Czech University of Life Sciences Prague

Faculty of Economics and Management

Department of Statistics

Master's Thesis

Time-Series Analysis of Cryptocurrencies

Tural Dadashzade

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

DIPLOMA THESIS ASSIGNMENT

Bc. Tural Dadashzade

Informatics

Thesis title

Time Series Analysis of Cryptocurrencies

Objectives of thesis

The aim of the thesis is to make the evaluation of selected Cryptocurrencies with highest market capitalization, finding relationship between the coins, their time series analysis and predictions for future prices together with market capitalizations.

Methodology

The methodology used in the thesis assumes statistical analysis of time series, with regard to the nature of the data used will be used methods with fixed as well as the adaptive methods, such as exponential smoothing or ARIMA models.

The proposed extent of the thesis

60 – 80 pages

Keywords

Time-Series Analysis, Statistical Theory, Multivariate Time Series Models.

Recommended information sources

ENDERS, W. *Applied econometric time series.* Hoboken: Wiley, 2015. ISBN 978-1-118-80856-6.
FIELD, A P. – MILES, J. – FIELD, Z. *Discovering statistics using R.* London: SAGE, 2012. ISBN 978-1-4462-0046-9.
GRABOWSKI, M. *Cryptocurrencies : A Primer on Digital Money. [elektronický zdroj] /.* Milton: Taylor & Francis Group, 2019. ISBN 9780429510144.
HATCHER, L. *Advanced statistics in research : reading, understanding, and writing up data analysis results.* Saginaw, MI: ShadowFinch Media, LLC, 2013. ISBN 978-0-9858670-0-3.
CHOWDHURY, N. *Inside Blockchain, Bitcoin, and Cryptocurrencies. [elektronický zdroj] /.* Milton: Auerbach Publishers, Incorporated, 2019. ISBN 9781000507706.
KOČENDA, E. – ČERNÝ, A. – UNIVERZITA KARLOVA. *Elements of time series econometrics: an applied approach.* Prague: Karolinum, 2014. ISBN 978-80-246-2315-3.
PESARAN, M H. *Time series and panel data econometrics.* Oxford: Oxford University Press, 2015. ISBN 978-0-19-875998-0.

Expected date of thesis defence

2022/23 SS – FEM

The Diploma Thesis Supervisor

Ing. Tomáš Hlavsa, Ph.D.

Supervising department

Department of Statistics

Electronic approval: 20. 6. 2022	Electronic approval: 28. 11. 2022
prof. Ing. Libuše Svatošová, CSc.	**doc. Ing. Tomáš Šubrt, Ph.D.**
Head of department	Dean

Prague on 07. 02. 2023

Declaration

I declare that I have worked on my master's thesis titled "Time-Series Analysis of Cryptocurrencies" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the master's thesis, I declare that the thesis does not break any copyrights.

In Prague on date of submission ______________________________

Acknowledgement

I would like to thank to my thesis supervisor Ing. Tomas Hlavsa, Ph.D for all the support and consultations provided during the course of this thesis.

Time-Series Analysis of Cryptocurrencies

Abstract

The present thesis experimentally analysed the time series of cryptocurrencies and the effect of different coefficients in models towards to them. Selection of the cryptocurrencies to investigate has been proceeded with checking the crypto market volumes. The selected cryptocurrencies which were among the coins with highest market capitalization have been chosen and they are the followings Bitcoin (BTC) and Ethereum (ETH). After finalizing the individual analysis, the predictions is made with choosing each other as exogenous variable. However, the thesis does not assess the relations between cryptos, it uses each other as a supporting variable in order to make accurate predictions. Individual observations are made and the statistical significances are tested through different time series models. Towards the end of the thesis, the predictions for the cryptocurrency prices are made and the fate of the market on the horizon is estimated based on the evaluation of training data set with different models.

Keywords: Time-Series Analysis, Statistical Theory, Multivariate Time Series Models, Cryptocurrencies, ARIMA, Python, ARCH, GARCH.

Analýza časových řad kryptoměn

Abstrakt

Tato práce experimentálně analyzovala časové řady kryptoměn a vliv různých koeficientů v modelech na ně. Výběr kryptoměn k prozkoumání byl proveden kontrolou objemů kryptotrhu. Byly vybrány vybrané kryptoměny, které patřily mezi coiny s nejvyšší tržní kapitalizací, a to Bitcoin (BTC) a Ethereum (ETH). Po dokončení individuální analýzy jsou provedeny predikce s tím, že se navzájem zvolí jako exogenní proměnné. Práce však neposuzuje vztahy mezi kryptoměnami, využívá se navzájem jako podpůrné proměnné pro přesné předpovědi. Provádějí se jednotlivá pozorování a statistická významnost se testuje prostřednictvím různých modelů časových řad. V závěru práce jsou provedeny predikce pro ceny kryptoměn a na základě vyhodnocení sady tréninkových dat s různými modely je odhadnut osud trhu na obzoru.

Klíčová slova: Analýza časových řad, statistická teorie, vícerozměrné modely časových řad, kryptoměny, ARIMA, Python, ARCH, GARCH

Table of content

1 Introduction

The concept of virtual money, which first emerged in 2009, has been an important development for states and individuals. There are different opinions about the emergence of the virtual currency system. The most accepted view is that the virtual currency system was discovered because of the loss of financial trust in financial markets and states after the 2008 global crisis and its popularity has gradually increased. In this context, it is of great importance to analyse Bitcoin, which is the most used cryptocurrency system and popular all over the world. Therefore, it is expected that this research will make an important contribution towards filling the gap in this field in the literature in terms of theory. Therefore, it is very important to conduct a study on the appearance of the Bitcoin ecosystem, which is an example of the concept of virtual money. In the study, after revealing the conceptual framework of cryptocurrencies, the structure of Bitcoin and other crypto systems, are examined practically and theoretically. In this thesis, after giving a comprehensive theoretical information about the topic, we covered a deep analysis of the market and chosen cryptos. Furthermore, other factors which potentially can have an impact or correlation to crypto market is investigated and concluded with statistical statements and predictions (Wayne, D, 2022).

In the first part of the study, the concept of virtual money so called cryptocurrencies, their features, classification, and the legal framework are examined. In this section, the selected cryptocurrencies will be covered, and their purpose and capabilities will be indicated. What crypto money is, how it emerged and why it spread rapidly are discussed in detail in the light of historical processes. In addition to these, information is given on the structure and use of the crypto monetary system. What kind of virtual money the coins are, what kind of system (Block Chain) is behind, where, and how it is used are discussed in detail. The collection of factual knowledge about each and single cryptocurrency is examined separately. Their characteristics, opportunities that they serve and the technology behind of each cryptocurrency is added. The dominance of the cryptocurrencies which are being neglected by investors is also comprehensively mentioned to deliver the importance and will be used in second part where the practical usage is shown. The paper includes data about the competitive advantage of the selected cryptocurrencies depending on the users' goals of utilizing cryptocurrencies. The advantages and disadvantages of virtual money are discussed in general and on coin level. The potential suspicious relations will be mentioned in the first part where the statistical questions will evolve

The second part of the thesis is about practically using the statistical models, tests and time series analysis in order to define the behaviour of the selected cryptocurrencies in the market and their correlations to each other. Selected cryptocurrencies are based on their market capitalizations and volumes which can be counted to 2. The individual data points, their historical prices, market volumes through various methods is shown as numeric and also graphical way. Again, similarly, all the investigation proceeds with certain statistical models. The graphs of historical changes and the comparison to the crypto prices is shown. The tests for the outcomes are completed separately to increase the reliability of the investigation and to give the assurance to make conclusions and the predictions. Towards to the end of the paper the results and predictions is combined and presented in collective form in the part of conclusion.

2 Objectives and Methodology

2.1 Objectives

The ultimate objective of the thesis is to evaluate, assess and the development of predictions for the cryptocurrencies. We will be using various methods to achieve this objective and that is the reason we will also divide our aim into so called sub objectives which will contain multiple milestones. One of the milestones will be the univariate analysis for selected crypto coins (BTC, ETH) to summarize their historical prices with various indicators. In this section the aim will be to build and compare various time-series models and then choose the best fitting model. Based on individual analysis the statistical tests are executed to increase the accuracy of the assessment as a different sub-goal of the thesis. After these steps, achieving the results of multivariate analysis with another time series method which can allow us using exogenous variables and tests will be the target to reach. Furthermore, the market volatility will be assessed and evaluated with other well-known time series models.

2.2 Methodology

To achieve the aim of the above-mentioned objectives there are various ways of evaluating the individual variables. The most important step in expressing an economic relationship econometrically is to make the relevant variables expressible with numbers. For this purpose, it is necessary to collect, compile and organize data about the variables to be included in the model. Since it is not possible to conduct an empirical study on a subject where data cannot be collected, it is important to first determine and obtain data related to the subject. Data collection for studies usually takes several forms. One of these is to take advantage of previously collected information. These are mostly in the form of statistical bulletins or statistical annuals. Another method is the direct observation method. There is a measurement process in this process. Measurement is done in different ways. The population that is the subject of the research is either completely measured, or in cases where it is very difficult or even impossible to measure the entire population, an estimate of the population is made with the help of a sample (Amadebai, N.D).

Description Statistics of Time Series:

Univariate Analysis is one of the most common analysis which is being used in almost all of the statistical researcehs. A single observation over a time period makes up the univariate time series. Multiple observations collected over time make up the multivariate time series. Taking into consideration that this analysis gives a great comprehensive information about the individual analysis of the variables, we will use this method in order to get general information on our variables. Methods of univariate descriptive analysis solve the problem of compressing the original information, its compact representation. As a rule, in the process of research, it is important to obtain the cumulative characteristics of individual objects through the prism of a particular property. Instead of a large number of individual indicators, we need one value that would be typical (representative) for the entire population of objects. Univariate descriptive analysis uses methods such as Construction of frequency distributions, graphical representation of the behavior of the analyzed variable and obtaining statistical characteristics of the distribution of the analyzed variable. Many services can now be provided in real time thanks to the growth of time series applications. There are numerous issues that arise as the amount of time series data grows. Time series analysis mechanisms are necessary to ensure the accuracy of the forecast. The AR, MA, ARMA, and ARIMA methods can only be used with univariate time series data, despite their advantages and disadvantages for time series analysis(W.Palma, 2016).

Time – Series Analysis:

Time series, as a rule, arise because of measuring some indicator. These can be both characteristics of technical systems and indicators of natural, socio-economic phenomena and processes. For example, the dynamics of the exchange rate or the stock price, in the analysis of which they try to determine the main direction of development, i.e. trend. Or, for example, an analysis of the company's sales dynamics in order to plan stock balances. The main purpose of time series analysis is to build a forecast of its values for future periods. And the main tasks of time series analysis are to understand under the influence of which components the value of the time series is formed, and to build a mathematical model for each component or their combination. Any time series can be decomposed into the following components: trend, seasonal component, cyclical component, and random component. The first three components form a non-random component of the time series. The random component is present in any time series. But the presence in the structure of the time series of components of a non-random component is not necessary. Time series modelling approaches can be divided into two areas. Modelling of a non-random component in the aggregate and Composition of the time series into constituent components and modelling the values of each component separately. Statistical forecasting methods are divided into algorithmic methods and analytical methods. Algorithmic methods include simple and weighted moving average methods. Analytical methods include predictive extrapolation methods based on growth curves as functions of time. If there is a seasonal or cyclical component in the time series, an analysis of periodic fluctuations or a spectral analysis of the time series is carried out. Time series are classified into stationary and non-stationary. To analyse and build a forecast for a stationary time series, special methods are used(W.Palma, 2016).

Moving average models (MA models), autoregressive models (AR models) or mixed models (ARMA) or integrated moving average and autoregressive models (ARIMA). The formulas for the above-mentioned models are as follows:

AR model:

Equation 1

$$Yt = \beta_1 * y_{-1} + \beta_2 * y_{t-2} + \beta_3 * y_{t-3} + \ldots\ldots\ldots\ldots + \beta_k * y_{t-k}$$

MA model:

Equation 2

$$Yt = \alpha_1 * \varepsilon_{t-1} + \alpha_2 * \varepsilon_{t-2} + \alpha_3 * \varepsilon_{t-3} + \ldots\ldots\ldots\ldots + \alpha_k * \varepsilon_{t-k}$$

ARMA model:

Equation 3

$$Yt = \beta_1 * y_{t-1} + \alpha_1 * \varepsilon_{t-1} + \beta_2 * y_{t-2} + \alpha_2 * \varepsilon_{t-2} + \beta_3 * y_{t-3} + \alpha_3 * \varepsilon_{t-3} + \ldots\ldots\ldots\ldots + \beta_k * y_{t-k} + \alpha_k * \varepsilon_{t-k}$$

(Shetty, C, 2020)

A separate direction in forecasting is adaptive forecasting models. In addition, when studying multifactorial time series, conventional regression models can be used to build a forecast, with time series reduced to a stationary form. Forecasting is closely related to planning and is used for effective decision making. Forecasting can provide an answer to the various questions like what is most likely to be expected in the future regarding the process under study? Or what needs to be done to achieve a given state of the forecast object under study?

The series formed by the observations of a variable at equal time intervals is called the "time series". In the time series obtained by ordering these observation results according to the options of a time attribute such as year, week, and day, there are observation values opposite the time attribute, and in this way, the variability of the event that is the subject of statistical research over time is observed. Time series data is usually compiled and collected at daily, weekly, monthly, quarterly, semi-annual, annual, and longer-term intervals. In general, the time series is represented as Zt, t= 1, 2, …, T, with T being the sample size. Accordingly, the first observed data is Z1, the second observed data is Z2, the last observed data is expressed as ZT. Series with data that can be recorded continuously over time is called "continuous time series", and series with data that can only be obtained at certain intervals are called "discrete time series". While series belonging to engineering fields such as electrical signals, voltage, and sound vibrations are continuous time series; Economic series such as interest rate, sales amount, and production are examples of discrete time series. However, in the purpose of the aims of this thesis the most important 2 forms are as below:

Economic and financial time series: Most of the economic and financial data consists of time series. Examples of these are series such as daily exchange rate, stock return, annual interest rate, and inflation rate.

Business time series: Data such as sales analysis of businesses, profitability ratios, and cost calculations observed in different periods are used effectively in determining, directing, or changing business policies (W.Palma, 2016).

White Noise: White noise is an important concept in time series analysis and making predictions. In a nutshell, white noise indicates whether your data is predictable or not. Also, it tells you if the model should be further optimized or not. Because it is a random number sequence, white noise is an unpredictable series. If you build a model and the residuals (the difference between predicted and actual values) look like white noise, you know you did everything possible to improve the model. On the other hand, if there are visible patterns in the residuals, you have a better fitting model for your dataset. It is significant for 2 reasons:

- Predictability: If your time series is white noise, it is random by definition. You can't reasonably model it and predict it.
- Model Diagnostics: A time series forecast model's series of errors should ideally be white noise.

Time series forecasting relies heavily on model diagnostics. On top of the signal generated by the underlying process, time series data are expected to contain some white noise.

For a time-series to be classified as white noise, the following conditions must be met:

- The average (mean) value is zero.
- The standard deviation remains constant over time.
- The relationship between time series and their lag is not significant. We would want to see if there is a significant correlation between the current time series and the same time series that has been shifted by N periods.

There are three (simple) ways to determine whether a time series resembles white noise:

- By displaying the series
- By making comparison the average and standard deviation over time
- Assessing autocorrelations

Once a time series forecast model has made predictions, they can be collected and analysed. Ideally, the series of forecast errors should be white noise. When forecast errors are white noise, it means that the model has used all of the signal information in the time series to make predictions. All that remains are the uncontrollable random fluctuations. A sign that model predictions are not white noise indicates that the forecast model can be improved further (Brownlee, J, 2017).

Stationarity: One of the basic operations in time series analysis is "stationary" (constant) distributional ensembles. A stationary process includes the mean and variance of which do not change over time, and the covariance between two periods depends on the distance between the periods, not the turning point. According to our definition, a stationary time series is a series whose mean, variance and covariance are independent of time. Such a series exhibits constant width oscillations around its mean. This property is also called mean reversion. Such stationary series can be encountered in the literature with different names:

- weak stationary.
- covariance stationary.
- second-order stationary.

In empirical studies with time series, it is assumed that the data are "stationary". However, most of the time series are not stationary. In order for the relationships between the variables to be meaningful, the time series we use must show stationary properties. Although there are no significant relationships between the two variables, it may seem as if there is a relationship between them. When we establish a regression model with these series, a high R^2 value can be obtained even if there is no relationship between them. In this case, the spurious regression problem will arise. The source of this problem is that if both time series have a strong trend, the reason for the high R^2 observed between them is this strong trend relationship, not the linear relationship between the two variables. Therefore, when analysis is made with non-stationary series, it gives misleading results with traditional R^2 and tests (Palachy, S, 2019).

Seasonality: is a phenomenon that predicts that the price is subject to similar and predictable changes in the same period in each calendar year. These changes can be in a particular meteorological season, growing season, quarterly, monthly, holiday or off-peak period. Seasonality often happens in the commodity market. For example, there is a seasonal trend in the demand for heating oil, with prices increasing when demand increases and lower when demand decreases. There is a seasonal trend in soybean supply (related to planting, growing and harvesting). Seasonality can also be found in other markets such as stocks, indices and Forex, and there is usually some underlying reason behind it. Finding seasonal patterns and using them to predict a trend, filter trade ideas, or identify a tradable opportunity can provide advantages to a trader. Please note that the character of each year and therefore seasonality may change. Used alone or in combination with other techniques, seasonality is a useful tool in the technical analyst's toolbox. There are many different kinds of seasons, for instance:

- Moment in Time
- Daily.
- Weekly.
- Monthly.
- Yearly.

Therefore, it is subjective to determine whether your time series problem contains a seasonality component. Plotting and reviewing your data, possibly at various scales and with the addition of trend lines, is the simplest method for determining whether seasonality is present (Brownlee, J, 2016).

Autocorrelation: Autocorrelation, or self-correlation, is the correlation of a signal between its values at different times. In other words, it is the expression of similarity between observed values as a function of time delay. Autocorrelation analysis is a mathematical tool used for purposes such as recognizing repeating patterns, detecting the missing fundamental frequency of a signal. It is frequently used for the analysis of functions or sequences in signal processing. In multiple regression analysis, autocorrelation describes the relationship between successive values of the error term. This is a deviation from an important assumption of the general linear regression model. As a general linear regression model assumption, there is no relationship between the error terms.

ACF/PACF: In the exploratory data analysis of time series forecasting, autocorrelation analysis is an essential step. The autocorrelation analysis aids in pattern recognition and randomness detection. Because it helps determine the parameters of an autoregressive–moving-average (ARMA) model, this is especially crucial if you intend to use it for forecasting. The Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots are examined during the analysis. When assessing a time series, the autocorrelation function (ACF) and the partial autocorrelation function (PACF, also known as partial ACF) are crucial functions. Plots that help determine the values of p, q, and r for Autoregressive (AR) and Moving Average (MA) models are typically produced. The average correlation between data points in a time series and previous values of the series measured for various lag lengths is measured and plotted in an ACF. The only difference between a PACF and an ACF is that each partial correlation takes into account any correlations between observations with shorter lag lengths. Due to the fact that they both measure the correlation between data points at time t and data points at time t-1, the values of an ACF and a PACF at the first lag are identical. However, the PACF measures the same correlation after controlling for the correlation between data points at time t and those at time t-1, whereas the ACF measures the correlation between data points at time t and data points at time t-2 at the second lag(Monigatti, L, 2022).

Returns and Normalization: As intuitive as it may seem to some of you, the term returns represented the percentage change between the values for two consecutive periods. It's important to note that positive returns show a rise in price, while negative returns show a fall. As a result, if investors anticipate positive returns in the future, they would rather hold onto their stocks as their value rises. The process of rescaling the data from the original range to ensure that all values fall within the range of 0 to 1 is known as normalization. When you have time series data with input values that have different scales, normalization can be helpful and even necessary for some machine learning algorithms. Linear regression and artificial neural networks that weight input values, as well as algorithms like k-Nearest neighbors, may necessitate it. You must either know or be able to accurately estimate the minimum and maximum observable values in order to normalize. From the data you have, you might be able to make some guesses about these values. Estimating these expected values may be challenging if your time series is trending upward or downward, and normalization may not be the best approach for your problem(Brownlee, J, 2016).

Residual Analysis: In a time, series model, the "residuals" are the leftovers from fitting the model. The difference between the observations and the residuals is the same for many, but not all, time series models. When determining whether a model has adequately captured the data's information, residuals are helpful. The following properties of a reliable forecasting method will be found in residuals:

- There are no correlated residuals. There should be information in the residuals that can be used to make forecasts if there are correlations between them.
- There is no mean in the residuals. The forecasts are biased if the mean of the residuals is not zero.

Any forecasting technique that does not meet these requirements can be enhanced. However, this does not preclude further development of forecasting techniques that meet these requirements. For the same data set, it is possible to have multiple forecasting methods that all meet these properties. It is important to check these properties to see if a method is making use of all of the information that is available, but this is not a good way to choose a forecasting method. The forecasting method can be altered to provide better forecasts if either of these properties is not met (Hyndman, R, N.D).

Importance of Partition: Splitting data into two or more subsets is known as data splitting. A split with two parts typically serves to train the model while the first part is used to evaluate or test the data. The training data set is used to train and develop models in a basic two-part data split. Estimating various parameters and comparing the performance of various models are two common uses for training sets. After the training is finished, the testing data set is used. To ensure that the final model functions properly, the training and test data are compared. Data is typically divided into three or more sets when using machine learning. The dev set is the third set, and its purpose is to alter the parameters of the learning process. There are various choices when it comes to the ratio between training and testing dataset. Depending on the number of the observation in dataset the choice can be made between 70/30 or 80/20. Particularly in our time series data set in both Bitcoin and Ethereum we will use the partition of 80/20.

AR Models:

Autoregression models, also known as AR models, are normally used to predict ex-post variables (observations whose values we know exactly) at specific moments in time in chronological order. When we want to make a projection, the dependent variable should always be at least a later time than the independent variable. Autoregressive models, as the name suggests, are models that return to themselves. That is, the dependent variable and the explanatory variable are the same except that the dependent variable will be at a later time (t) than the independent variable (t-1). We say chronologically ordered because we are now at time (t). If we go forward one period, we go to (t+1), and if we go back one period, we go to (t-1). When we want to project using autoregression, our attention must be focused on the type of variable, the frequency of its observations, and the time horizon of the projection. They are colloquially known as AR(p), where p is labelled 'order' and is equal to the number of periods we will return to to perform the estimation of our variable. We must take into account that the more periods we go back or the more orders we place on the model, the more potential information will appear in our forecast (W.Palma, 2016).

MA models:

The moving average is an important indicator used as a trend follower and is frequently used in technical analysis. Moving averages show the current direction with a lag, rather than giving a direction on where prices will go. It is delayed as it is an indicator based on past prices. Moving averages are used in most of the forex market indicators. For example, Bollinger bands, moving averages are included in the calculations of indicators such as MACD. A moving average is an indicator calculated by averaging n prices. Moving averages are considered an important indicator for trend tracking. This is because moving averages consist of past price movements. Moving averages also help in identifying support and resistance points. For example, the 200-day moving average moves more slowly than the 20-day moving average and indicates a more lagging forecast. Short-term moving averages are used by short-term traders, while long-term moving averages are used by long-term investors. The 200-day moving average, which is frequently used by traders, is carefully followed as an important signal and support resistance level. In some cases, moving averages with more than one time frame are used together to have an opinion about the direction of the market. The longer the time to look at the moving averages, the greater the lag. For example, looking at the 10-day moving average, the lag is less as it considers more recent prices. There are 3 main types of MA models:

- Simple Moving Average: It is the moving average created by averaging the price movements of a financial product within the specified period. The simple moving average considers the closing prices. For example, The 5-day simple moving average is obtained by adding the 5-day closing prices and dividing by 5.
- Weighted Moving Average: It is the moving average calculated by averaging the price movements of a financial product within the specified period according to the determined weights.
- Exponential Moving Average: It is a moving average calculated by taking the average of the price movements of a financial product within the specified period, giving more weight to the price movements in the recent period. Because of the weighting, the exponential moving average counts as a less lagged moving average.

Moving averages, which are widely used in technical analysis, are more effective when used together. For example, the cross between the 50-day moving average and the 200-day moving average produces a technical analysis signal. Generally, a combination of short-term moving averages and long-term moving averages gives better results. When the short-term moving average crosses the long-term moving average upwards, it signals that price may move upwards in the short term. In the literature, this “golden cross” is known as the “golden cross”. On the contrary, if the short-term moving average cuts the long-term moving average downwards, it generates the signal that prices may move downwards. In the literature, this "death cross" is referred to as "dead cross"(W. Palma, 2016).

ARMA Models:

ARMA models are used for modelling stationary time series and are a combination of AR and MA models. In these models, the observation value for any period of a time series is expressed as a linear combination of a certain number of previous observation values and the error term. If the ARMA model is a combination of the p-term AR and the q-term MA model, it contains p+q terms and is written as ARMA(p,q)(W.Palma, 2016).

ARIMA Models:

Most of the series encountered in practice, especially the economic time series, are not stationary. The stationarity of these series is disturbed by factors such as trends, seasonal and cyclical fluctuations, and random causes. Modelling of non-stationary time series depends on providing stationarity in the series. To ensure stability, these factors must first be identified and then eliminated. If the observation values of a time series are not stationary around the mean value of this series, stationarity is achieved by taking the appropriate differences of the series. The degree of difference is represented by d, and in practice d usually takes the value 1 and at most 2. Models applied to series that are not stationary but converted to stationary by differencing are called integrated models or "non-stationary stochastic models". If the degree of the autoregression parameter is p and the degree of the moving average parameter is q and the difference is made d times, this model is called the (p,d,q) order autoregressive integrated moving average model and is written as ARIMA (p,d,q). ARIMA models, also known as Box and Jenkins, are one of the statistical methods used for predicting the future. The Box-Jenkins (B.J) method is used in the forward estimation of univariate time series. This shows a systematic approach to establishing forward forecast models and making forecasts of discrete and stationary time series consisting of observation values obtained at equal time intervals. The fact that the series consisting of the observation values obtained with equal time intervals is discrete and stationary is B.J. important assumption of the method. The difference between the Box Jenkins estimation method from other estimation methods is that it does not require any prior knowledge about the structure of the time series or the general development trend. In addition, while the use of other methods requires the series to have a certain trend, the Box-Jenkins method can also be applied to complex time series since there is no such restriction in these models. An important advantage of the method is that it uses past observation values as an explanatory variable. Unlike econometric models, the Box Jenkins method does not provide a behavioural explanation for the studied variable, so it does not fit into the theoretical framework. It considers the internal dynamics of the time series.

The Box and Jenkins methodology is summarized in four phases:

• The first phase consists of identifying the possible ARIMA model that follows the series, which requires the Decision on which transformations to apply to convert the observed series into a stationary series. Then determine an ARMA model for the stationary series, that is, the p and q orders of its autoregressive and moving average structure.

• The second phase: After provisionally selecting a model for the stationary series, the second stage of estimation is passed, where the AR and MA parameters of the model are estimated by maximum likelihood and their standard errors and model residuals are obtained.

• The third phase is the diagnosis, where it is verified that the residuals do not have a dependency structure and follow a white noise process. If the residuals show structure, the model is modified to incorporate it and the previous steps are repeated until an adequate model is obtained.

• The fourth phase is the prediction, once an adequate model has been obtained, predictions are made with it (W.Palma, 2016).

ARIMAX:

An Autoregressive Integrated Moving Average with Explanatory Variable (ARIMAX) model is a multiple regression model that includes one or more autoregressive (AR) and/or moving average (MA) terms. This method is appropriate for forecasting when the data is stationary/nonstationary, multivariate, and has any type of data pattern, i.e., level/trend/seasonality/cyclicity. ARIMAX is related to the ARIMA technique, but ARIMA is appropriate for univariate datasets ARIMAX is appropriate for analyses with additional explanatory variables in categorical and/or numeric format (multivariate). This model incorporates exogenous variables, or we use external data in our forecast. Exogenous variables in the real world include the gold price, oil price, outdoor temperature, and exchange rate. It's fascinating to consider that all exogenous factors are still technically modelled indirectly in the historical model forecast. However, if we include external data, the model will respond to its effect much faster than if we rely on the influence of lagging terms (Smarten, 2018).

ARCH and GARCH:

ARCH and GARCH models are one of the most well-known models in time series in order to assess the volatility in the market and even to predict the stability for the future. In the practical part to assess the volatility we will create a squared version of returns which we will refer as the volatility (Torben, G, 2013). A time series' variance can be modelled using an ARCH model, which stands for autoregressive conditionally heteroscedastic. To describe a fluctuating, possibly volatile variance, ARCH models are utilized. Although an ARCH model could be used to describe a gradual increase in variance over time, most of the time, it is used to describe brief periods of increased variation. It may be more effective to transform the variable to deal with the gradually increasing variance associated with the gradually increasing mean level (Engle, R, n.d).

Log likelihood ratio test:

In the final step of every fitted model we will run a log likelihood ratio test in order to decide on which model performs better. The likelihood ratio (LLR) test is a hypothesis test in which two different maximum likelihood estimates of a parameter are compared to determine whether or not to reject a parameter restriction. The likelihood ratio test (LLR) is a statistical test used to compare the goodness-of-fit of two models. A relatively more complex model is compared to a simpler model to see if it significantly better fits a specific dataset. If this is the case, the more complex model's additional parameters are frequently used in subsequent analyses. This test is only useful when comparing hierarchically nested models. That is, the more complex model must only differ from the simple model by one or more parameters. Increasing the number of parameters will always result in a higher likelihood score. However, there comes a point when adding more parameters is no longer justified in terms of significantly improving a model's fit to a specific dataset and the equation is as following (Evomics, N.D):

$$LR = 2*(\ln L1-\ln L2)$$

Equation 4

Furthermore, in order to implement this in Python we will create the following function:

```
def LLR_test(mod_1, mod_2, DF = 1):
    L1 = mod_1.fit().llf
    L2 = mod_2.fit().llf
    LR = (2*(L2-L1))
    p = chi2.sf(LR, DF).round(3)
    return p
```

Code Chunk 1

3 Literature Review

In this section of the thesis, we will describe the entities which are being analyzed and also the comparing indicators. By describing, the detailed information will be delivered to the reader on crypto currencies, blockchain technology stock markets etc. Furthermore, comprehensive knowledge is aimed to be passed regarding the dominance factor of selected crypto coins and their way of working.

3.1 Crypto Currencies

Cryptocurrencies are digital assets that are used as virtual currency and do not exist in any physical form. They are secured by cryptography, which is called encryption, and this prevents the act of “double spending”, which means counterfeiting or making multiple transactions with the same cryptocurrency, which has become almost impossible.

The world's first cryptocurrency was Bitcoin, created in 2008. Bitcoin was followed by other types of cryptocurrencies, with hundreds of variations today. Unlike currencies in the classical sense, cryptocurrencies are not issued by a central authority. This feature is perhaps the most attractive aspect of cryptocurrencies for investors. Because in this way, most cryptocurrencies remain immune from government regulation or manipulation.

