In the case of the ARIMA(p, d, q) class, for $d = 1$,

$$y_t = \Delta x_t = x_t - x_{t-1} = x_t - Bx_t = (1 - B)x_t \quad (3.14)$$

follows an ARMA(p, q) model. Therefore, an ARIMA($p, 1, q$) process can be written as

$$\varphi(B)(1 - B)x_t = \theta(B)\varepsilon_t. \quad (3.15)$$

Similarly, for $d = 2$,

$$\begin{aligned} y_t &= \Delta^2 x_t = x_t - 2x_{t-1} + x_{t-2} \\ &= x_t - 2Bx_t + B^2x_t \\ &= (1 - 2B + B^2)x_t = (1 - B)^2x_t \end{aligned} \quad (3.16)$$

follows an ARMA(p, q) model. Therefore, an ARIMA($p, 2, q$) can be written as

$$\varphi(B)(1 - B)^2x_t = \theta(B)\varepsilon_t. \quad (3.17)$$

In general, an ARIMA(p, d, q) process can be written as

$$\varphi(B)(1 - B)^d x_t = \theta(B)\varepsilon_t. \quad (3.18)$$

3.1.7 Homoscedasticity & heteroscedasticity

In statistics, a sequence (or vector) of random variables is *homoscedastic* when each of its random variables has the same finite variance. This property is alternatively referred to as homogeneity of variance. Conversely, the complementary concept is termed *heteroscedasticity*, also recognized as heterogeneity of variance. Homoskedasticity is an essential assumption in many statistical models, including the family of ARIMA models. The violation of this assumption while using such models can lead to biased and inconsistent estimates of the model parameters, which can affect the accuracy and reliability of the model predictions.

3.2 Conditional variance models

Conditional variance models are used to model the heteroscedasticity observed in time series data. These models address the limitations of assuming constant variance, allowing for a more realistic representation of the dynamics of volatility over time. More specifically, consider the time series

$$x_t = \mu + \varepsilon_t, \quad (3.19)$$

where x_t is the time series variable, μ is the unconditional mean of the time series, and ε_t are the innovations modeled by

$$\varepsilon_t = \sigma_t z_t \quad (3.20)$$

assuming that σ_t is the standard deviation and $z_t \sim WN(0, \sigma_w^2)$. The dynamic evolution of the innovation variance is specified by a conditional variance model

$$\sigma_t^2 = \text{Var}(\varepsilon_t | H_{t-1}), \quad (3.21)$$

where H_{t-1} is the history of the time series. The history includes:

- Past variances, $\sigma_1^2, \sigma_2^2, \dots, \sigma_{t-1}^2$.
- Past innovations, $\varepsilon_1^2, \varepsilon_2^2, \dots, \varepsilon_{t-1}^2$.

Conditional variance models are appropriate for time series that do not exhibit significant autocorrelation, but are serially dependent. The innovation series ε_t is uncorrelated because:

- $E[\varepsilon_t] = 0$, and

- $E[\varepsilon_t \varepsilon_{t-h}] = 0$ for all t and $h \neq 0$.

However, if σ_t^2 depends on σ_{t-1}^2 , for example, then ε_t depends on ε_{t-1} , even though they are uncorrelated. This kind of dependence exhibits itself as autocorrelation in the squared innovation series, ε_t^2 .

3.2.1 Autoregressive conditional heteroskedasticity model

The Auto-Regressive Conditional Heteroscedastic (ARCH) model [16] is a conditional variance model used to capture the heteroskedasticity in time series data. It models the innovation variance at the current time point in a time series as a function of the actual sizes of the previous time periods' innovations. An ARCH(q) model of order q is modeled as follows

$$\begin{aligned}\varepsilon_t &= \sigma_t z_t, \\ \sigma_t^2 &= \alpha_0 + \sum_{i=1}^q \alpha_i \varepsilon_{t-i}^2,\end{aligned}\quad (3.22)$$

where ε_t denotes the innovations, $z_t \sim WN(0, \sigma_w^2)$, and σ_t is the standard deviation characterizing the typical size of the innovations. Also, we assume that $\alpha_0 > 0$, and $\alpha_i \geq 0, i > 0$. The ARCH model is appropriate when the innovation variance in a time series follows an AR model.

3.2.2 Generalized autoregressive conditional heteroskedasticity model

The Generalized Auto-Regressive Conditional Heteroscedastic (GARCH) model [17] is an extension of the ARCH model in case an ARMA model is assumed for the innovation variance. A GARCH(p, q) model includes p GARCH coefficients associated with lagged variances, determining the order of GARCH terms, and q ARCH coefficients associated with lagged squared innovations, determining the order of ARCH terms. It is given by

$$\begin{aligned}\varepsilon_t &= \sigma_t z_t, \\ \sigma_t^2 &= \kappa + \sum_{i=1}^p \gamma_i \sigma_{t-i}^2 + \sum_{j=1}^q \alpha_j \varepsilon_{t-i}^2,\end{aligned}\quad (3.23)$$

where ε_t denotes the innovations, $z_t \sim WN(0, \sigma_w^2)$, and σ_t is the standard deviation characterizing the typical size of the innovations. For stationarity and positivity, the GARCH model has the following constraints:

- $\kappa > 0$,
- $\gamma_i \geq 0, \alpha_j \geq 0$, and
- $\sum_{i=1}^p \gamma_i + \sum_{j=1}^q \alpha_j < 1$.

3.2.3 Autoregressive conditional heteroscedastic test

A time series exhibiting conditional heteroscedasticity, or else, varying conditional variance over time, is said to have ARCH effects. Consider the time series

$$x_t = \mu + \varepsilon_t, \quad (3.24)$$

where x_t is the time series variable, μ is the unconditional mean of the time series, and ε_t are the innovations modeled by

$$\varepsilon_t = \sigma_t z_t \quad (3.25)$$

assuming that σ_t is the standard deviation and $z_t \sim WN(0, \sigma_w^2)$. Engle's ARCH test [16] is a Lagrange multiplier test used to assess the significance of ARCH effects in the regression of squared innovations

$$\varepsilon_t^2 = \alpha_0 + \sum_{i=1}^q \alpha_i \varepsilon_{t-i}^2. \quad (3.26)$$

The null and alternative hypotheses for the ARCH test are as stated below.

- Null hypothesis (H_0): all the coefficients α_i are equal to zero, indicating the absence of ARCH effects.
- Alternative hypothesis (H_1): at least one of the coefficients α_i is not equal to zero, indicating the presence of ARCH effects.

3.3 Conditional mean & conditional variance model

Composite *conditional mean and conditional variance models* are models that simultaneously model both the conditional mean and conditional variance of a time series, enhancing the accuracy of representing the underlying patterns in time series data and, thus, leading to more precise forecasts. Widely employed in time series analysis, particularly within financial econometrics, the ARMA-GARCH model is the most commonly used composite conditional mean and conditional variance model. It is obtained by combining an ARMA(p, q) model for the conditional mean with a GARCH(p', q') model for the conditional variance and has the following form:

$$x_t = \sum_{i=1}^p \varphi_i x_{t-i} + \sum_{j=0}^q \theta_j \varepsilon_{t-j},$$

assuming $\theta_0 = 1$, and

$$\varepsilon_t = \sigma_t z_t,$$

$$\sigma_t^2 = \kappa + \sum_{i=1}^{p'} \gamma_i \sigma_{t-i}^2 + \sum_{j=1}^{q'} \alpha_j \varepsilon_{t-i}^2,$$

where ε_t denotes the innovations, $z_t \sim WN(0, \sigma_w^2)$, and σ_t is the standard deviation characterizing the typical size of the innovations.

3.4 Model selection criteria

In an attempt of comparing several time series models in this work we used the AIC and BIC selection criteria as statistical measures for the most adequate model. First of all, we present the method of MLE that was used to estimate the parameters of the initially partially specified models by their fitting to the observed data. Then, we demonstrate the two most commonly used information criteria, AIC and BIC, both based on MLE, utilized for the final comparison of the fully specified models. Every criterion seeks to strike a balance between the goodness of fit and the complexity of a model, penalizing more complex models and encouraging those that are accurate, but also parsimonious, preventing both *underfitting* and *overfitting*. In addition, we introduce the Anderson-Darling test, which we employed after selecting the best candidate, to either validate or reject the assumption made for the distribution of residuals in the competing models.

- **Underfitting** refers to a situation where a model is too generic and fails to capture the underlying patterns or relationships present in the data.
- **Overfitting** occurs when a model tries to fit the data too closely and ends up memorizing the data patterns, as well as the noise, and becoming overly complex.

3.4.1 Maximum likelihood estimation

The most frequently employed technique for estimation of unknown parameters in time series models is the method of *maximum likelihood*. The *likelihood function* L is a function that describes the joint distribution of the observed data $\{x_1, x_2, \dots, x_n\}$. However, it is viewed as a function of the model parameters with the observed data being fixed. Therefore, when we maximize the likelihood function with respect to the model parameters, we are finding the values of the parameters, i.e., the estimates, that are most consistent with the observed data.

To illustrate how maximum likelihood estimates are obtained, consider the $AR(1)$ model

$$x_t = \varphi x_{t-1} + \varepsilon_t,$$

where $\varepsilon_t \sim WN(0, \sigma_w^2)$. There are two parameters in this model, φ and σ_w^2 . The probability density function (pdf) of ε_t is

$$f(\varepsilon_t) = \frac{1}{\sigma_w \sqrt{2\pi}} \exp(-\varepsilon_t^2 / 2\sigma_w^2). \quad (3.27)$$

Because $\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n$ are all independent, the joint pdf of $\varepsilon_2, \varepsilon_3, \dots, \varepsilon_n$ is given by

$$\begin{aligned} f(\varepsilon_2, \varepsilon_3, \dots, \varepsilon_n) &= \prod_{t=2}^n f(\varepsilon_t) = \prod_{t=2}^n \frac{1}{\sigma_w \sqrt{2\pi}} \exp(-\varepsilon_t^2 / 2\sigma_w^2) \\ &= (2\pi\sigma_w^2)^{-(n-1)/2} \exp\left(-\frac{1}{2\sigma_w^2} \sum_{t=2}^n \varepsilon_t^2\right). \end{aligned} \quad (3.28)$$

To write out the joint pdf of $\mathbf{x} = (x_1, x_2, \dots, x_n)$, we can first perform a multivariate transformation using

$$\begin{aligned} x_2 &= \varphi x_1 + \varepsilon_2 \\ x_3 &= \varphi x_2 + \varepsilon_3 \\ &\vdots \\ x_n &= \varphi x_{n-1} + \varepsilon_n, \end{aligned}$$

with x_1 being fixed. This will give us the (conditional) joint distribution of x_2, x_3, \dots, x_n given x_1 . Applying the laws of conditioning, the joint pdf of \mathbf{x} ; i.e., the likelihood function $L \equiv L(\varphi, \sigma_w^2 | \mathbf{x})$, is given by

$$L = L(\varphi, \sigma_w^2 | \mathbf{x}) = f(x_2, x_3, \dots, x_n | x_1) f(x_1). \quad (3.29)$$

where

$$f(x_2, x_3, \dots, x_n | x_1) = (2\pi\sigma_w^2)^{-(n-1)/2} \exp\left\{-\frac{1}{2\sigma_w^2} \sum_{t=2}^n (x_t - \varphi x_{t-1})^2\right\}, \quad (3.30)$$

$$f(x_1) = \left[\frac{1}{2\pi\sigma_w^2/(1-\varphi^2)}\right]^{1/2} \exp\left[-\frac{x_1^2}{2\sigma_w^2/(1-\varphi^2)}\right]. \quad (3.31)$$

Multiplying these pdfs and simplifying, we get

$$L = L(\varphi, \sigma_w^2 | \mathbf{x}) = (2\pi\sigma_w^2)^{-n/2} (1-\varphi^2)^{1/2} \exp\left[-\frac{S(\varphi)}{2\sigma_w^2}\right], \quad (3.32)$$

where

$$S(\varphi) = (1-\varphi^2)x_1^2 + \sum_{t=2}^n (x_t - \varphi x_{t-1})^2. \quad (3.33)$$

For this AR(1) model, the maximum likelihood estimators (MLEs) of φ and σ_w^2 are the values which maximize $L(\varphi, \sigma_w^2 | \mathbf{x})$.

The approach to finding MLEs in any stationary ARMA(p, q) model is the same as what we have just outlined in the special AR(1) case. The likelihood function L becomes more complex in larger models. That is a matter of no concern, though, as we use the Matlab software to do the estimation.

3.4.2 Akaike information criterion

According to the Akaike Information Criterion (AIC) [18], [19], the most suitable model for a time series $\{x_t\}$ is the one that minimizes

$$\text{AIC} = -2 \ln L + 2k, \quad (3.34)$$

where $\ln L$ is the natural logarithm of the maximized likelihood function, computed under a distributional assumption for $\{x_1, x_2, \dots, x_n\}$, and k is the number of parameters in the model.

- The $-2 \ln L$ term penalizes models that do not fit the data well. The likelihood function L quantifies how well the model explains the observed data. The higher the likelihood, the better the fit.
- The $2k$ term penalizes models with a larger number of parameters (adhering to the Principle of Parsimony). The idea is to prevent overfitting by favoring simpler models. The penalty increases with the number of parameters k .