The concept of crypto money has been in our lives for many years. For example, we used cryptocurrencies instead of physical banknotes in every transaction we made with debit cards, virtual cards, or over the internet. Transactions were made on digital basis, without physical money transfer between banks. So, from a point of view, cryptocurrencies were also used in these transactions. Because of these transactions, there were only numerical changes in the financial systems. The new generation cryptocurrencies, on the other hand, differ from their ancestors primarily by not physically existing, besides being used primarily in digital transactions. In addition, as we mentioned above, it is different from the previous versions in that they are not subject to the rules of a state or organization and the transactions are made with the consensus of all units in the system. The main reason why it attracts so much attention and love compared to other currencies in the world is that it has a distributed structure. As such, transactions are not carried out under the control of a single authority, but through the control and approval of all users. This feature also ensures that this currency is referred to as more secure.

Cryptocurrencies are created through a process called mining. Individuals with special hardware (hardware) are rewarded by a network with tokens or cryptocurrencies such as Bitcoin in return for their services. In this decentralized competitive process, if too many people try to mine a coin, it will become increasingly difficult to profit with each new addition to the network. This is one of the main reasons why Bitcoin, which can be produced on a limited basis, has increased in value over time with its increasing popularity (Chowdhury, N, 2019).

3.1.1 Blockchain Technology

We are involved in many networks in our lives. Messaging through our social media accounts, sending e-mails, transferring through a bank, or trading stocks in the stock market. All these are the networks we use in our daily lives and some of the transactions we perform on these networks. It is also known that there is an agent that manages the relevant network on all these networks. For any financial transfer, it is necessary to confirm that there is enough in the account of the transferee, and to create records containing the time information of this transfer, the amount sent, the sender and the sending party information. In short, intermediaries who manage the relevant networks are needed to ensure that transactions in all these networks can be carried out smoothly and be recorded, that transactions are verified in case of a problem, and that disputes are resolved. Blockchain technology enables these functions on networks to be performed in a decentralized manner and at lower costs. Blockchain is a database system made up of interconnected blocks. Any information involving a transaction can be processed into this database. New transactions are added on top of the previous block and a new block is created. These blocks are linked chronologically. In this way, the new incoming block also confirms the information in the previous blocks which is increasing the security of the records. Every single input on these blocks is encrypted and therefore has a distributed structure. Due to this, it is almost impossible to change or remove the data, as it would require a person to change the historical records on this network on all the other blocks in the chain. The larger the network, the more separate records it has and that provides more secure to the data in the blockchain. This eliminates the verification and auditing costs mentioned above and provides an accountable and reliable structure. To put it more simply, let's think of blocks on the blockchain as ledgers. Let a copy of these notebooks be distributed to everyone on the network. Each new transaction is recorded in these books simultaneously, so the records of the transactions are kept in many places, not just in one or a few places. In a centralized structure, since there is only one record, it is a significant cost to ensure the security of these records exist and it is reliable. However, as it is not possible to change all records in the decentralized blockchain structure, a more secure system is formed. In a centralized system, the privacy risk created by the data that the intermediaries need to access while transacting is another disadvantage of this structure. Transactions made through a broker often need to be shared because of the verification process. This increases the possibility of using the data outside of its real purpose in the network. In addition, the security of the data held on the intermediary institution that manages the network creates a different problem and cost. In Blockchain, such information leaks are prevented as users can verify without sharing information with another person or institution. Blockchain technology has the potential to change the processes performed over networks in many different sectors and fields such as finance, health, science, and industry in the future. This potential excites all institutions in the relevant sectors and investments in blockchain technology are increasing day by day for these reasons (Chowdhury, N, 2019). There are numerous methods for constructing a blockchain network. They can be public, private, permissioned, or built by a consortium of individuals.

Public blockchain networks are permissionless networks and allow anyone to join. All members of the blockchain have equal rights to read, edit and verify the blockchain. Common blockchain networks are mainly used for trading and mining cryptocurrencies such as Bitcoin, Ethereum and Litecoin.

Private blockchains, also referred to as managed blockchains, are controlled by a single entity. This authority decides who can become a member and what rights the members have in the network. Private blockchains are only partially decentralized because they contain access restrictions. Ripple, a digital currency exchange network for businesses, is an example of a private blockchain.

Hybrid blockchains combine some features of both private and public networks. Companies can set up private, permission-based systems as well as a common system. Thus, they control access to certain data stored on the blockchain while keeping the rest of the data public. They use smart contracts to allow members of the common system to check whether private transactions have been completed. For example, hybrid blockchains can allow shared access to digital currency, while keeping bank-owned currency private. Consortium blockchain networks are managed by a group of organizations. Pre-selected organizations share responsibility for maintaining the continuity of the blockchain and determining data access rights. Consortium blockchain networks are generally preferred in sectors where many organizations have a common goal and can benefit from responsibility sharing. For example, the Global Shipping Business Network Consortium is a non-profit blockchain consortium that aims to digitize the shipping industry and increase collaboration among organizations in the shipping industry (Parizo, 2021).

3.1.2 Bitcoin (BTC)

After the 2008 Mortgage crisis, Satoshi Nakamoto published a technical paper (Whitepaper) on Bitcoin, an end-to-end electronic payment system. With the whitepaper, Bitcoin, which has a decentralized and transparent structure, emerged as a cryptocurrency. The Bitcoin blockchain was started to be used with the first transfer made in January 2009 and was named "1st generation blockchain" with the popularity it gained in a short time. Thanks to its distributed, decentralized, and transparent structure, Bitcoin has risen against today's financial order in a very short time. With the increase in the use of Bitcoin, the limited supply, and the technology it brings, it has been adopted by many investors and financial institutions. Bitcoin has enabled the development of many leading sectors and technologies with its pioneering nature and technology in the crypto currency world. The fact that the Bitcoin blockchain structure is transparent and its supply is limited, in addition to the technological revolution brought by Bitcoin, has caused it to be seen as an investment with low inflation and high potential for many investors. After its birth, Bitcoin caused the birth of many different cryptocurrencies due to its inability to provide sufficient capacity in terms of both speed and scalability. These cryptocurrencies are called "alternative coins", in other words "altcoins". These cryptocurrencies, which are developed in a similar or different structure with the Bitcoin blockchain, can be programmable and have a faster structure. While creating alternative cryptocurrencies, competitive advantage has been taken advantage of by having different features at various points and new crypto money types have emerged. The main differences between these cryptocurrencies are the maximum amount of supply that can be produced in general, the algorithms used and the types of blockchains (private/shared, permissioned/unauthorized consensus) are examples. As all revolutionary technologies, there are Bitcoin predecessors and sources that its creator refers to in his article. Wei Dai B-Cash, Nick Szabo BitGold, David Chaum Digicash are the most primitive and government-blocked versions of digital currencies. Satoshi Nakamoto managed to keep his identity secret because of these negative experiences of his predecessors. There is controversy over whether it is his real name, pseudonym, or a team title. The maximum number of Bitcoins that can be produced is limited to 21 million by specifying in the genesis block. The first Bitcoin transfer was between Satoshi Nakamoto, cryptographer Hal Finney, who helped him develop it. The first purchase was made on May 22, 2010, with the purchase of a pizza for 10,000 Bitcoins. As of July 22, 2013, the total value of Bitcoins in circulation was already 1.2 billion dollars as of 26.07.2020 this value was 182,967,290 dollars. The total amount of Bitcoin currently produced has reached the level of 19,209,775 according to Binance which is one of the most widely used cryptocurrency trade platforms.

Bitcoins, which are not produced from any centre, show a point-to-point distributed network feature similar to Bittorent networks. Payments made in this network reach other points instantly, so that the payment from which address to which address is recorded. Thus, the collected records are located in structures called blocks. By applying a hash algorithm that requires high processing on each block, it is desired to find the expression that starts with a certain number of zeros. The first user to perform this transaction, which corresponds to approximately every 10 minutes, is rewarded from zero to 50 BTC (currently 12.5 BTC). Thus, Bitcoins are driven to emission. Each block contains the hash expression of the last block before it. This creates a blockchain that is very hard to break (except for the 51% attack). The aim is to avoid double spending and to keep records of submissions. The process of creating the coin is called mining. Mining is the general name of the process of performing mathematical operations using computational power. To make these transactions, the nodes in the bitcoin network that download the offered bitcoin software and perform operations that require intensive processing power on their hardware (usually video cards) are called "miners".

The first block of the system was named "genesis block" and was produced on January 4, 2009. As such, the first transaction in the block is a private transaction and is initiated by the creator of the new money block. This is an incentive system for miners to participate in the network, so that money can enter the system distributed as desired, which does not have a central authority to print the money. In this way, miners make a profit both by generating and driving new bitcoins into the system, and by receiving bitcoins from the system in exchange for services to perform pending transactions. The regular addition of new money to the system is likened to gold miners finding gold and putting it into circulation, hence the name mining. In the current process, miners continue to produce the amount of bitcoin that will come into circulation each year at a decreasing and predictable rate. In the system, production will continue until a total of 21 million bitcoins are in circulation, then the production process will stop, and miners will continue to be supported only at transaction costs.

Bitcoin is used as a payment and investment tool in some countries. Bitcoins have a value because they can be used like money, and some funds are also known to be interested in this product with the expectation that its value will increase as its popularity grows in the future. The value of Bitcoin is determined by the supply and demand conditions in the market. When the demand increases, the price increases and the price decreases. There is a limited number of bitcoins in circulation and there is a limit and procedure for generating new bitcoins. The biggest threats to Bitcoin's market value are technical difficulties, legislative changes due to the approach of countries to this money, and the negative change in people's desire and trust in this money.

Besides its advantages, Bitcoin also has a few disadvantages. The Disadvantages of Bitcoin are as follows:

- High price volatility and high risk to invest
- Bitcoin transaction speed and capacity remain quite low compared to its competitors
- The high energy usage required to run the Bitcoin blockchain
- For these reasons, it is very important for people who want to trade Bitcoin to act by considering Bitcoin Advantages and Bitcoin Disadvantages.

Although there are many cryptocurrencies in the market, no cryptocurrency has managed to surpass Bitcoin in terms of market dominance. Bitcoin Market Dominance or Bitcoin Dominance is a data that expresses the ratio of the market value of Bitcoin to the overall value of the cryptocurrency market. When historical data are examined, it is seen that the dominance of Bitcoin, which was over 96 percent in 2013, decreased to 32 percent in 2018. Underlying this major decline is the increase in transaction volumes of other cryptocurrencies, especially Ethereum. One of the main reasons for the rise of Bitcoin Market Dominance can be shown as the rise in Bitcoin price, increasing the transaction volume by attracting more users and, accordingly, more demand than other cryptocurrencies. It would not be wrong to say that the cryptocurrency market is largely driven by Bitcoin. The increase in demand leading to increased Bitcoin dominance may also be a sign of users moving away from low-volume and relatively riskier cryptocurrencies and switching to Bitcoin. Being the first cryptocurrency enables Bitcoin to shape the cryptocurrency industry. Although the number of cryptocurrencies in the market is very large, there has not yet been a cryptocurrency that can compete with Bitcoin's weight in the market (Grabowski, M, 2019)

3.2 Ethereum and its system

Ethereum is a system that was first introduced at the North American Bitcoin Conference by Ethereum founder Vitalik Buterin. Although it is generally seen as an altcoin, Ethereum is an innovative system that aims to develop blockchain technology and use it in more areas. After the Ethereum development process, it was released in July 2015 and quickly gained popularity. Ethereum official website can be visited via ethereum.org/tr/ link.

In 2016, due to a software bug, hackers stole approximately $50 million (3.6 million ETH) from the DAO (Decentralized Autonomous Organization), a smart contract-operated venture fund. After this hack, the Ethereum blockchain was hard forked, rewinding the hack and moving on. In this process, the old chain continued its life as Ethereum Classic. In 2017, the ERC-20 standard was created on the Ethereum blockchain, making it easier for developers to develop tokens compatible with applications. In 2017, MakerDao, the first decentralized finance application on Ethereum, launched and launched the DAI stable cryptocurrency. Also in the same year, ETH exceeded the level of $100 for the first time.

In 2018, platforms such as Compound and Uniswap were launched in the field of decentralized finance. ETH broke above $1000 for the first time in January 2018, then fell back below $100.

With the popularity of DeFi in 2020, many applications have emerged on Ethereum. Ethereum has announced that it will launch the Beacon chain for ETH 2.0 migration in 2020. With Ethereum 2.0, it was planned to switch from the Proof of Work algorithm to the Proof of Stake algorithm. With the rise in the market after the Covid-19 epidemic in 2020, Ethereum rose to the level of 4500 dollars in 2021.

Ethereum is a blockchain project that emerged after Bitcoin and allows creating smart contracts on the blockchain. With the announcement of the Ethereum project in 2014, it developed rapidly and allowed the creation of many existing sectors and new tokens in the crypto money world. The Ethereum project has encountered many difficulties in the process and has allowed many new areas to be born. With the ability to create smart contracts on the Ethereum blockchain, it started the decentralized finance trend and helped this field become a very large industry. The Ethereum project first started to work with the Proof of Work algorithm, then switched to the Proof of Stake algorithm in September 2022. This process, called Ethereum Merge, is the beginning of the steps taken to make the Ethereum project more scalable, fast, cheap and decentralized. Ethereum aims to provide a faster and cheaper experience to its users with a more scalable structure. In addition to all these goals, unlike other projects, Ethereum aims to create this structure in a decentralized way. Ethereum was developed by Vitalik Buterin in 2014 and has been supported by many names such as Mihai Alisie, Anthony Di Iorio, Charles Hoskinson and Gawin Wood in the following period. After the development phase, the non-profit Ethereum Foundation was established to ensure the functionality of the blockchain. Initial capital investments in the Ethereum project were made through online bookkeeping in July 2014. In this demand collection, Ethereum purchases were made by barter with Bitcoin. Ethereum operates in a worldwide distributed manner, thanks to users participating as “nodes” instead of a central server, as in the Bitcoin network. This way of working makes the blockchain network decentralized and highly resistant to attacks. Thus, if a node on the Ethereum blockchain does not work, other nodes on the network are able to keep the system alive (Grabowski, M.2019, p.42-68). Ethereum is basically a decentralized system that runs a computer called the Ethereum Virtual Machine (EVM). Each validator that creates a node on the Ethereum blockchain contributes to decentralization while keeping a copy of every transaction on the network. These copies, which are kept by different people, can be updated after each block and synchronize with each other. Actions performed on the network are considered "transactions" and are stored in blocks on the Ethereum blockchain. Verifiers check the transaction history and records for accuracy before connecting these blocks to the network. The Proof of Stake algorithm is used to maintain consensus on transaction accuracy on the blockchain. With this algorithm, min 32 ETH must be locked to become a validator node. After the locked 32 ETH, it is synchronized with the network and blocks can be added to the blockchain. Ethereum has a transparent blockchain structure just like Bitcoin. All transactions on the network can be viewed and examined by third parties (etherscan.io). However, despite all this transparency, the Ethereum blockchain also provides anonymity for users. To be able to transact on the Ethereum blockchain, you must have some ETH in your wallet. Each transaction that takes place on the network is realized with some transaction fee according to the current supply, demand and density balance. Each transaction comes with a fee called "gas" which is paid by the user who initiated that transaction. Gas essentially acts as a limit, restricting the number of actions a user can take per transaction. It also has a very deterrent function to prevent gas fee network spam. ETH does not have a limited supply like other cryptocurrencies. The supply of ETH increases annually according to a certain inflation rate. This rate is inversely proportional to the ETHs that are currently locked to be validators. In the Pos mechanism, new ETHs will circulate in the form of staking rewards. However, ETH can take on a deflationary structure in some cases, as a part of each transaction fee is burned according to the network usage on Ethereum. Ethereum transaction fees can be quite high depending on network activity. This is because the total gas capacity of a block is limited. As a result, users who want to perform their transactions quickly can perform their transactions faster by paying a higher gas fee. If this is done by many users, transaction fees generally rise on the network to reduce demand. By shaping the transaction fees according to the demand, the network can work more healthily and for a long time. ERC-20 is a standard format used to create tokens on the Ethereum blockchain. The ERC-20 standard, proposed by Ethereum developer Fabian Vogelsteller in 2015, describes the basis of many sets of rules, such as how a token will work in the Ethereum ecosystem, how much it will supply. In simpler terms, ERC-20 is defined as a standard in which the basic rules are determined for creating tokens with the same characteristics on the Ethereum blockchain. Ethereum hosts many tokens on its own blockchain with the ERC-20 standard. Some popular Ethereum-based altcoins are mentioned below to show the importance and usage of ERC-20((Reiff, N, 2022):

Tether USD (USDT)
USD Coin (USDC)
Shiba Inu (SHIB)
Binance USD (BUSD)
BNB (BNB)
DAI Stablecoin (DAI)
HEX (HEX)
Bitfinex LEO (LEO)
MAKER (MKR)

4 Practical Part

In the practical part we will implement the methods which have been discussed in methodology section. The aim will be to apply the time series models to our data with splitting it into training and testing.

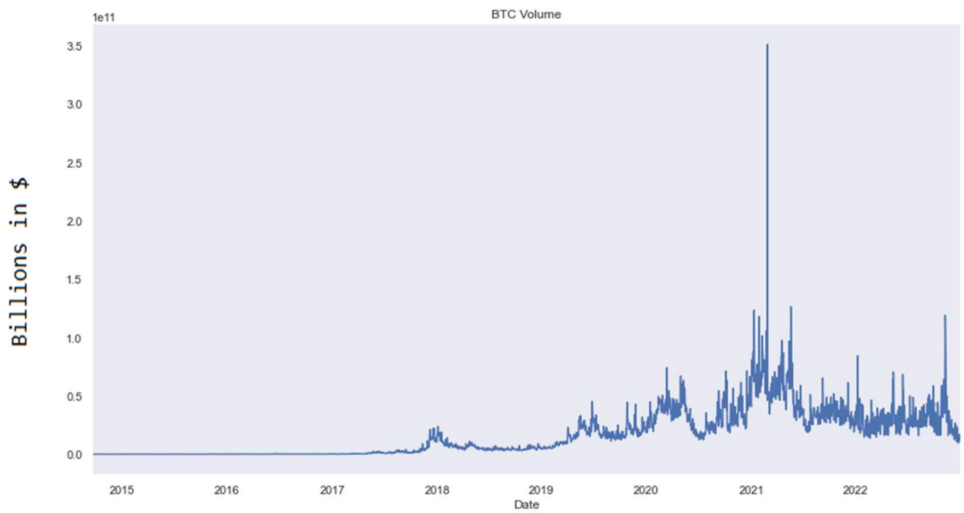
4.1 Analysis of BTC

In this section we will analyse the biggest crypto currency BTC since its first signed day in Yahoo finance taking into consideration that we have already separated the data into training and testing dataset in the ratio of 80/20. To start the analysis firstly let's have a look on the price plot of BTC since the beginning:

Graph 1

Looking to the plot, it seems there was a significant expansion of BTC in the beginning of 2021 which is following a sharp decrease towards to the middle of the same year and increase again at the end of the period. Furthermore, 2022 also shows a significant fall in the BTC prices.

Next, we can have a look to the BTC Market capitalization:

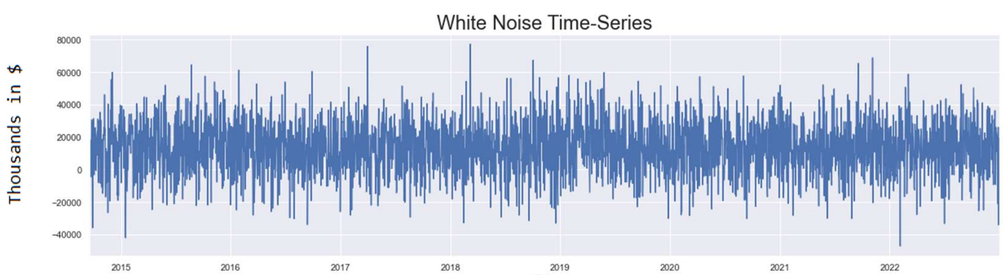


Graph 2

Looking to the above plot we can detect that the Market volume of BTC was relatively stable until the end of 2019 and fluctuating since then until now. Both plots for BTC market volume and BTC market prices can be signs of high volatility which we will investigate in further sections of the thesis.

4.1.1 White Noise, Stationarity and Seasonality

In first step, we will try to examine the white noise of our dataset. To do that in python we have set the dates as our indexes, configured frequency as days.



Graph 3

The white noise here is telling us if our data is predictable or not. As we know, one of the conditions of White Noise is its mean to be 0. To prove that our data is not white noise we can have a look on the mean which is 13385 and based on this we can clearly state that the data is not white noise.

```
print(wn.mean())
```

```
13384.990595423738
```

Figure 1 (thousands in $)

In the next step we will assess the stationarity of our dataset with dicky fuller test and we will set our Null hypothesis as “The data is stationary”:

```
sts.adfuller(btcdata.Close)
```

```
(-1.6366271507566819,
 0.4640237426676581,
 29,
 2998,
 {'1%': -3.4325330913621452,
  '5%': -2.862504548608965,
  '10%': -2.5672834546224057},
 48504.776278974656)
```

Figure 2

1st line test statistic and 5th 6th 7th are the respective critical values. As our test statistic is greater than all the critical values, we do not have enough evidence for stationarity.
2nd line is p-value which states that there is 46 percent chance of not accepting the Null Hypothesis so we cannot confirm that the data is stationary. So, we reject the Null hypothesis.
3rd line shows the number of lags the utilized in the regression when determining the T statistic. As we have 29 it means there is some autocorrelation going back 29 periods.

In the next step we will check the seasonality of the dataset with additive and multiplicative decompositions.

Additive:

Graph 4

When we check the seasonal part of the above plot, we can see that it is a rectangle. This happens when the values are constantly oscillating back and forth and the figure sizes too small. In our case, the linear change results from constantly switching up and down between negative -20 and 20 one for every period. Therefore, there is no concrete cyclical pattern determined by using naive decomposition. And finally, residuals are indicating the difference between the predicted values and actual values. From the plot we see that the residuals are quite high in 2018, 2021 and 2022.

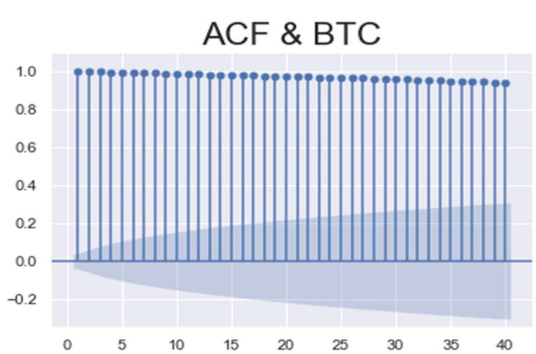
Multiplicative:

Graph 5

Checking the multiplicative decomposition method to be sure on seasonality we can see the similar results which states that there is no seasonal cycle in our dataset.

4.1.2 Auto Correlation Function and Partial Correlation Function

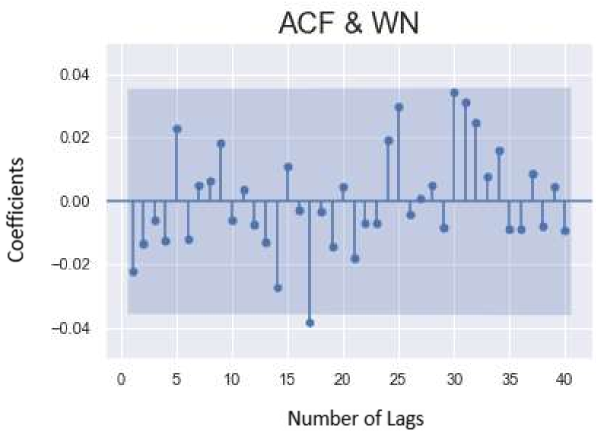
Next, we will examine the autocorrelation coefficients for BTC prices. To do so we have again used Python for visualization where we have set the period daily.



Graph 6

As we can see from the plot at the top all the coefficients are higher than the given significance level which is the area in blue figure. As each lag shows how the prices differ from each other one period ago we can state that there is an autocorrelation between lags. In simple words, it means that prices one period ago can still assist us in forecasting the future prices.

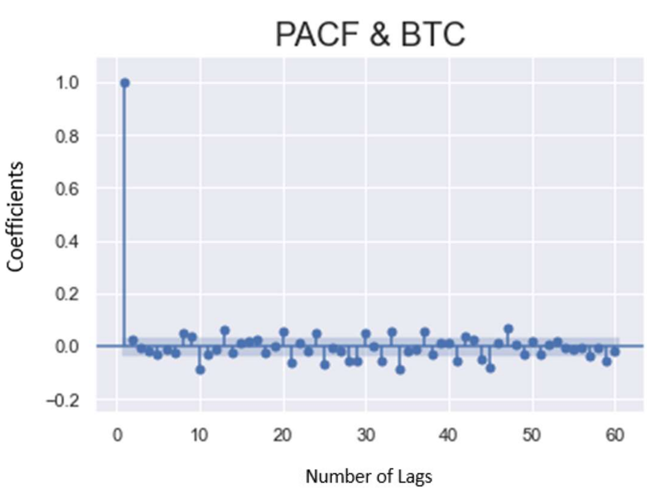
Furthermore, as a second level confirmation of being our data white noise, we can plot ACF of our white noise data:



Graph 7

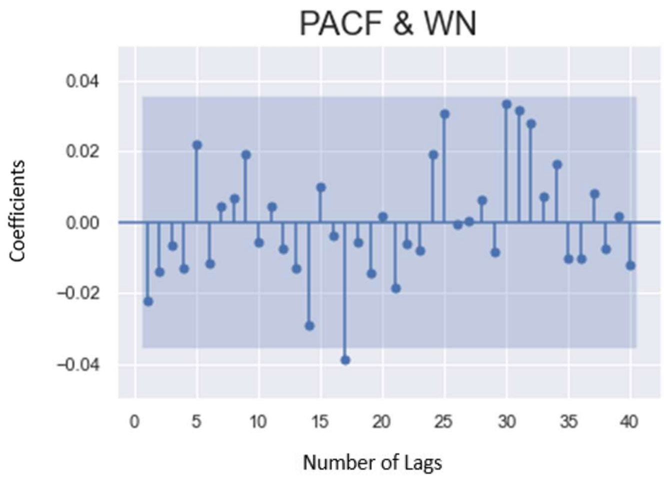
Here we can clearly see that almost all the lags are in within significance level (blue figure) which enables us to easily confirm that there is no auto correlation in white noise data which is indeed one of the assumptions for White-Noise.

In this step we will also analyse PACF for BTC close prices using order least squared method in Python.



Graph 8

As PACF shows direct effects of the prices from past period we can notice a completely different plot than auto correlation function. Looking at the plot we can also notice positive and negative values which is somewhat random without any lasting effects.
Moreover, we can examine the PACF graph for our white noise data:



Graph 9

Again, from the plot above although there is one lag which is out of significance level, we can claim that it is completely random. Taking this into consideration we can prove one more time that WN data has no autocorrelation.

4.1.3 AR model

In this section we will start our model building process where we will use Auto regression models

First, in order to choose the number of lags that we will use in AR model it is important to examine ACF&PACF and derive the result from there. As we have already plotted our graphs in univariate, we can go through them again. As per the ACF graph, the more lags we include, the better our model will fit to our data set however this can create an overfitting problem which might cause incorrect predictions of the future prices. As per the PACF plot we can remember that there are existing negative and positive coefficients and some coefficients which are not in significance level. We can see that, after 48^{th} lag the coefficients are more likely to be significant and that is the reason, we should have less than 48 lags in our AR model.

In the next step we are starting to implement AR model with 1 lag and then going forward slowly to detect the best model. Also, we will use the log likelihood test to compare the models with different lags in order to define the best one. Here we define the Null hypothesis as the second (more complex model) does not perform better than the first one:

AR model with 4 lags:

Dep. Variable:	Close	**No. Observations:**	2418
Model:	ARIMA(4, 0, 0)	**Log Likelihood**	-18683.763
Date:	Tue, 31 Jan 2023	**AIC**	37379.527
Time:	07:48:13	**BIC**	37414.271
Sample:	09-21-2014	**HQIC**	37392.162
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	7178.9367	6.904	1039.814	0.000	7165.405	7192.468
ar.L1	1.0116	0.008	124.820	0.000	0.996	1.027
ar.L2	0.0141	0.011	1.332	0.183	-0.007	0.035
ar.L3	0.0395	0.012	3.409	0.001	0.017	0.062
ar.L4	-0.0654	0.008	-8.575	0.000	-0.080	-0.050
sigma2	3.041e+05	2016.363	150.792	0.000	3e+05	3.08e+05

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	143806.34
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	4083.98	**Skew:**	1.28
Prob(H) (two-sided):	0.00	**Kurtosis:**	40.69

Table 1

AR model with 3 lags:

Dep. Variable:	Close	**No. Observations:**	2418
Model:	ARIMA(3, 0, 0)	**Log Likelihood**	-18688.728
Date:	Tue, 31 Jan 2023	**AIC**	37387.457
Time:	07:48:14	**BIC**	37416.410
Sample:	09-21-2014	**HQIC**	37397.986
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	7178.9363	5.791	1239.567	0.000	7167.585	7190.287
ar.L1	1.0128	0.008	125.997	0.000	0.997	1.029
ar.L2	0.0139	0.010	1.330	0.184	-0.007	0.034
ar.L3	-0.0270	0.007	-3.758	0.000	-0.041	-0.013
sigma2	3.02e+05	1981.120	152.450	0.000	2.98e+05	3.06e+05

Ljung-Box (L1) (Q):	0.02	**Jarque-Bera (JB):**	144742.42
Prob(Q):	0.89	**Prob(JB):**	0.00
Heteroskedasticity (H):	4293.11	**Skew:**	1.38
Prob(H) (two-sided):	0.00	**Kurtosis:**	40.80

Table 2

Log likelihood test for models with 1 and 4 lags (for visibility as 1 lag model performed well as well):

```
LLR_test(ar_model, ar_model_1)
```

```
0.0
```

Figure 3

After several trials until 7th lag it is obvious that we should stop at 4th lag based on the LLR test results. Although, we have one value which is insignificant in AR model with 4 lags we log likelihood test provides greater log likelihood in model 4 and that is the reason we will make our decision on 4 lags. So, we reject the Null hypothesis. Furthermore, after testing the log likelihood test between 1 lag and 4 lag we see that adding 4 more lags does not affect the model in a negative way.

Next, as it is more reliable to use stationary data in AR models which are having constant mean, variance, and autocorrelation we will try to use returns which is a percentage representation of the changes in prices. To obtain that, we have created a new column in both training and test data set and used some python methods to calculate the percentages:

```
btc_train['returns'] = btc_train.Close.pct_change(1).mul(100)
btc_test['returns'] = btc_test.Close.pct_change(1).mul(100)
btc_train = btc_train.iloc[1:]
```

Code Chunk 2

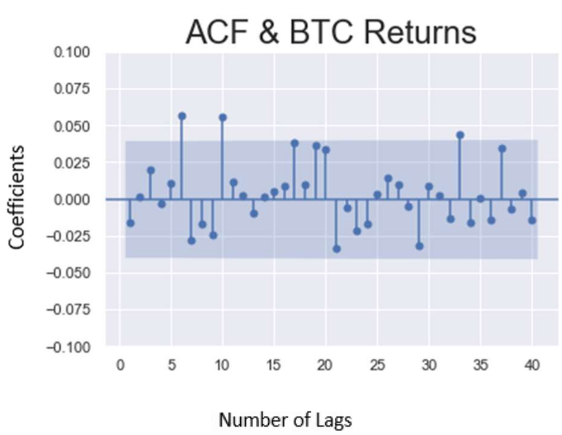
To test the stationarity, we will use the Dickey-Fuller test for training data set where the Null hypothesis is that the data is stationary:

```
(-14.87779616981684,
 1.626558183779799e-27,
 9,
 2410,
 {'1%': -3.4330662982661715,
  '5%': -2.8627400264482548,
  '10%': -2.5674088238838864},
 13274.840659930018)
```

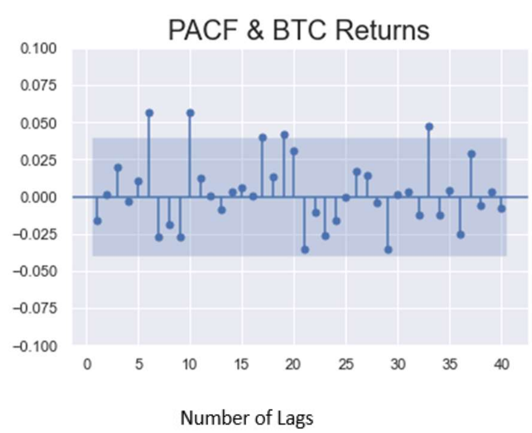
Figure 4

From the above we can see that our test statistic -14 is smaller than all the three critical values in different confidence levels. Because of that we can state that the data that we have is stationary now.

Now we can go ahead and examine ACF and PACF respectively for the return values:



Graph 10



Graph 11

In both autocorrelation and partial autocorrelation suggests indicates us that there are some coefficients which are positive, negative, also within confidence level and some outside of confidence level which allows us to state that the data has no autocorrelation.

As we have fitted the models for close prices of BTC we will follow the similar approach in returns. We will try to fit several models with different lags and apply log likelihood test in order to assess the best model fit. Here we define the Null hypothesis as the second (more complex model) does not perform better than the first one:

AR model with 7 lags

Dep. Variable:	returns	No. Observations:	2417
Model:	ARIMA(7, 0, 0)	Log Likelihood	-6699.913
Date:	Tue, 31 Jan 2023	AIC	13417.826
Time:	14:37:51	BIC	13469.939
Sample:	09-22-2014	HQIC	13436.778
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2787	0.083	3.340	0.001	0.115	0.442
ar.L1	-0.0144	0.013	-1.117	0.264	-0.040	0.011
ar.L2	0.0015	0.016	0.095	0.924	-0.030	0.033
ar.L3	0.0176	0.017	1.058	0.290	-0.015	0.050
ar.L4	-0.0008	0.016	-0.051	0.959	-0.032	0.030
ar.L5	0.0112	0.016	0.684	0.494	-0.021	0.043
ar.L6	0.0561	0.016	3.465	0.001	0.024	0.088
ar.L7	-0.0275	0.015	-1.872	0.061	-0.056	0.001
sigma2	14.9702	0.207	72.344	0.000	14.565	15.376

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	6308.91
Prob(Q):	0.98	Prob(JB):	0.00
Heteroskedasticity (H):	1.51	Skew:	-0.16
Prob(H) (two-sided):	0.00	Kurtosis:	10.91

Table 3

AR model with 1 lag:

Dep. Variable:	returns	No. Observations:	2417
Model:	ARIMA(1, 0, 0)	Log Likelihood	-6705.236
Date:	Tue, 31 Jan 2023	AIC	13416.471
Time:	14:32:42	BIC	13433.842
Sample:	09-22-2014	HQIC	13422.789
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2787	0.078	3.568	0.000	0.126	0.432
ar.L1	-0.0153	0.013	-1.227	0.220	-0.040	0.009
sigma2	15.0368	0.194	77.352	0.000	14.656	15.418

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	6613.33
Prob(Q):	1.00	Prob(JB):	0.00
Heteroskedasticity (H):	1.50	Skew:	-0.16
Prob(H) (two-sided):	0.00	Kurtosis:	11.10

Table 4

```
LLR_test(ar_model_ret_1, ar_model_ret_2)
0.001
```

Figure 5

Based on our trials we can see that second AR model with 7 lags have greater log likelihood and as LLR test value is less than 0.01, we can indeed state that the second model is better than the first model with 1 lag and reject the Null hypothesis.

In the next step, we can also test if normalized prices would result in stationary data that we can use in our AR models in python:

```
benchmark = btc_train.Close.iloc[0]
```

```
btc_train['norm'] = btc_train.Close.div(benchmark).mul(100)
btc_test['norm'] = btc_test.Close.div(benchmark).mul(100)
```

Code Chunk 3

As usual, let's try to assess the stationarity again where we define claim the Null hypothesis as the data is stationary:

```
sts.adfuller(btc_train.norm)
```

```
(2.7238760327780485,
 0.9990879052948666,
 27,
 2392,
 {'1%': -3.4330867606360274,
  '5%': -2.862749062318083,
  '10%': -2.5674136347538057},
 30276.790079294624)
```

Figure 6

Noticing the fact that the test statistic is greater than our critical values we can state that the data is non-stationary (rejecting Null hypothesis) thus we will not use normalized prices for our AR model.

However, when we normalize returns, we are able obtain a stationary data. That is the reason we will implement the similar approach to normalized returns and use Dicky-Fuller test to check the stationarity where we define claim the Null hypothesis as the data is stationary:

```
benchmark_ret = btc_train.returns.iloc[0]
btc_train['norm_ret'] = btc_train.returns.div(benchmark_ret).mul(100)
```

```
sts.adfuller(btc_train.norm_ret)
```

```
(-14.87779616981684,
 1.626558183779799e-27,
 9,
 2410,
 {'1%': -3.4330662982661715,
  '5%': -2.8627400264482548,
  '10%': -2.5674088238838864},
 26007.506934147772)
```

Figure 7

As our test statistic falls to the left of our critical values, we can easily complete our argument with stating the data is stationary. So we accept the Null hypothesis.

As we have done for prices and returns, we will try AR models with different lags and use log likelihood test to compare our models. Here we define the Null hypothesis as the second (more complex model) does not perform better than the first one:

Dep. Variable:	norm_ret	No. Observations:	2418
Model:	ARIMA(6, 0, 0)	Log Likelihood	-13138.849
Date:	Tue, 31 Jan 2023	AIC	26293.697
Time:	15:30:23	BIC	26340.023
Sample:	09-21-2014	HQIC	26310.544
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.9744	1.228	-3.238	0.001	-6.380	-1.568
ar.L1	-0.0159	0.013	-1.251	0.211	-0.041	0.009
ar.L2	0.0006	0.016	0.038	0.970	-0.031	0.032
ar.L3	0.0178	0.017	1.073	0.283	-0.015	0.050
ar.L4	-0.0011	0.016	-0.068	0.946	-0.032	0.030
ar.L5	0.0114	0.016	0.695	0.487	-0.021	0.043
ar.L6	0.0567	0.016	3.539	0.000	0.025	0.088
sigma2	3070.1127	41.484	74.006	0.000	2988.805	3151.421

Ljung-Box (L1) (Q):	0.01	Jarque-Bera (JB):	6448.55
Prob(Q):	0.94	Prob(JB):	0.00
Heteroskedasticity (H):	1.52	Skew:	0.16
Prob(H) (two-sided):	0.00	Kurtosis:	10.99

Table 5

Dep. Variable:	norm_ret	No. Observations:	2418
Model:	ARIMA(10, 0, 0)	Log Likelihood	-13132.818
Date:	Tue, 31 Jan 2023	AIC	26289.635
Time:	15:30:31	BIC	26359.124
Sample:	09-21-2014	HQIC	26314.906
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.9745	1.220	-3.257	0.001	-6.366	-1.583
ar.L1	-0.0138	0.013	-1.070	0.285	-0.039	0.011
ar.L2	0.0021	0.016	0.128	0.898	-0.030	0.034
ar.L3	0.0213	0.017	1.270	0.204	-0.012	0.054
ar.L4	-0.0035	0.016	-0.220	0.826	-0.034	0.027
ar.L5	0.0109	0.017	0.656	0.512	-0.022	0.044
ar.L6	0.0569	0.016	3.512	0.000	0.025	0.089
ar.L7	-0.0283	0.015	-1.926	0.054	-0.057	0.001
ar.L8	-0.0188	0.018	-1.031	0.303	-0.054	0.017
ar.L9	-0.0276	0.018	-1.540	0.124	-0.063	0.008
ar.L10	0.0561	0.017	3.240	0.001	0.022	0.090
sigma2	3055.6268	42.201	72.406	0.000	2972.914	3138.340

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	6365.20
Prob(Q):	0.97	Prob(JB):	0.00
Heteroskedasticity (H):	1.51	Skew:	0.16
Prob(H) (two-sided):	0.00	Kurtosis:	10.94

Table 6

```
LLR_test(ar_model_norm_ret_1, ar_model_norm_ret_2)
0.001
```

Figure 8

From the above results we can see that the model which has 10 lags have higher log likelihood and also LLR test suggests that there is no significant impact adding extra 4 more lags to the model with 6 lags. So we reject Null hyptothesis and accept that the more complex model performs better than the simpler one. However, comparing our results to the unnormalized returns that we did earlier we can see that normalizing does not have any significant impact on model selection.

In the last section of AR models we will examine the residuals for the created models. In order to do so firstly we create the residual columns in the datasets:

```
btc_train['res_price'] = results_ar_model_1.resid
btc_test['res_price'] = results_ar_model_1.resid
```

```
btc_train.res_price.mean()
```

```
16.652310445018966
```

```
btc_train.res_price.var()
```

```
319367.01910491264
```

Code Chunk 4

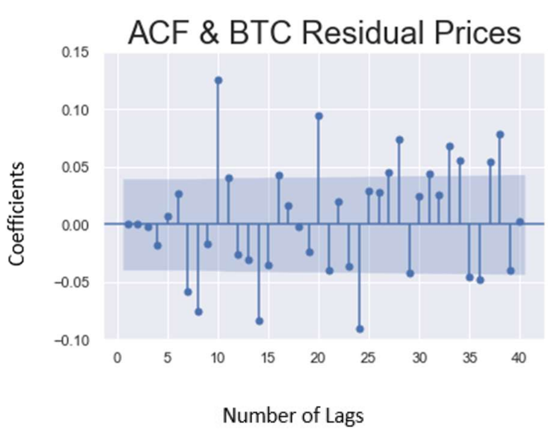
Next, we will check teh stationarity where we define claim the Null hypothesis as the data is stationary::

```
sts.adfuller(btc_train.res_price[1:])
```

```
(-8.330642483276229,
 3.3759266971884524e-13,
 27,
 2389,
 {'1%': -3.433090201041693,
  '5%': -2.862750581542575,
  '10%': -2.567414443618994},
 36799.08153158113)
```

Figure 9

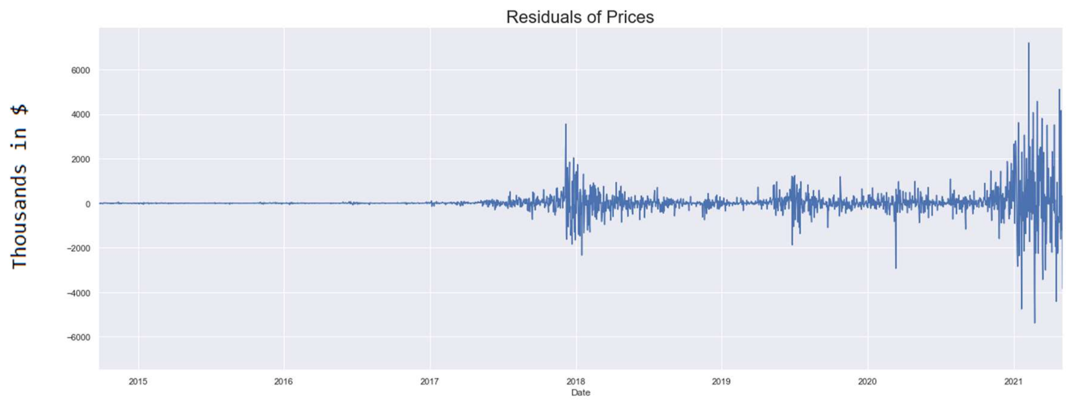
Seeing test statistic falling on the left side of our critical values we can state that the data is stationary(accepting Null hyptohesis). Now we can check the ACF for our data in order to analyze residuals



Graph 12

From the above plot we can see many coefficients which are outside of confidence level(blue region) which makes us to believe that there is a better predictor than residuals.

Finally, we must plot the residual numbers to determine whether they match what we are accustomed to anticipating from white noise data with ordinary plot function in python.



Graph 13

When we compare the above graph to actual prices which we initally plotted for BTC prices we can see the correct patterns. This is another indicator that our model is correct.

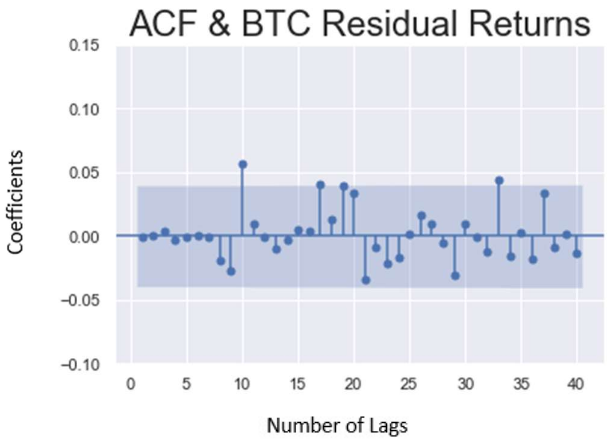
Finally, we will analyze residuals of returns in the same way as we did for prices where we define claim the Null hypothesis as the data is stationary:

```
btc_train['res_price_ret'] = results_ar_model_ret_2.resid
btc_test['res_price_ret'] = results_ar_model_ret_2.resid

btc_train.res_price_ret.mean()

-8.325419120685814e-05

btc_train.res_price_ret.var()

14.97719249322223

sts.adfuller(btc_train.res_price_ret[1:])

(-49.12376273495132,
 0.0,
 0,
 2416,
 {'1%': -3.43305954530467,
  '5%': -2.862737044430077,
  '10%': -2.5674072362026337},
 13253.803837517575)
```

Figure 10

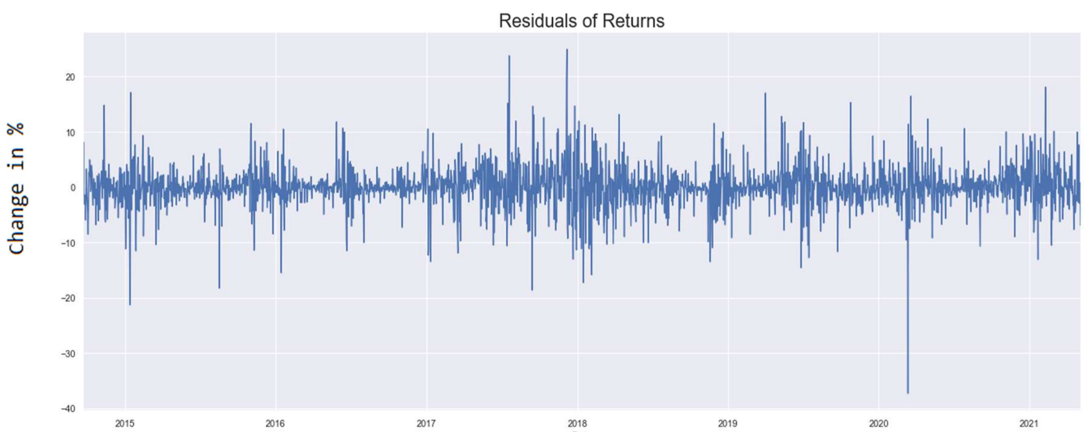
We can again see that data is stationary based on the dicky fuller test statistic and p-value which means we accept the Null hyptothesis.

While examining the ACF we see less coefficients which are outside of the confidence level which means our model is a good predictor but we still have a steady evidence that there is a better one which is in existence.



Graph 14

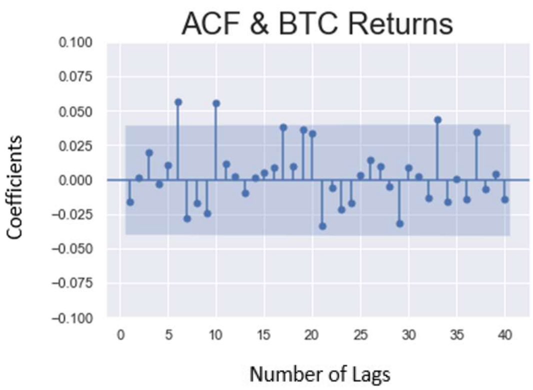
Finally, when we plot the residuals of returns we can see the below graph which shows the price volatility in different times. As there was a market crash in the second half of 2020, the prices fell down significantly which was not predicted by many investors.



Graph 15

4.1.4 MA models

Firstly, we will start from setting up our expectation on how many lags should be used. In order to do so, we will need to check the ACF for Return close prices again:



Graph 16

From the above plot we can remind ourselves that 6th and 10th lag seems to be statistically significant, and after 32nd lag the lags becomes following insignificance. So we can assume our model to have less than 35 lags.

In the next step we will fit the models. As expected from ACF plot, we will use the models with 7 and 10 lags, then calculate the LLR test to see which performs better. Here we define the Null hypothesis as the second (more complex model) does not perform better than the first one:

MA model with 7 lags

Dep. Variable:	returns	**No. Observations:**	2420
Model:	ARIMA(0, 0, 7)	**Log Likelihood**	-6708.664
Date:	Wed, 01 Feb 2023	**AIC**	13435.328
Time:	06:28:17	**BIC**	13487.452
Sample:	09-19-2014	**HQIC**	13454.283
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2759	0.083	3.329	0.001	0.113	0.438
ma.L1	-0.0161	0.013	-1.252	0.211	-0.041	0.009
ma.L2	0.0024	0.016	0.149	0.882	-0.030	0.034
ma.L3	0.0248	0.017	1.489	0.136	-0.008	0.058
ma.L4	-0.0089	0.016	-0.574	0.566	-0.039	0.022
ma.L5	0.0103	0.016	0.625	0.532	-0.022	0.042
ma.L6	0.0581	0.016	3.583	0.000	0.026	0.090
ma.L7	-0.0308	0.015	-2.118	0.034	-0.059	-0.002
sigma2	14.9762	0.207	72.339	0.000	14.570	15.382

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	6291.73
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	-0.17
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.89

Table 7

MA model with 10 lags

Dep. Variable:	returns	**No. Observations:**	2420
Model:	ARIMA(0, 0, 10)	**Log Likelihood**	-6703.508
Date:	Wed, 01 Feb 2023	**AIC**	13431.016
Time:	06:21:35	**BIC**	13500.515
Sample:	09-19-2014	**HQIC**	13456.289
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2758	0.085	3.257	0.001	0.110	0.442
ma.L1	-0.0161	0.013	-1.233	0.218	-0.042	0.009
ma.L2	0.0035	0.016	0.214	0.830	-0.029	0.036
ma.L3	0.0236	0.017	1.402	0.161	-0.009	0.057
ma.L4	-0.0052	0.016	-0.328	0.743	-0.036	0.026
ma.L5	0.0089	0.017	0.535	0.593	-0.024	0.041
ma.L6	0.0563	0.016	3.468	0.001	0.024	0.088
ma.L7	-0.0332	0.015	-2.271	0.023	-0.062	-0.005
ma.L8	-0.0166	0.018	-0.910	0.363	-0.052	0.019
ma.L9	-0.0273	0.018	-1.529	0.126	-0.062	0.008
ma.L10	0.0583	0.018	3.317	0.001	0.024	0.093
sigma2	14.9123	0.207	71.930	0.000	14.506	15.319

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	6322.16
Prob(Q):	0.97	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	-0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.91

Table 8

LLR test with 3 degrees of freedom

```
LLR_test(model_ret_ma_1,model_ret_ma_2, DF = 3)
```

```
0.016
```

Figure 11

As the result from LLR_test is less than 0,05 we can state that the model with 10 lags performs better than the model with 7 lags which means we reject the Null hyptothesis. Also, we can see a slight better log likelihood in the model with 10 lags.

In the next step, we will examine the residuals for the model that performed better with creating another column for residuals with MA models:

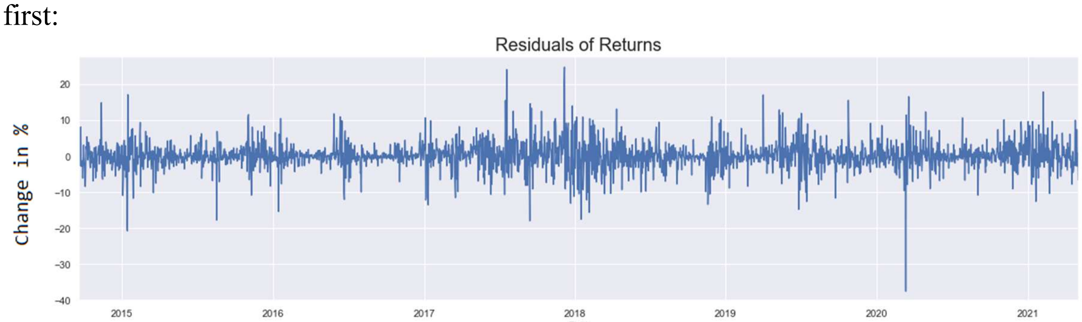
```
btc_train['res_ret_ma_2'] = results_ret_ma_2.resid[1:]
```

```
print("mean is " + str(round(btc_train.res_ret_ma_2.mean(),3)))
print("variance is " + str(round(btc_train.res_ret_ma_2.var(),3)))
print("Standard deviation is " + str(round(sqrt(btc_train.res_ret_ma_2.var()), 3)))
```

```
mean is 0.003
variance is 14.903
Standard deviation is 3.86
```

Code Chunk 5

In order to decide if our model is good or not, we need to check the graph for the residuals first:



Graph 17

Looking from the above graph it seems that the residuals are rather random than following certain pattern. In order to test this randomness, we can run Dickey-Fuller test and make sure if our residuals are stationary where our Null hypothesis that data is stationary:

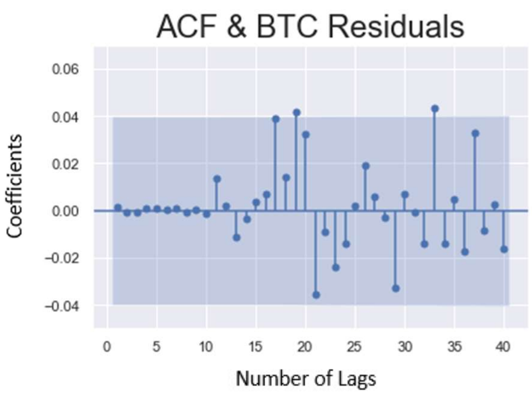
```
sts.adfuller(btc_train.res_ret_ma_2[2:])
```

```
(-49.055967881609746,
 0.0,
 0,
 2418,
 {'1%': -3.4330573017728736,
  '5%': -2.8627360537147197,
  '10%': -2.5674067087278276},
 13251.566606107284)
```

Figure 12

As our p-value is 0.0 we can state that the data is stationary which means we can accept the Null hypothesis.

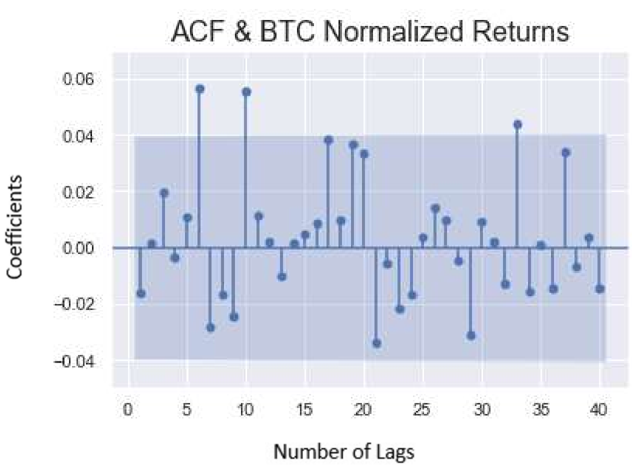
Furthermore, we can examine ACF of the residuals of our model in order to find out if the return residuals is White Noise or not:



Graph 18

From the above graphs we can see a many of the coefficients which are in significance level. As we have added the first 10 legs to our model it was expected to have the coefficients of those close to zero. And coefficients of the following 7 lags are also not significant which is a sign on how well our model performs.

In this step, we will investigate how the MA models forecasts normalized values (*which we have created earlier*) with plotting autocorrelation function. The purpose of this step is being able to compare BTC values with other crypto currency values towards to the end of the thesis as all 5 Crpytos have completely different range of prices.



Graph 19

From the above plot we can have an idea on what number of lags to use in firring the model to normalized returns based on the coefficients which are not within confidence level. However, we will fit the model and examine the results to make sure if normalizing has any effect on model selection or not:

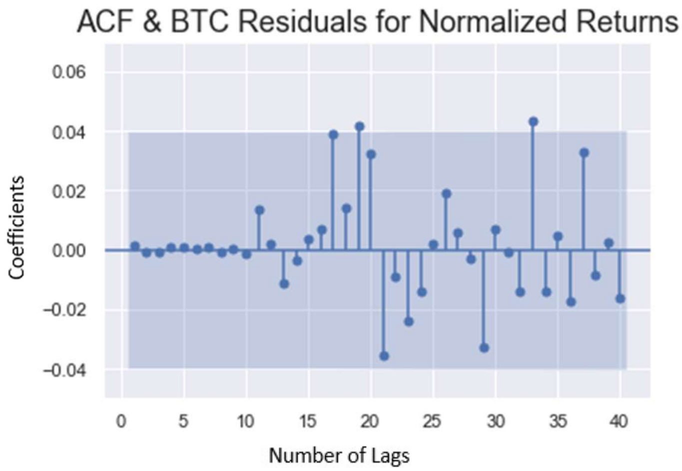
Dep. Variable:	norm_ret	**No. Observations:**	2420
Model:	ARIMA(0, 0, 10)	**Log Likelihood**	-13073.251
Date:	Wed, 01 Feb 2023	**AIC**	26170.503
Time:	08:03:41	**BIC**	26240.001
Sample:	09-19-2014	**HQIC**	26195.775
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.8366	1.181	-3.249	0.001	-6.151	-1.522
ma.L1	-0.0161	0.013	-1.230	0.219	-0.042	0.010
ma.L2	0.0035	0.017	0.213	0.831	-0.029	0.036
ma.L3	0.0236	0.017	1.399	0.162	-0.009	0.057
ma.L4	-0.0052	0.016	-0.328	0.743	-0.036	0.026
ma.L5	0.0089	0.017	0.534	0.593	-0.024	0.041
ma.L6	0.0563	0.016	3.458	0.001	0.024	0.088
ma.L7	-0.0332	0.015	-2.264	0.024	-0.062	-0.004
ma.L8	-0.0166	0.018	-0.907	0.364	-0.053	0.019
ma.L9	-0.0273	0.018	-1.525	0.127	-0.062	0.008
ma.L10	0.0583	0.018	3.308	0.001	0.024	0.093
sigma2	2890.5087	40.296	71.732	0.000	2811.530	2969.488

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	6322.17
Prob(Q):	0.97	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.91

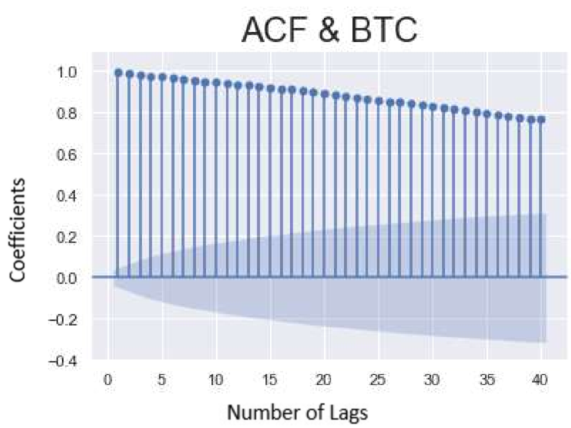
Table 9

From the above, we can see that we have almost the exact same results of MA model with 10 lags in normalized returns to non-normalized returns. Taking this into consideration we can state that normalizing values have no impact on model selection. Lastly to prove that our model was the correct choice we will plot the residuals and ACF of residuals again. From the below we can see many coefficients falling under confidence level even after the 10^{th} lag. That is the reason our model is indeed correct.



Graph 20

In the last part, we will try examining if close prices of BTC can be predicted using MA models. As we have already done in earlier sections as well, we will start with plotting ACF for Prices to determine the number of lags:



Graph 21

From the above we can clearly see that all the coefficients are higher than the confidence level which derives the assumption that any higher lag model will perform better than the one with less lags. Also, we can derive another theory that infinite number of lags would perform better in such cases. Since there is no possibility of adding infinite lags, we can presume that MA models are definitely not the best to predict the real close prices.