3.4.3 Bayesian information criterion

According to Bayesian Information Criterion (BIC) [20], the most suitable model for a time series $\{x_t\}$ is the one that minimizes

$$\text{BIC} = -2 \ln L + k \ln n, \quad (3.35)$$

where $\ln L$ is the natural logarithm of the maximized likelihood function, k and n represent the number of parameters in the model and the sample size, respectively.

- Compared to AIC, BIC similarly requires the maximization of a log likelihood function, but additionally offers a harsher penalty for overparameterized models.

3.4.4 Anderson-Darling test

The Anderson-Darling goodness-of-fit test is commonly used to assess whether a data sample comes from a hypothesized distribution [21]. Even if we do not fully specify the distribution parameters, the test estimates any unknown parameters from the data sample. The null and alternative hypotheses for the Anderson-Darling test are as stated below:

- Null hypothesis (H_0): The data follow a specified distribution.
- Alternative hypothesis (H_1): The data do not follow the specified distribution.

The Anderson-Darling test is based on the calculation of the *Anderson-Darling test statistic*, which belongs to the family of quadratic empirical distribution function statistics, that measure the distance between the hypothesized distribution, $F(x)$, and the

Empirical Cumulative Distribution Function (ECDF), $F_n(x)$, as

$$n \int_{-\infty}^{\infty} (F_n(x) - F(x))^2 w(x) dF(x), \quad (3.36)$$

where $w(x)$ is a weight function and n is the number of data points in the sample. The weight function for the Anderson-Darling test is given by

$$w(x) = [F(x)(1 - F(x))]^{-1}, \quad (3.37)$$

which places greater weight on the observations in the tails of the distribution, thus making the test more sensitive to outliers. The Anderson-Darling test statistic is defined as

$$A^2 = -n - \sum_{i=1}^n \frac{2i-1}{n} [\ln(F(x_i)) + \ln(1 - F(x_{n+1-i}))], \quad (3.38)$$

where $\{x_1 < \dots < x_n\}$ are the ordered sample data points and n is the number of data points in the sample. If the absolute value of the Anderson-Darling test statistic is greater than the critical value of the specified theoretical distribution, the null hypothesis is rejected.

3.5 Time series forecasting

We begin with a sample of values up to time t , say $\{x_1, x_2, \dots, x_t\}$. These are our *observed data*. *Forecasting* is the practice of predicting future values of the time series, i.e.,

$$x_{t+1}, x_{t+2}, x_{t+3}, \dots$$

Generally, x_{t+l} is the value of the time series at time $t+l$, where $l \geq 1$. We call t the *forecast origin* and l the *lead time*. The value x_{t+l} is " l -steps ahead" of the most recently observed value x_t .

It is essential to adopt a mathematical criterion for the computation of model forecasts. The chosen criterion relies on the *mean squared error of prediction* [22]

$$MSEP = E\{[x_{t+l} - h(x_1, x_2, \dots, x_t)]^2\}. \quad (3.39)$$

Supposing that we have a sample of observed data $\{x_1, x_2, \dots, x_t\}$ and that we would like to predict x_{t+l} , we choose the function $h(x_1, x_2, \dots, x_t)$ that minimizes MSEP. This function will be our forecasted value of x_{t+l} . The general solution to this minimization problem is

$$h(x_1, x_2, \dots, x_t) = E[x_{t+l} | x_1, x_2, \dots, x_t], \quad (3.40)$$

the *conditional expectation* of x_{t+l} , given the observed data $\{x_1, x_2, \dots, x_t\}$. Using standard notation, we write

$$\hat{x}_t(l) = E[x_{t+l}|x_1, x_2, \dots, x_t]. \quad (3.41)$$

This is called the *minimum mean squared error* (MMSE) [23] forecast. That is, $\hat{x}_t(l)$ is the MMSE forecast of x_{t+l} .

3.5.1 One-step ahead prediction

The present work focuses on *one-step ahead forecasting*. Below we estimate the MMSE forecast when $l = 1$ for the special cases of AR(1) and MA(1) models, as well as for the generalized ARMA(p, q) and ARIMA(p, d, q) models.

• AR(1)

Consider the AR(1) model

$$x_t = \varphi x_{t-1} + \varepsilon_t, \quad (3.42)$$

where $\varepsilon_t \sim WN(0, \sigma_w^2)$. The one-step ahead MMSE forecast of x_{t+1} is

$$\begin{aligned} \hat{x}_t(1) &= E(x_{t+1}|x_1, x_2, \dots, x_t) \\ &= E[\varphi x_t + \varepsilon_{t+1}|x_1, x_2, \dots, x_t] \\ &= E[\varphi x_t|x_1, x_2, \dots, x_t] + E[\varepsilon_{t+1}|x_1, x_2, \dots, x_t]. \end{aligned} \quad (3.43)$$

From the properties of conditional expectation:

- $E[\varphi x_t|x_1, x_2, \dots, x_t] = \varphi x_t$, because φx_t is a function of x_1, x_2, \dots, x_t .
- $E[\varepsilon_{t+1}|x_1, x_2, \dots, x_t] = E[\varepsilon_{t+1}] = 0$, because ε_{t+1} is independent of x_1, x_2, \dots, x_t .

Thus, the MMSE forecast of x_{t+1} is

$$\hat{x}_t(1) = \varphi x_t. \quad (3.44)$$

• MA(1)

Consider the MA(1) model

$$x_t = \varepsilon_t + \theta \varepsilon_{t-1}, \quad (3.45)$$

where $\varepsilon_t \sim WN(0, \sigma_w^2)$. The one-step ahead MMSE forecast of x_{t+1} is

$$\begin{aligned} \hat{x}_t(1) &= E(x_{t+1}|x_1, x_2, \dots, x_t) \\ &= E[\varepsilon_{t+1} + \theta \varepsilon_t|x_1, x_2, \dots, x_t] \\ &= \underbrace{E[\varepsilon_{t+1}|x_1, x_2, \dots, x_t]}_{=E[\varepsilon_{t+1}]=0} + \underbrace{E[\theta \varepsilon_t|x_1, x_2, \dots, x_t]}_{=(*)}. \end{aligned} \quad (3.46)$$

To compute (*) we rewrite the MA(1) model as

$$\varepsilon_t = x_t - \theta\varepsilon_{t-1}. \quad (3.47)$$

Then we write

$$\varepsilon_t = x_t - \theta(x_{t-1} - \theta\varepsilon_{t-2}) = x_t - \theta x_{t-1} + \theta^2\varepsilon_{t-2}. \quad (3.48)$$

Using repeated similar substitution, we take

$$\varepsilon_t = x_t - \theta x_{t-1} + \theta^2 x_{t-2} - \theta^3 x_{t-3} + \dots, \quad (3.49)$$

a weighted (theoretically infinite) linear combination of x_{t-j} for $j = 0, 1, 2, \dots$. That is, ε_t can be expressed as a function of x_1, x_2, \dots, x_t and therefore

$$E[\theta\varepsilon_t | x_1, x_2, \dots, x_t] = \theta\varepsilon_t. \quad (3.50)$$

Hence,

$$\hat{x}_t(1) = \theta\varepsilon_t. \quad (3.51)$$

• ARMA(p, q)

Consider the ARMA(p, q) model

$$x_t = \sum_{i=1}^p \varphi_i x_{t-i} + \sum_{j=0}^q \theta_j \varepsilon_{t-j},$$

where $\varepsilon_t \sim WN(0, \sigma_w^2)$ and assuming $\theta_0 = 1$. The one-step ahead MMSE forecast of x_{t+1} is

$$\begin{aligned} \hat{x}_t(1) &= E[x_{t+1} | x_1, x_2, \dots, x_t] \\ &= E[\sum_{i=1}^p \varphi_i x_{t+1-i} + \sum_{j=0}^q \theta_j \varepsilon_{t+1-j} | x_1, x_2, \dots, x_t] \\ &= \varphi_1 E[x_t | x_1, x_2, \dots, x_t] + \dots + \varphi_p E[x_{t+1-p} | x_1, x_2, \dots, x_t] \\ &\quad + \theta_0 E[\varepsilon_{t+1} | x_1, x_2, \dots, x_t] + \dots + \theta_q E[\varepsilon_{t+1-q} | x_1, x_2, \dots, x_t], \end{aligned} \quad (3.52)$$

where $E[x_{t+1-i} | x_1, x_2, \dots, x_t] = x_{t+1-i}$ for every $1 \leq i \leq p$ and every $p \geq 1$ and

$$E[\varepsilon_{t+1-k} | x_1, x_2, \dots, x_t] = \begin{cases} 0, & \text{for } 1 - k > 0, \\ \varepsilon_{t+1-k}, & \text{for } 1 - k \leq 0, \end{cases} \quad (3.53)$$

for $k = 0, 1, \dots, q$.

• ARMAX(p, q)

Consider the ARMAX(p, q) model

$$x_t = \sum_{i=1}^p \varphi_i x_{t-i} + \sum_{k=1}^r \beta_k z_{t,k} + \sum_{j=0}^q \theta_j \varepsilon_{t-j},$$

where $\varepsilon_t \sim WN(0, \sigma_w^2)$ and $\theta_0 = 1$. Assuming that the exogenous variables $z_{1,t}, \dots, z_{r,t}$ are deterministic and known at each time t , the one-step ahead MMSE forecast of x_{t+1} is

$$\begin{aligned} \hat{x}_t(1) &= E[x_{t+1}|x_1, x_2, \dots, x_t] \\ &= E[\sum_{i=1}^p \varphi_i x_{t+1-i} + \sum_{k=1}^r \beta_k z_{t,k} + \sum_{j=0}^q \theta_j \varepsilon_{t+1-j}|x_1, x_2, \dots, x_t] \\ &= \varphi_1 E[x_t|x_1, x_2, \dots, x_t] + \dots + \varphi_p E[x_{t+1-p}|x_1, x_2, \dots, x_t] \\ &\quad + \beta_1 E[z_{t,1}|x_1, x_2, \dots, x_t] + \dots + \beta_r E[z_{t,r}|x_1, x_2, \dots, x_t] \\ &\quad + \theta_0 E[\varepsilon_{t+1}|x_1, x_2, \dots, x_t] + \dots + \theta_q E[\varepsilon_{t+1-q}|x_1, x_2, \dots, x_t], \end{aligned} \quad (3.54)$$

where $E[x_{t+1-p}|x_1, x_2, \dots, x_t] = x_{t+1-p}$ for every p , $E[z_{t,r}|x_1, x_2, \dots, x_t] = z_{t,r}$ for every r and

$$E[\varepsilon_{t+1-k}|x_1, x_2, \dots, x_t] = \begin{cases} 0, & \text{for } 1 - k > 0, \\ \varepsilon_{t+1-k}, & \text{for } 1 - k \leq 0, \end{cases} \quad (3.55)$$

for $k = 0, 1, \dots, q$.

• ARMA(p, q) - GARCH(p', q')

Consider the composite ARMA(p, q)-GARCH(p', q') model

$$x_t = \sum_{i=1}^p \varphi_i x_{t-i} + \sum_{j=0}^q \theta_j \varepsilon_{t-j},$$

where $\varepsilon_t \sim WN(0, \sigma_w^2)$ and assuming $\theta_0 = 1$,

$$\varepsilon_t = \sigma_t z_t,$$

$$\sigma_t^2 = \kappa + \sum_{i=1}^{p'} \gamma_i \sigma_{t-i}^2 + \sum_{j=1}^{q'} \alpha_j \varepsilon_{t-i}^2,$$

where ε_t denotes the innovations, z_t is a white noise process that follows either the Gaussian or Student's t distribution, and σ_t is the standard deviation characterizing the typical size of the innovations. The one-step ahead MMSE forecast of x_{t+1} is

$$\begin{aligned} \hat{x}_t(1) &= E[x_{t+1}|x_1, x_2, \dots, x_t] \\ &= E[\sum_{i=1}^p \varphi_i x_{t+1-i} + \sum_{j=0}^q \theta_j \varepsilon_{t+1-j}|x_1, x_2, \dots, x_t] \\ &= \varphi_1 E[x_t|x_1, x_2, \dots, x_t] + \dots + \varphi_p E[x_{t+1-p}|x_1, x_2, \dots, x_t] \\ &\quad + \theta_0 E[\varepsilon_{t+1}|x_1, x_2, \dots, x_t] + \dots + \theta_q E[\varepsilon_{t+1-q}|x_1, x_2, \dots, x_t], \end{aligned} \quad (3.56)$$

where $E[x_{t+1-p}|x_1, x_2, \dots, x_t] = x_{t+1-p}$ for every p and

$$E[\varepsilon_{t+1-k}|x_1, x_2, \dots, x_t] = \begin{cases} 0, & \text{for } 1 - k > 0, \\ \sigma_{t+1-k} z_{t+1-k}, & \text{for } 1 - k \leq 0, \end{cases} \quad (3.57)$$

for $k = 0, 1, \dots, q$.

• **ARIMA(p, d, q)**

For invertible ARIMA(p, d, q) models with $d \geq 1$, MMSE forecasts are computed using the same approach as in the stationary case. To show why, suppose that $d = 1$, so we have the following model

$$\varphi(B)(1 - B)x_t = \theta(B)\varepsilon_t,$$

where $(1 - B)x_t = \Delta x_t$ is the series of first differences. Given that

$$\begin{aligned} \varphi(B)(1 - B) &= (1 - \varphi_1 B - \varphi_2 B^2 - \dots - \varphi_p B^p)(1 - B) & (3.58) \\ &= (1 - \varphi_1 B - \varphi_2 B^2 - \dots - \varphi_p B^p) - (B - \varphi_1 B^2 - \varphi_2 B^3 - \dots - \varphi_p B^{p+1}) \\ &= \underbrace{1 - (1 + \varphi_1)B - (\varphi_2 - \varphi_1)B^2 - \dots - (\varphi_p - \varphi_{p-1})B^p + \varphi_p B^{p+1}}_{\varphi^*(B)}. \end{aligned}$$

We can thus rewrite the ARIMA($p, 1, q$) model as

$$\varphi^*(B)x_t = \theta(B)\varepsilon_t, \quad (3.59)$$

a nonstationary ARMA($p + 1, q$) model. We then estimate MMSE forecasts the same way as in the stationary case.

3.5.2 Prediction intervals

A *prediction interval* for a future observation is a range of values that, with a certain level of confidence, is expected to contain the true value of that observation. For a future time series observation x_{t+l} it can be expressed as

$$[\hat{x}_{t+l} - z_c \cdot \text{SE}, \hat{x}_{t+l} + z_c \cdot \text{SE}], \quad (3.60)$$

where \hat{x}_{t+l} is the forecasted value, z_c is the *critical value*, and SE, standing for *standard error*, is the standard deviation of the forecast errors. The critical value z_c represents the number of standard deviations from the mean in the distribution of errors and corresponds to a certain confidence level for the prediction interval. SE accounts for the uncertainty associated with the forecasted values, considering both the uncertainty in estimating model parameters and the inherent randomness in the data.

The width of a prediction interval is influenced by the variability of the data and the desired level of confidence. Increased variability in the data or larger forecast errors leads to wider intervals. Higher confidence levels also result in wider prediction intervals.

3.5.3 Evaluation metrics

To assess the performance and reliability of time series forecasting models various *evaluation metrics* can be employed. Below we present those used in this work.

- **MSE**

The Mean Square Error (MSE) is a commonly used metric in time series analysis. It measures the average of the squared differences between predicted values \hat{x}_t and actual values x_t . Given N observations, we have

$$MSE = \frac{1}{N} \sum_{t=1}^N (x_t - \hat{x}_t)^2 \quad (3.61)$$

- **RMSE**

The Root Mean Squared Error (RMSE) is derived from the MSE and provides a measure of the average magnitude of the errors between predicted values \hat{x}_t and actual values x_t . Given N observations, RMSE is calculated as follows

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N} \sum_{t=1}^N (x_t - \hat{x}_t)^2} \quad (3.62)$$

- **MAPE**

The Mean Absolute Percentage Error (MAPE) is another widely used metric that measures the average percentage difference between predicted values \hat{x}_t and actual values x_t . Given N observations, it is computed as

$$MAPE = \frac{1}{N} \sum_{t=1}^N \left(\frac{|x_t - \hat{x}_t|}{|x_t|} \right) \times 100\% \quad (3.63)$$

Data Analysis & Results

In this chapter, we present in detail the steps that we followed for the next-day Bitcoin closing price using the ARIMA-GARCH time series model. These steps include in chronological order: 1. Data collection, 2. Data cleaning, 3. Exploratory data analysis, 4. Data splitting, 5. Data transformation, 6. Model estimation, 7. Model selection, 8. One-step ahead forecasting, and 9. Model evaluation. Along with the results of the application of the proposed forecasting methodology shown in figure 4.1, we provide the corresponding results of three other research works that employ the forecasting methods of LSTM, Bi-LSTM, GRU, and Bi-GRU towards the same goal. Also, we investigate if using the historical prices of popular stock market indices as exogenous variables of the ARIMA-GARCH model can improve its forecasting performance.

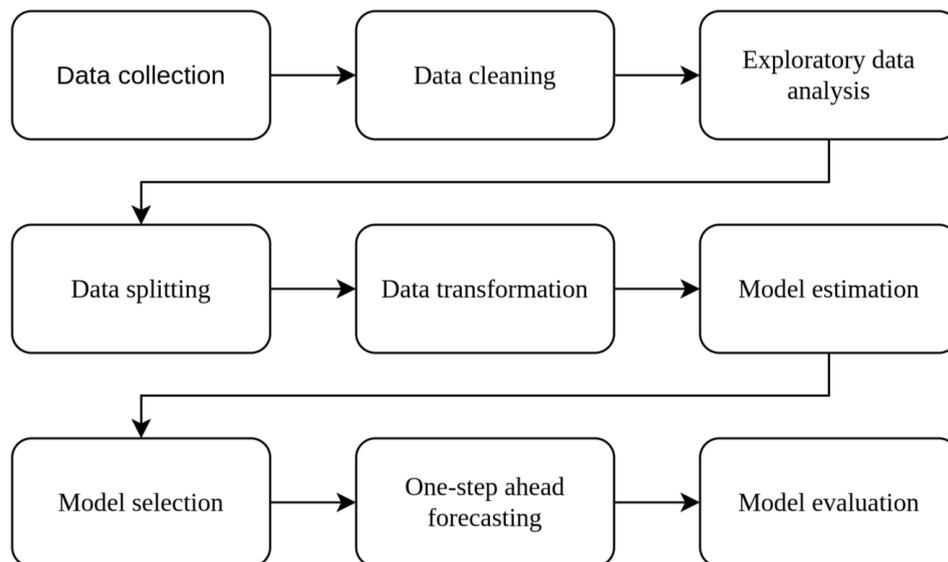


FIGURE 4.1: Overview of the proposed forecasting methodology.

4.1 Data collection & cleaning

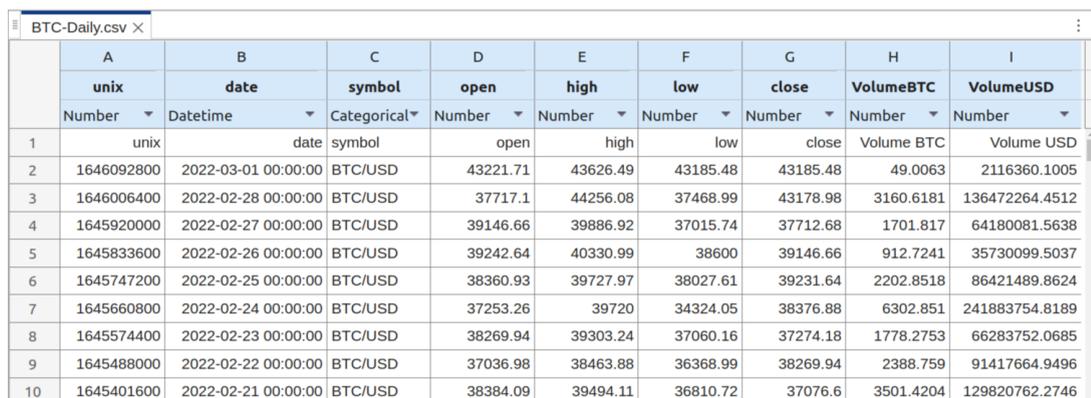
We begin with data collection, followed by data cleaning. Both are critical for ensuring the reliability of the data. Well-collected and carefully prepared data form the basis for the development of accurate predictive time series models.

Regarding data collection, firstly, we have to determine a time period for our data analysis. This could be daily, monthly, quarterly, or any other relevant time interval based on the nature of the data and our research objectives. Then, in order to obtain the time series data, we must locate trustworthy sources and also make sure the data contain all the variables that are essential for our study and the corresponding timestamps. For

the purposes of this work we used the historical data of daily Bitcoin closing price, as well as the historical daily closing prices of the NASDAQ Composite (COMP), Dow Jones Industrial Average (DJIA) and S&P500 (SPX) indices.

The Bitcoin dataset was obtained from [Kaggle](#) in the form of a .csv file and contains a total of 2651 samples. In figure 4.2 we can see a part of the initial dataset. Each sample record has nine (9) fields/features displayed in columns labelled "unix", "date", "symbol", "open", "high", "low", "close", "Volume BTC" and "Volume USD". The data is from November 28, 2014, to March 1, 2022. The dataset features are defined as follows:

1. Unix: Timestamp useful for conversion to the local timezone.
2. Date: The time the data were recorded in UTC timezone.
3. Symbol: The symbol to which the data refer.
4. Open: The the opening price of the date.
5. High: The highest price of the date.
6. Low: The lowest price of the date.
7. Close: The closing price of the data.
8. Volume BTC: The total trade volume on a specific date in BTC.
9. Volume USD: The total trade volume on a specific date in USD.



	A	B	C	D	E	F	G	H	I
	unix	date	symbol	open	high	low	close	VolumeBTC	VolumeUSD
	Number	Datetime	Categorical	Number	Number	Number	Number	Number	Number
1	unix	date	symbol	open	high	low	close	Volume BTC	Volume USD
2	1646092800	2022-03-01 00:00:00	BTC/USD	43221.71	43626.49	43185.48	43185.48	49.0063	2116360.1005
3	1646006400	2022-02-28 00:00:00	BTC/USD	37717.1	44256.08	37468.99	43178.98	3160.6181	136472264.4512
4	1645920000	2022-02-27 00:00:00	BTC/USD	39146.66	39886.92	37015.74	37712.68	1701.817	64180081.5638
5	1645833600	2022-02-26 00:00:00	BTC/USD	39242.64	40330.99	38600	39146.66	912.7241	35730099.5037
6	1645747200	2022-02-25 00:00:00	BTC/USD	38360.93	39727.97	38027.61	39231.64	2202.8518	86421489.8624
7	1645660800	2022-02-24 00:00:00	BTC/USD	37253.26	39720	34324.05	38376.88	6302.851	241883754.8189
8	1645574400	2022-02-23 00:00:00	BTC/USD	38269.94	39303.24	37060.16	37274.18	1778.2753	66283752.0685
9	1645488000	2022-02-22 00:00:00	BTC/USD	37036.98	38463.88	36368.99	38269.94	2388.759	91417664.9496
10	1645401600	2022-02-21 00:00:00	BTC/USD	38384.09	39494.11	36810.72	37076.6	3501.4204	129820762.2746

FIGURE 4.2: The Bitcoin dataset as obtained from [Kaggle](#) in CSV format.

The historical data of the S&P500 index were collected from [MarketWatch](#) in the form of .csv files that we merged together into one .csv file such that it contains 1826 samples of S&P500 data from 28-11-2014 to 01-03-2022. Each sample has five (5) features, "Date", "Open", "High", "Low" and "Close" as shown in figure 4.3.

Regarding the NASDAQ index, its historical data were gathered from [Nasdaq](#) in the form of a .csv file, and consists of 2537 samples for the period from 28-11-2014 until

	A	B	C	D	E
	Date	Open	High	Low	Close
	Datetime	Number	Number	Number	Number
1	Date	Open	High	Low	Close
2	27-Nov-2015	2,088.82	2,093.29	2,084.13	2,090.11
3	25-Nov-2015	2,089.30	2,093.00	2,086.30	2,088.87
4	24-Nov-2015	2,084.42	2,094.12	2,070.29	2,089.14
5	23-Nov-2015	2,089.41	2,095.61	2,081.39	2,086.59
6	20-Nov-2015	2,082.82	2,097.06	2,082.82	2,089.17
7	19-Nov-2015	2,083.70	2,086.74	2,078.76	2,081.24
8	18-Nov-2015	2,051.99	2,085.31	2,051.99	2,083.58
9	17-Nov-2015	2,053.67	2,066.69	2,045.90	2,050.44
10	16-Nov-2015	2,022.08	2,053.22	2,019.39	2,053.19

FIGURE 4.3: The S&P500 index dataset as obtained from [MarketWatch](#) in CSV format.

01-03-2022. A part of the initial dataset is shown in figure 4.4. Each sample has 5 columns consisting of "Date", "Close/Last", "Open", "High" and "Low", that have the same interpretation as the respective features of Bitcoin.

	A	B	C	D	E
	Date	Close_Last	Open	High	Low
	Datetime	Number	Number	Number	Number
1	Date	Close/Last	Open	High	Low
2	11/30/2023	14226.22	14265.05	14289.2	14127.05
3	11/29/2023	14258.49	14367.11	14423.22	14247.89
4	11/28/2023	14281.76	14224.63	14303.65	14195.72
5	11/27/2023	14241.02	14239.31	14305.71	14205.24
6	11/24/2023	14250.85	14238.03	14270.28	14214.13
7	11/22/2023	14265.86	14283.82	14359.61	14226.63
8	11/21/2023	14199.98	14217.23	14237.5	14146.31
9	11/20/2023	14284.53	14134.75	14309.8	14134.2
10	11/17/2023	14125.48	14101.44	14154.32	14063.93

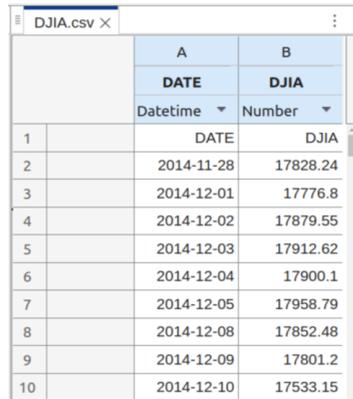
FIGURE 4.4: The NASDAQ index dataset as obtained from [Nasdaq](#) in CSV format.

The historical data of the DJIA index were acquired from [FRED](#) in CSV format, and has a total of 1894 samples from 28-11-2014 to 01-03-2022. The data in figure 4.4 are part of the initial dataset. Each sample has 2 columns, "DATE" and "DJIA", with the later representing the price of the index on the datetime of "DATE".