4.1.5 ARMA models

First, we will start with fitting ARMA models to the returns and interpret the results. We could use the 7 and 10 lags for each part of the model taking into consideration that they were the chosen lags for each model separately, this kind of complicated model in ARMA will be very time consuming and inefficient for computers to calculate. That is the reason choosing a model with less lags is more preferable. We can start to fit (3,3) and (4,4) models and check the results:

Dep. Variable:	returns	**No. Observations:**	2420
Model:	ARIMA(4, 0, 4)	**Log Likelihood**	-6710.810
Date:	Wed, 01 Feb 2023	**AIC**	13441.620
Time:	15:43:14	**BIC**	13499.535
Sample:	09-19-2014	**HQIC**	13462.681
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2791	0.079	3.555	0.000	0.125	0.433
ar.L1	-0.5407	1.559	-0.347	0.729	-3.596	2.514
ar.L2	-0.5609	1.381	-0.406	0.685	-3.267	2.145
ar.L3	-0.5386	1.432	-0.376	0.707	-3.345	2.267
ar.L4	0.3307	1.319	0.251	0.802	-2.255	2.916
ma.L1	0.5229	1.554	0.336	0.737	-2.523	3.569
ma.L2	0.5716	1.357	0.421	0.674	-2.088	3.232
ma.L3	0.5357	1.450	0.370	0.712	-2.305	3.377
ma.L4	-0.3441	1.327	-0.259	0.795	-2.945	2.257
sigma2	15.0146	0.199	75.288	0.000	14.624	15.405

Ljung-Box (L1) (Q):	0.01	**Jarque-Bera (JB):**	6719.47
Prob(Q):	0.92	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.50	**Skew:**	-0.15
Prob(H) (two-sided):	0.00	**Kurtosis:**	11.16

Table 10

Dep. Variable:	returns	**No. Observations:**	2420
Model:	ARIMA(3, 0, 3)	**Log Likelihood**	-6710.047
Date:	Wed, 01 Feb 2023	**AIC**	13436.095
Time:	15:39:32	**BIC**	13482.427
Sample:	09-19-2014	**HQIC**	13452.944
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2680	0.089	3.020	0.003	0.094	0.442
ar.L1	0.3481	0.124	2.798	0.005	0.104	0.592
ar.L2	-0.4338	0.074	-5.878	0.000	-0.578	-0.289
ar.L3	0.9335	0.123	7.562	0.000	0.692	1.175
ma.L1	-0.3507	0.129	-2.719	0.007	-0.604	-0.098
ma.L2	0.4414	0.076	5.840	0.000	0.293	0.590
ma.L3	-0.9265	0.128	-7.233	0.000	-1.178	-0.675
sigma2	14.9923	0.201	74.551	0.000	14.598	15.386

Ljung-Box (L1) (Q):	0.43	**Jarque-Bera (JB):**	6416.59
Prob(Q):	0.51	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	-0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.97

Table 11

Looking to the above models we can see that the lags in the model (3,3) is statistically significant whereas the same values in the model (4,4) were much above from the significance level. That is the reason we presume that the model with 3 lags in each side would be much better fit for our data. Furthermore, from the LLR test we can also prove that the first model with 3,3 is doing much better than the model with 4,4 lags:

```
LLR_test(model_ret_arma_1, model_ret_arma_2, DF = 2)
1.0
```

Figure 13

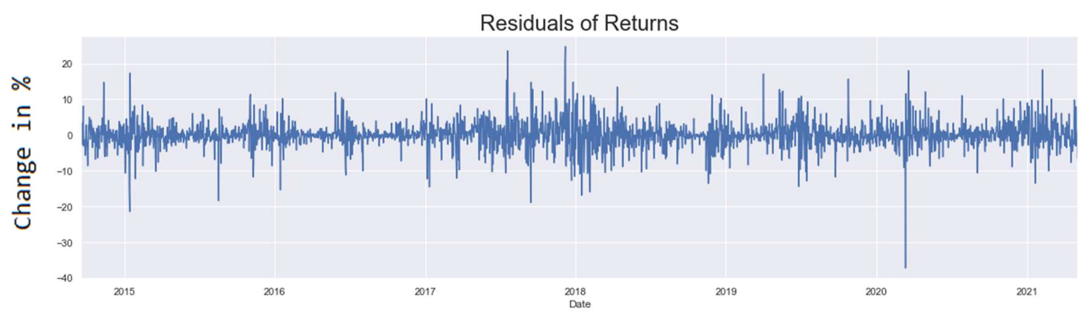
Finally, when we compare AIC values, we can see that ARMA (3,3) has less information criteria which is a better indicator:

```
print("ARMA(4,4) AIC Value is " + str(results_ret_arma_2.aic))
print("ARMA(3,3) AIC Value is " + str(results_ret_arma_1.aic))
```

```
ARMA(4,4) AIC Value is 13441.619914324332
ARMA(3,3) AIC Value is 13436.094945896086
```

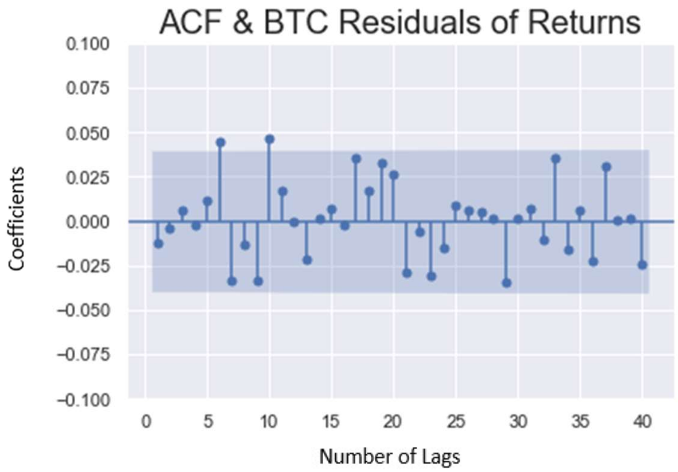
Figure 14

As we have done previously for other models, we will also analyse the residuals of ARMA model too. In order to do so we will create another column with residuals derived from our ARMA model (3,3) and plot it:



Graph 22

The results are quite similar on what we have obtained from AR and MA models previously. This recommends that the volatility in returns cannot be fully understood in case of using only ARMA model. However, we will still need to make sure if the residuals are random by plotting the autocorrelation function:



Graph 23

Looking at the ACF we can see that the majority of the lags are falling within the confidence level which enables us stating the residuals are random.

Lastly, we will use ARMA models in close prices of BTC and examine how well it performs on stationary data.

In order to do so, we will fit the ARMA models (3,3) which was our choice for returns and also ARMA model (3,6) which is the model until we obtain some coefficients above significance level:

Dep. Variable:	Close	No. Observations:	2421
Model:	ARIMA(3, 0, 6)	Log Likelihood	-18673.037
Date:	Wed, 01 Feb 2023	AIC	37368.073
Time:	16:23:28	BIC	37431.785
Sample:	09-18-2014	HQIC	37391.241
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	7170.5465	6.440	1113.375	0.000	7157.924	7183.169
ar.L1	0.5315	0.005	116.439	0.000	0.523	0.540
ar.L2	-0.4875	0.005	-95.766	0.000	-0.497	-0.477
ar.L3	0.9557	0.004	238.521	0.000	0.948	0.964
ma.L1	0.4855	0.010	48.887	0.000	0.466	0.505
ma.L2	1.0109	0.009	108.931	0.000	0.993	1.029
ma.L3	0.0918	0.011	7.988	0.000	0.069	0.114
ma.L4	0.0392	0.011	3.644	0.000	0.018	0.060
ma.L5	0.0471	0.009	5.431	0.000	0.030	0.064
ma.L6	0.0278	0.008	3.474	0.001	0.012	0.043
sigma2	2.967e+05	2183.294	135.912	0.000	2.92e+05	3.01e+05

Ljung-Box (L1) (Q):	0.01	Jarque-Bera (JB):	128907.71
Prob(Q):	0.94	Prob(JB):	0.00
Heteroskedasticity (H):	3812.82	Skew:	1.30
Prob(H) (two-sided):	0.00	Kurtosis:	38.65

Table 12

Dep. Variable:	Close	No. Observations:	2421
Model:	ARIMA(3, 0, 3)	Log Likelihood	-18695.052
Date:	Wed, 01 Feb 2023	AIC	37406.104
Time:	16:23:31	BIC	37452.439
Sample:	09-18-2014	HQIC	37422.953
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	7170.5465	1.09e-10	6.6e+13	0.000	7170.547	7170.547
ar.L1	-0.5941	0.006	-102.277	0.000	-0.605	-0.583
ar.L2	0.6075	0.004	155.027	0.000	0.600	0.615
ar.L3	0.9865	0.006	164.015	0.000	0.975	0.998
ma.L1	1.6159	0.010	161.532	0.000	1.596	1.636
ma.L2	1.0141	0.016	63.318	0.000	0.983	1.046
ma.L3	0.0293	0.009	3.353	0.001	0.012	0.046
sigma2	3.008e+05	7.32e-09	4.11e+13	0.000	3.01e+05	3.01e+05

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	146715.78
Prob(Q):	0.99	Prob(JB):	0.00
Heteroskedasticity (H):	4776.46	Skew:	1.35
Prob(H) (two-sided):	0.00	Kurtosis:	41.04

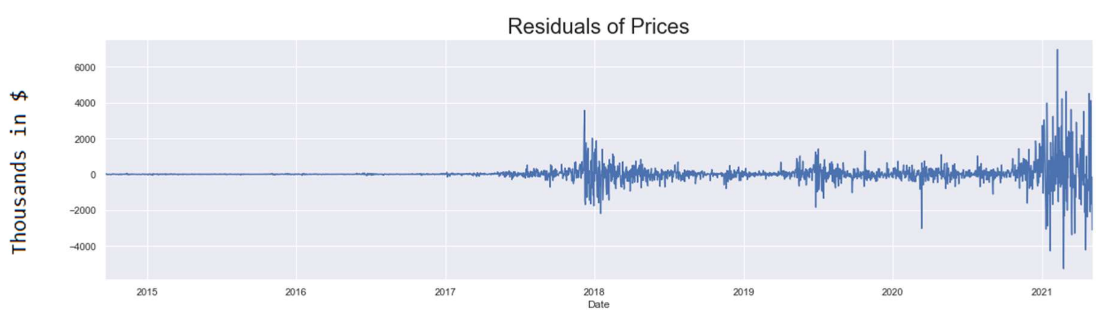
Table 13

```
LLR_test(model_arma_1, model_arma_2, DF = 3)
0.0
```

Figure 15

Looking at the LLR test the ARMA (3,6) performs better than ARMA (3,3). Furthermore, smaller AIC value in suggests that the model 3,6 is a better fit to our data.

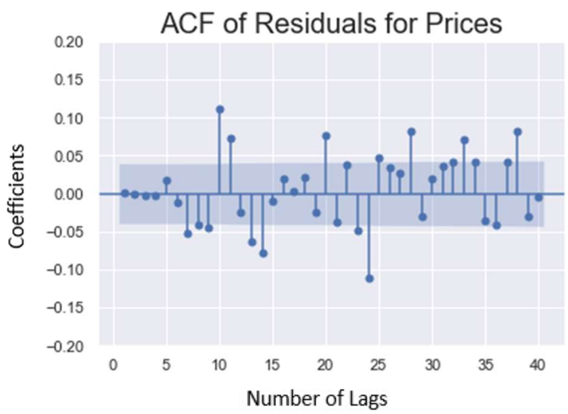
In the final part, we can analyse the residual values of BTC for close prices.



Graph 24

Looking at the above plot the prices have similar trends in the previous residual graphs for prices. Seems like, there might be some volatility in 2018 and 2021 where there was a big difference between expected and actual values.

Last but not least, we plot the ACF of residuals and examine the results:

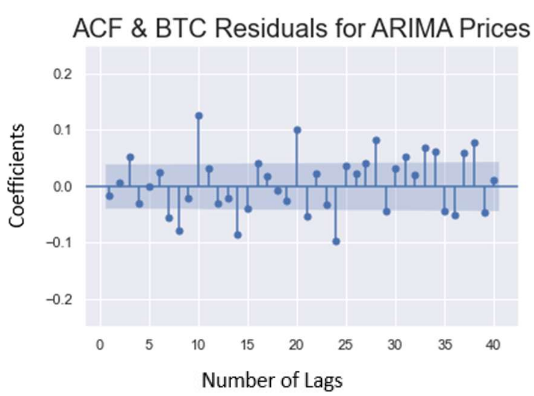


Graph 25

From the above graph we can see that there are many lags which are significantly non-zero. Taking this into consideration we can claim that the residuals for the prices are non-random.

4.1.6 ARIMA models

In this section we will fit different ARIMA models to our data and examine the results. In order to choose the lags we can start from plotting the ACF of residuals for ARIMA (1,1,1):



Graph 26

Looking at the autocorrelation graph we can see that it might be helpful to add 3rd or 7th lag to our model as they are significant. As we generally prefer simpler models lets try to fit various models with 3 lags and compare their result with likelihood, AIC and log likelihood ratio test:

Fitted models:

```
# 1
model_arima_1 = ARIMA(btc_train.Close[1:], order = (1,1,1))
results_model_arima1 = model_arima1.fit()
results_model_arima1.summary()
# 2
model_arima_2 = ARIMA(btc_train.Close[1:], order = (1,1,2))
results_model_arima_2 = model_arima_2.fit()
results_model_arima_2.summary()
# 3
model_arima_3 = ARIMA(btc_train.Close[1:], order = (1,1,3))
results_model_arima_3 = model_arima_3.fit()
results_model_arima_3.summary()
# 4
model_arima_4 = ARIMA(btc_train.Close[1:], order = (2,1,1))
results_model_arima_4 = model_arima_4.fit()
# 5
model_arima_5 = ARIMA(btc_train.Close[1:], order = (3,1,1))
results_model_arima_5 = model_arima_5.fit()

# 6
model_arima_6 = ARIMA(btc_train.Close[1:], order = (3,1,2))
results_model_arima_6 = model_arima_6.fit()
```

Code Chunk 6

Results of calculation of LL and AIC:

```
ARIMA(1,1,1):     LL =  -18690.930198249596       AIC =  37387.86039649919
ARIMA(1,1,2):     LL =  -18689.748010232594       AIC =  37387.49602046519
ARIMA(1,1,3):     LL =  -18686.541816054236       AIC =  37383.08363210847
ARIMA(2,1,1):     LL =  -18689.894258525324       AIC =  37387.78851705065
ARIMA(3,1,1):     LL =  -18686.098516218015       AIC =  37382.19703243603
ARIMA(3,1,2):     LL =  -18655.94414688421        AIC =  37323.88829376842
```

Figure 16

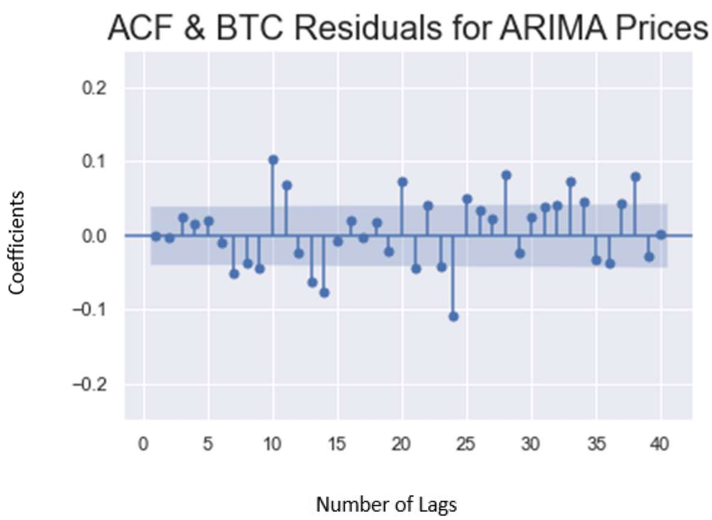
As we can see that ARIMA (3,1,2) has higher LL and lower AIC, we can make the conclusion that this model may perform better than the others. To make this statement sure we can finally run the LLR test and make our statement where we define the Null hypothesis as the ARIMA (3,1,2) does not perform better than the others:

```
print("\nLLR test p-value = " + str(LLR_test(model_arima_5, model_arima_6)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_4, model_arima_6, DF = 2)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_2, model_arima_6, DF = 2)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_1, model_arima_6, DF = 3)))
```

```
LLR test p-value = 0.0

LLR test p-value = 0.0

LLR test p-value = 0.0

LLR test p-value = 0.0
```

Figure 17

From the results it is obvious that the ARIMA (3,1,2) outperforms the other models. So we reject the Null hypothesis. Lastly, lets plot the residuals for this model and examine the results:



Graph 27

In this autocorrelation graph we can see less coefficients which are insignificant in comparison with the simple ARIMA model which shows that how the new model performs better in actual and expected prices. However, 10^{th} lag is still highly significant which is the sign that there might be a better model existing with 10 lags.

In the next step, we will try to use higher level of integration. As we know in order to use higher integration levels our data needs to come from a non-stationary process. In order to find if integrated data is stationary or not, we will create manually an integrated delta prices column using diff function in python and then use Dicky-Fueller test where we define claim the Null hypothesis as the data is stationary:

```
btc_train['delta_prices']=btc_train.Close.diff(1)
```

```
sts.adfuller(btc_train.delta_prices[1:])
```

```
(-8.177641199656662,
 8.293887243613951e-13,
 27,
 2392,
 {'1%': -3.4330867606360274,
  '5%': -2.862749062318083,
  '10%': -2.5674136347538057},
 36839.442886560086)
```

Figure 18

From the above results we see that our test statistic is greater than our critical values in all 3 levels. Furthermore, p-value is also very close to 0 which enables us to state that the data is stationary. So we accept the Null hypothesis. Taking this into consideration we can easily recommend not to use higher integrated levels in ARIMA models as 1 level of integration will be sufficient.

4.1.7 ARIMAX models

In the next step we will use ARIMAX model to include outside factors which has impact on prices. So called exogenous variables can be used in model fitting in python. To do that we will use the prices of Ethereum to see if there is correlation between ETH and BTC.

Dep. Variable:	Close	No. Observations:	354
Model:	ARIMA(3, 1, 3)	Log Likelihood	-2649.440
Date:	Thu, 09 Feb 2023	AIC	5314.879
Time:	07:26:50	BIC	5345.811
Sample:	11-09-2017	HQIC	5327.187
	- 10-28-2018		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
Close	8.8986	0.392	22.711	0.000	8.131	9.667
ar.L1	-0.0342	0.365	-0.094	0.925	-0.749	0.681
ar.L2	-0.7492	0.090	-8.300	0.000	-0.926	-0.572
ar.L3	-0.1546	0.195	-0.793	0.428	-0.537	0.228
ma.L1	0.2556	0.367	0.696	0.486	-0.464	0.976
ma.L2	0.6644	0.111	5.972	0.000	0.446	0.882
ma.L3	0.2654	0.168	1.577	0.115	-0.064	0.595
sigma2	1.855e+05	5965.581	31.090	0.000	1.74e+05	1.97e+05

Ljung-Box (L1) (Q):	0.27	Jarque-Bera (JB):	3110.45
Prob(Q):	0.61	Prob(JB):	0.00
Heteroskedasticity (H):	0.02	Skew:	1.00
Prob(H) (two-sided):	0.00	Kurtosis:	17.40

Table 14

Dep. Variable:	Close	No. Observations:	354
Model:	ARIMA(1, 1, 1)	Log Likelihood	-2651.841
Date:	Thu, 09 Feb 2023	AIC	5311.681
Time:	07:27:56	BIC	5327.147
Sample:	11-09-2017	HQIC	5317.835
	- 10-28-2018		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
Close	8.1114	0.417	19.457	0.000	7.294	8.929
ar.L1	-0.3027	0.082	-3.676	0.000	-0.464	-0.141
ma.L1	0.5463	0.084	6.535	0.000	0.382	0.710
sigma2	1.984e+05	6336.192	31.307	0.000	1.86e+05	2.11e+05

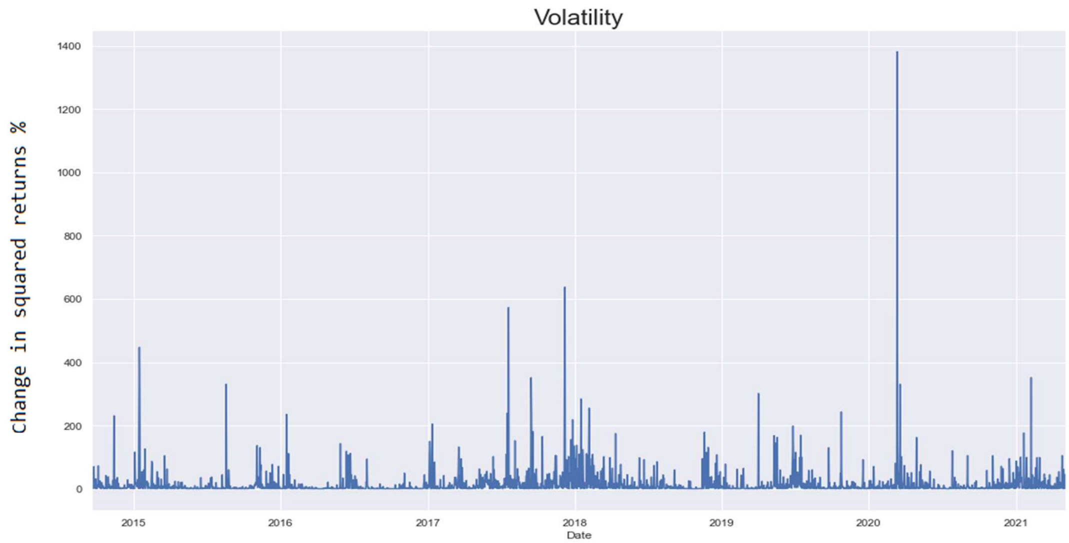
Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	2346.59
Prob(Q):	0.98	Prob(JB):	0.00
Heteroskedasticity (H):	0.02	Skew:	0.98
Prob(H) (two-sided):	0.00	Kurtosis:	15.48

Table 15

As it can be seen above the p value of the close prices of ETH is statistically significant for our prices in BTC. Comparing the models we can see higher likelihood in the complex model with 3 lags in each side which can be the sign of better performance.

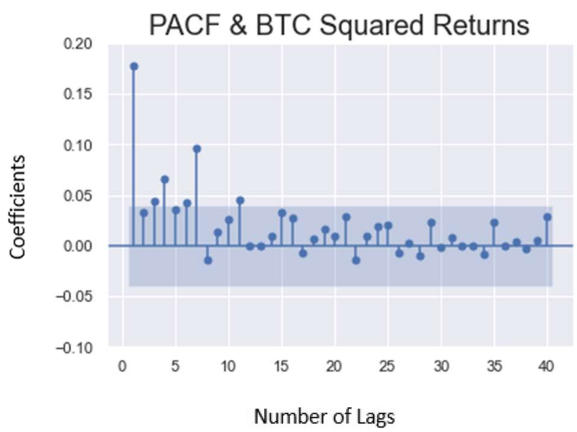
4.1.8 ARCH and GARCH models

In this section we will use ARCH models in order to analyse the volatility of returns. Before starting to try the models, we will create another column where we will create the squared of returns as our volatility values. From the below plot we can see that the returns for BTC seems to have high volatility as expected.



Graph 28

Next even though PACF is not able to assist us in defining the number of lags to be used in ARCH model, we can still get a lot of valuable data by looking at it:



Graph 29

As we can see from the above results, out 7 lags in the beginning 5 of them are statistically significant. Such high values in PACF might be a convention that there can be short term trends in variances.

Now, we will fit the ARCH model with constant mean with 5 iterations

Constant Mean - ARCH Model Results

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	ARCH	**Log-Likelihood:**	-6641.33
Distribution:	Normal	**AIC:**	13288.7
Method:	Maximum Likelihood	**BIC:**	13306.0
		No. Observations:	2418
Date:	Sun, Jan 29 2023	**Df Residuals:**	2417
Time:	13:32:49	**Df Model:**	1

Mean Model

	coef	**std err**	**t**	**P>|t|**	**95.0% Conf. Int.**
mu	0.2983	7.493e-02	3.982	6.843e-05	[0.151, 0.445]

Volatility Model

	coef	**std err**	**t**	**P>|t|**	**95.0% Conf. Int.**
omega	12.4061	1.125	11.025	2.887e-28	[10.201, 14.612]
alpha[1]	0.1696	4.138e-02	4.099	4.154e-05	[8.850e-02, 0.251]

Table 16

From the above results we can see that both adjusted and not adjusted R squared are 0.00. As R squared is the way to measure explanatory variation compared to the mean it means that for our ARCH model it will not be very useful in explaining the deviation. Moving to Log likelihood we can see a higher value in log likelihood in ARCH models in comparison with our previous AR, MA, ARMA and ARIMA models which means that simple ARCH model performs already well. Secondly, we will fit another ARCH model with 3 lags and compare their results to the simpler one.

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	ARCH	**Log-Likelihood:**	-6596.32
Distribution:	Normal	**AIC:**	13202.6
Method:	Maximum Likelihood	**BIC:**	13231.6
		No. Observations:	2420
Date:	Thu, Feb 02 2023	**Df Residuals:**	2419
Time:	15:41:28	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2894	6.694e-02	4.324	1.534e-05	[0.158, 0.421]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	9.2991	1.314	7.077	1.469e-12	[6.724, 11.874]
alpha[1]	0.1623	4.209e-02	3.855	1.155e-04	[7.977e-02, 0.245]
alpha[2]	0.0830	4.529e-02	1.833	6.675e-02	[-5.735e-03, 0.172]
alpha[3]	0.1733	6.294e-02	2.753	5.899e-03	[4.994e-02, 0.297]

Table 17

From the first glance it is already visible that the log likelihood has increased while AIC decreased when we used 3 lags. These both are already indicators that second model is outperforms the first one. Lastly, checking the coefficients (p-values) we see that all the figures are statistically significant with the exception of alpha 2. Judging overall we can still claim that the second model with 3 lags performs better than the first one in estimating the market volatility.

In the last section of this sub-chapter, we will fit GARCH models which are extension or ARCH and also referred as “ARMA Equivalent” of ARCH which is generally expected to have better performance. We will fit simple and multi lag GARCH models and compare the results:

Constant Mean - GARCH Model Results

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	GARCH	**Log-Likelihood:**	-6486.73
Distribution:	Normal	**AIC:**	12981.5
Method:	Maximum Likelihood	**BIC:**	13004.6
		No. Observations:	2420
Date:	Thu, Feb 02 2023	**Df Residuals:**	2419
Time:	16:11:03	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2392	6.322e-02	3.783	1.547e-04	[0.115, 0.363]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	0.6753	0.258	2.616	8.891e-03	[0.169, 1.181]
alpha[1]	0.1294	3.184e-02	4.065	4.794e-05	[6.703e-02, 0.192]
beta[1]	0.8373	2.924e-02	28.633	2.575e-180	[0.780, 0.895]

Table 18

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	GARCH	**Log-Likelihood:**	-6485.59
Distribution:	Normal	**AIC:**	12981.2
Method:	Maximum Likelihood	**BIC:**	13010.1
		No. Observations:	2420
Date:	Thu, Feb 02 2023	**Df Residuals:**	2419
Time:	16:11:08	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2425	6.245e-02	3.883	1.031e-04	[0.120, 0.365]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	0.8094	0.312	2.594	9.477e-03	[0.198, 1.421]
alpha[1]	0.1583	3.515e-02	4.503	6.709e-06	[8.938e-02, 0.227]
beta[1]	0.5403	0.218	2.477	1.323e-02	[0.113, 0.968]
beta[2]	0.2617	0.201	1.299	0.194	[-0.133, 0.656]

Table 19

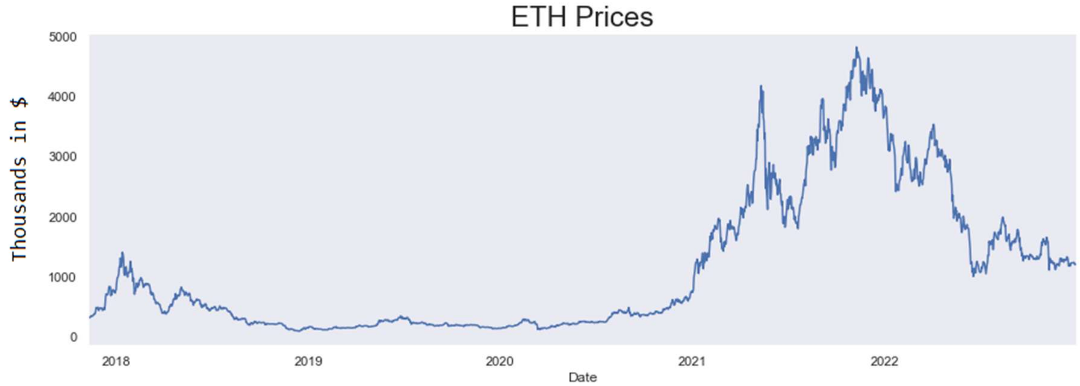
Looking at the above we can see that simple GARCH model with (1,1) performs better than more complex GARCH model based on having beta [2] greater than 0,5 which means it is not significant. Taking this into consideration we will stick to the GARCH (1,1).

Finally, when we compare the GARCH (1,1) with our ARCH model with 3 lags, we can see a better log likelihood in ARCH model. From that perspective we can state that in order to estimate the volatility of BTC it ARCH model would be a better fit.

4.2 Analysis of ETH

In the next section of the thesis, we will analyse another popular and practical cryptocurrency which is Ethereum. We will proceed with the same strategy as we have done for BTC and will define the best model to predict Ethereum's prices.

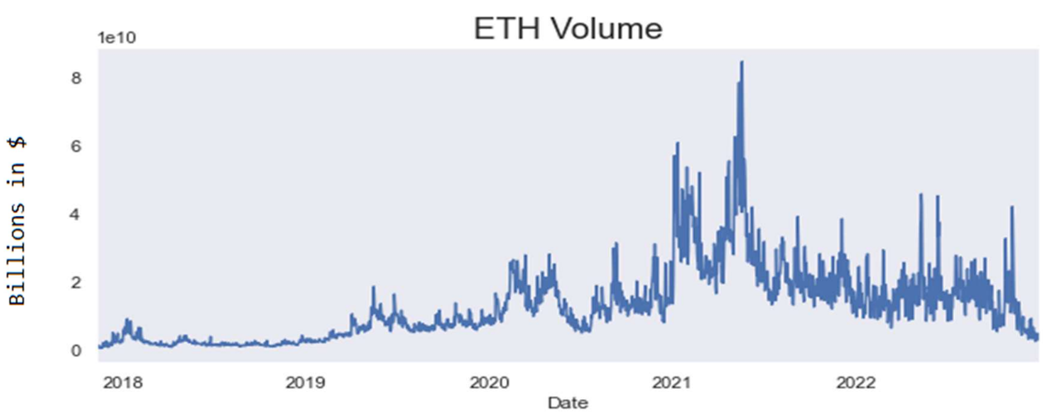
Initially, we can start by plotting the prices of ETH from its first registered date in Yahoo finance.



Graph 30

From the above plot of prices, we can observe that there was a sharp increase of ETH since the beginning of 2021 which is following a fall towards the middle of the same year and expansion again at the end of the same year. Furthermore, 2022 also shows a significant fall in ETH prices. In general, at the first glance, the trends in ETH prices since 2021 remind the trends in BTC prices.

Next, we can plot the market capitalization of ETH and check the trends as well:

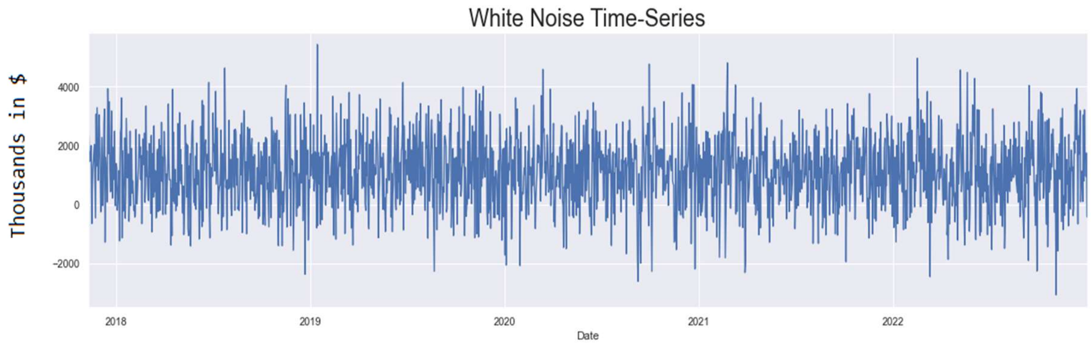


Graph 31

The graph shows us that the Market volume of ETH was relatively stable until the middle of 2020 and fluctuating since then until now. Both plots for ETH market volume and ETH prices can be signs of high volatility which we will investigate in further sections of the thesis.

4.2.1 White Noise, Stationarity and Seasonality

We'll attempt to look at our dataset's *white noise* in the following step as we have done for BTC.



Graph 32

Here, the white noise indicates whether or not our data is predictable. We may look at the mean, which in our instance is 1111.73, to demonstrate that the data is not white noise because time series data with white noise would always have a mean of zero.

```
print(wn.mean())
```

```
1111.7392034831769
```

Figure 19

The dataset's *stationarity* will be examined using the Dicky Fuller test in the following step and set our Null hypothesis as “The data is stationary”:

```
sts.adfuller(ethdata.Close)
```

```
(-1.4089708365742828,
 0.5779679105330212,
 17,
 1861,
 {'1%': -3.4338687226315336,
  '5%': -2.863094318475046,
  '10%': -2.5675974634086765},
 21446.440104463112)
```

Figure 20

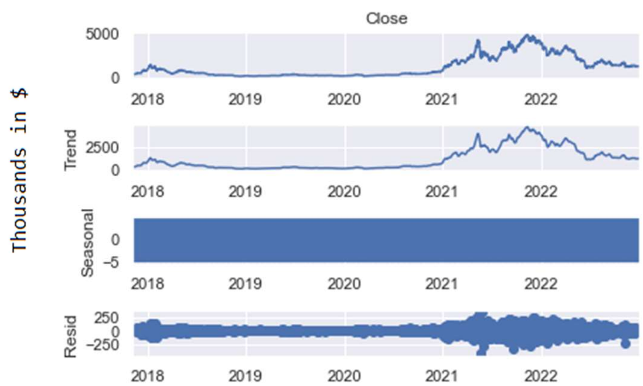
Our test statistic exceeds all critical values, hence there is insufficient support for stationarity.