Data cleaning means, first of all, isolating the information in the data that is relevant to our study and removing what is redundant. Here, we isolated the columns "date" and "close" from the Bitcoin dataset, "Date" and "Close" from the S&P500 dataset, "Date" and "Close/Last" from the NASDAQ dataset, as well as "DATE" and "DJIA" from the DJIA dataset. Then, we parsed these columns from .csv files into variables and used them to create the desired datasets to work with later on.

Another essential part of the data cleaning step is removing any duplicate data points, as it is important to ensure that each timestamp corresponds to a unique and meaningful data point. In our case, none of the datasets had any duplicate date values.

Moreover, data cleaning includes checking for and handling missing values. This might involve *interpolation*, *imputation*, or removal of data points with missing values,



	A	B
	DATE	DJIA
	Datetime	Number
1	DATE	DJIA
2	2014-11-28	17828.24
3	2014-12-01	17776.8
4	2014-12-02	17879.55
5	2014-12-03	17912.62
6	2014-12-04	17900.1
7	2014-12-05	17958.79
8	2014-12-08	17852.48
9	2014-12-09	17801.2
10	2014-12-10	17533.15

FIGURE 4.5: The DJIA index dataset as obtained from [FRED](#) in CSV format.

depending on the nature of the missing values and their impact on the data analysis. Interpolation is used to estimate the values between known data points, creating a continuous representation of the data. Some common interpolation methods include *linear interpolation*, *polynomial interpolation*, *nearest-neighbor interpolation* and *kriging*. We will not get further into the interpolation, as it has not been utilized in this study. Imputation, on the other hand, is a broader term that encompasses various techniques that aim to replace missing values in a dataset with estimated values, ensuring completeness for subsequent analysis. Here we used *forward fill*, which is basically filling missing values with the most recent observed value. This imputation technique is often employed in time series data where values tend to hold over time.

The Bitcoin dataset did not have any missing dates or closing price values, while the NASDAQ, DJIA and S&P500 datasets had several missing dates with their corresponding closing price values due to the closing of the stock market over the weekends and stock market holidays. We had to identify missing dates and create entries with empty closing price values. Then, we appended them to the original dataset and sorted the dataset based on dates. We used forward fill to replace each empty closing price value with the most recent one for the existing dates.

4.2 Exploratory data analysis

After data collection and effective data cleaning we proceed with Exploratory Data Analysis (EDA). EDA is crucial for understanding the underlying patterns, trends, and characteristics of a time series. It provides an initial glimpse of the time series data via a combination of visualizations and statistical tests. These are essential for defining the kind of preprocessing the data need to undergo before the subsequent ARIMA modeling. Specifically, EDA includes:

- Time series decomposition that allows the understanding of the underlying structure.
- The ARCH test for heteroskedasticity.

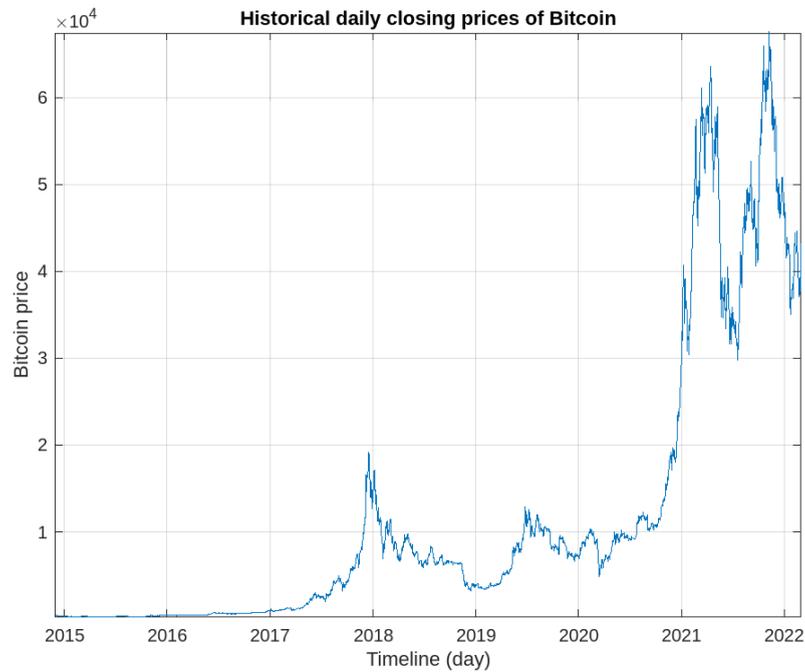


FIGURE 4.6: The Bitcoin time series.

- Employing the ACF and PACF plots to identify autocorrelation in the time series data.
- Stationarity assessment of the time series via the ADF, KPSS, and PP tests.
- Examining the correlation between the subject time series and other time series that could potentially influence it.

So, we began with the decomposition of the Bitcoin time series, shown in figure 4.6, as the first step of EDA. An additive decomposition resulted in a trend proportional to the level of the time series, indicating that a multiplicative model would be more suitable. So, we log-transformed the time series and then used the additive decomposition. From the product property of the logarithm this was equivalent to assuming a multiplicative decomposition of the time series. The components of the decomposition are shown in figure 4.7. The trend component appears to be strong, whereas seasonality is negligible. The ARCH test showed that the time series exhibits heteroskedasticity. The ACF and PACF plots are shown in figure 4.8. A value of 1 in all lags of the ACF plot indicates a strong correlation with previous time points. This suggests a high autocorrelation, which is a typical characteristic of a time series with a trend. The outcomes of the ADF, KPSS, and PP tests are shown in tables 4.1, 4.2 and 4.3, respectively, indicating non-stationarity. Finally, the correlation coefficients between the Bitcoin time series and the COMP, DJIA, and SPX indices time series were 0.91, 0.86, and 0.91, respectively, signifying a robust positive linear relationship in all three cases. Despite the identified correlations, definitive conclusions regarding the causal link between changes in the specific financial indices and Bitcoin cannot be drawn. Various other factors may contribute, and a more in-depth

analysis is required to establish causation, but we will not delve further into investigating this causation.

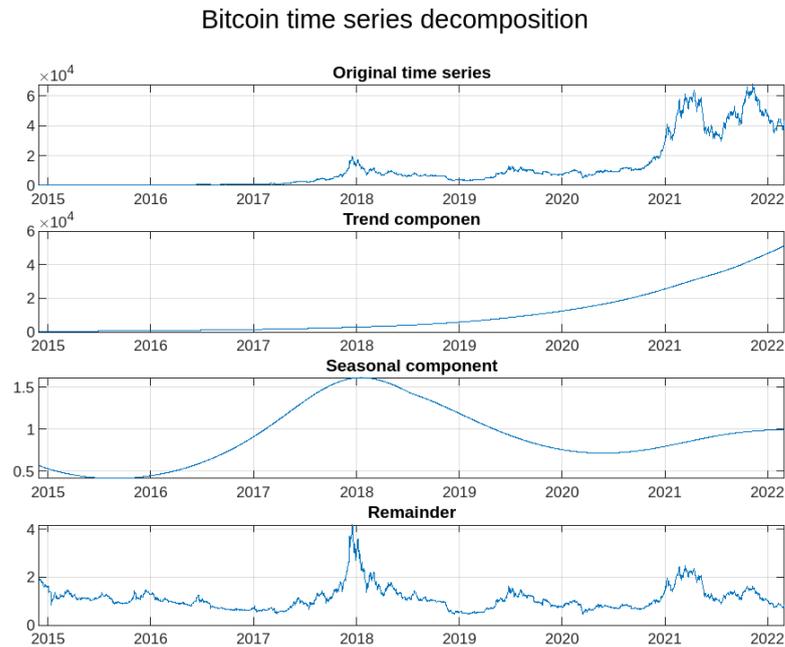


FIGURE 4.7: Additive decomposition of Bitcoin time series.

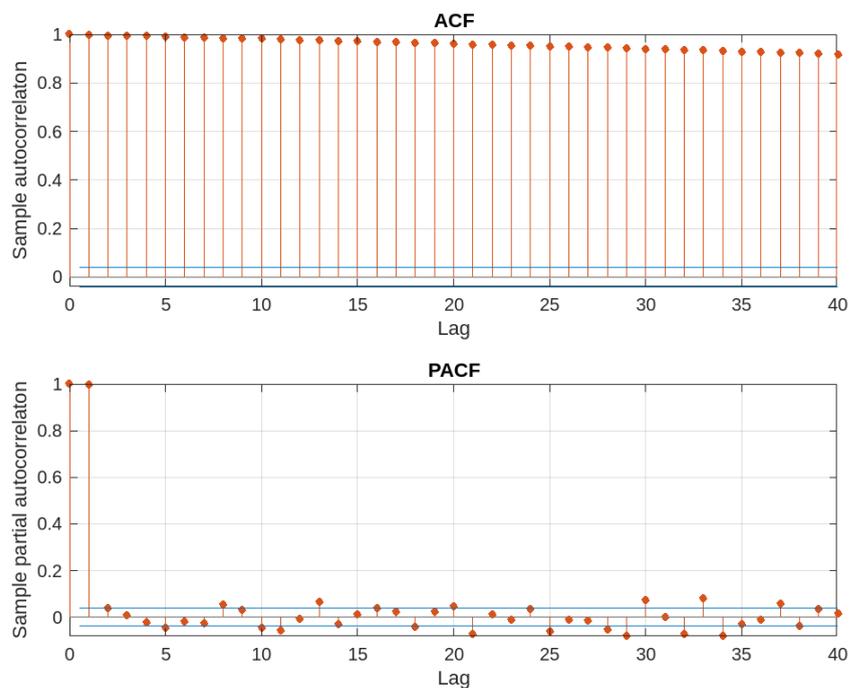


FIGURE 4.8: ACF and PACF plots of Bitcoin time series.

Reject H0	Test statistic	Critical value
true	0.02999	-2.1571

TABLE 4.1: ADF test results for the Bitcoin time series.

Reject H0	Test statistic	Critical value
true	34.641	0.146

TABLE 4.2: KPSS test results for the Bitcoin time series.

Reject H0	Test statistic	Critical value
true	-2.1571	-1.9416

TABLE 4.3: PP test results for the Bitcoin time series.

4.3 Data splitting

Before any preprocessing, we first had to perform a two-part data split into *training* and *test sets*. The training set is the largest subset of the data that is used for model fitting. A common rule of thumb for the size of the training set is to use 60-80% of the data. The training set should be a representative sample of the time series that we aim to model and predict. The test set is the remaining smaller portion of the data that we use to assess the final performance the chosen model or models. It consists of data points unseen by the training process. The size of the test set is correspondingly 20-40% of the data. The test set should reflect the real-world conditions and scenarios that the model will face. Various methods can be utilized to divide data into training and test sets. Here we applied the most fundamental and widely used splitting approach in time series forecasting, *time-based splitting*, that separates the dataset into training and test sets based on a specific point in time. This technique simulates a real-world scenario where the model is trained on historical data and then tested on new future data, and therefore it is crucial for assessing the generalization ability of a forecasting model.

To compare our methodology with each of research works [3], [4], and [5], we opted for three distinct data splits for the Bitcoin dataset based on the intervals used as back-testing periods in the specific pieces of literature. More precisely, the intervals 27 June 2018 - 27 June 2019, 26 February 2020 - 26 February 2021, and 5 February 2018 - 5 June 2019 constituted the back-testing periods for the three research works of interest, respectively. These intervals consequently served as the test sets in the application of our methodology. The remaining portion of the Bitcoin dataset in each application was utilized as the maximum available training set. We conducted experiments with various training set sizes. For the first partition, we used the intervals 28 November 2014 - 26 June 2018, 26 February 2016 - 26 June 2018, and 26 December 2016 - 26 June 2018 as training sets, resulting in training and test set ratios of 80:20, 70:30, and 60:40, respectively. In the second partition, the training sets comprised the intervals 25 February 2016 - 25 February 2020, 25 October 2017 - 25 February 2020, and 25 August 2018 - 25 February 2020, with corresponding training and test set ratios of 80:20, 70:30, and 60:40. For the third partition, the training sets were based on the intervals the intervals 28 November 2014 - 4 February 2018 and 4 February 2016 - 4 February 2018, resulting in training and test set ratios of 70:30 and 60:40, respectively.

4.4 Data transformation

In this step, our goal is to find a suitable transformation or combination of transformations for the training data in order to facilitate accurate ARIMA modeling. ARIMA models rely on the stability of statistical properties over time. In case the time series data are heteroscedastic, this assumption is violated. We then examine the effectiveness of the

logarithmic and Box-Cox transformations in stabilizing the variance. If neither of them results in homoscedastic data, we turn to the GARCH model to address non-constant variance. In such situations, the logarithmic or Box-Cox transformation can be used to stabilize the variance to some extent, and then the GARCH model to further capture and model any remaining time-varying volatility. In case the data have unstable mean, we apply differencing to stabilize it. While ARIMA internally uses differencing to handle non-stationarity, determining the most appropriate order of differencing is crucial for effective modeling. To evaluate homoscedasticity, we utilize the ARCH test. To assess the stationarity of the mean, we employ the ADF, KPSS, and PP statistical tests.

Concerning the first partition of the Bitcoin dataset and the choice of 80:20 as the ratio of the training data to the test data, an initial ARCH test suggested heteroscedasticity. Also, ADF, KPSS and PP stationarity tests indicated non-stationarity, as shown in tables 4.4, 4.5 and 4.6, respectively. PP and KPSS tests are robust to heteroscedasticity, unlike the ADF test, which might be affected by non-constant variance. Next, neither the logarithmic nor the Box-Cox transformation (utilizing the optimal parameter $\lambda = -0.3589$) proved effective in addressing the heteroscedasticity in the data according to the ARCH test. These results implied that a GARCH model would be more suitable for modeling the variance of the time series. Regarding the mean of the time series, after combining the results of the ADF, KPSS, and PP stationarity tests for the first differences of the logarithmic and Box-Cox transformations applied to the training data, we concluded that differencing once is sufficient for achieving a stable mean.

Regarding the choices of 70:30, 60:40 as ratios of the training data to the test data in the same dataset partition, as well as each case of the chosen ratios in the second and third dataset partitions, we obtained similar data transformation results to the above case. The Box-Cox transformation, aimed at stabilizing the variance, and the application of first-order differences, intended for mean stabilization, were the selected transformation candidates for subsequent ARIMA-GARCH modeling.

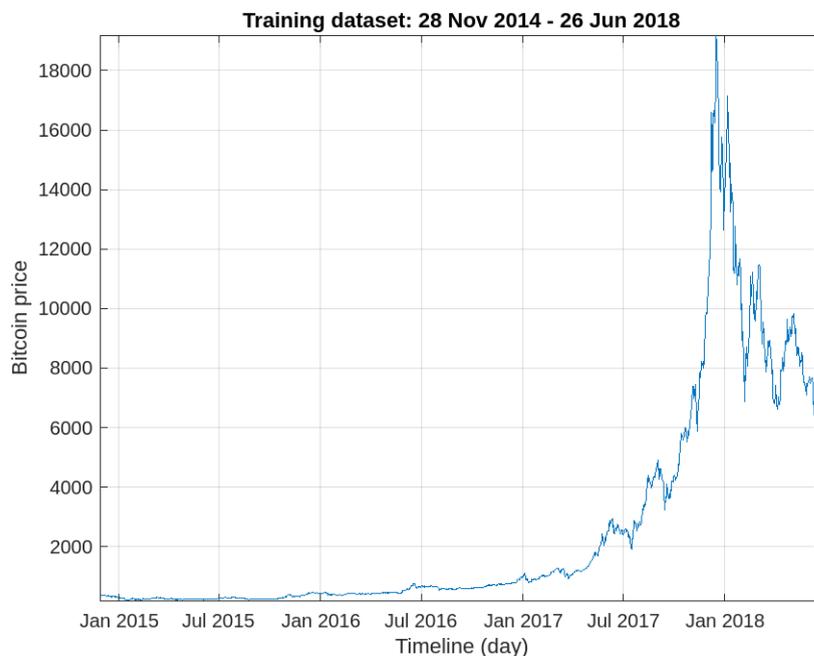


FIGURE 4.9: The training dataset from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
false	0.12496	-1.5017

TABLE 4.4: ADF test results for the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	17.874	0.146

TABLE 4.5: KPSS test results for the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
false	-1.5017	-1.9416

TABLE 4.6: PP test results for the training data from 28 November 2014 to 26 June 2018.

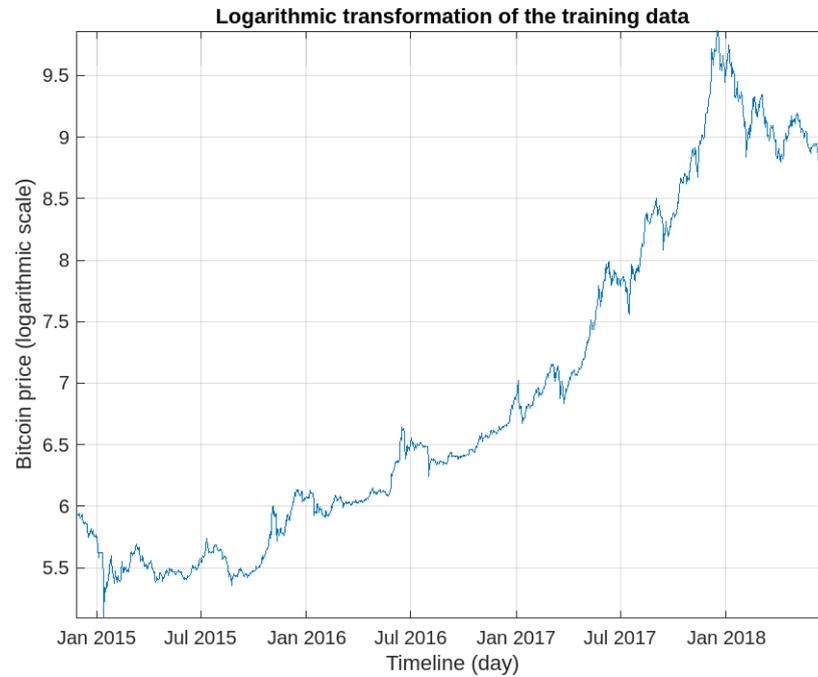


FIGURE 4.10: Logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	0.034626	-2.0985

TABLE 4.7: ADF test results for the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	21.712	0.146

TABLE 4.8: KPSS test results for the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	-2.0985	-1.9416

TABLE 4.9: PP test results for the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.

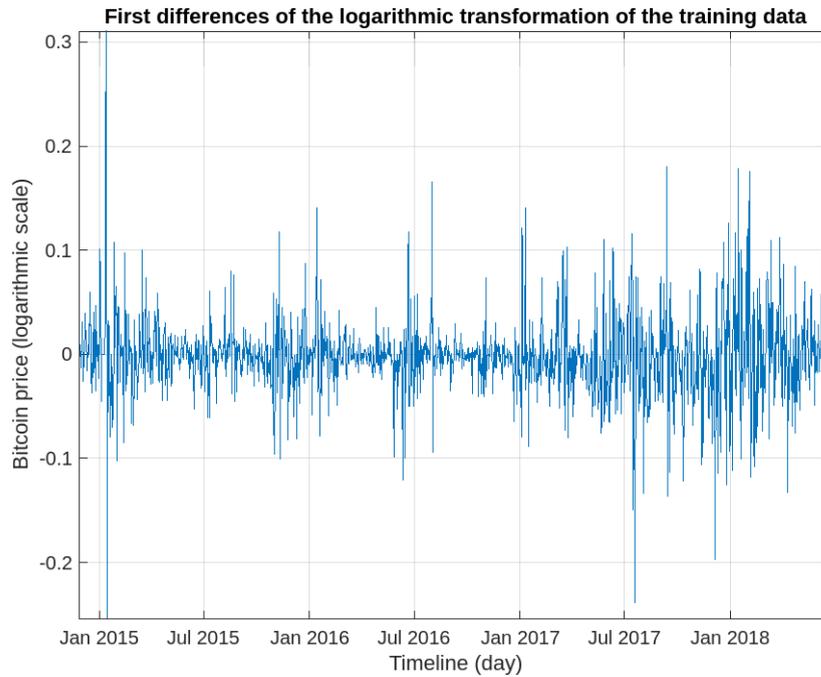


FIGURE 4.11: First differences of the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	0.001	-36.486

TABLE 4.10: ADF test results for the the first differences of the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
false	0.14422	0.146

TABLE 4.11: KPSS test results for the the first differences of the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	-36.486	-1.9416

TABLE 4.12: PP test results for the the first differences of the logarithmic transformation of the training data from 28 November 2014 to 26 June 2018.

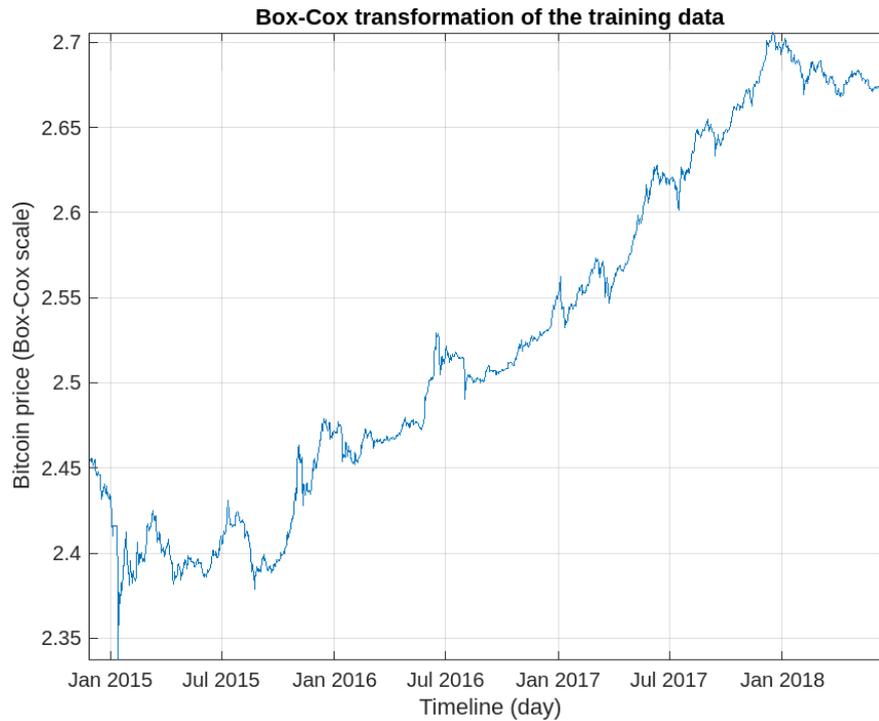


FIGURE 4.12: Box-Cox transformation of the training data.

Reject H0	Test statistic	Critical value
false	0.092613	-1.6552

TABLE 4.13: ADF test results for the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	11.579	0.146

TABLE 4.14: KPSS test results for the the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
false	-1.6552	-1.9416

TABLE 4.15: PP test results for the the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.

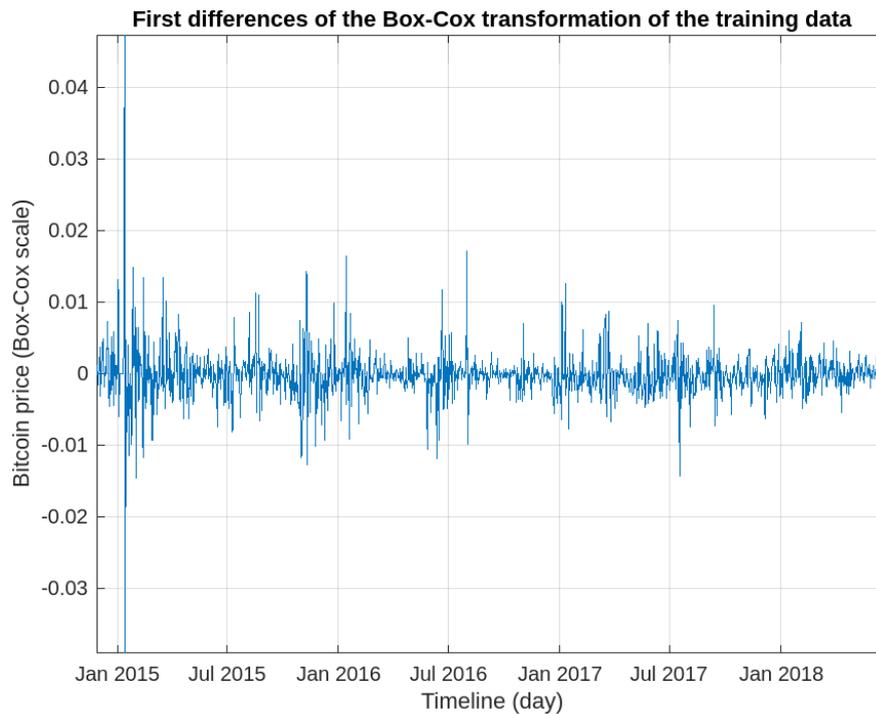


FIGURE 4.13: First differences of the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	0.001	-37.657

TABLE 4.16: ADF test results for the first differences of the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
false	0.11862	0.146

TABLE 4.17: KPSS test results for the first differences of the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.

Reject H0	Test statistic	Critical value
true	-37.657	-1.9416

TABLE 4.18: PP test results for the first differences of the Box-Cox transformation of the training data from 28 November 2014 to 26 June 2018.

4.5 Model estimation & selection

Here, we proceed on examining the ACF and PACF plots that can be useful to determine an ARIMA model or models suitable for the transformed training data. In table 4.19 is a handy cheat sheet for ACF and PACF plots interpretation [24]. Inspecting the ACF and PACF plots is a valuable starting point for identifying potential orders of an ARIMA model, but it is not always sufficient to guarantee the best modeling choice.

	ACF $\rho(k)$	PACF $\varphi_{k,k}$
AR(p)	Damped exponential and/or sine functions	$\varphi_{k,k} = 0$ for $k > p$
MA(q)	$\rho(k) = 0$ for $k > p$	Dominated by damped exponential and/or sine functions
ARMA(p, q)	Damped exponential and/or sine functions after lag $q - p$	Dominated by damped exponential and/or sine functions after lag $p - q$

TABLE 4.19: Characteristics for the autocorrelation functions.

Assuming that ACF and PACF plot inspection does not indicate a certain model suitable for the examined time series, we have to employ other methods towards effective modeling. We start with creating a number of different ARIMA models combining the orders of p, d , and q , while also making an assumption about a specific distribution for the residuals. We choose p and q to take values according to ACF and PACF plot inspection, while the possible effective orders of differencing, d , are defined by the findings in the previous step. If the time series exhibits heteroscedasticity, we use GARCH(p', q') models by combining the orders p' and q' to model the non-constant variance of the time series. Additionally, by incorporating the values of COMP, DJIA, and SPX indices as exogenous variables into the partly specified ARIMA or ARIMA-GARCH model, we can fit an ARIMAX or ARIMAX-GARCH model to the transformed training data. Note that the exogenous variables must align in scale with the endogenous one.