We cannot confirm that the data is stationary since, according to the second line of the p-value, there is a 57% chance that the null hypothesis will be accepted. So we reject the

Null hypothesis. From the number of lags, we can see that there is some autocorrelation that is going 17 periods back.

In the following phase, the dataset's *seasonality* will be examined using additive and multiplicative decompositions.

Additive:



Graph 33

We can observe that the plot above is a rectangle when we look at the seasonal portion. As we mentioned during BTC analysis, this occurs when the figures are too small, and the values oscillate back and forth constantly. In our situation, the linear change is caused by a steady up-and-down movement between -250 and 250 for each period. As a result, no actual cyclical pattern based on naïve decomposition can be found. Lastly, from the residual part of the plot, we can see that they are relatively high in 2018, 2021, and 2022.

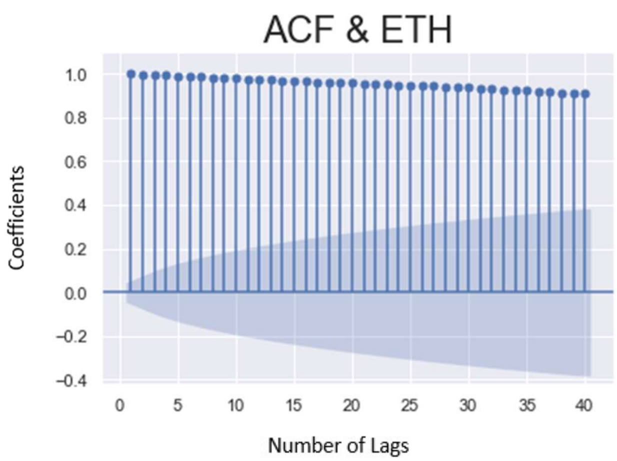
Multiplicative:

Graph 34

We may find the same results when we use the multiplicative decomposition method to confirm the absence of seasonality in our dataset.

4.2.2 Auto Correlation Function and Partial Correlation Function

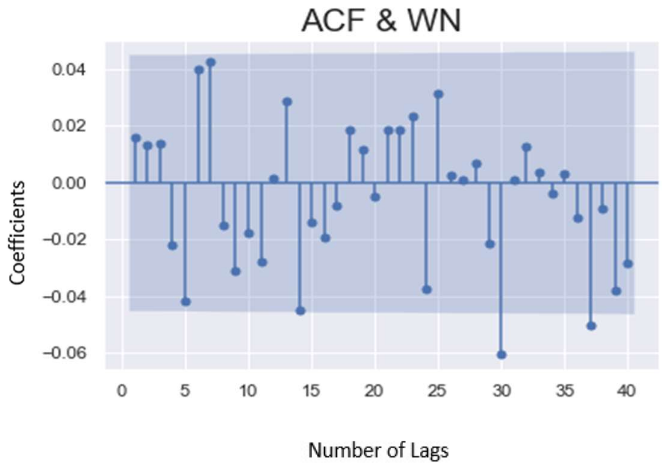
We will now look at the *autocorrelation* of ETH prices. To do this, we once more used Python for visualization, setting the period to daily.



Graph 35

All of the lags are higher than the specified significance threshold, which is represented by the area in blue in the plot at the top, as can be seen. We can say that there is an autocorrelation between lags since each lag demonstrates how the prices differ from one another one period ago. Simply put, it indicates that we can still predict future prices using prices from a previous time period.

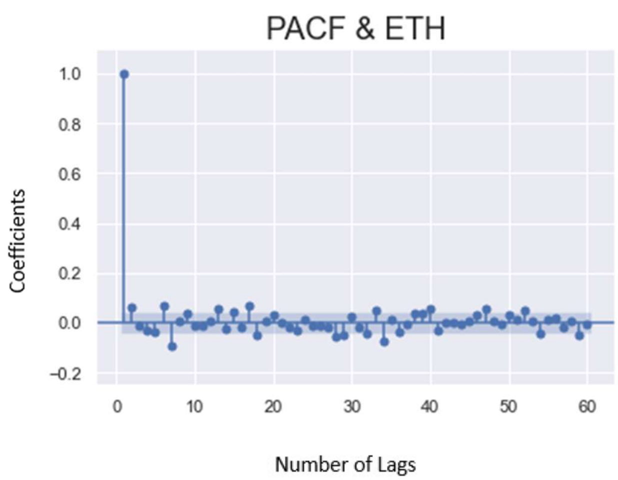
Additionally, we may display the ACF of our white noise data as a secondary assurance that the data is indeed white noise:



Graph 36

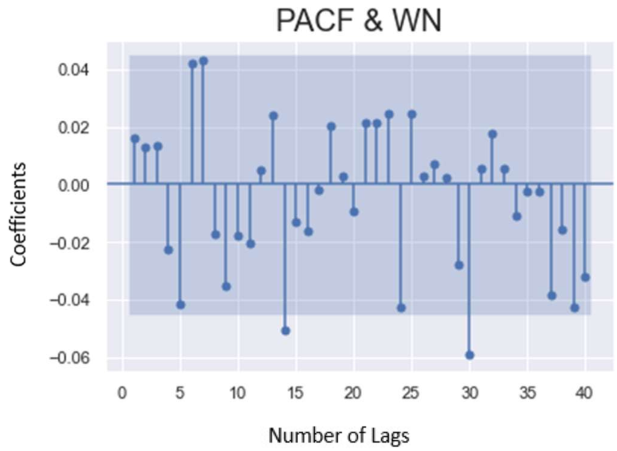
Here, it is easy to see that almost all of the lags are inside the threshold of significance (blue figure), allowing us to confidently conclude that there is no autocorrelation in the white noise data, which is one of the WN's underlying assumptions.

In this stage, we'll also use Python's order least squares approach to analyse PACF for BTC close prices to check *partial autocorrelation*.



Graph 37

We can see a quite different graphic from the auto correlation function since the PACF illustrates the direct effects of the prices from the previous period. When examining the plot, we can also see positive and negative numbers, which are fairly arbitrary and have no long-term consequences.
Finally, we can do the same investigation in White noise data:



Graph 38

Again, based on the plot above, we can conclude that everything is perfectly random, despite the fact that there is one lag that is not statistically significant. By taking this into account, we can demonstrate once more that WN data lacks autocorrelation.

4.2.3 AR models

This part will serve as the beginning of our model-building process, using auto-regression models taking into consideration that we have already separated the data into training and testing dataset in the ratio of 80/20.

First, it's crucial to look at ACF&PACF and take the result from there to decide how many lags to employ in the AR model. We can go over them again since we have already plotted our graphs in univariate. The ACF graph demonstrates that the more lags we add, the better our model will fit our data set, however, this can lead to an overfitting issue that could result in inaccurate projections of future prices. We may recall that there are existent negative and positive coefficients as well as some coefficients that are not in a significant level based on the PACF plot. We should have less than 30 since, as we can see, the coefficients are more likely to be significant after the 30 lag.

The following phase involves implementing an AR model with a single lag before moving slowly forward to choose the optimal model. Additionally, in order to determine the optimal model, we will evaluate models with various lags using the log-likelihood test. Here we define the Null hypothesis as the second (more complex model) does not perform better than the first one:

AR model with 2 lags:

Dep. Variable:	Close	**No. Observations:**	1502
Model:	ARIMA(2, 0, 0)	**Log Likelihood**	-8630.028
Date:	Sat, 04 Feb 2023	**AIC**	17268.055
Time:	19:34:08	**BIC**	17289.313
Sample:	11-10-2017	**HQIC**	17275.974
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	901.5974	4705.739	0.192	0.848	-8321.481	1.01e+04
ar.L1	0.8998	0.013	71.623	0.000	0.875	0.924
ar.L2	0.0993	0.013	7.898	0.000	0.075	0.124
sigma2	5706.7400	61.354	93.014	0.000	5586.489	5826.991

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	35366.99
Prob(Q):	0.99	**Prob(JB):**	0.00
Heteroskedasticity (H):	13.85	**Skew:**	-0.96
Prob(H) (two-sided):	0.00	**Kurtosis:**	26.69

Table 20

AR model with 3 lags:

Dep. Variable:	Close	**No. Observations:**	1502
Model:	ARIMA(3, 0, 0)	**Log Likelihood**	-8630.011
Date:	Sat, 04 Feb 2023	**AIC**	17270.021
Time:	19:36:07	**BIC**	17296.594
Sample:	11-10-2017	**HQIC**	17279.920
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	901.5950	4672.272	0.193	0.847	-8255.890	1.01e+04
ar.L1	0.9003	0.013	70.713	0.000	0.875	0.925
ar.L2	0.1035	0.015	6.842	0.000	0.074	0.133
ar.L3	-0.0047	0.010	-0.453	0.651	-0.025	0.016
sigma2	5706.3579	63.493	89.874	0.000	5581.914	5830.802

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	35121.95
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	13.85	**Skew:**	-0.95
Prob(H) (two-sided):	0.00	**Kurtosis:**	26.61

Table 21

Log-likelihood test for models with 2 and 3 lags (for visibility as 1 lag model performed well as well):

```
LLR_test(ar_model, ar_model_1)
```

```
0.851
```

Figure 21

Based on the findings of the LLR test, it is clear that we should stop at the 3rd lag after a number of trials up to the 4th lag. We will choose 2 lags because the log likelihood test shows that simpler model has a better log likelihood. So we accept the Null hyptothesis. Next, since returns, which are a percentage representation of price changes, are more dependable than stationary data in AR models since they have constant mean, variance, and autocorrelation, we will try to employ returns. To do that, we added a new column to the training and test data sets and calculated the percentages using simple Python methods:

```
eth_train['returns'] = eth_train.Close.pct_change(1).mul(100)
eth_test['returns'] = eth_test.Close.pct_change(1).mul(100)
eth_train = eth_train.iloc[1:]
```

Code Chunk 7

And we will test the stationarity with the Dickey-Fueller test where we define claim the Null hypothesis as the data is stationary:

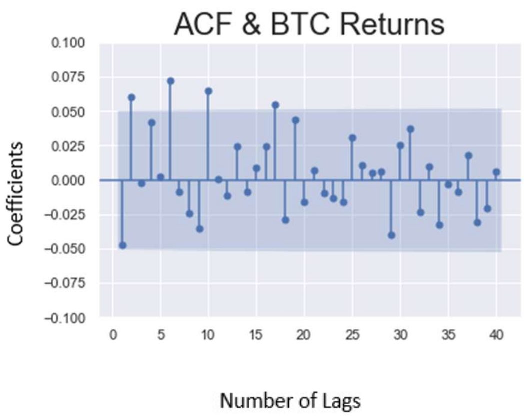
```
sts.adfuller(eth_train.returns)
```

```
(-11.412390521316967,
 7.197384243781305e-21,
 9,
 1492,
 {'1%': -3.434740473427213,
  '5%': -2.863479112458789,
  '10%': -2.5678023610641922},
 9054.48520801563)
```

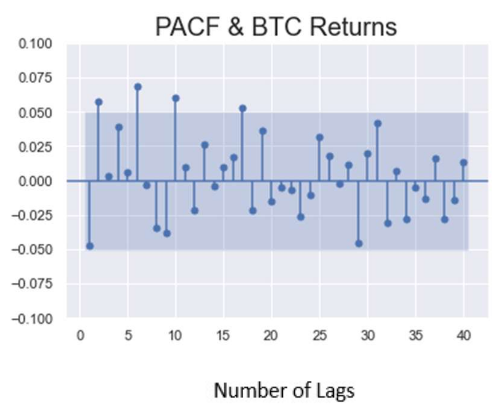
Figure 22

We can see from the above that our test statistic, which is -11.41, is less than each of the three essential values at various confidence levels. As a result, we can say that the data is stationary with accepting the Null hypthesis.

Now we can go ahead and examine ACF and PACF respectively for the return values:



Graph 39



Graph 40

Both autocorrelation and partial autocorrelation imply that some coefficients are positive, some are negative, some are also inside the confidence interval and some are outside the confidence interval, allowing us to conclude that the data do not exhibit autocorrelation.

We will approach returns in a similar manner as we did when we trained the models for ETH prices. In order to determine which model fits the data best, we will perform the log-likelihood test and attempt to fit many models with various lags:

Dep. Variable:	returns	**No. Observations:**	1500
Model:	ARIMA(4, 0, 0)	**Log Likelihood**	-4593.996
Date:	Sat, 04 Feb 2023	**AIC**	9199.991
Time:	20:09:57	**BIC**	9231.870
Sample:	11-12-2017	**HQIC**	9211.867
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3060	0.143	2.133	0.033	0.025	0.587
ar.L1	-0.0444	0.020	-2.278	0.023	-0.083	-0.006
ar.L2	0.0555	0.024	2.295	0.022	0.008	0.103
ar.L3	0.0039	0.023	0.170	0.865	-0.041	0.049
ar.L4	0.0389	0.020	1.912	0.056	-0.001	0.079
sigma2	26.7740	0.552	48.481	0.000	25.692	27.856

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	1879.90
Prob(Q):	0.99	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.90	**Skew:**	-0.25
Prob(H) (two-sided):	0.23	**Kurtosis:**	8.46

Table 22

Dep. Variable:	returns	**No. Observations:**	1500
Model:	ARIMA(8, 0, 0)	**Log Likelihood**	-4589.558
Date:	Sat, 04 Feb 2023	**AIC**	9199.117
Time:	20:10:01	**BIC**	9252.249
Sample:	11-12-2017	**HQIC**	9218.911
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3061	0.149	2.051	0.040	0.014	0.599
ar.L1	-0.0450	0.020	-2.246	0.025	-0.084	-0.006
ar.L2	0.0552	0.025	2.191	0.028	0.006	0.105
ar.L3	0.0038	0.024	0.162	0.872	-0.042	0.050
ar.L4	0.0365	0.021	1.780	0.075	-0.004	0.077
ar.L5	0.0094	0.022	0.428	0.669	-0.034	0.052
ar.L6	0.0699	0.024	2.909	0.004	0.023	0.117
ar.L7	-0.0050	0.020	-0.250	0.803	-0.044	0.034
ar.L8	-0.0349	0.025	-1.420	0.156	-0.083	0.013
sigma2	26.6151	0.555	47.984	0.000	25.528	27.702

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	1975.82
Prob(Q):	0.96	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.90	**Skew:**	-0.26
Prob(H) (two-sided):	0.23	**Kurtosis:**	8.60

Table 23

```
LLR_test(ar_model_ret_1, ar_model_ret_2)
```

```
0.003
```

Figure 23

Our tests show that the second AR model with 8 lags has a higher log-likelihood, and since the LLR test value is smaller than 0.01, we can conclude that the second model is superior to the first model with 4 lags.

The following step allows us to evaluate whether normalized prices would produce stationary data that we could incorporate into our Python-based AR models:

```
benchmark = eth_train.Close.iloc[0]
```

```
eth_train['norm'] = eth_train.Close.div(benchmark).mul(100)
eth_test['norm'] = eth_test.Close.div(benchmark).mul(100)
```

Code Chunk 8

Testing stationarity as usual where we define claim the Null hypothesis as the data is stationary:

```
sts.adfuller(eth_train.norm)
```

```
(0.6069429080168565,
 0.9877855155545913,
 17,
 1483,
 {'1%': -3.4347671645756304,
  '5%': -2.86349089226533,
  '10%': -2.5678086339403325},
 13543.131431078897)
```

Figure 24

We can conclude that the data is non-stationary because the test statistic is higher than our critical values, hence we won't use normalized prices for our AR model so we reject the Null hypothesis.

But when we normalize the results, we can get stationary data. That's why we'll utilize a method akin to normalized returns and the Dicky-Fuller test to determine stationarity where we define claim the Null hypothesis as the data is stationary.

```
benchmark_ret = eth_train.returns.iloc[0]
eth_train['norm_ret'] = eth_train.returns.div(benchmark_ret).mul(100)
eth_test['norm_ret'] = eth_test.returns.div(benchmark_ret).mul(100)
```

```
sts.adfuller(eth_train.norm_ret)
```

```
(-11.416317801866343,
 7.048091371372087e-21,
 9,
 1491,
 {'1%': -3.434743423170358,
  '5%': -2.8634804142964025,
  '10%': -2.567803054306163},
 17802.319193813113)
```

Figure 25

We can simply conclude that the data is stationary by noting that our test statistic falls to the left of our critical values where we accept the Null hypothesis/

We will experiment with AR models with various lags and use the log-likelihood test to compare our models, just as we did for pricing and returns. Here we define the Null hypothesis as the second (more complex model) does not perform better than the first one:

Dep. Variable:	norm_ret	**No. Observations:**	1501
Model:	ARIMA(2, 0, 0)	**Log Likelihood**	-9048.715
Date:	Sat, 04 Feb 2023	**AIC**	18105.430
Time:	20:21:27	**BIC**	18126.686
Sample:	11-11-2017	**HQIC**	18113.348
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	5.9962	2.660	2.254	0.024	0.782	11.211
ar.L1	-0.0445	0.019	-2.296	0.022	-0.082	-0.007
ar.L2	0.0579	0.024	2.404	0.016	0.011	0.105
sigma2	1.009e+04	190.612	52.954	0.000	9720.050	1.05e+04

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2025.50
Prob(Q):	0.99	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.90	**Skew:**	-0.26
Prob(H) (two-sided):	0.24	**Kurtosis:**	8.67

Table 24

Dep. Variable:	norm_ret	**No. Observations:**	1501
Model:	ARIMA(10, 0, 0)	**Log Likelihood**	-9039.367
Date:	Sat, 04 Feb 2023	**AIC**	18102.734
Time:	20:21:15	**BIC**	18166.501
Sample:	11-11-2017	**HQIC**	18126.489
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	5.9962	2.993	2.004	0.045	0.131	11.862
ar.L1	-0.0443	0.020	-2.216	0.027	-0.084	-0.005
ar.L2	0.0575	0.025	2.274	0.023	0.008	0.107
ar.L3	0.0074	0.024	0.309	0.757	-0.040	0.054
ar.L4	0.0325	0.021	1.580	0.114	-0.008	0.073
ar.L5	0.0099	0.023	0.437	0.662	-0.034	0.054
ar.L6	0.0679	0.024	2.821	0.005	0.021	0.115
ar.L7	-0.0026	0.020	-0.132	0.895	-0.042	0.036
ar.L8	-0.0399	0.025	-1.613	0.107	-0.088	0.009
ar.L9	-0.0351	0.026	-1.367	0.172	-0.085	0.015
ar.L10	0.0601	0.024	2.545	0.011	0.014	0.106
sigma2	1e+04	208.680	47.926	0.000	9592.210	1.04e+04

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2088.42
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.88	**Skew:**	-0.28
Prob(H) (two-sided):	0.14	**Kurtosis:**	8.75

Table 25

```
LLR_test(ar_model_norm_ret_1, ar_model_norm_ret_2)
```

```
0.0
```

Figure 26

The model with 10 lags has a greater log likelihood, as can be seen from the data above, and the LLR test indicates that the addition of 8 additional lags to the model with 2 lags has no high effect. So we reject the Null hypothesis. Also, we can see that normalizing has no appreciable influence on model choice by comparing our results to the unnormalized returns that we did earlier.

We will look at the residuals for the developed models in the final section of AR models. We first establish the residual columns in the datasets in order to accomplish this:

```
eth_train['res_price'] = results_ar_model_1.resid
eth_test['res_price'] = results_ar_model_1.resid
```

```
eth_train.res_price.mean()
```

```
2.245538426615259
```

```
eth_train.res_price.var()
```

```
5947.85755665728
```

Code Chunk 9

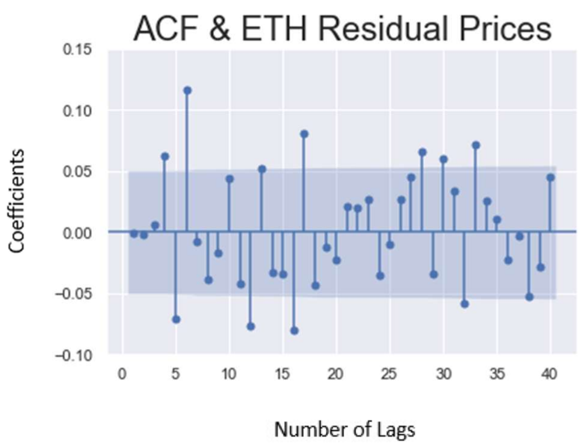
We will check the stationarity as usual where we define claim the Null hypothesis as the data is stationary:

```
sts.adfuller(eth_train.res_price[1:])
```

```
(-9.90433891325914,
 3.293878938828608e-17,
 16,
 1483,
 {'1%': -3.4347671645756304,
  '5%': -2.86349089226533,
  '10%': -2.5678086339403325},
 16916.052118069485)
```

Figure 27

We can say that the data is stationary as the test statistic falls on the left side of our critical values. So, we accept the Null hypothesis. Now that our data has been checked for the ACF, we can study the residuals:



Graph 41

Numerous coefficients outside of the confidence interval (blue region) may be seen in the figure above, leading us to conclude that there is a stronger predictor than residuals. Lastly, we will conduct a similar analysis of return residuals as we did for price and test the stationarity where we define claim the Null hypothesis as the data is stationary:

```
eth_train['res_price_ret'] = results_ar_model_ret_2.resid
eth_test['res_price_ret'] = results_ar_model_ret_2.resid
```

```
eth_train.res_price_ret.mean()
```

```
-1.857860593810645e-05
```

```
eth_train.res_price_ret.var()
```

```
26.63373497151356
```

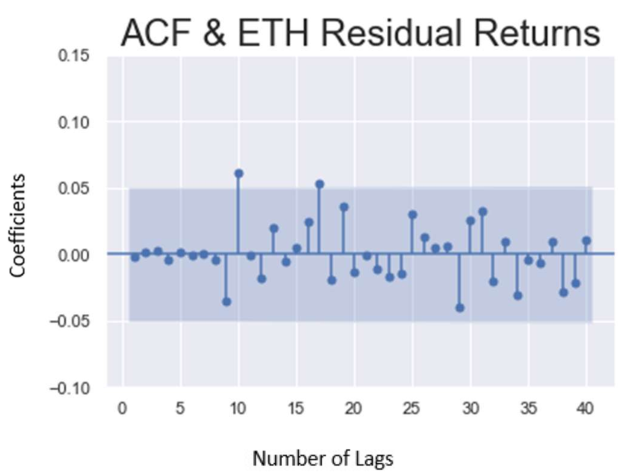
```
sts.adfuller(eth_train.res_price_ret[1:])
```

```
(-38.746625407896836,
 0.0,
 0,
 1499,
 {'1%': -3.4347199356122493,
  '5%': -2.86347004827819,
  '10%': -2.567797534300163},
 9031.09628608121)
```

Table 26

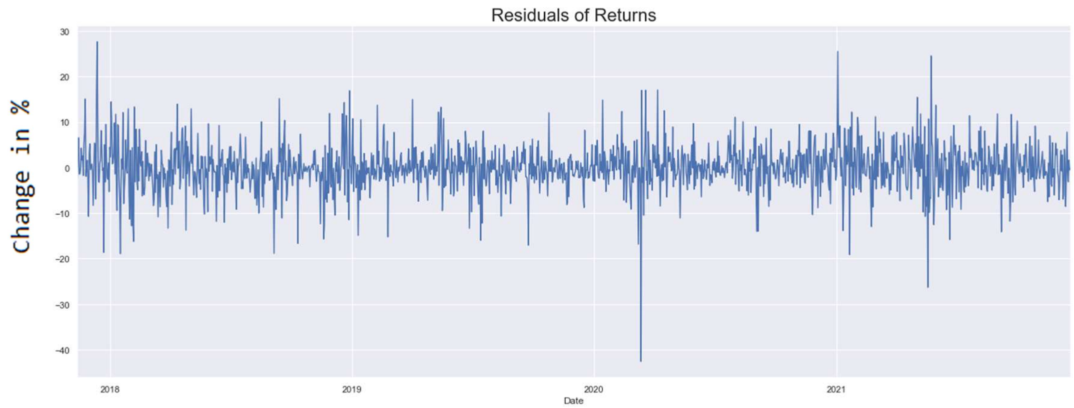
We can see that data is stationary based on the dicky fuller test statistic and 0.0 p-value which means we accept the Null hyptothesis.

We notice fewer coefficients outside the confidence interval when looking at the ACF, which indicates that our model is a solid predictor but that there is still a better one out there.



Graph 42

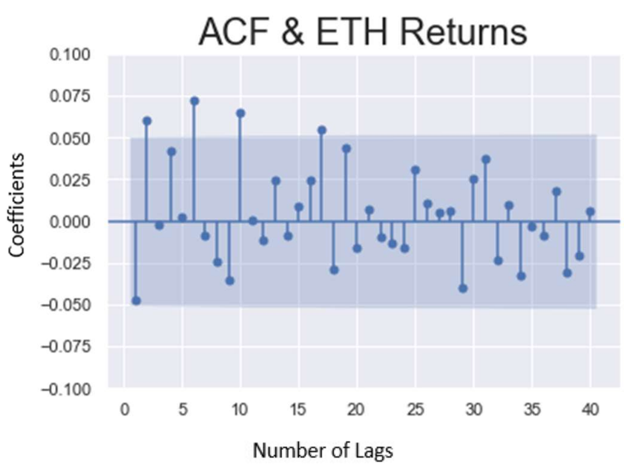
Finally, the graph below, which displays the price volatility over time, may be seen when we plot the residuals of returns. Prices substantially decreased in the second half of 2020 due to a market meltdown that few investors had anticipated.



Graph 43

4.2.4 MA models

First, we will establish our expectations for the number of lags that should be used. We must once more verify the ACF for Return closing prices in order to accomplish this.



Graph 44

The accompanying plot serves as a helpful reminder that the 2^{nd}, 6^{th} lags appear to be statistically significant, and that after the 17th lag, the lags become statistically insignificant. We can therefore assume that our model has fewer than 17 lags.

We will fit the models in the following phase. We will employ the models with 7 and 10 lags, as predicted by the ACF plot, and compute the LLR test to determine which model performs better:

MA mode with 2 lags **MA model with 6 lags:**

Dep. Variable:	returns	No. Observations:	1500
Model:	ARIMA(0, 0, 2)	Log Likelihood	-4595.308
Date:	Sun, 05 Feb 2023	AIC	9198.617
Time:	09:58:50	BIC	9219.869
Sample:	11-12-2017	HQIC	9206.534
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3059	0.137	2.236	0.025	0.038	0.574
ma.L1	-0.0445	0.019	-2.283	0.022	-0.083	-0.006
ma.L2	0.0558	0.024	2.316	0.021	0.009	0.103
sigma2	26.8212	0.507	52.930	0.000	25.828	27.814

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	2037.74
Prob(Q):	1.00	Prob(JB):	0.00
Heteroskedasticity (H):	0.90	Skew:	-0.26
Prob(H) (two-sided):	0.25	Kurtosis:	8.69

Table 27

Dep. Variable:	returns	No. Observations:	1500
Model:	ARIMA(0, 0, 6)	Log Likelihood	-4590.325
Date:	Sun, 05 Feb 2023	AIC	9196.649
Time:	10:02:30	BIC	9239.155
Sample:	11-12-2017	HQIC	9212.484
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3062	0.154	1.987	0.047	0.004	0.608
ma.L1	-0.0443	0.020	-2.219	0.027	-0.083	-0.005
ma.L2	0.0606	0.025	2.396	0.017	0.011	0.110
ma.L3	0.0038	0.024	0.161	0.872	-0.043	0.050
ma.L4	0.0321	0.020	1.565	0.118	-0.008	0.072
ma.L5	0.0093	0.022	0.428	0.669	-0.033	0.052
ma.L6	0.0737	0.024	3.100	0.002	0.027	0.120
sigma2	26.6425	0.552	48.261	0.000	25.560	27.724

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	2027.82
Prob(Q):	0.98	Prob(JB):	0.00
Heteroskedasticity (H):	0.90	Skew:	-0.26
Prob(H) (two-sided):	0.26	Kurtosis:	8.67

Table 28

```
print("\nLLR Test P-value = " + str(LLR_test(model_ret_ma_1,model_ret_ma_2, DF = 4)))
```

```
LLR Test P-value = 0.041
```

Figure 28

After number of trials, we can see that models are performing worse than 2 lag models until the 6 th lag models. This is because on the 6th lag there is an additional statistically significant coefficient. Taking this into consideration even though there are insignificant coefficients we will be using the model with 6 lags.

We will study the residuals for the model that performed better in the following step after adding a new column for residuals from MA models:

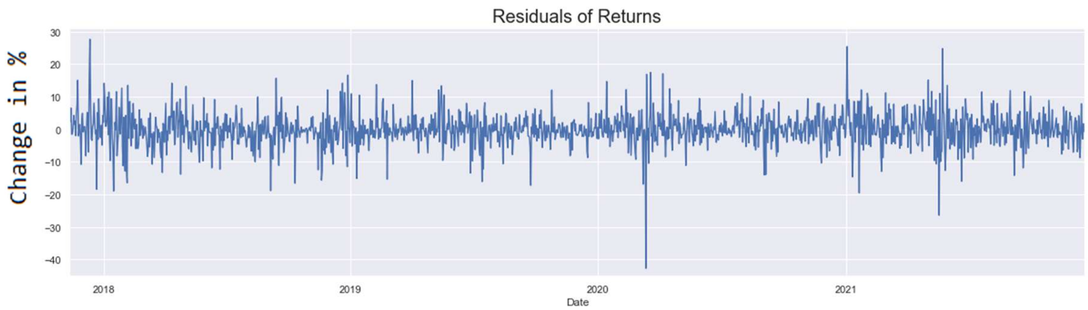
```
eth_train['res_ret_ma_2'] = results_ret_ma_2.resid[1:]
```

```
print("mean is " + str(round(eth_train.res_ret_ma_2.mean(),3)))
print("variance is " + str(round(eth_train.res_ret_ma_2.var(),3)))
print("Standard deviation is " + str(round(sqrt(eth_train.res_ret_ma_2.var()), 3)))
```

```
mean is 0.002
variance is 26.675
Standard deviation is 5.165
```

Code Chunk 10

To determine whether our model is sound or not, we must first look for residuals in the graph:



Graph 45

From the graph, it appears that the residuals are more random than they do consistently. We may use the Dicky Fuller test to check for stationary residuals and test for this randomness where we define claim the Null hypothesis as the data is stationary:

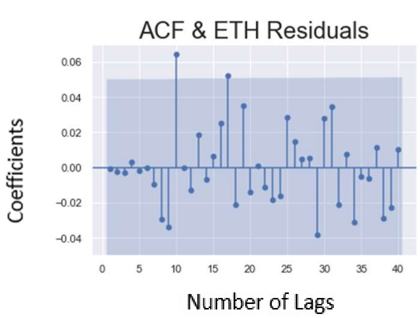
```
sts.adfuller(eth_train.res_ret_ma_2[2:])
```

```
(-38.70434351882141,
 0.0,
 0,
 1498,
 {'1%': -3.4347228578139943,
  '5%': -2.863471337969528,
  '10%': -2.5677982210726897},
 9027.645156416089)
```

Figure 29

As our p-value is 0.0, test statistic is smaller than the critical values we can state that the data is stationary and we accept the Null hypothesis.