Subsequently, we estimate the parameters of every partly specified model using MLE and proceed on comparing the fully specified models, aiming to find the most appropriate ARIMA or ARIMA-GARCH model for the time series. AIC and BIC criteria, followed by residual diagnostics for confirmation, finally determine the best candidate. Note that, sometimes, the model indicated as the best by model selection criteria does not exhibit residuals that align with the desired behavior and the assumed distribution. So, we have to repeat the procedure as many times as needed to finally conclude to the best model.

Concerning the first partition of the Bitcoin dataset and the choice of 80:20 as the ratio of the training data to the test data, the ACF and PACF plots of the first differences of the Box-Cox transformation of the training data from the previous step are shown in figure 4.14. The values of the ACF at lags 2 and 18 clearly slightly exceed the level of statistical significance, suggesting potential autocorrelation at these lags, while the value at lag 12 is marginal and can more easily be considered negligible. In all three cases, the proximity to the significance threshold indicates that the evidence for potential autocorrelation is relatively weak and could be influenced by noise. Upon examination of the PACF, a comparable pattern emerges. Subtle but distinctive departures from the significance level are evident at lags 2 and 18, indicating potential autocorrelation. However, the value at lag 38 hovers around the margin and could be more easily considered insignificant. Similarly to the ACF interpretation, this consistency in findings emphasizes the cautious interpretation of potential autocorrelation, considering the proximity to the significance threshold. Any of these ACF and PACF values might be attributable to noise rather than indicating a specific strong association. Thus, examining the ACF and PACF plots suggested the presence of potential autocorrelation, but it did not offer clear guidance on determining the specific orders of an ARIMA model for effectively modeling the mean of the transformed training data.

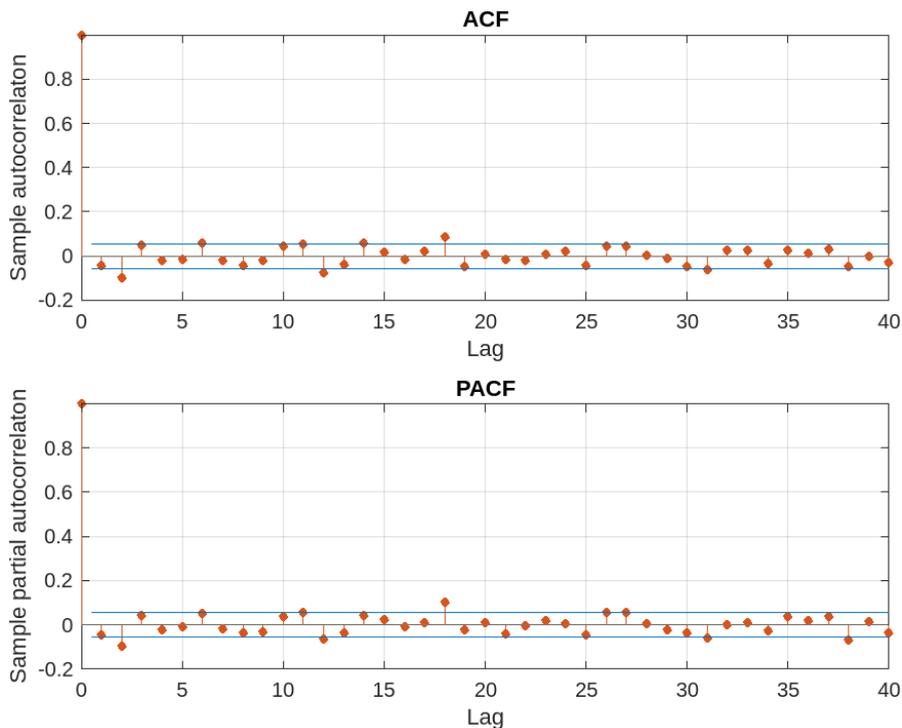


FIGURE 4.14: ACF and PACF plots of the first differences of the Box-Cox transformation of the training data.

We proceeded with creating 100 different ARIMA(p, q) models with p and q taking values from 1 to 10, and d equal to 1. To model the time-varying variance, we started by using the ARCH(1) or GARCH(0,1) model. We assumed that the residuals follow

a t-distribution. If necessary, we could then explore the effectiveness of more complex models. Subsequently, we fitted each composite ARIMA(p, d, q)-ARCH(1) model to the transformed training data, aiming to choose the best candidate based on the AIC and BIC information criteria. Both criteria led to the selection of the ARIMA(3,1,1)-ARCH(1) as the optimal model.

In figure 4.15, the ACF and PACF plots of the residuals reveal an absence of correlation. Along with a zero mean, the residuals resemble white noise, suggesting that the ARIMA(3,1,1)-ARCH(1) model has effectively captured the underlying patterns and structure in the data. Regarding the adequacy of the ARCH(1) model in addressing heteroscedasticity, the standardized residuals were found to have a zero mean, identified as homoscedastic through the ARCH test, as well as uncorrelated based on their ACF and PACF plots. To verify whether the residuals align with our initial assumption of a t-distribution, we fitted a t-distribution to the residuals and then conducted an Anderson-Darling test, which provided evidence that the residuals indeed follow a t-distribution. Additionally, comparing their distribution with a fitted t-distribution, as shown in figure 4.16, indicated a good fit. Therefore, the presented evidence collectively suggested the suitability of the ARIMA(3,1,1)-ARCH(1) model for modeling the Bitcoin time series in the first partition of the Bitcoin dataset with an 80:20 ratio of training data to test data.

Similarly, using 70:30 and 60:40 as ratios of the training data to the test data in the same dataset partition, we identified ARIMA(8,1,8)-ARCH(1) and ARIMA(5,1,5)-ARCH(1) as the best candidate models. In the second partition, also with training and test set ratios of 80:20, 70:30, and 60:40, ARIMA(1,1,7)-ARCH(1), ARIMA(9,1,9)-ARCH(1), and ARIMA(9,1,10)-ARCH(1) appeared to be the optimal choices for modeling the Bitcoin time series. In the third partition, using training and test set ratios of 70:30 and 60:40, ARIMA(7,1,7)-ARCH(1) and ARIMA(7,1,2)-ARCH(1) emerged as the most suitable models, respectively. Finally, taking into account the values of COMP, DJIA, and SPX indices as exogenous variables, we also selected the most suitable candidate ARIMAX-ARCH(1) models, which will be discussed in the next section. In each scenario, the ARCH(1) model effectively addressed the remaining heteroscedasticity, rendering more complex ARCH models unnecessary.

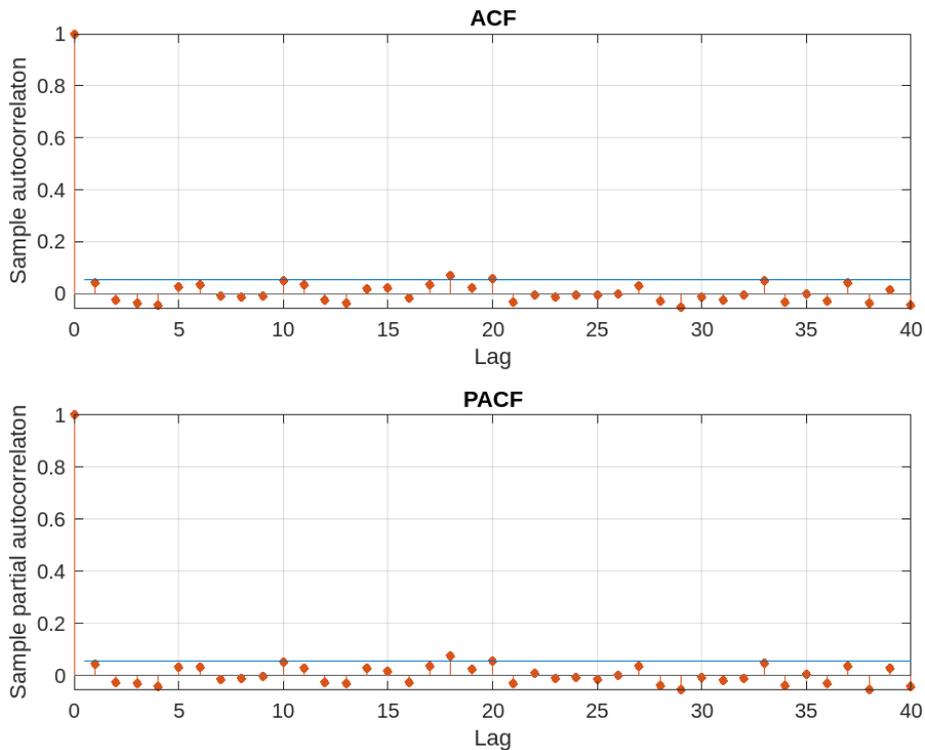


FIGURE 4.15: ACF and PACF plots of the residuals.

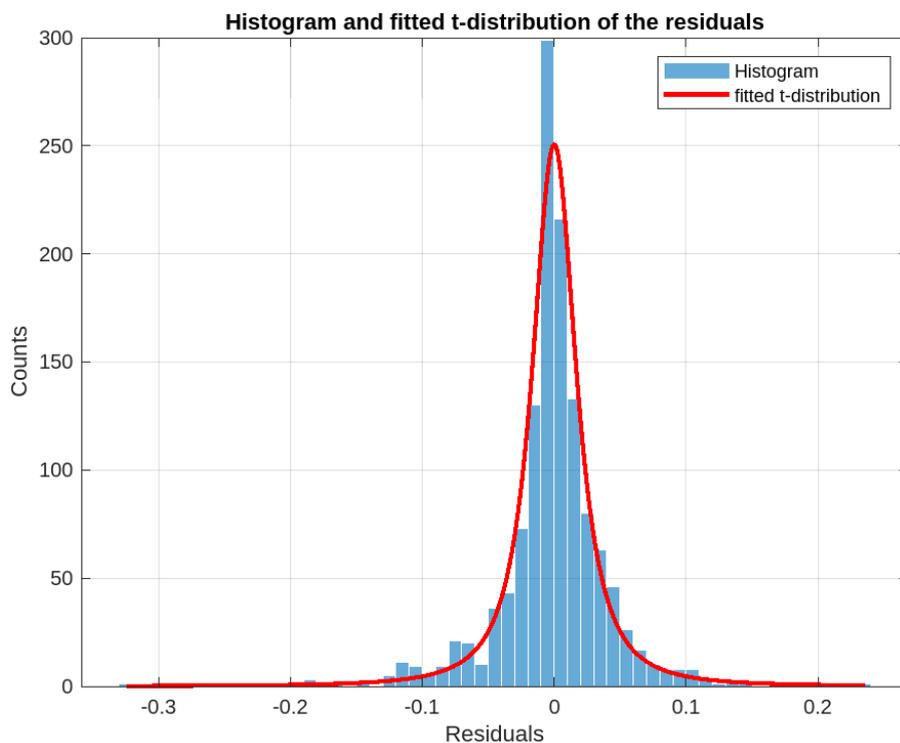


FIGURE 4.16: Histogram and fitted t-distribution of the residuals.

4.6 One-step ahead forecasting results

Eventually, after selecting the model or models that best fit the transformed training set of the time series data, we can generate prediction values for each day of the test set. In this process, we provide not only point predictions but also 95% prediction intervals, offering a range within which the true values are likely to fall. Specifically, we employ one-step ahead forecasting, meaning we forecast the value at each subsequent time point using information available up to the current time point. In the context of daily data, one-step ahead forecasting is equivalent to predicting the value for the next day. As outlined in the corresponding theoretical section, for each one-step ahead prediction using the $ARIMA(p, d, q)$ model, we require at least the p previous values of the time series. Note that as predictions are made using a model fitted to transformed data, the prediction intervals are also computed in the transformed scale. To ensure interpretability in terms of the original data, we back-transform them to the original scale. The final evaluation of the forecasting performance of each model is conducted using metrics MSE, RMSE and MAPE, based on the values predicted by the model and their corresponding actual values in the test set.

In figures 4.17, 4.18 and 4.19, we present the one-step ahead forecasting outcomes using the models $ARIMA(3,1,1)$ - $ARCH(1)$, $ARIMA(1,1,7)$ - $ARCH(1)$, and $ARIMA(7,1,7)$ - $ARCH(1)$. The specific models were selected based on distinct training to test set ratios for predicting the Bitcoin prices in the time intervals spanning from 27 June 2018 to 27 June 2019, 26 February 2020 to 26 February 2021, and 5 February 2018 to 5 June 2019, respectively. Tables 4.20, 4.21, and 4.22 display the MSE, RMSE, and MAPE scores for each tested model in our study, along with the *coverage probability* (CP) of the 95% prediction intervals, considering the forecasting time interval and the training to test data ratio. The best performing models for predicting the Bitcoin prices in each scenario are highlighted in yellow. Furthermore, we include the corresponding results for the same intervals obtained from literature works [3], [4], and [5] for comparison. $ARIMAX_1$, $ARIMAX_2$, and $ARIMAX_3$ represent $ARIMA$ models with SPX, DJIA, and COMP indices, respectively, as exogenous variables.

Referring to the data presented in table 4.20, the $ARIMA(3,1,1)$ - $ARCH(1)$ model, given an 80:20 training to test set ratio, yielded the best MSE, RMSE, and MAPE scores for the forecasting time interval spanning from 27 June 2018 to 27 June 2019. Additionally, it slightly outperformed the optimal LSTM model in terms of MSE and RMSE metrics, as reported in research study [3]. Notably, the 80:20 split consistently outperformed the 70:30 and 60:40 splits. The coverage probability of 100%, exceeding the expected 95%, signifies that the constructed prediction intervals consistently encompass the actual future observations. This deviation is attributed to employing a t-distribution with a small number of degrees of freedom and the discrete nature of the sample size, leading to wider intervals than anticipated.

In table 4.21, the $\text{ARIMAX}_3(1,1,7)\text{-ARCH}(1)$ model, with an 80:20 training to test set ratio, achieved the lowest MSE, RMSE, and MAPE scores. Although its MSE and RMSE values slightly outperformed those of Bi-LSTM, they appeared worse compared to those of the LSTM, GRU, and Bi-GRU models presented in study [4]. Similar to the first forecasting period, the 80:20 split consistently produced better scores than the 70:30 and 60:40 splits, while the coverage probability of 99.72% exceeded the expected 95% for the same reasons mentioned above.

Lastly, turning attention to table 4.22, the $\text{ARIMAX}_2(3,1,2)\text{-ARCH}(1)$ model, also given an 70:30 training to test set ratio, demonstrated superior performance, achieving the best MSE, RMSE, and MAPE scores between the compared models, and also surpassing those of the $\text{ARIMA}(1,1,0)$, LSTM, and GRU models proposed in [5]. The models performed equally well with both the 70:30 and 60:40 splits, achieving optimal scores an equal number of times, while coverage probability also exceeded the expected 95%, reaching 100% and 99.79%, respectively.

The findings from the first forecasting period illustrate that the inclusion of any of the specified exogenous variables (SPX, DJIA, and COMP indices) in the ARIMA models did not result in improved outcomes. However, incorporating them in the second and third forecasting periods, led to enhanced prediction results compared to ARIMA models without these variables as exogenous factors. This suggests that information from the SPX, DJIA, and COMP stock market indices can be beneficial in modeling the Bitcoin time series, contributing to more accurate predictions. That is an indication that the stock market could potentially play a role in influencing the Bitcoin price. Yet, further analysis is required to explore the nature and extent of this relationship.

In the second testing period, we observe an abrupt increase in prices, with corresponding RMSE scores for the various ARIMA-ARCH and ARIMAX-ARCH models significantly higher compared to those in the first and third forecasting periods, where fluctuations are milder. Notably, the static nature of the proposed ARIMA-ARCH model, which assumes a constant underlying structure of the data over time, becomes a weakness when there are sudden, significant alterations in the values of the data or changes in the pattern or characteristics of the data. In such cases, the model may struggle to adapt quickly, leading to less accurate forecasts.

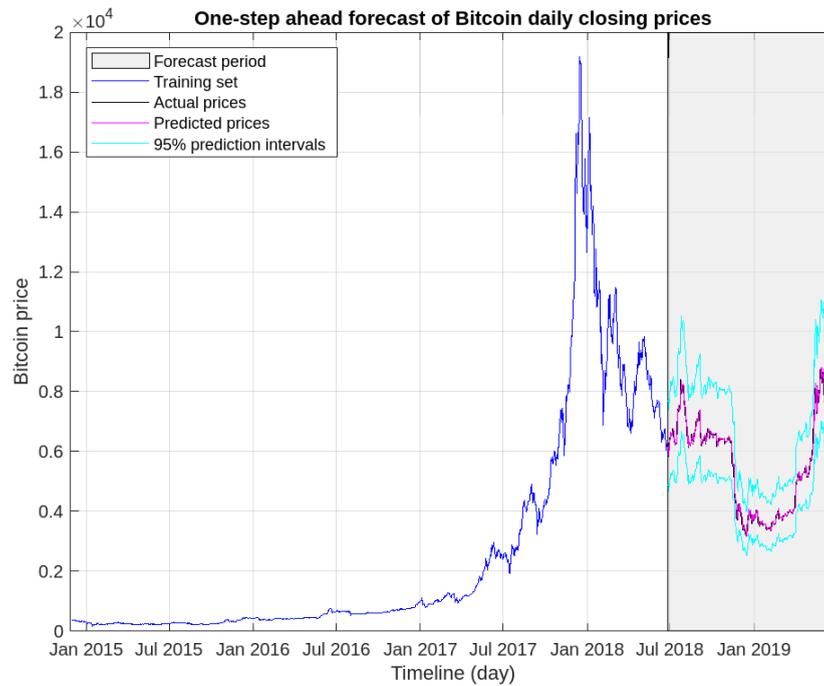


FIGURE 4.17: One-step ahead forecast of Bitcoin daily closing prices from 27 June 2018 to 27 June 2019 using the ARIMA(3,1,1)-ARCH(1) model with an 80:20 ratio of training to test data split.

Ratio	Model	MSE	RMSE	MAPE	CP
80/20	ARIMA(3,1,1)-ARCH(1)	54,766.40	234.02	2.32	100
70/30	ARIMA(8,1,8)-ARCH(1)	57,924.25	240.67	2.37	100
60/40	ARIMA(5,1,5)-ARCH(1)	59,111.35	243.12	2.59	100
80/20	ARIMAX ₁ (3,1,1)-ARCH(1)	54,904.06	234.31	2.33	100
70/30	ARIMAX ₁ (8,1,8)-ARCH(1)	61,294.71	247.57	2.48	100
60/40	ARIMAX ₁ (9,1,9)-ARCH(1)	63,219.01	251.43	2.82	100
80/20	ARIMAX ₂ (3,1,1)-ARCH(1)	54,984.46	234.48	2.33	100
70/30	ARIMAX ₂ (8,1,8)-ARCH(1)	61,541.61	248.07	2.48	100
60/40	ARIMAX ₂ (9,1,9)-ARCH(1)	63,279.05	251.55	2.81	100
80/20	ARIMAX ₃ (3,1,1)-ARCH(1)	54,911.61	234.33	2.33	100
70/30	ARIMAX ₃ (8,1,8)-ARCH(1)	62,248.27	249.49	2.50	100
60/40	ARIMAX ₃ (9,1,9)-ARCH(1)	62,824.54	250.64	2.79	100
80/20	LSTM	83,284.18	288.59	—	—

TABLE 4.20: MSE, RMSE, MAPE, and coverage probability results of one-step ahead forecast of Bitcoin daily closing prices from 27 June 2018 to 27 June 2019.

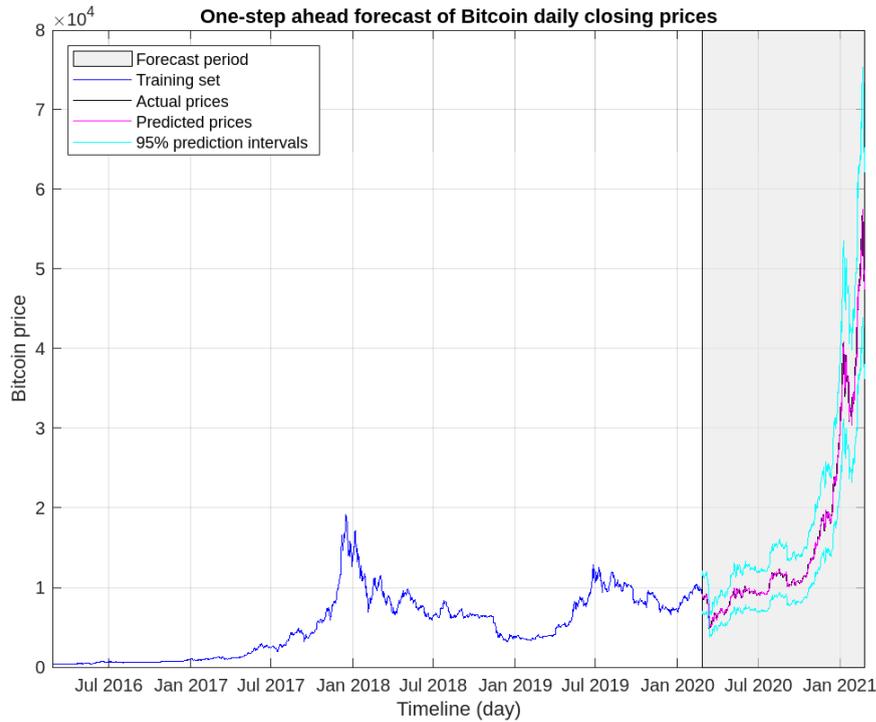


FIGURE 4.18: One-step ahead forecast of Bitcoin daily closing prices from 26 February 2020 to 26 February 2021 using the ARIMA(1,1,7)-ARCH(1) model with an 80:20 ratio of training to test data split.

Ratio	Model	MSE	RMSE	MAPE	CP
80/20	ARIMA(1,1,7)-ARCH(1)	922,434.37	960.43	2.75	99.72
70/30	ARIMA(9,1,9)-ARCH(1)	970,246.17	985.01	2.94	99.72
60/40	ARIMA(9,1,10)-ARCH(1)	923,102.65	960.78	2.82	99.72
80/20	ARIMAX ₁ (1,1,7)-ARCH(1)	918,473.05	958.37	2.74	99.72
70/30	ARIMAX ₁ (9,1,9)-ARCH(1)	1,024,872.76	1,012.36	3.08	99.72
60/40	ARIMAX ₁ (9,1,10)-ARCH(1)	920,525.11	959.44	2.81	99.72
80/20	ARIMAX ₂ (10,1,10)-ARCH(1)	992,853.78	996.42	2.85	99.72
70/30	ARIMAX ₂ (10,1,10)-ARCH(1)	1,138,363.10	1,066.94	3.11	99.72
60/40	ARIMAX ₂ (9,1,10)-ARCH(1)	920,768.43	959.56	2.81	99.72
80/20	ARIMAX ₃ (1,1,7)-ARCH(1)	917,492.91	957.85	2.73	99.72
70/30	ARIMAX ₃ (9,1,9)-ARCH(1)	1019,866.70	1,009.88	3.07	99.72
60/40	ARIMAX ₃ (9,1,10)-ARCH(1)	919,286.40	958.79	2.8	99.72
80/20	LSTM	578,497.14	760.59	—	—
80/20	Bi-LSTM	962,714.19	981.18	—	—
80/20	GRU	593,608.61	770.46	—	—
80/20	Bi-GRU	183,055.62	427.85	—	—

TABLE 4.21: MSE, RMSE, MAPE, and coverage probability results of one-step ahead forecast of Bitcoin daily closing prices from 26 February 2020 to 26 February 2021.

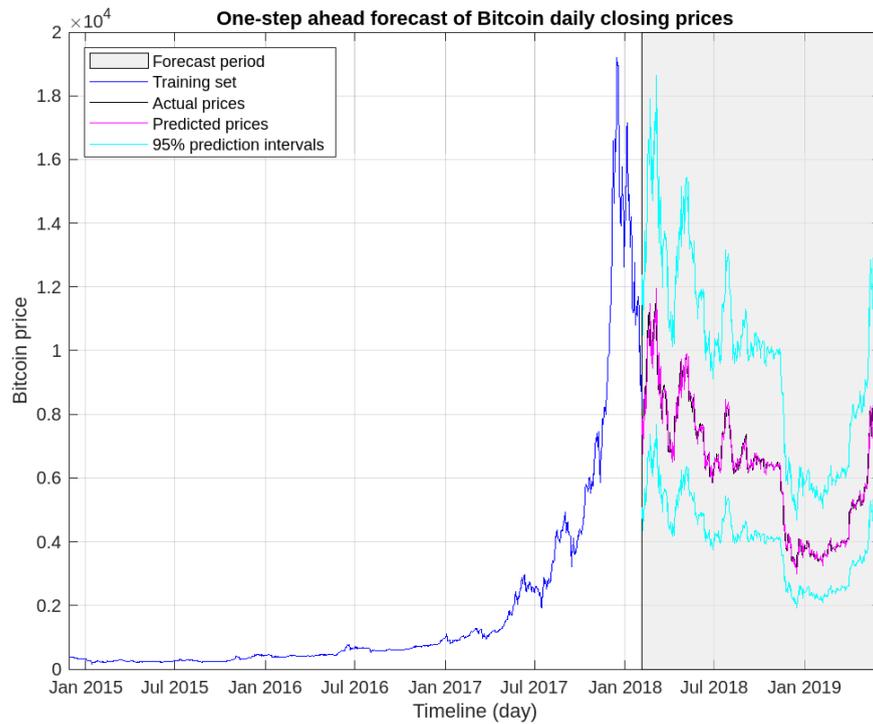


FIGURE 4.19: One-step ahead forecast of Bitcoin daily closing prices from 5 February 2018 to 5 June 2019 using the ARIMA(7,1,7)-ARCH(1) model with an 70:30 ratio of training to test data split.