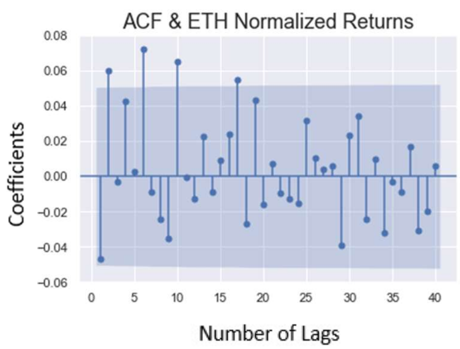
In order to determine whether the return residuals are White Noise or not, we can also look at the ACF of the residuals from our model:



Graph 46

We can see that several of the coefficients in the above graphs are significant. The first 10 legs of our model were added; therefore it was predicted that their coefficients would be near to zero. Additionally, the subsequent seven lags are similarly not significant, which shows how effectively our model works.

In this stage, we will examine how the MA models predict the previously constructed normalized data by visualizing the autocorrelation function. As each of the five cryptos has a completely separate price range, the goal of this step is to enable comparisons between ETH values and BTC near the conclusion of the thesis.



Graph 47

We can determine from above how many lags to employ when fitting the model to normalized returns based on coefficients that are outside of the confidence interval. To determine whether normalizing affects model selection, we will fit the model and look at the results:

Dep. Variable:	norm_ret	**No. Observations:**	1500
Model:	ARIMA(0, 0, 6)	**Log Likelihood**	-9037.983
Date:	Sun, 05 Feb 2023	**AIC**	18091.966
Time:	10:22:37	**BIC**	18134.472
Sample:	11-12-2017	**HQIC**	18107.801
	- 12-20-2021		
Covariance Type:	opg		

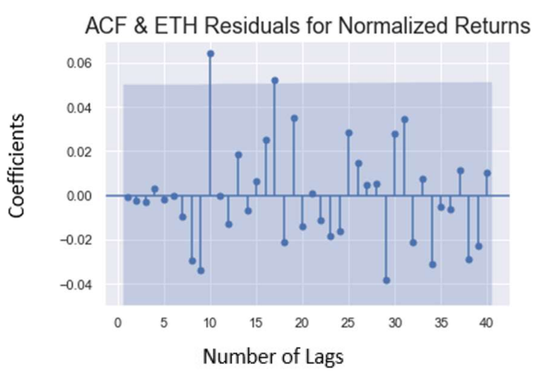
	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	5.9335	2.994	1.982	0.047	0.065	11.802
ma.L1	-0.0443	0.020	-2.214	0.027	-0.084	-0.005
ma.L2	0.0606	0.025	2.392	0.017	0.011	0.110
ma.L3	0.0038	0.024	0.161	0.872	-0.043	0.050
ma.L4	0.0321	0.021	1.562	0.118	-0.008	0.072
ma.L5	0.0093	0.022	0.427	0.669	-0.033	0.052
ma.L6	0.0736	0.024	3.094	0.002	0.027	0.120
sigma2	1.004e+04	208.508	48.167	0.000	9634.619	1.05e+04

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2027.81
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.90	**Skew:**	-0.26
Prob(H) (two-sided):	0.26	**Kurtosis:**	8.67

Table 29

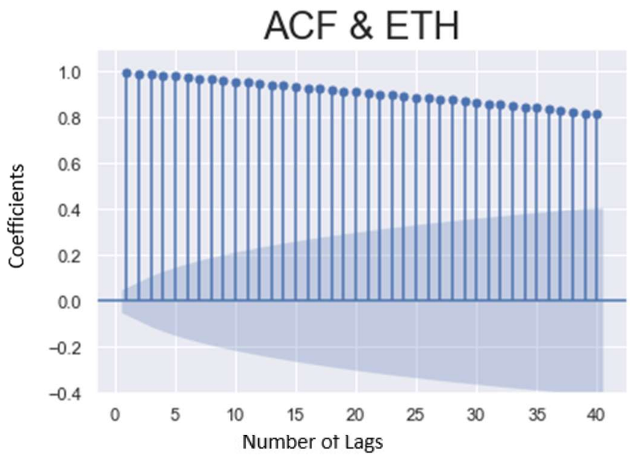
We can observe from the above that the outcomes of the MA model with 6 lags in normalized returns and non-normalized returns are nearly identical. Given this, we may conclude that normalizing values has no effect on model choice.

Finally, we will plot the residuals and ACF of residuals once more to demonstrate that our model was the right one. From the graph below, we can observe that even after the tenth lag, many coefficients are below the confidence level. Because of this, our model is accurate.



Graph 48

In the final section, we'll try to determine whether BTC closing prices can be forecast using MA models. We will begin by graphing ACF for Prices to establish the number of lags, much as we did in the preceding sections:



Graph 49

The assumption that any higher lag model will perform better than the one with fewer lags is derived from the fact that all of the coefficients are higher than the confidence level as shown above. Another notion that we can come up with is that situations like this would benefit from an endless number of lags. Since adding infinite lags is not possible, we can assume that MA models are not the best for predicting real close prices.

4.2.5 ARMA models

We will look into ARMA models in this part and attempt to fit them to our datasets.

We will first begin by fitting ARMA models to the returns and analysing the outcomes. We may be fitting the (2,1) and (4,3) models and evaluating the outcomes:

Dep. Variable:	returns	**No. Observations:**	1500
Model:	ARIMA(2, 0, 1)	**Log Likelihood**	-4594.296
Date:	Sun, 05 Feb 2023	**AIC**	9198.593
Time:	19:28:40	**BIC**	9225.159
Sample:	11-12-2017	**HQIC**	9208.489
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3075	0.156	1.972	0.049	0.002	0.613
ar.L1	0.7088	0.146	4.868	0.000	0.423	0.994
ar.L2	0.0790	0.022	3.637	0.000	0.036	0.122
ma.L1	-0.7570	0.146	-5.182	0.000	-1.043	-0.471
sigma2	26.7859	0.508	52.753	0.000	25.791	27.781

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	1996.32
Prob(Q):	0.96	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.91	**Skew:**	-0.25
Prob(H) (two-sided):	0.28	**Kurtosis:**	8.63

Table 30

Dep. Variable:	returns	**No. Observations:**	1500
Model:	ARIMA(4, 0, 3)	**Log Likelihood**	-4592.187
Date:	Sun, 05 Feb 2023	**AIC**	9202.374
Time:	19:30:16	**BIC**	9250.193
Sample:	11-12-2017	**HQIC**	9220.189
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3060	0.136	2.244	0.025	0.039	0.573
ar.L1	0.4299	0.172	2.493	0.013	0.092	0.768
ar.L2	-0.0078	0.185	-0.042	0.966	-0.371	0.355
ar.L3	-0.6632	0.161	-4.125	0.000	-0.978	-0.348
ar.L4	0.0355	0.030	1.202	0.229	-0.022	0.093
ma.L1	-0.4733	0.173	-2.737	0.006	-0.812	-0.134
ma.L2	0.0662	0.200	0.331	0.741	-0.326	0.459
ma.L3	0.6228	0.176	3.531	0.000	0.277	0.968
sigma2	26.7031	0.540	49.487	0.000	25.646	27.761

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	1830.23
Prob(Q):	0.97	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.89	**Skew:**	-0.27
Prob(H) (two-sided):	0.20	**Kurtosis:**	8.39

Table 31

The lags in the models (2,1) are statistically significant, however, the identical values in the models (4,3) were significantly above the significance level when comparing the models. For this reason, we assume that the first model will match our data crucially better. Additionally, the LLR test demonstrates that the first model performs significantly better than the second model with 4,3 lags respectively:

```
LLR_test(model_ret_arma_1, model_ret_arma_2, DF = 4)
```

```
0.377
```

Figure 30

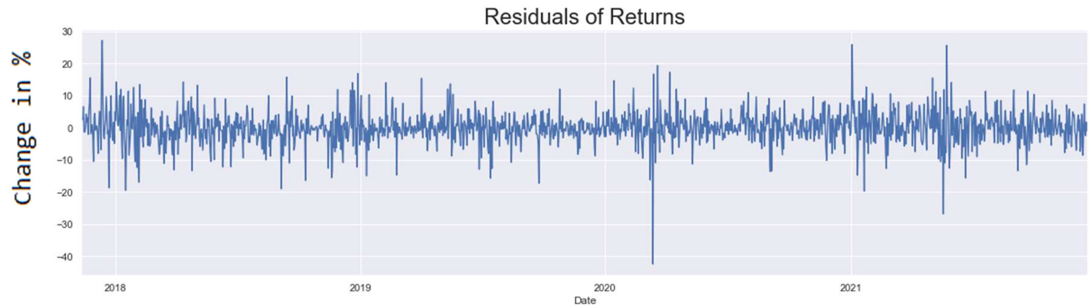
Lastly, when we compare AIC values we can see that ARMA (2,2) has less information criteria which is a better indicator:

```
print("ARMA(2,1) AIC Value is " + str(results_ret_arma_1.aic))
print("ARMA(4,3) AIC Value is " + str(results_ret_arma_2.aic))
```

```
ARMA(2,1) AIC Value is 9198.592608872896
ARMA(4,3) AIC Value is 9202.37437426682
```

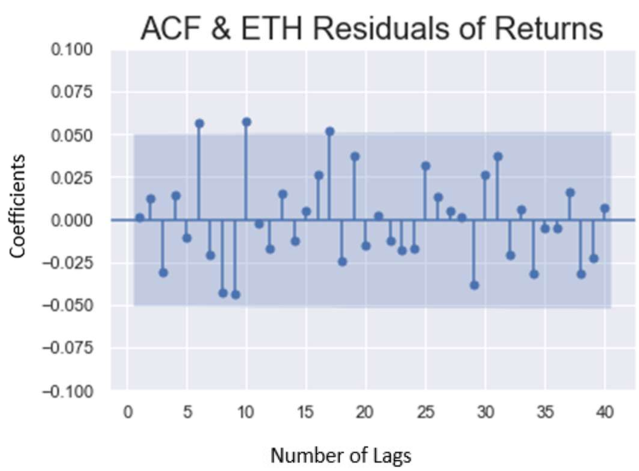
Figure 31

We will analyse the ARMA model's residuals in the same way that we have in the past for other models. To do this, we will plot another column that contains the residuals from our ARMA model (2,1):



Graph 50

The outcomes are comparable to what we previously discovered using AR and MA models. This suggests that if the ARMA model is used alone, the volatility in returns cannot be fully comprehended. We must still plot the autocorrelation function to determine whether the residuals are random.



Graph 51

Most of the lags are falling within the confidence level, which allows us to conclude that the residuals are random, according to the ACF, which we can see.

Finally, we will test the performance of ARMA models on stationary data by using them near BTC prices. We will fit the ARMA models (2,1), which were our choice for returns, as well as the ARMA model (4,3), which is the model till we receive some coefficients over the significance level and also define our Null hypothesis for the LLR test that the more complex model (4,3) performs better than eh ARMA (2,1):

Dep. Variable:	Close	**No. Observations:**	1501
Model:	ARIMA(2, 0, 1)	**Log Likelihood**	-8621.952
Date:	Sun, 05 Feb 2023	**AIC**	17253.903
Time:	19:50:01	**BIC**	17280.473
Sample:	11-11-2017	**HQIC**	17263.801
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	901.9933	4246.570	0.212	0.832	-7421.130	9225.117
ar.L1	0.1776	0.038	4.723	0.000	0.104	0.251
ar.L2	0.8206	0.038	21.762	0.000	0.747	0.895
ma.L1	0.7546	0.045	16.701	0.000	0.666	0.843
sigma2	5689.9125	64.767	87.852	0.000	5562.972	5816.853

Ljung-Box (L1) (Q):	0.94	**Jarque-Bera (JB):**	32269.73
Prob(Q):	0.33	**Prob(JB):**	0.00
Heteroskedasticity (H):	13.78	**Skew:**	-0.87
Prob(H) (two-sided):	0.00	**Kurtosis:**	25.65

Graph 52

Dep. Variable:	Close	**No. Observations:**	1501
Model:	ARIMA(4, 0, 3)	**Log Likelihood**	-8604.261
Date:	Sun, 05 Feb 2023	**AIC**	17226.521
Time:	19:50:11	**BIC**	17274.346
Sample:	11-11-2017	**HQIC**	17244.337
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	901.9887	7667.143	0.118	0.906	-1.41e+04	1.59e+04
ar.L1	-0.4027	0.039	-10.316	0.000	-0.479	-0.326
ar.L2	-0.0239	0.021	-1.150	0.250	-0.065	0.017
ar.L3	0.6351	0.021	30.485	0.000	0.594	0.676
ar.L4	0.7892	0.040	19.785	0.000	0.711	0.867
ma.L1	1.2980	0.046	28.035	0.000	1.207	1.389
ma.L2	1.3148	0.033	40.120	0.000	1.251	1.379
ma.L3	0.6887	0.047	14.568	0.000	0.596	0.781
sigma2	5628.1247	69.517	80.961	0.000	5491.874	5764.375

Ljung-Box (L1) (Q):	0.50	**Jarque-Bera (JB):**	29975.64
Prob(Q):	0.48	**Prob(JB):**	0.00
Heteroskedasticity (H):	13.15	**Skew:**	-0.84
Prob(H) (two-sided):	0.00	**Kurtosis:**	24.83

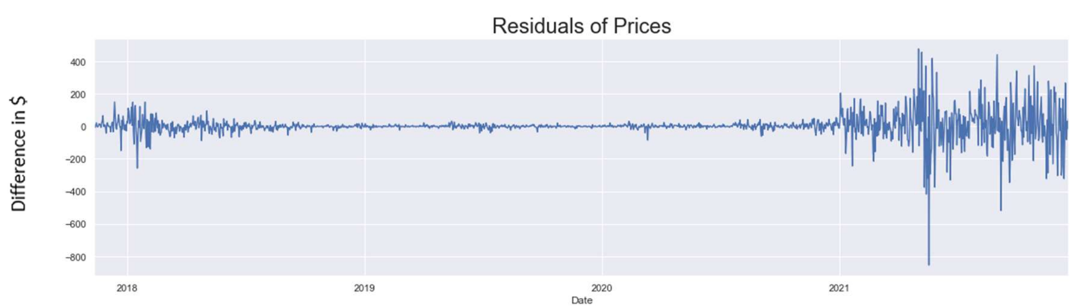
Graph 53

```
LLR_test(model_arma_1, model_arma_2, DF = 4)
0.0
```

Figure 32

From the above results, we can see that even though model 2,1 performed better for the returns it is not the case when it comes to the prices. For prices, the higher lag model has better results in the LLR test and the lower AIC value confirms our conclusion. So we are accepting the Null hypothesis.

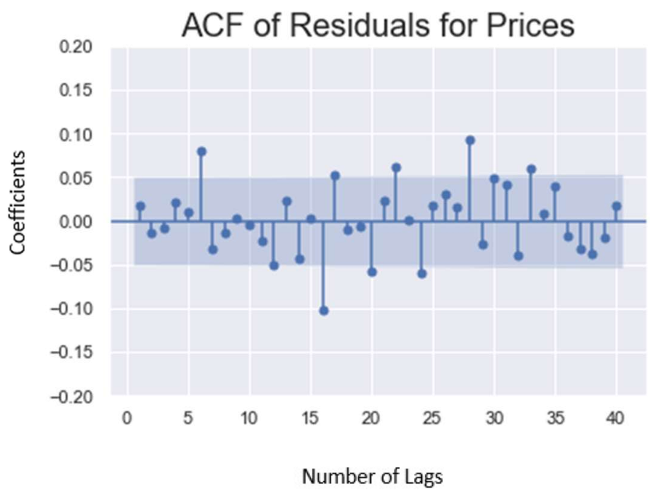
In the final part of ARMA models, we can analyse the residual values of ETH for close prices.



Graph 54

In the preceding residual price graphs, which can be seen by looking at the above plot, the prices exhibit similar tendencies. While there was a significant discrepancy between projected and actual numbers in 2018 and 2021, it appears that there may be some volatility in those years.

Last but not least, we plot the ACF of residuals and review the outcomes:

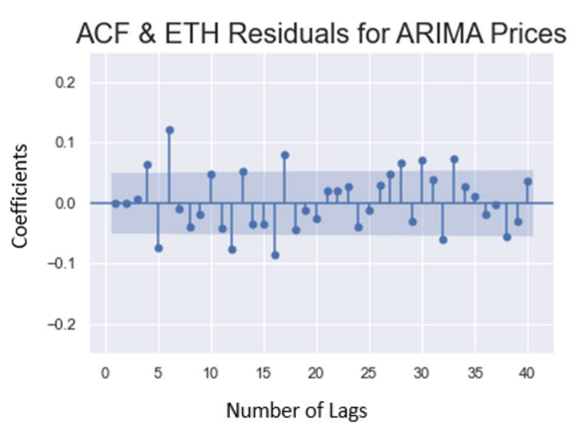


Graph 55

We can observe from the graph above that there are numerous lags that are noticeably non-zero. By taking this into account, we can state that the price residuals are not random.

4.2.6 ARIMA models

In this section, we will analyse the outcomes of fitting various ARIMA models to our data. We can begin by plotting the ACF of residuals for ARIMA (1,1,1) in order to determine the lags:



Graph 56

We can see from the autocorrelation graph that it could be beneficial to include the 5th or 6th lags in our model because they are substantial. Let's try fitting several models with three delays as we often favor simpler models and evaluate the results using likelihood, AIC, and log likelihood ratio tests. Here we define the Null hypothesis as the model ARIMA (1,1,3) perform better than the others:

Fitted Models:

```
# 1
model_arima_1 = ARIMA(eth_train.Close[1:], order = (1,1,1))
results_model_arima_1 = model_arima_1.fit()
results_model_arima1.summary()
# 2
model_arima_2 = ARIMA(eth_train.Close[1:], order = (1,1,2))
results_model_arima_2 = model_arima_2.fit()
results_model_arima_2.summary()
# 3
model_arima_3 = ARIMA(eth_train.Close[1:], order = (1,1,3))
results_model_arima_3 = model_arima_3.fit()
results_model_arima_3.summary()
# 4
model_arima_4 = ARIMA(eth_train.Close[1:], order = (2,1,1))
results_model_arima_4 = model_arima_4.fit()
# 5
model_arima_5 = ARIMA(eth_train.Close[1:], order = (3,1,1))
results_model_arima_5 = model_arima_5.fit()

# 6
model_arima_6 = ARIMA(eth_train.Close[1:], order = (3,1,2))
results_model_arima_6 = model_arima_6.fit()
```

Code Chunk 11

Results of LL and AIC:

```
print("ARIMA(1,1,1):  \t LL = ", results_model_arima_1.llf, "\t AIC = ", results_model_arima_1.aic)
print("ARIMA(1,1,2):  \t LL = ", results_model_arima_2.llf, "\t AIC = ", results_model_arima_2.aic)
print("ARIMA(1,1,3):  \t LL = ", results_model_arima_3.llf, "\t AIC = ", results_model_arima_3.aic)
print("ARIMA(2,1,1):  \t LL = ", results_model_arima_4.llf, "\t AIC = ", results_model_arima_4.aic)
print("ARIMA(3,1,1):  \t LL = ", results_model_arima_5.llf, "\t AIC = ", results_model_arima_5.aic)
print("ARIMA(3,1,2):  \t LL = ", results_model_arima_6.llf, "\t AIC = ", results_model_arima_6.aic)
```

```
ARIMA(1,1,1):    LL =  -8610.795256554851       AIC =  17227.590513109702
ARIMA(1,1,2):    LL =  -8610.12597071617        AIC =  17228.25194143234
ARIMA(1,1,3):    LL =  -8605.290078714881       AIC =  17220.580157429762
ARIMA(2,1,1):    LL =  -8607.247994566285       AIC =  17222.49598913257
ARIMA(3,1,1):    LL =  -8610.15517355717        AIC =  17230.31034711434
ARIMA(3,1,2):    LL =  -8607.247371915331       AIC =  17226.494743830663
```

Figure 33

Because ARIMA (1,1,3) has a lower AIC and a larger LL, we can infer that it might perform better than the other models. Finally, we can use the LLR test to validate this assertion and make the following claim:

```
print("\nLLR test p-value = " + str(LLR_test(model_arima_1, model_arima_3, DF = 2)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_2, model_arima_3)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_4, model_arima_3)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_5, model_arima_3)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_6, model_arima_3)))
```

```
LLR test p-value = 0.004

LLR test p-value = 0.002

LLR test p-value = 0.048

LLR test p-value = 0.002

LLR test p-value = 0.048
```

Figure 34

From the results it is obvious that the ARIMA (1,1,3) which is the respective third model at the above outperforms the other models. So, we accept the Null hypothesis. Lastly, lets plot the residuals for this model and examine the results:

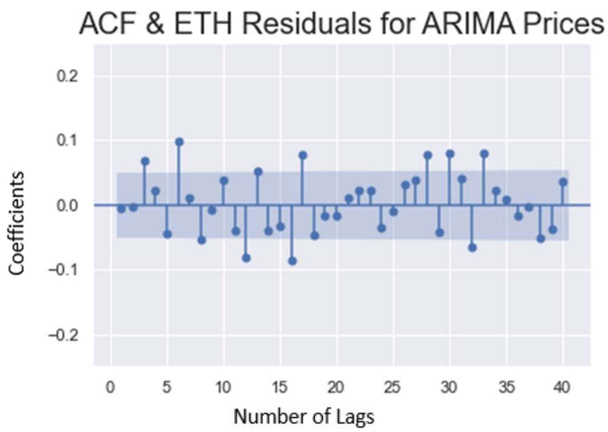


Figure 35

Our data must originate from a non-stationary process, as we are aware, in order to leverage higher integration levels. The Dicky-Fueller test will be used to determine whether integrated data is stationary or not after manually creating an integrated delta pricing column using Python's diff function where we define claim the Null hypothesis as the data is stationary:

```
eth_train['delta_prices']=eth_train.Close.diff(1)
```

```
sts.adfuller(eth_train.delta_prices[1:])
```

```
(-10.144321399562491,
 8.262347504369866e-18,
 16,
 1483,
 {'1%': -3.4347671645756304,
  '5%': -2.86349089226533,
  '10%': -2.5678086339403325},
 16914.969030305292)
```

Figure 36

We can see from the results above that, at all three levels, our test statistic is higher than our critical values. Additionally, the p-value is quite close to 0, allowing us to conclude that the data is stationary. So, we accept the Null hypothesis. Due to the fact that, 1 level of integration in ARIMA models is sufficient, we can readily advise against using more integrated levels.

4.2.7 ARIMAX model

The ARIMAX model will be used in the following phase to integrate external factors that have an impact on prices. Python allows for the use of so-called exogenous variables when fitting models. To check for a correlation between ETH and BTC, we will use the price of Ethereum.

Dep. Variable:	Close	**No. Observations:**	1501
Model:	ARIMA(2, 1, 2)	**Log Likelihood**	-8069.353
Date:	Mon, 06 Feb 2023	**AIC**	16150.706
Time:	19:41:53	**BIC**	16182.586
Sample:	11-11-2017	**HQIC**	16162.583
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
Close	0.0541	0.001	87.804	0.000	0.053	0.055
ar.L1	0.0408	0.065	0.624	0.533	-0.087	0.169
ar.L2	0.7108	0.062	11.434	0.000	0.589	0.833
ma.L1	-0.0804	0.074	-1.089	0.276	-0.225	0.064
ma.L2	-0.6386	0.068	-9.448	0.000	-0.771	-0.506
sigma2	2759.2403	39.691	69.518	0.000	2681.447	2837.033

Ljung-Box (L1) (Q):	0.29	**Jarque-Bera (JB):**	23797.11
Prob(Q):	0.59	**Prob(JB):**	0.00
Heteroskedasticity (H):	8.54	**Skew:**	0.22
Prob(H) (two-sided):	0.00	**Kurtosis:**	22.51

Table 32

Dep. Variable:	Close	**No. Observations:**	1501
Model:	ARIMA(1, 1, 1)	**Log Likelihood**	-8070.893
Date:	Thu, 09 Feb 2023	**AIC**	16149.787
Time:	07:19:59	**BIC**	16171.040
Sample:	11-11-2017	**HQIC**	16157.704
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
Close	0.0539	0.001	91.295	0.000	0.053	0.055
ar.L1	-0.8277	0.040	-20.582	0.000	-0.906	-0.749
ma.L1	0.7707	0.048	16.070	0.000	0.677	0.865
sigma2	2760.8465	36.778	75.067	0.000	2688.762	2832.931

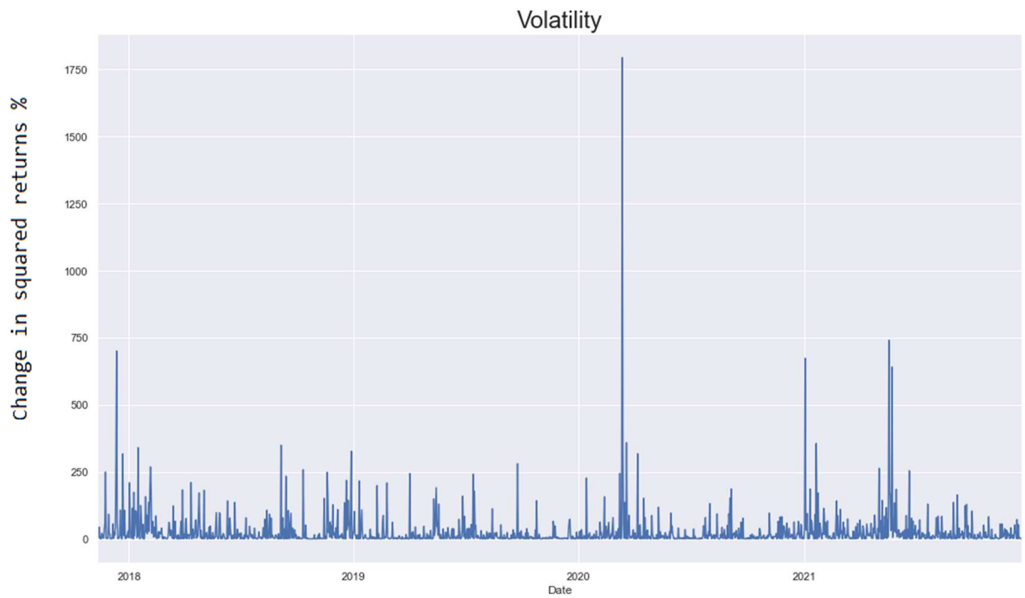
Ljung-Box (L1) (Q):	1.53	**Jarque-Bera (JB):**	24575.34
Prob(Q):	0.22	**Prob(JB):**	0.00
Heteroskedasticity (H):	8.57	**Skew:**	0.31
Prob(H) (two-sided):	0.00	**Kurtosis:**	22.82

Table 33

As it can be seen above the p-value of the close prices of BTC is statistically significant for our prices in ETH in the model where we have 2 lags in each side of model(AR, MA). Furthermore, looking at the log likelihood we expect a great fit to the dataset in the first model which we will illustrate in the forecasting section of the ETH.

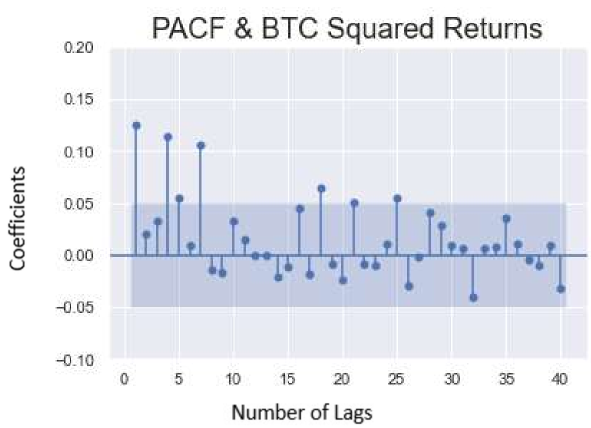
4.2.8 ARCH and GARCH Models

In this part, we will analyze the return volatility using ARCH models. We'll make another column and use the squared of returns as our volatility values before we start testing the models. The plot below shows that, as may be predicted, the returns for ETH appear to have substantial volatility.



Graph 57

Next, although though PACF cannot help us determine the number of delays to be utilized in the ARCH model, it may still provide us with a wealth of useful information:



Graph 58

The data above show that, of the seven initial lags, only 4 are statistically significant. Such high PACF scores may indicate that short-term patterns in variances are common.

Now, we will fit the ARCH model with constant mean with 5 iterations

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	ARCH	**Log-Likelihood:**	-4584.43
Distribution:	Normal	**AIC:**	9174.85
Method:	Maximum Likelihood	**BIC:**	9190.79
		No. Observations:	1500
Date:	Mon, Feb 06 2023	**Df Residuals:**	1499
Time:	18:18:59	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2893	0.128	2.267	2.339e-02	[3.919e-02, 0.539]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	24.0554	2.094	11.485	1.566e-30	[19.950, 28.160]
alpha[1]	0.1080	4.793e-02	2.253	2.429e-02	[1.403e-02, 0.202]

Table 34

We can see from the results above that R squared is zero when adjusted and not adjusted. R squared, which is used to assess explanatory variation in relation to the mean, indicates that it will not be particularly helpful in explaining the deviation for our ARCH model. Moving on to log likelihood, we can observe that ARCH models have a higher log likelihood value than our prior AR, MA, ARMA, and ARIMA models, indicating that simpler ARCH models can outperform complicated ARIMA models in estimations.

Second, we will construct a ARCH model with 2 lags and contrast the outcomes with the first.

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	ARCH	**Log-Likelihood:**	-4581.12
Distribution:	Normal	**AIC:**	9170.24
Method:	Maximum Likelihood	**BIC:**	9191.49
		No. Observations:	1500
Date:	Mon, Feb 06 2023	**Df Residuals:**	1499
Time:	18:24:40	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.3124	0.126	2.477	1.325e-02	[6.522e-02, 0.560]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	22.5594	2.610	8.643	5.458e-18	[17.444, 27.675]
alpha[1]	0.1037	4.948e-02	2.097	3.600e-02	[6.777e-03, 0.201]
alpha[2]	0.0644	5.161e-02	1.247	0.212	[-3.678e-02, 0.166]

Table 35

When we employed 2 lags, the log-likelihood increased while the AIC fell, which is immediately apparent. These two are already signs that the second model performs better than the previous one. Finally, when looking at the coefficients (p-values), we can see that all of the figures—aside from alpha 2—are statistically significant. Overall, we can still say that the second model, which has three delays, outperforms the first one in terms of estimating market volatility.

We will fit GARCH models, which are an extension of ARCH and are also known as the "ARMA Equivalent" of ARCH and are typically predicted to perform better, in the final section of this sub-chapter. We will compare the outcomes of fitting both basic and multi-lag GARCH models:

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	GARCH	**Log-Likelihood:**	-4539.37
Distribution:	Normal	**AIC:**	9086.75
Method:	Maximum Likelihood	**BIC:**	9108.00
		No. Observations:	1500
Date:	Mon, Feb 06 2023	**Df Residuals:**	1499
Time:	18:27:21	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2676	0.119	2.245	2.478e-02	[3.397e-02, 0.501]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	1.5002	0.898	1.671	9.464e-02	[-0.259, 3.259]
alpha[1]	0.0727	3.256e-02	2.233	2.556e-02	[8.884e-03, 0.137]
beta[1]	0.8727	5.708e-02	15.291	8.800e-53	[0.761, 0.985]

Table 36

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	GARCH	**Log-Likelihood:**	-4539.29
Distribution:	Normal	**AIC:**	9088.58
Method:	Maximum Likelihood	**BIC:**	9115.15
		No. Observations:	1500
Date:	Mon, Feb 06 2023	**Df Residuals:**	1499
Time:	18:28:00	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2692	0.119	2.263	2.367e-02	[3.600e-02, 0.502]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	1.6650	0.918	1.813	6.985e-02	[-0.135, 3.465]
alpha[1]	0.0825	3.462e-02	2.382	1.721e-02	[1.462e-02, 0.150]
beta[1]	0.6945	0.384	1.809	7.050e-02	[-5.809e-02, 1.447]
beta[2]	0.1625	0.362	0.449	0.654	[-0.547, 0.872]

Table 37

Looking at the above, it is clear that a simple GARCH model with the parameters (1,1) outperforms a more complicated GARCH model based on having beta[2] bigger than 0,5, which denotes that the difference is not significant. In light of this, we will continue to use the GARCH (1,1).