Ratio	Model	MSE	RMSE	MAPE	CP
70/30	ARIMA(7,1,7)-ARCH(1)	94,463.66	307.34	2.96	100
60/40	ARIMA(7,1,2)-ARCH(1)	77,035.15	277.55	2.67	100
70/30	ARIMAX ₁ (7,1,7)-ARCH(1)	108,516.12	329.41	3.18	99.79
60/40	ARIMAX ₁ (7,1,2)-ARCH(1)	77,007.06	277.50	2.68	100
70/30	ARIMAX ₂ (3,1,2)-ARCH(1)	75,483.48	274.74	2.65	100
60/40	ARIMAX ₂ (7,1,2)-ARCH(1)	76,974.74	277.44	2.68	100
70/30	ARIMAX ₃ (3,1,2)-ARCH(1)	75,569.11	274.89	2.65	100
60/40	ARIMAX ₃ (7,1,2)-ARCH(1)	77,011.88	277.51	2.68	100
70/30	ARIMA(1,1,0)	91,524.40	302.53	2.76	—
70/30	LSTM	364,429.54	603.68	6.80	—
70/30	GRU	145,420.19	381.34	3.97	—

TABLE 4.22: MSE, RMSE, MAPE, and coverage probability results of one-step ahead forecast of Bitcoin daily closing prices from 5 February 2018 to 5 June 2019.

Conclusion & Future Work

In this thesis we explored the modeling efficiency and forecasting capabilities of the ARIMA and ARIMA-GARCH models for the prediction of the Bitcoin daily closing price. The training data used involve daily closing prices over a number of months. The significant presence of heteroscedasticity in the Bitcoin time series data rendered the ARIMA models unsuitable. On the other hand, the composite ARIMA-ARCH(1) models, effectively addressing heteroscedasticity, proved to be successful in capturing the patterns of variability and the correlation structure of the Bitcoin time series. To evaluate predictive performance, we conducted comparisons of one-day-ahead forecasts for three forecasting scenarios which involve different time periods. For each scenario, we experimented with various training-to-test ratios. The comparisons involved several ARIMA-ARCH(1) models with different $AR(p)$ and $MA(q)$ orders, as well as ARIMAX-ARCH(1) models that incorporate the SPX, DJIA, and COMP stock market indices as exogenous variables. After identifying the optimal ARIMA-ARCH(1) models (using model selection methods), we compared their predictive performance with that of the advanced LSTM, Bi-LSTM, GRU, and Bi-GRU RNN models proposed in other research works.

More specifically, for the scenario involving the test period from 27 June 2018 to 27 June 2019, we found that the ARIMA(3,1,1)-ARCH(1) model, given an 80:20 training-test split, exhibited superior predictive accuracy (MSE, RMSE, MAPE) compared to LSTM (using the same training-test split). For the scenario which involves the test period from 26 February 2020 to 26 February 2021, we determined that the ARIMAX₃(1,1,7)-ARCH(1) model, employing an 80:20 training-test split, slightly outperformed Bi-LSTM. However, it fell short in comparison to the accuracy achieved by LSTM, GRU, and Bi-GRU (using the same training-test split). For the scenario involving the test period spanning from 5 February 2018 to 5 June 2019, given a 70:30 training-test split, the ARIMAX₂(3,1,2)-ARCH(1) model exhibited enhanced predictive capabilities, surpassing the poor performance of ARIMA(1,1,0), as expected, as well as the performance of LSTM and GRU implementations.

The results collectively emphasize that the optimal ARIMA-GARCH and ARIMAX-GARCH models depend on the specific time periods and the training-test split. Additionally, our results demonstrate that ARIMA-GARCH and ARIMAX-GARCH models can achieve comparable, and in some cases, superior forecasting performance compared to RNN models. Last but not least, the performance scores of the employed ARIMAX-ARCH(1) model reveals potential influence of the stock market on the price of Bitcoin and, generally, on the cryptocurrency market.

It is crucial to acknowledge that ARIMA models provide a transparent and interpretable framework for time series forecasting. Conversely, both of the RNN variants, LSTM and GRU, characterized by intricate architectures, exhibit escalated complexity and diminished interpretability. Thus, in cases where our primary focus is on understanding the factors that influence predictions, the preference leans towards ARIMA models over the intricate "black box" nature inherent in RNNs. Moreover, ARIMA models come with a straightforward method for estimating uncertainty by constructing prediction intervals based on the distribution of residuals. In contrast, RNNs require more sophisticated techniques to quantify uncertainty and establish prediction intervals.

Future studies could delve more deeply into the relationship between Bitcoin price and traditional financial markets. In addition, one can explore other auxiliary factors capable of enhancing the predictive power of ARIMA-GARCH models, including the price of gold, the popularity of Bitcoin derived from social media or news, or the price of competing cryptocurrencies. Beyond this, varying the forecasting horizon could offer valuable insights into how ARIMA-GARCH models can be strategically employed for different investment goals, from short-term trading to long-term wealth accumulation. It is worth noting that the proposed methodology can be used for the prediction of not only the daily closing prices, but also the daily opening, lowest, and highest prices of Bitcoin. Moreover, its applicability extends to other cryptocurrencies, enabling the prediction of their respective daily opening, closing, lowest, and highest prices as well.

In this work we used a fixed ARIMA-GARCH model to generate forecasts for future time steps. This means that the model parameters were not updated during the forecasting process. An alternative approach worth exploring is a recursive forecasting strategy. In this strategy, after each forecast, the actual observation at the current time step is included in the sample set, and the model is re-estimated using this new information. The model update can be estimated using either a fixed-size window of historical data or all available data up to the current point. Another possibility is a hybrid update strategy in which the model is re-estimated after a certain number of steps. This strategy strikes a balance between a fully static approach—where the model is never updated—and a fully recursive approach—where the model is updated at every step. It allows the model to adapt to changes in the data but does so at less frequent intervals, providing a compromise between model stability and responsiveness to evolving patterns. Finally, averaging ARIMA-GARCH forecasts with those of an RNN incorporating attention mechanisms could also be an intriguing avenue for future research. An RNN with attention mechanisms can excel at capturing complex, non-linear relationships, as well as be more robust to irregularities and outliers in the data. Combining the strengths of both models might result in a more versatile and generalized ensemble model with improved forecasting accuracy and robustness.

In conclusion, we have contributed valuable insights regarding the capabilities and limitations of static ARIMA and ARIMA-GARCH models in predicting the next-day price

of the Bitcoin cryptocurrency. The findings presented herein open new directions for future research that can deepen our comprehension of the intricate dynamics governing the cryptocurrency market. Further legalization, along with clearer regulatory frameworks, have the potential to reduce uncertainty in the cryptocurrency market. This increased transparency can attract more participants and contribute to a more stable and mature market environment. Nevertheless, it is crucial to acknowledge that forecasting financial markets, particularly cryptocurrencies, inherently entails uncertainty as unforeseen events and market sentiment are influential factors shaping price movements. Consequently, a cautious approach and an awareness of inherent risks should always accompany cryptocurrency forecasts.

Bibliography

- [1] Kate Murray et al. “On Forecasting Cryptocurrency Prices: A Comparison of Machine Learning, Deep Learning, and Ensembles”. In: *Forecasting* 5.1 (2023), pp. 196–209. ISSN: 2571-9394. DOI: [10.3390/forecast5010010](https://doi.org/10.3390/forecast5010010).
- [2] I Made Wirawan, Triyanna Widiyaningtyas, and Muchammad Maulana Hasan. “Short Term Prediction on Bitcoin Price Using ARIMA Method”. In: *2019 International Seminar on Application for Technology of Information and Communication (iSemantic)*. 2019, pp. 260–265. DOI: [10.1109/ISEMANTIC.2019.8884257](https://doi.org/10.1109/ISEMANTIC.2019.8884257).
- [3] Ferdiansyah Ferdiansyah et al. “A LSTM-Method for Bitcoin Price Prediction: A Case Study Yahoo Finance Stock Market”. In: *2019 International Conference on Electrical Engineering and Computer Science (ICECOS)*. 2019, pp. 206–210. DOI: [10.1109/ICECOS47637.2019.8984499](https://doi.org/10.1109/ICECOS47637.2019.8984499).
- [4] Ashish Singh, Abhinav Kumar, and Zahid Akhtar. “Bitcoin Price Prediction: A Deep Learning Approach”. In: *2021 8th International Conference on Signal Processing and Integrated Networks (SPIN)*. 2021, pp. 1053–1058. DOI: [10.1109/SPIN52536.2021.9565988](https://doi.org/10.1109/SPIN52536.2021.9565988).
- [5] Peter T. Yamak, Li Yujian, and Pius K. Gadosey. “A Comparison between ARIMA, LSTM, and GRU for Time Series Forecasting”. In: *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*. 2020, pp. 49–55. ISBN: 9781450372619. DOI: [10.1145/3377713.3377722](https://doi.org/10.1145/3377713.3377722).
- [6] C. Chatfield. *The Analysis of Time Series: An Introduction, Sixth Edition*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, 2016, p. 33. ISBN: 9780203491683. URL: <https://books.google.gr/books?id=qKzyAbdaDFAC>.
- [7] Emanuel Parzen. “An Approach to Time Series Analysis”. In: *The Annals of Mathematical Statistics* 32.4 (1961), pp. 951–989. DOI: [10.1214/aoms/1177704840](https://doi.org/10.1214/aoms/1177704840).
- [8] R.J. Hyndman and G. Athanasopoulos. *Forecasting: Principles and Practice*. OTexts, 2014. ISBN: 9780987507105. URL: <https://books.google.gr/books?id=nmTQwAEACAAJ>.
- [9] Nina Golyandina, Vladimir Nekrutkin, and Anatoly Zhigljavsky. “Analysis of Time Series Structure: SSA and Related Techniques”. In: *Monographs on Statistics and Applied Probability* 90 (Jan. 2001). DOI: [10.1201/9781420035841](https://doi.org/10.1201/9781420035841).
- [10] D. Dickey and Wayne Fuller. “Distribution of the Estimators for Autoregressive Time Series With a Unit Root”. In: *JASA. Journal of the American Statistical Association* 74 (June 1979). DOI: [10.2307/2286348](https://doi.org/10.2307/2286348).
- [11] James Mackinnon. “Critical Values for Cointegration Tests”. In: *Long-Run Economic Relationships* (Feb. 1990).
- [12] Denis Kwiatkowski et al. “Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?” In: *Journal of Econometrics* 54.1 (1992), pp. 159–178. ISSN: 0304-4076. DOI: [https://doi.org/10.1016/0304-4076\(92\)90104-Y](https://doi.org/10.1016/0304-4076(92)90104-Y).
- [13] Peter C. B. Phillips and Pierre Perron. “Testing for a Unit Root in Time Series Regression”. In: *Biometrika* 75.2 (1988), pp. 335–346. ISSN: 00063444. URL: <http://www.jstor.org/stable/2336182> (visited on 02/17/2024).
- [14] H. L. S. “A Study in the Analysis of Stationary Time Series. By Herman Wold. [Pp. 214 viii. Almqvist and Wiksells Boktryckeri-A.-B., Uppsala. 1938. Price kr. 6.]” In: *Journal of the Institute of Actuaries* 70.1 (1939), pp. 113–115. DOI: [10.1017/S0020268100011574](https://doi.org/10.1017/S0020268100011574).
- [15] Robert H. Shumway and David S. Stoffer. *Time Series Analysis and Its Applications (Springer Texts in Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2005. ISBN: 0387989501.

- [16] Robert F. Engle. "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation". In: *Econometrica* 50.4 (1982), pp. 987–1007. ISSN: 00129682, 14680262. URL: <http://www.jstor.org/stable/1912773>.
- [17] Tim Bollerslev. "Generalized autoregressive conditional heteroskedasticity". In: *Journal of Econometrics* 31.3 (1986), pp. 307–327. ISSN: 0304-4076. DOI: [https://doi.org/10.1016/0304-4076\(86\)90063-1](https://doi.org/10.1016/0304-4076(86)90063-1).
- [18] Hirotugu Akaike. "A new look at the statistical model identification". In: *IEEE Transactions on Automatic Control* 19 (1974), pp. 716–723. URL: <https://api.semanticscholar.org/CorpusID:411526>.
- [19] Kenneth Burnham and David Anderson. "Model Selection and Multimodel Inference". In: *A Practical Information-theoretic Approach* (Jan. 2004). DOI: [10.1007/978-0-387-22456-5_5](https://doi.org/10.1007/978-0-387-22456-5_5).
- [20] Ernst Wit, Edwin van den Heuvel, and Jan-Willem Romeijn. "All models are wrong...': an introduction to model uncertainty". English. In: *Statistica Neerlandica* 66.3 (Aug. 2012), pp. 217–236. ISSN: 0039-0402. DOI: [10.1111/j.1467-9574.2012.00530.x](https://doi.org/10.1111/j.1467-9574.2012.00530.x).
- [21] T. W. Anderson and D. A. Darling. "Asymptotic Theory of Certain "Goodness of Fit" Criteria Based on Stochastic Processes". In: *The Annals of Mathematical Statistics* 23.2 (1952), pp. 193–212. DOI: [10.1214/aoms/1177729437](https://doi.org/10.1214/aoms/1177729437).
- [22] R.S. Pindyck and D.L. Rubinfeld. *Econometric Models and Economic Forecasts*. International student edition. McGraw-Hill, 1976. ISBN: 9780070500952. URL: <https://books.google.gr/books?id=5EwEAQAIAAJ>.
- [23] H. Pishro-Nik. *Introduction to Probability, Statistics, and Random Processes*. Kappa Research, LLC, 2014. ISBN: 9780990637202. URL: https://books.google.gr/books?id=3yq_oQECAAJ.
- [24] Henrik Madsen. *Time Series Analysis*. Oct. 2008. ISBN: 978-1-4200-5967-0. DOI: [10.1201/9781420059687](https://doi.org/10.1201/9781420059687).