Finally, we can see that the ARCH model has a higher log probability when compared to the GARCH (1,1) model with three lags. From that angle, we may say that the ARCH model would be a better fit to estimate the volatility of ETH.

5 Results and Discussion

In this section we will share the predictions of each of the cryptocurrencies for AR, MA, ARMA and ARIMAX models implemented and then derive the results.

5.1 Forecasting of BTC

In this section we will use python predict method to make forecasts with the models that we have built and plot the predictions vs actual values from our testing dataset.

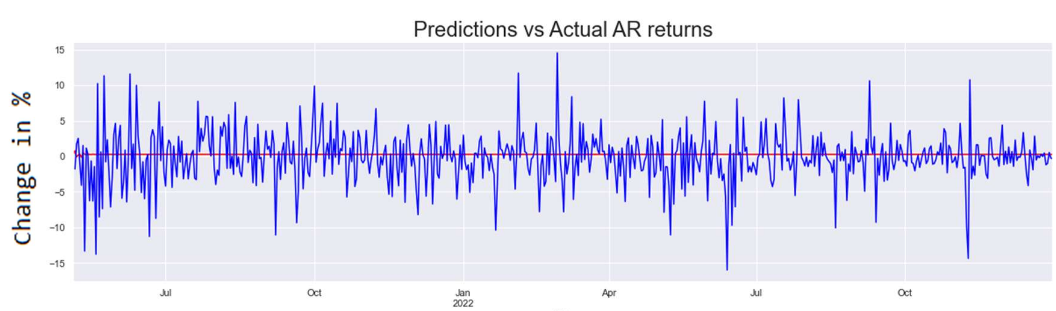
5.1.1 Forecasting with AR model

Initially we can start with our AR models for BTC prices:

Graph 59

From the above graph we can see that AR model is not performing well with prices. Here the main reason is because AR models are based on constants and it performs poorly with non-stationary datasets.

As we have already found out that the returns of BTC prices are stationary we can plot our predictions with the chosen AR return model and plot the predicted and actual values.

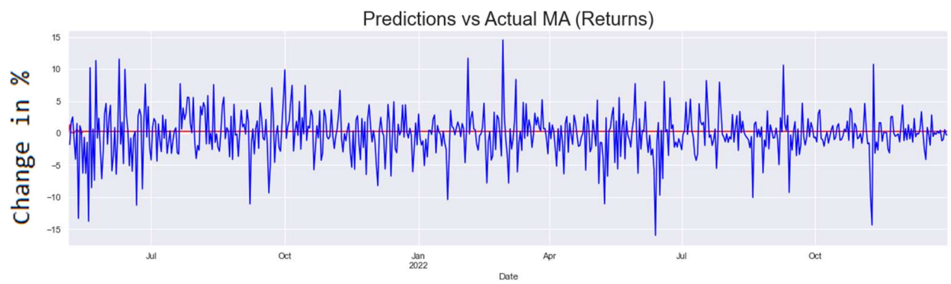


Graph 60

From the plot we can see that our model (red line) makes no assumptions as it predicts the future returns will be either 0 or very close to 0.

5.1.2 Forecasting with MA model

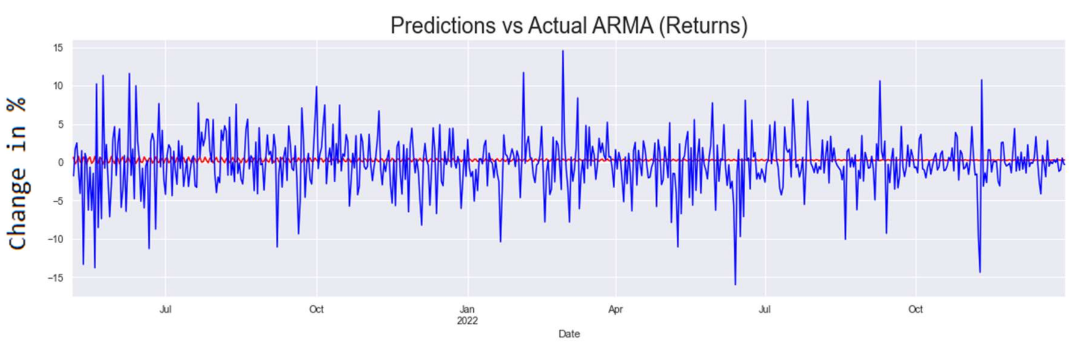
When also try MA models in order to see how well they perform in forecasting of returns we have the similar results as we had in AR models where can see the poor performance by the model with constant prediction



Graph 61

5.1.3 Forecasting with ARMA model

Next, we can also try to predict using the ARMA model and examine the results:

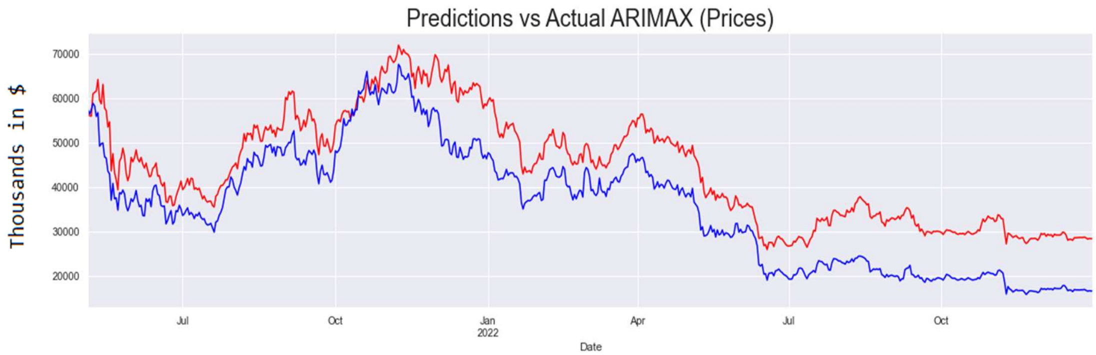


Graph 62

From ARMA model, even though it does not have one constant value for the whole period we cannot still conclude that this model performs well when it comes to predicting the returns.

5.1.4 Forecasting with ARIMAX

Finally, we will use the Ethereum data as exogenous variable and try to use our ARIMAX model in order to to our forecasts and see how well the model performs:



Graph 63

Looking at the above plot we can see that our model with ARIMAX perform significantly better than other models that we had before. It shows the correct trends even though sometimes it does overperform and sometimes underperform. From the forecast we can also conclude that adding more exogenous variables can increase the performance of model significantly.

5.2 Forecasting of ETH

In this section, we'll create predictions using the models we've built using the Python predict method and plot those predictions against the actual values from our testing dataset.

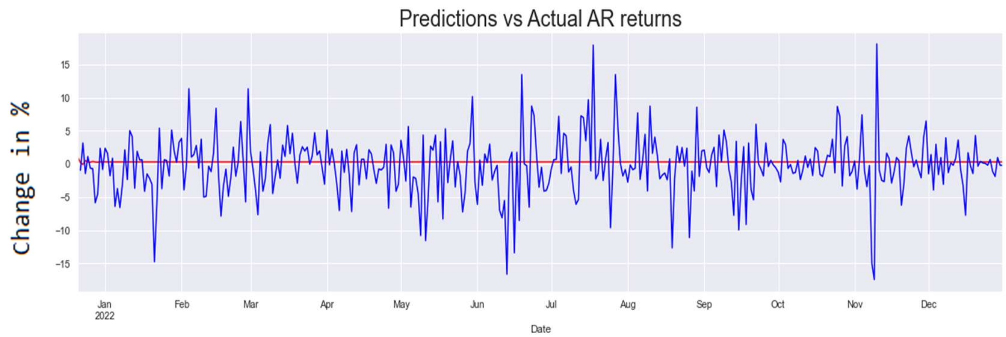
5.2.1 Forecasting with AR model

We can start with our AR models for ETH pricing initially:

Graph 64

We can observe from the graph above that the AR model does not work well with prices. Here, the fundamental cause is that AR models' performance with non-stationary datasets is weak because they are reliant on constants.

We may plot our predictions using the selected AR return model and plot the projected and actual values because we have already established that the returns of ETH prices are stationary.

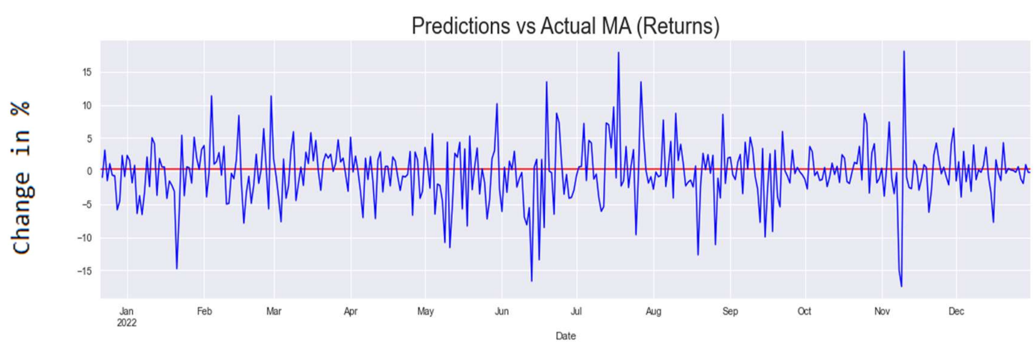


Graph 65

The plot shows that our model (red line), which forecasts that future returns would either be zero or extremely close to zero, contains no assumptions.

5.2.2 Forecasting with MA model

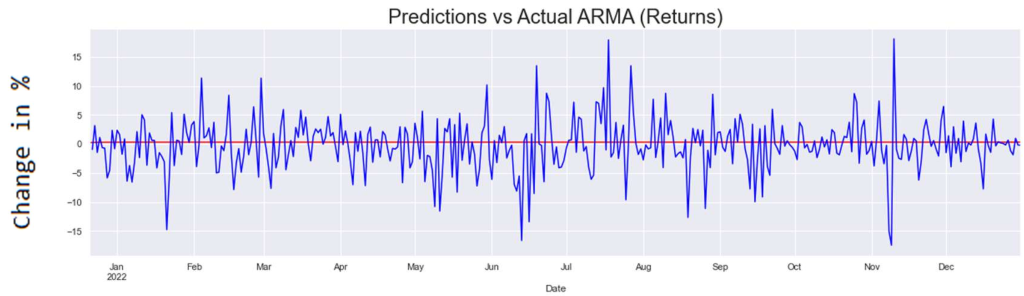
We get comparable outcomes to what we saw with AR models, where we could show the model's poor performance with constant prediction, when we also test MA models to see how well they do in projecting returns.



Graph 66

5.2.3 Forecasting with ARMA model

Next, we can also try to predict using the ARMA model and examine the results:

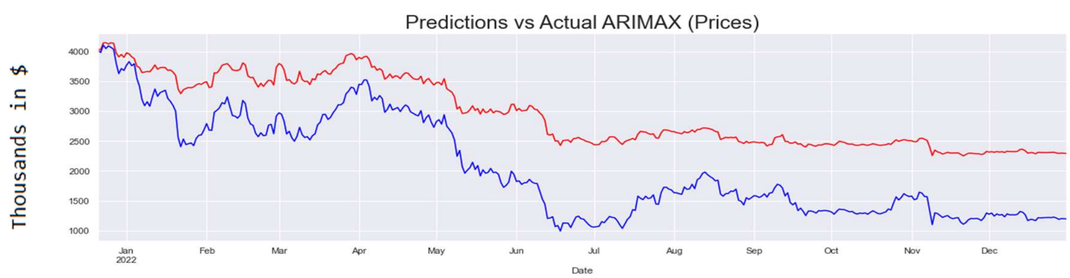


Graph 67

We cannot infer from the ARMA model that this model is effective at predicting returns despite the fact that it does not have a single constant value for the entire period.

5.2.4 Forecasting wiht ARIMAX

Finally, we will attempt to use our ARIMAX model to make our forecasts using the BTC data as an exogenous variable and test the model's performance:



Graph 68

As seen in the aforementioned graphic, our model with ARIMAX performs noticeably better than other models we had previously. Despite the fact that it occasionally outperforms and occasionally underperforms, it always displays the right trends. We can infer from the forecast that significantly more exogenous factors can improve the model's performance.

6 Conclusion

In conclusion it is important to mention that after assessing different time-series models it may be useful to do detailed multivariate analysis and understand the relationship between other factors to in order to make more accurate predictions. Taking into consideration the fact that this thesis mainly covered the univariate analysis and only used one exogenous variable during model fitting, the ARIMAX model performed the best. In spite of the fact of having not ideal predictions on prices, using ARIMAX, we were able to detect the trends on testing dataset very accurately which points one of the aims of the paper. Furthermore, thesis also explained that it is crucial to analyse the time series analysis of cryptocurrencies with more than 1 exogenous variable and heavily leaning on multivariate analysis of these coins. Thesis covered detailed analysis of each crypto currency the existence of seasonality, stationarity and other important indicators which enables us to have more enlightened comprehension on the BTC, ETH and in crypto world generally considering these coins as leading coins. Different models of time series also showed us on which kind of datasets can be the best fit for those models based on the characteristics of the respective datasets. As those characters can play a significant role on the course of the analysis and also have a great impact on the decision to be made on the model. Other time series properties such as auto correlation functions and partial autocorrelation functions helped us to define the number of lags and their importance to the data set in modelling. As crypto market is considered as one of the most volatile financial instruments to be traded in the current world it will be important to mention the final note which is the necessity of mentioning the description of thesis on the functionalities of time series with different models and assessing the volatility with ARCH and GARCH models. Using all the indicators such as returns, residuals and prices we were able to confirm that the volatility in the selected crypto currencies indeed exists.

7 References

Parizo, C. (2021). "What are the 4 different types of blockchain technology?", Techtarget https://www.techtarget.com/searchcio/feature/What-are-the-4-different-types-of-blockchain-technology

Brownlee, J, (2017), "White Noise Time Series", machinelearninfmastery https://machinelearningmastery.com/white-noise-time-series-python/

Daniel, D. (2016). "Applied Univariate, Bivariate and Multivariate Statistics"
ISBN 978-1-118-63233-8

Prabhakaran, S. (2019). "Vector Autoregression (VAR) – Comprehensive Guide with Examples in Python", Machinelearningplus
https://www.machinelearningplus.com/time-series/vector-autoregression-examples-python/

Amadebai, E.(N.D). "5 Methods of Collecting Data", analyticsfordecision.
https://www.analyticsfordecisions.com/methods-of-collecting-data/

Palma, W. (2016). "Time Series Analysis"
ISBN 978-1-118-63432-5

Grabowski, M. (2019). "Cryptocurrencies: A Primer on Digital Money"
ISBN 978-0-367-19267-9

Reiff, N. (2022, July 6). "What Are ERC-20 Tokens on the Ethereum Network?, investopedia
https://www.investopedia.com/news/what-erc20-and-what-does-it-mean-ethereum/

Janssen, J. (2013). "VaR methodology for Nan-Gaussian Finance"
ISBN 978-1-84821-464-4

Brownlee, J. (2018). "A Gentle Introduction to Exponential Smoothing for Time Series Forecasting in Python", machinelearningmastery
https://machinelearningmastery.com/exponential-smoothing-for-time-series-forecasting-in-python/

Chowdhury, N. (2019). “Inside Blockchain, Bitcoin, and Cryptocurrencies”.
ISBN 978-1-00050-770-6.

Cointelegraph, (N.D). “Ripple (XRP): A beginner's guide to the digital asset built for global payments”
https://cointelegraph.com/blockchain-for-beginners/what-is-ripple-a-beginners-guide-for-understanding-ripple

CFI team, (2022). “Binance Coin(BNB)”, corporatefinanceinstitute
https://corporatefinanceinstitute.com/resources/cryptocurrency/binance-coin-bnb/#:~:text=Binance%20Coin%20(BNB)%20is%20a,1.4%20million%20transactions%20per%20second.

Coinbase. (N.D). “What is Cardano?”
https://www.coinbase.com/learn/crypto-basics/what-is-cardano

Sankrit, K. (2022). “What is Bitcoin dominance?”. MoonPay
https://www.moonpay.com/blog/what-is-bitcoin-dominance

Shetty, C. (2020) “Time Series Models”. towardsdatascience
https://towardsdatascience.com/time-series-models-d9266f8ac7b0

Engle, R (N.D) “An Introduction to the Use of ARCH/GARCH models in Applied Econometrics”
https://web-static.stern.nyu.edu/rengle/GARCH101.PDF

Brownlee, J(2016). “How to Normalize and Standardize Time Series Data in Python”. Machinelearningmastery
https://machinelearningmastery.com/normalize-standardize-time-series-data-python/

Hyndman R(2021). “Forecasting: Principles and Practice”. Otexts
https://otexts.com/fpp3/intro.html

Monigatti, L, (2022). “Interpreting ACF and PACF Plots for Time Series Forecasting”. Towardsdatascience
https://towardsdatascience.com/interpreting-acf-and-pacf-plots-for-time-series-forecasting-af0d6db4061c

Brownlee, J(2016). “How to Identify and Remove Seasonality fromTime Series Data with Python”. Machinelearningmastery
https://machinelearningmastery.com/time-series-seasonality-with-python/

Palachy, S, (2019). “Stationarity in time series analysis”. Towardsdatascience
https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322

Smarten, (2018). “What is ARIMAX Forecasting and how is it used for Enterprise Analysis? ”.
https://www.elegantjbi.com/blog/what-is-arimax-forecasting-and-how-is-it-used-for-enterprise-analysis.htm

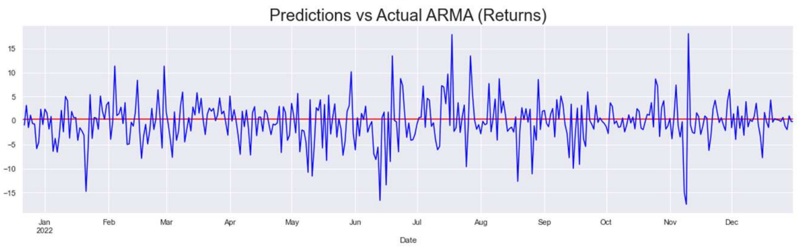
Torben, G, (2013). “Financial Risk Measurement for Financial Risk Management”. ScienceDirect
https://www.sciencedirect.com/science/article/abs/pii/B9780444594068000172

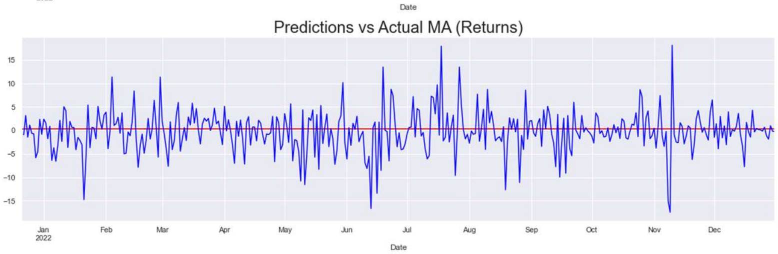
Evomics, (N.D), “Likelihood Ratio Test”. Evolution and Genomics
https://evomics.org/resources/likelihood-ratio-test/

Wayne, D, “The History of Bitcoin, First Cryptocurrency”. Usnews
https://money.usnews.com/investing/articles/the-history-of-bitcoin

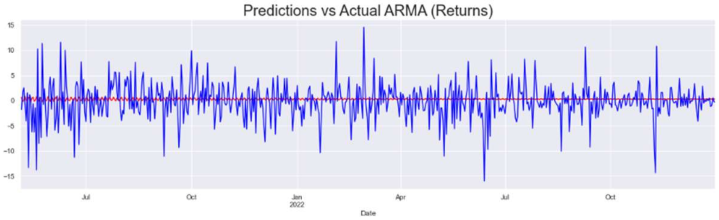
8 List of pictures, tables, graphs and abbreviations

8.1 List of Graphs

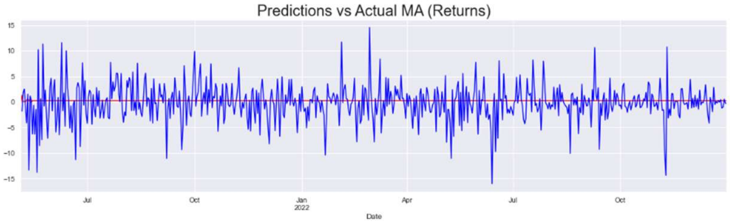




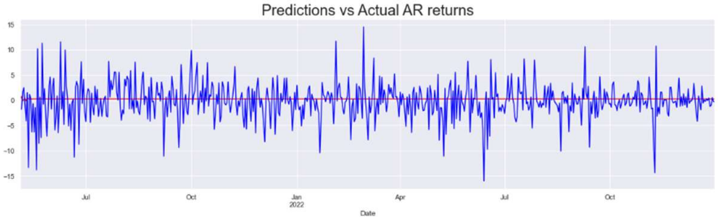
Predictions vs Actual ARMA (Returns)



Predictions vs Actual MA (Returns)

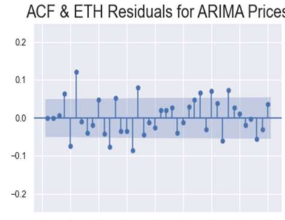


Predictions vs Actual AR returns

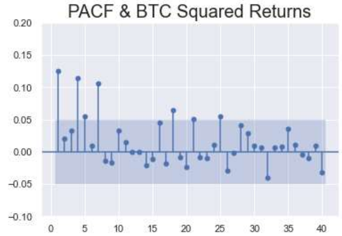


Predictions vs Actual AR prices

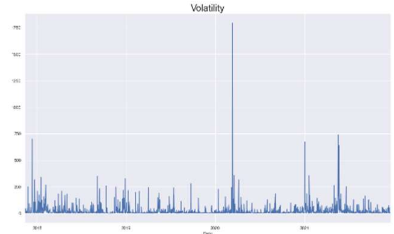
ACF & ETH Residuals for ARIMA Prices



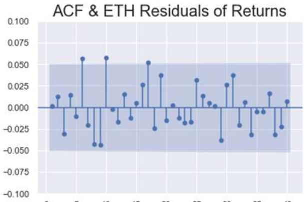
PACF & BTC Squared Returns



Volatility

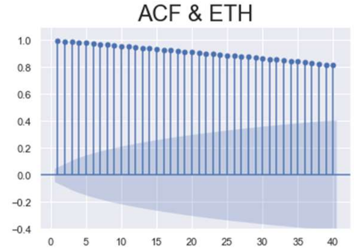


ACF & ETH Residuals of Returns



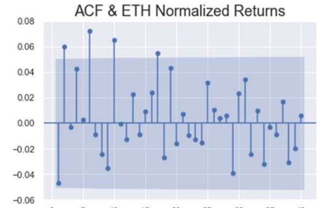
ACF & ETH Residuals for ARIMA Prices

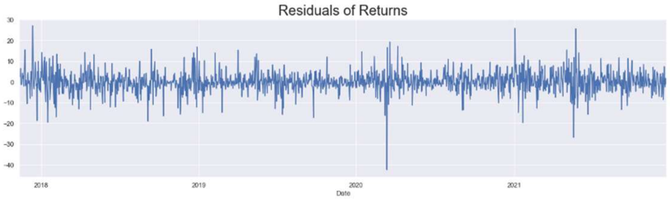
ACF & ETH

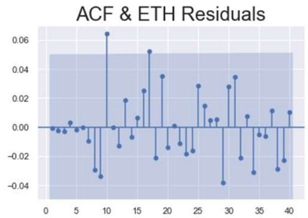


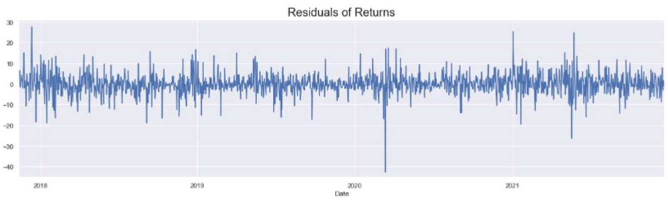
Residuals of Prices

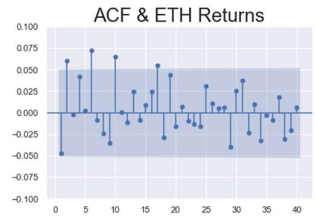
ACF & ETH Normalized Returns

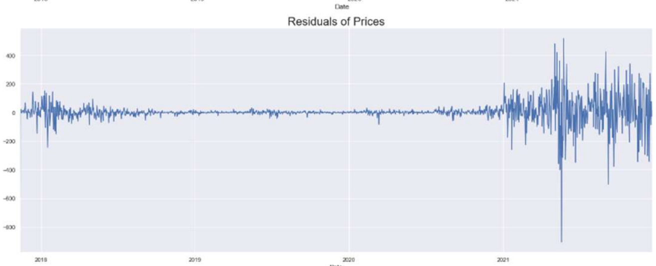


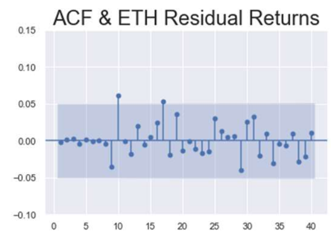

Residuals of Returns
Date

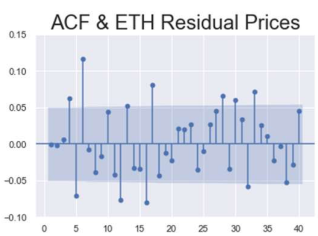

ACF & ETH Residuals

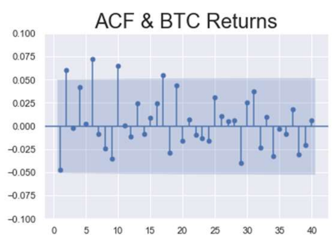

Residuals of Returns
Date

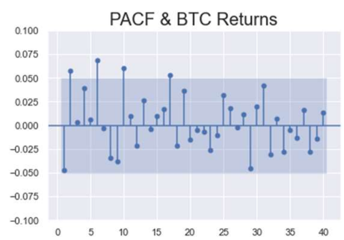

ACF & ETH Returns

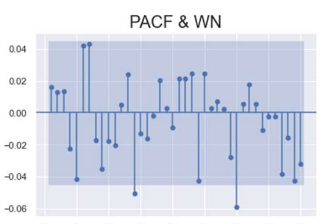

Residuals of Prices
Date

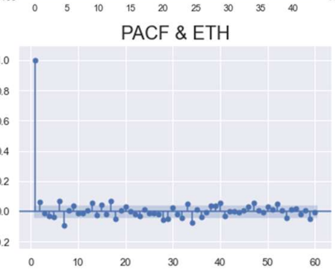

ACF & ETH Residual Returns

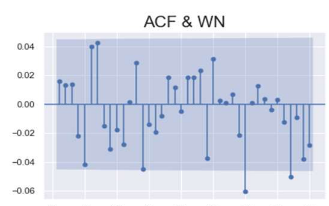

ACF & ETH Residual Prices

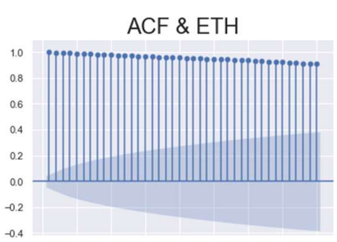

ACF & BTC Returns

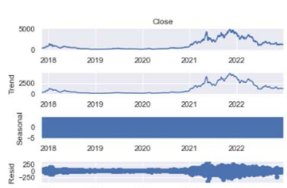

PACF & BTC Returns

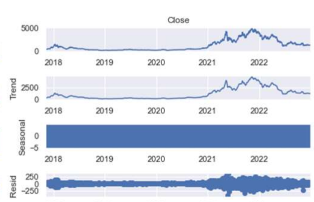

PACF & WN

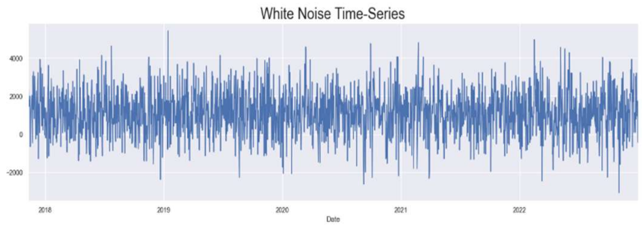

PACF & ETH

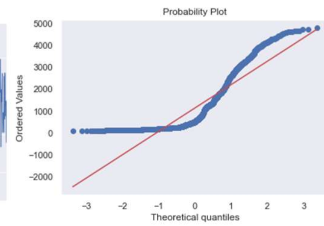

ACF & WN

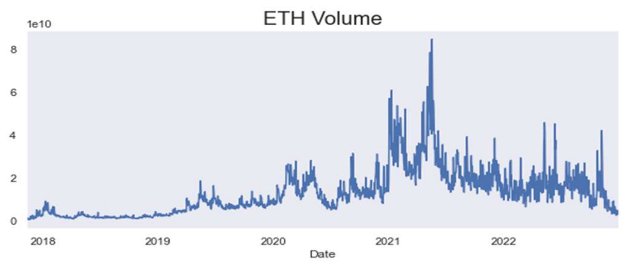

ACF & ETH

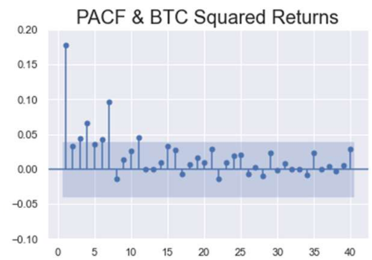

Close
Trend
Seasonal
Resid

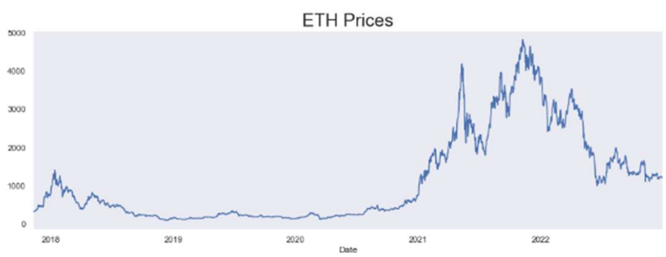

Close
Trend
Seasonal
Resid

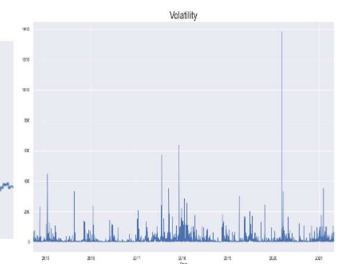

White Noise Time-Series
Date

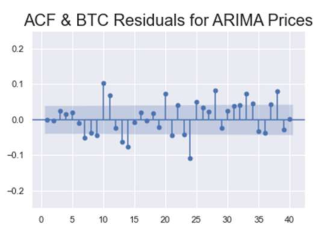

Probability Plot
Ordered Values
Theoretical quantiles

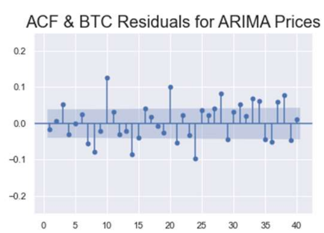

ETH Volume
1e10
Date


PACF & BTC Squared Returns


ETH Prices
Date

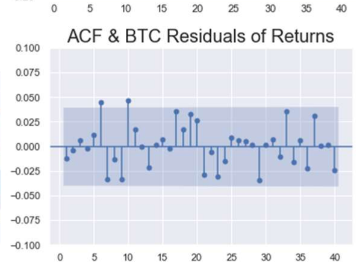

Volatility


ACF & BTC Residuals for ARIMA Prices

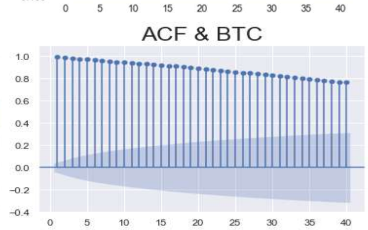

ACF & BTC Residuals for ARIMA Prices

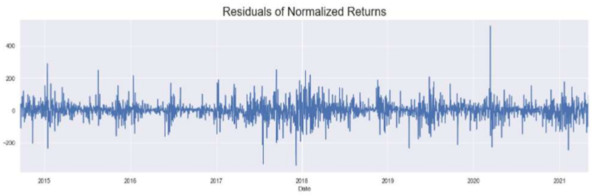

ACF of Residuals for Prices

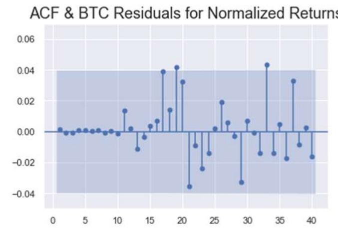

Residuals of Prices
Date

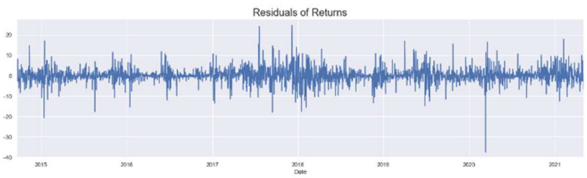

ACF & BTC Residuals of Returns

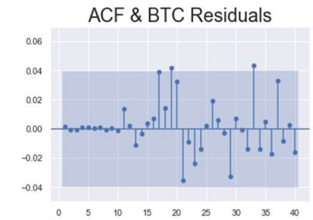

Residuals of Returns
Date

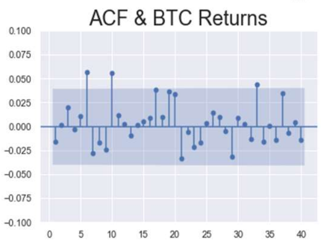

ACF & BTC

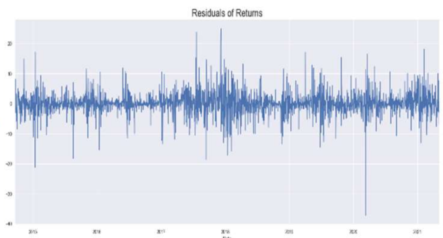

Residuals of Normalized Returns
Date

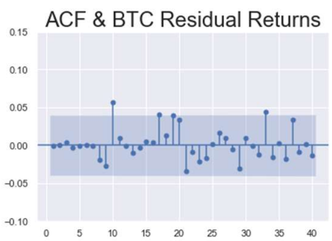

ACF & BTC Residuals for Normalized Returns

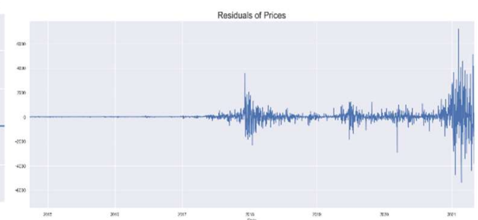

Residuals of Returns
Date

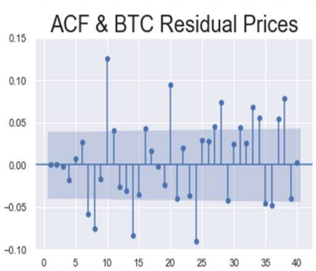

ACF & BTC Residuals

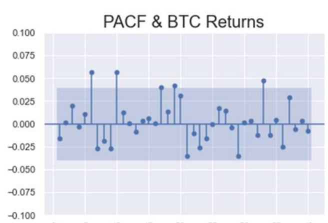

ACF & BTC Returns

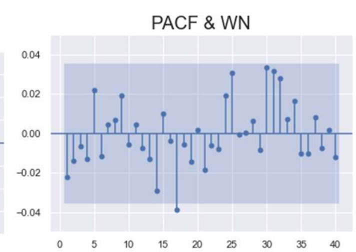

Residuals of Returns


ACF & BTC Residual Returns

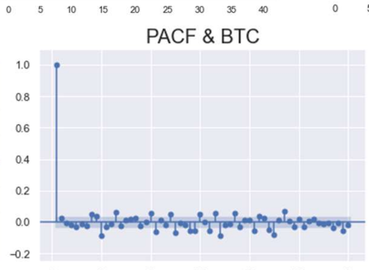

Residuals of Prices

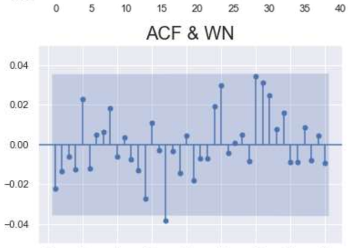

ACF & BTC Residual Prices

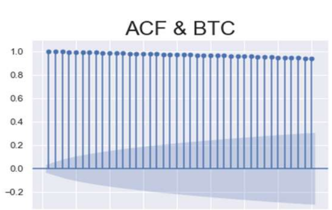

PACF & BTC Returns

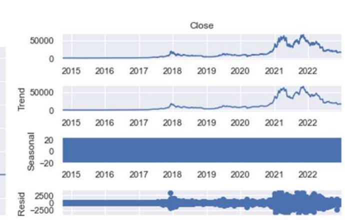

PACF & WN

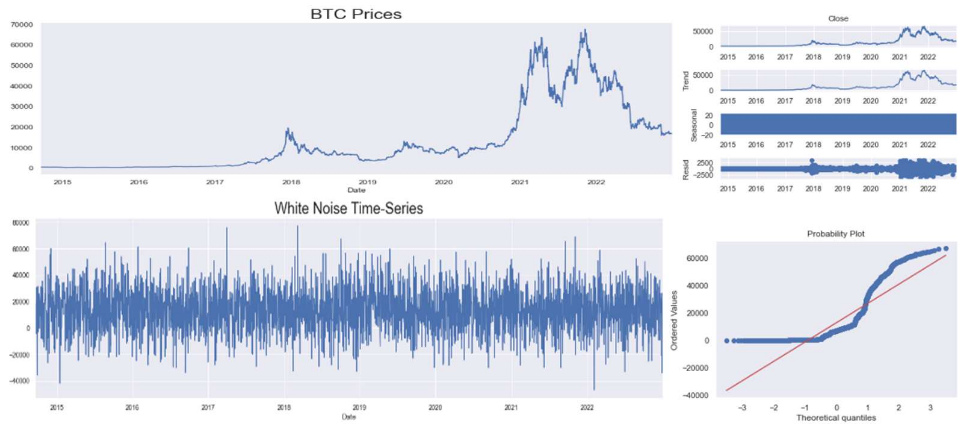

ACF & BTC Returns


PACF & BTC


ACF & WN


ACF & BTC


Close
Trend
Seasonal
Resid



8.2 List of Tables

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	GARCH	**Log-Likelihood:**	-4539.37
Distribution:	Normal	**AIC:**	9086.75
Method:	Maximum Likelihood	**BIC:**	9108.00
		No. Observations:	1500
Date:	Mon, Feb 06 2023	**Df Residuals:**	1499
Time:	18:27:21	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2676	0.119	2.245	2.478e-02	[3.397e-02, 0.501]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	1.5002	0.898	1.671	9.464e-02	[-0.259, 3.259]
alpha[1]	0.0727	3.256e-02	2.233	2.556e-02	[8.884e-03, 0.137]
beta[1]	0.8727	5.708e-02	15.291	8.800e-53	[0.761, 0.985]

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	GARCH	**Log-Likelihood:**	-4539.29
Distribution:	Normal	**AIC:**	9088.58
Method:	Maximum Likelihood	**BIC:**	9115.15
		No. Observations:	1500
Date:	Mon, Feb 06 2023	**Df Residuals:**	1499
Time:	18:28:00	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2692	0.119	2.263	2.367e-02	[3.600e-02, 0.502]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	1.6650	0.918	1.813	6.985e-02	[-0.135, 3.465]
alpha[1]	0.0825	3.462e-02	2.382	1.721e-02	[1.462e-02, 0.150]
beta[1]	0.6945	0.384	1.809	7.050e-02	[-5.809e-02, 1.447]
beta[2]	0.1625	0.362	0.449	0.654	[-0.547, 0.872]

Dep. Variable:	returns	R-squared:	0.000
Mean Model:	Constant Mean	Adj. R-squared:	0.000
Vol Model:	ARCH	Log-Likelihood:	-4581.12
Distribution:	Normal	AIC:	9170.24
Method:	Maximum Likelihood	BIC:	9191.49
		No. Observations:	1500
Date:	Mon, Feb 06 2023	Df Residuals:	1499
Time:	18:24:40	Df Model:	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.3124	0.126	2.477	1.325e-02	[6.522e-02, 0.560]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	22.5594	2.610	8.643	5.458e-18	[17.444, 27.675]
alpha[1]	0.1037	4.948e-02	2.097	3.600e-02	[6.777e-03, 0.201]
alpha[2]	0.0644	5.161e-02	1.247	0.212	[-3.678e-02, 0.166]

Dep. Variable:	returns	R-squared:	0.000
Mean Model:	Constant Mean	Adj. R-squared:	0.000
Vol Model:	ARCH	Log-Likelihood:	-4584.43
Distribution:	Normal	AIC:	9174.85
Method:	Maximum Likelihood	BIC:	9190.79
		No. Observations:	1500
Date:	Mon, Feb 06 2023	Df Residuals:	1499
Time:	18:18:59	Df Model:	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2893	0.128	2.267	2.339e-02	[3.919e-02, 0.539]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	24.0554	2.094	11.485	1.566e-30	[19.950, 28.160]
alpha[1]	0.1080	4.793e-02	2.253	2.429e-02	[1.403e-02, 0.202]

Dep. Variable:	Close	No. Observations:	1501
Model:	ARIMA(2, 1, 2)	Log Likelihood	-8069.353
Date:	Mon, 06 Feb 2023	AIC	16150.706
Time:	19:41:53	BIC	16182.586
Sample:	11-11-2017	HQIC	16162.583
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
Close	0.0541	0.001	87.804	0.000	0.053	0.055
ar.L1	0.0408	0.065	0.624	0.533	-0.087	0.169
ar.L2	0.7108	0.062	11.434	0.000	0.589	0.833
ma.L1	-0.0804	0.074	-1.089	0.276	-0.225	0.064
ma.L2	-0.6386	0.068	-9.448	0.000	-0.771	-0.506
sigma2	2759.2403	39.691	69.518	0.000	2681.447	2837.033

Ljung-Box (L1) (Q):	0.29	Jarque-Bera (JB):	23797.11
Prob(Q):	0.59	Prob(JB):	0.00
Heteroskedasticity (H):	8.54	Skew:	0.22
Prob(H) (two-sided):	0.00	Kurtosis:	22.51

Dep. Variable:	Close	No. Observations:	1501
Model:	ARIMA(4, 0, 3)	Log Likelihood	-8604.261
Date:	Sun, 05 Feb 2023	AIC	17226.521
Time:	19:50:11	BIC	17274.346
Sample:	11-11-2017	HQIC	17244.337
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	901.9887	7667.143	0.118	0.906	-1.41e+04	1.59e+04
ar.L1	-0.4027	0.039	-10.316	0.000	-0.479	-0.326
ar.L2	-0.0239	0.021	-1.150	0.250	-0.065	0.017
ar.L3	0.6351	0.021	30.485	0.000	0.594	0.676
ar.L4	0.7892	0.040	19.785	0.000	0.711	0.867
ma.L1	1.2980	0.046	28.035	0.000	1.207	1.389
ma.L2	1.3148	0.033	40.120	0.000	1.251	1.379
ma.L3	0.6887	0.047	14.568	0.000	0.596	0.781
sigma2	5628.1247	69.517	80.961	0.000	5491.874	5764.375

Ljung-Box (L1) (Q):	0.50	Jarque-Bera (JB):	29975.64
Prob(Q):	0.48	Prob(JB):	0.00
Heteroskedasticity (H):	13.15	Skew:	-0.84
Prob(H) (two-sided):	0.00	Kurtosis:	24.83

Dep. Variable:	Close	**No. Observations:**	1501
Model:	ARIMA(2, 0, 1)	**Log Likelihood**	-8621.952
Date:	Sun, 05 Feb 2023	**AIC**	17253.903
Time:	19:50:01	**BIC**	17280.473
Sample:	11-11-2017	**HQIC**	17263.801
	- 12-20-2021		
Covariance Type:	opg		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	901.9933	4246.570	0.212	0.832	-7421.130	9225.117
ar.L1	0.1776	0.038	4.723	0.000	0.104	0.251
ar.L2	0.8206	0.038	21.762	0.000	0.747	0.895
ma.L1	0.7546	0.045	16.701	0.000	0.666	0.843
sigma2	5689.9125	64.767	87.852	0.000	5562.972	5816.853

Ljung-Box (L1) (Q):	0.94	**Jarque-Bera (JB):**	32269.73
Prob(Q):	0.33	**Prob(JB):**	0.00
Heteroskedasticity (H):	13.78	**Skew:**	-0.87
Prob(H) (two-sided):	0.00	**Kurtosis:**	25.65

Dep. Variable:	returns	**No. Observations:**	1500
Model:	ARIMA(2, 0, 1)	**Log Likelihood**	-4594.296
Date:	Sun, 05 Feb 2023	**AIC**	9198.593
Time:	19:28:40	**BIC**	9225.159
Sample:	11-12-2017	**HQIC**	9208.489
	- 12-20-2021		
Covariance Type:	opg		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	0.3075	0.156	1.972	0.049	0.002	0.613
ar.L1	0.7088	0.146	4.868	0.000	0.423	0.994
ar.L2	0.0790	0.022	3.637	0.000	0.036	0.122
ma.L1	-0.7570	0.146	-5.182	0.000	-1.043	-0.471
sigma2	26.7859	0.508	52.753	0.000	25.791	27.781

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	1996.32
Prob(Q):	0.96	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.91	**Skew:**	-0.25
Prob(H) (two-sided):	0.28	**Kurtosis:**	8.63

Dep. Variable:	returns	**No. Observations:**	1500
Model:	ARIMA(4, 0, 3)	**Log Likelihood**	-4592.187
Date:	Sun, 05 Feb 2023	**AIC**	9202.374
Time:	19:30:16	**BIC**	9250.193
Sample:	11-12-2017	**HQIC**	9220.189
	- 12-20-2021		
Covariance Type:	opg		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	0.3060	0.136	2.244	0.025	0.039	0.573
ar.L1	0.4299	0.172	2.493	0.013	0.092	0.768
ar.L2	-0.0078	0.185	-0.042	0.966	-0.371	0.355
ar.L3	-0.6632	0.161	-4.125	0.000	-0.978	-0.348
ar.L4	0.0355	0.030	1.202	0.229	-0.022	0.093
ma.L1	-0.4733	0.173	-2.737	0.006	-0.812	-0.134
ma.L2	0.0662	0.200	0.331	0.741	-0.326	0.459
ma.L3	0.6228	0.176	3.531	0.000	0.277	0.968
sigma2	26.7031	0.540	49.487	0.000	25.646	27.761

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	1830.23
Prob(Q):	0.97	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.89	**Skew:**	-0.27
Prob(H) (two-sided):	0.20	**Kurtosis:**	8.39

Dep. Variable:	norm_ret	**No. Observations:**	1500
Model:	ARIMA(0, 0, 6)	**Log Likelihood**	-9037.983
Date:	Sun, 05 Feb 2023	**AIC**	18091.966
Time:	10:22:37	**BIC**	18134.472
Sample:	11-12-2017	**HQIC**	18107.801
	- 12-20-2021		
Covariance Type:	opg		

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	5.9335	2.994	1.982	0.047	0.065	11.802
ma.L1	-0.0443	0.020	-2.214	0.027	-0.084	-0.005
ma.L2	0.0606	0.025	2.392	0.017	0.011	0.110
ma.L3	0.0038	0.024	0.161	0.872	-0.043	0.050
ma.L4	0.0321	0.021	1.562	0.118	-0.008	0.072
ma.L5	0.0093	0.022	0.427	0.669	-0.033	0.052
ma.L6	0.0736	0.024	3.094	0.002	0.027	0.120
sigma2	1.004e+04	208.508	48.167	0.000	9634.619	1.05e+04

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2027.81
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.90	**Skew:**	-0.26
Prob(H) (two-sided):	0.26	**Kurtosis:**	8.67

Dep. Variable:	returns	**No. Observations:**	1500
Model:	ARIMA(0, 0, 2)	**Log Likelihood**	-4595.308
Date:	Sun, 05 Feb 2023	**AIC**	9198.617
Time:	09:58:50	**BIC**	9219.869
Sample:	11-12-2017	**HQIC**	9206.534
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3059	0.137	2.236	0.025	0.038	0.574
ma.L1	-0.0445	0.019	-2.283	0.022	-0.083	-0.006
ma.L2	0.0558	0.024	2.316	0.021	0.009	0.103
sigma2	26.8212	0.507	52.930	0.000	25.828	27.814

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2037.74
Prob(Q):	1.00	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.90	**Skew:**	-0.26
Prob(H) (two-sided):	0.25	**Kurtosis:**	8.69

Dep. Variable:	returns	**No. Observations:**	1500
Model:	ARIMA(0, 0, 6)	**Log Likelihood**	-4590.325
Date:	Sun, 05 Feb 2023	**AIC**	9196.649
Time:	10:02:30	**BIC**	9239.155
Sample:	11-12-2017	**HQIC**	9212.484
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3062	0.154	1.987	0.047	0.004	0.608
ma.L1	-0.0443	0.020	-2.219	0.027	-0.083	-0.005
ma.L2	0.0606	0.025	2.396	0.017	0.011	0.110
ma.L3	0.0038	0.024	0.161	0.872	-0.043	0.050
ma.L4	0.0321	0.020	1.565	0.118	-0.008	0.072
ma.L5	0.0093	0.022	0.428	0.669	-0.033	0.052
ma.L6	0.0737	0.024	3.100	0.002	0.027	0.120
sigma2	26.6425	0.552	48.261	0.000	25.560	27.724

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2027.82
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.90	**Skew:**	-0.26
Prob(H) (two-sided):	0.26	**Kurtosis:**	8.67

Dep. Variable:	norm_ret	**No. Observations:**	1501
Model:	ARIMA(2, 0, 0)	**Log Likelihood**	-9048.715
Date:	Sat, 04 Feb 2023	**AIC**	18105.430
Time:	20:21:27	**BIC**	18126.686
Sample:	11-11-2017	**HQIC**	18113.348
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	5.9962	2.660	2.254	0.024	0.782	11.211
ar.L1	-0.0445	0.019	-2.296	0.022	-0.082	-0.007
ar.L2	0.0579	0.024	2.404	0.016	0.011	0.105
sigma2	1.009e+04	190.612	52.954	0.000	9720.050	1.05e+04

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2025.50
Prob(Q):	0.99	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.90	**Skew:**	-0.26
Prob(H) (two-sided):	0.24	**Kurtosis:**	8.67

Dep. Variable:	norm_ret	**No. Observations:**	1501
Model:	ARIMA(10, 0, 0)	**Log Likelihood**	-9039.367
Date:	Sat, 04 Feb 2023	**AIC**	18102.734
Time:	20:21:15	**BIC**	18166.501
Sample:	11-11-2017	**HQIC**	18126.489
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	5.9962	2.993	2.004	0.045	0.131	11.862
ar.L1	-0.0443	0.020	-2.216	0.027	-0.084	-0.005
ar.L2	0.0575	0.025	2.274	0.023	0.008	0.107
ar.L3	0.0074	0.024	0.309	0.757	-0.040	0.054
ar.L4	0.0325	0.021	1.580	0.114	-0.008	0.073
ar.L5	0.0099	0.023	0.437	0.662	-0.034	0.054
ar.L6	0.0679	0.024	2.821	0.005	0.021	0.115
ar.L7	-0.0026	0.020	-0.132	0.895	-0.042	0.036
ar.L8	-0.0399	0.025	-1.613	0.107	-0.088	0.009
ar.L9	-0.0351	0.026	-1.367	0.172	-0.085	0.015
ar.L10	0.0601	0.024	2.545	0.011	0.014	0.106
sigma2	1e+04	208.680	47.926	0.000	9592.210	1.04e+04

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2088.42
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.88	**Skew:**	-0.28
Prob(H) (two-sided):	0.14	**Kurtosis:**	8.75

Dep. Variable:	returns	No. Observations:	1500
Model:	ARIMA(4, 0, 0)	Log Likelihood	-4593.996
Date:	Sat, 04 Feb 2023	AIC	9199.991
Time:	20:09:57	BIC	9231.870
Sample:	11-12-2017	HQIC	9211.867
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3060	0.143	2.133	0.033	0.025	0.587
ar.L1	-0.0444	0.020	-2.278	0.023	-0.083	-0.006
ar.L2	0.0555	0.024	2.295	0.022	0.008	0.103
ar.L3	0.0039	0.023	0.170	0.865	-0.041	0.049
ar.L4	0.0389	0.020	1.912	0.056	-0.001	0.079
sigma2	26.7740	0.552	48.481	0.000	25.692	27.856

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	1879.90
Prob(Q):	0.99	Prob(JB):	0.00
Heteroskedasticity (H):	0.90	Skew:	-0.25
Prob(H) (two-sided):	0.23	Kurtosis:	8.46

Dep. Variable:	returns	No. Observations:	1500
Model:	ARIMA(8, 0, 0)	Log Likelihood	-4589.558
Date:	Sat, 04 Feb 2023	AIC	9199.117
Time:	20:10:01	BIC	9252.249
Sample:	11-12-2017	HQIC	9218.911
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3061	0.149	2.051	0.040	0.014	0.599
ar.L1	-0.0450	0.020	-2.246	0.025	-0.084	-0.006
ar.L2	0.0552	0.025	2.191	0.028	0.006	0.105
ar.L3	0.0038	0.024	0.162	0.872	-0.042	0.050
ar.L4	0.0365	0.021	1.780	0.075	-0.004	0.077
ar.L5	0.0094	0.022	0.428	0.669	-0.034	0.052
ar.L6	0.0699	0.024	2.909	0.004	0.023	0.117
ar.L7	-0.0050	0.020	-0.250	0.803	-0.044	0.034
ar.L8	-0.0349	0.025	-1.420	0.156	-0.083	0.013
sigma2	26.6151	0.555	47.984	0.000	25.528	27.702

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	1975.82
Prob(Q):	0.96	Prob(JB):	0.00
Heteroskedasticity (H):	0.90	Skew:	-0.26
Prob(H) (two-sided):	0.23	Kurtosis:	8.60

Dep. Variable:	Close	No. Observations:	1502
Model:	ARIMA(2, 0, 0)	Log Likelihood	-8630.028
Date:	Sat, 04 Feb 2023	AIC	17268.055
Time:	19:34:08	BIC	17289.313
Sample:	11-10-2017	HQIC	17275.974
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	901.5974	4705.739	0.192	0.848	-8321.481	1.01e+04
ar.L1	0.8998	0.013	71.623	0.000	0.875	0.924
ar.L2	0.0993	0.013	7.898	0.000	0.075	0.124
sigma2	5706.7400	61.354	93.014	0.000	5586.489	5826.991

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	35366.99
Prob(Q):	0.99	Prob(JB):	0.00
Heteroskedasticity (H):	13.85	Skew:	-0.96
Prob(H) (two-sided):	0.00	Kurtosis:	26.69

Dep. Variable:	Close	No. Observations:	1502
Model:	ARIMA(3, 0, 0)	Log Likelihood	-8630.011
Date:	Sat, 04 Feb 2023	AIC	17270.021
Time:	19:36:07	BIC	17296.594
Sample:	11-10-2017	HQIC	17279.920
	- 12-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	901.5950	4672.272	0.193	0.847	-8255.890	1.01e+04
ar.L1	0.9003	0.013	70.713	0.000	0.875	0.925
ar.L2	0.1035	0.015	6.842	0.000	0.074	0.133
ar.L3	-0.0047	0.010	-0.453	0.651	-0.025	0.016
sigma2	5706.3579	63.493	89.874	0.000	5581.914	5830.802

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	35121.95
Prob(Q):	0.98	Prob(JB):	0.00
Heteroskedasticity (H):	13.85	Skew:	-0.95
Prob(H) (two-sided):	0.00	Kurtosis:	26.61

Constant Mean - GARCH Model Results

Dep. Variable:	returns	R-squared:	0.000
Mean Model:	Constant Mean	Adj. R-squared:	0.000
Vol Model:	GARCH	Log-Likelihood:	-6486.73
Distribution:	Normal	AIC:	12981.5
Method:	Maximum Likelihood	BIC:	13004.6
		No. Observations:	2420
Date:	Thu, Feb 02 2023	Df Residuals:	2419
Time:	16:11:03	Df Model:	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2392	6.322e-02	3.783	1.547e-04	[0.115, 0.363]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	0.6753	0.258	2.616	8.891e-03	[0.169, 1.181]
alpha[1]	0.1294	3.184e-02	4.065	4.794e-05	[6.703e-02, 0.192]
beta[1]	0.8373	2.924e-02	28.633	2.575e-180	[0.780, 0.895]

Dep. Variable:	returns	R-squared:	0.000
Mean Model:	Constant Mean	Adj. R-squared:	0.000
Vol Model:	GARCH	Log-Likelihood:	-6485.59
Distribution:	Normal	AIC:	12981.2
Method:	Maximum Likelihood	BIC:	13010.1
		No. Observations:	2420
Date:	Thu, Feb 02 2023	Df Residuals:	2419
Time:	16:11:08	Df Model:	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2425	6.245e-02	3.883	1.031e-04	[0.120, 0.365]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	0.8094	0.312	2.594	9.477e-03	[0.198, 1.421]
alpha[1]	0.1583	3.515e-02	4.503	6.709e-06	[8.938e-02, 0.227]
beta[1]	0.5403	0.218	2.477	1.323e-02	[0.113, 0.968]
beta[2]	0.2617	0.201	1.299	0.194	[-0.133, 0.656]

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	ARCH	**Log-Likelihood:**	-6596.32
Distribution:	Normal	**AIC:**	13202.6
Method:	Maximum Likelihood	**BIC:**	13231.6
		No. Observations:	2420
Date:	Thu, Feb 02 2023	**Df Residuals:**	2419
Time:	15:41:28	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2894	6.694e-02	4.324	1.534e-05	[0.158, 0.421]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	9.2991	1.314	7.077	1.469e-12	[6.724, 11.874]
alpha[1]	0.1623	4.209e-02	3.855	1.155e-04	[7.977e-02, 0.245]
alpha[2]	0.0830	4.529e-02	1.833	6.675e-02	[-5.735e-03, 0.172]
alpha[3]	0.1733	6.294e-02	2.753	5.899e-03	[4.994e-02, 0.297]

Constant Mean - ARCH Model Results

Dep. Variable:	returns	**R-squared:**	0.000
Mean Model:	Constant Mean	**Adj. R-squared:**	0.000
Vol Model:	ARCH	**Log-Likelihood:**	-6641.33
Distribution:	Normal	**AIC:**	13288.7
Method:	Maximum Likelihood	**BIC:**	13306.0
		No. Observations:	2418
Date:	Sun, Jan 29 2023	**Df Residuals:**	2417
Time:	13:32:49	**Df Model:**	1

Mean Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
mu	0.2983	7.493e-02	3.982	6.843e-05	[0.151, 0.445]

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	12.4061	1.125	11.025	2.887e-28	[10.201, 14.612]
alpha[1]	0.1696	4.138e-02	4.099	4.154e-05	[8.850e-02, 0.251]

```python
#will be done for other Cryptos

model_arimax = ARIMA(btc_train['2017-11-09':].Close, exog = ethdata[:"2021-05-04"].Close, order=(1,1,1)
results_arimax = model_arimax.fit()
results_arimax.summary()
```

SARIMAX Results

Dep. Variable:	Close	**No. Observations:**	1273
Model:	ARIMA(1, 1, 1)	**Log Likelihood**	-9925.590
Date:	Thu, 02 Feb 2023	**AIC**	19859.180
Time:	09:00:15	**BIC**	19879.774
Sample:	11-09-2017	**HQIC**	19866.915
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
Close	12.0532	0.116	104.125	0.000	11.826	12.280
ar.L1	0.2222	0.196	1.131	0.258	-0.163	0.607
ma.L1	-0.1597	0.198	-0.806	0.420	-0.548	0.228
sigma2	3.515e+05	4443.353	79.116	0.000	3.43e+05	3.6e+05

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	20743.18
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	4.02	**Skew:**	0.61
Prob(H) (two-sided):	0.00	**Kurtosis:**	22.75

Dep. Variable:	Close	**No. Observations:**	2421
Model:	ARIMA(3, 0, 6)	**Log Likelihood**	-18673.037
Date:	Wed, 01 Feb 2023	**AIC**	37368.073
Time:	16:23:28	**BIC**	37431.785
Sample:	09-18-2014	**HQIC**	37391.241
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	7170.5465	6.440	1113.375	0.000	7157.924	7183.169
ar.L1	0.5315	0.005	116.439	0.000	0.523	0.540
ar.L2	-0.4875	0.005	-95.766	0.000	-0.497	-0.477
ar.L3	0.9557	0.004	238.521	0.000	0.948	0.964
ma.L1	0.4855	0.010	48.887	0.000	0.466	0.505
ma.L2	1.0109	0.009	108.931	0.000	0.993	1.029
ma.L3	0.0918	0.011	7.988	0.000	0.069	0.114
ma.L4	0.0392	0.011	3.644	0.000	0.018	0.060
ma.L5	0.0471	0.009	5.431	0.000	0.030	0.064
ma.L6	0.0278	0.008	3.474	0.001	0.012	0.043
sigma2	2.967e+05	2183.294	135.912	0.000	2.92e+05	3.01e+05

Ljung-Box (L1) (Q):	0.01	**Jarque-Bera (JB):**	128907.71
Prob(Q):	0.94	**Prob(JB):**	0.00
Heteroskedasticity (H):	3812.82	**Skew:**	1.30
Prob(H) (two-sided):	0.00	**Kurtosis:**	38.65

Dep. Variable:	Close	**No. Observations:**	2421
Model:	ARIMA(3, 0, 3)	**Log Likelihood**	-18695.052
Date:	Wed, 01 Feb 2023	**AIC**	37406.104
Time:	16:23:31	**BIC**	37452.439
Sample:	09-18-2014	**HQIC**	37422.953
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	7170.5465	1.09e-10	6.6e+13	0.000	7170.547	7170.547
ar.L1	-0.5941	0.006	-102.277	0.000	-0.605	-0.583
ar.L2	0.6075	0.004	155.027	0.000	0.600	0.615
ar.L3	0.9865	0.006	164.015	0.000	0.975	0.998
ma.L1	1.6159	0.010	161.532	0.000	1.596	1.636
ma.L2	1.0141	0.016	63.318	0.000	0.983	1.046
ma.L3	0.0293	0.009	3.353	0.001	0.012	0.046
sigma2	3.008e+05	7.32e-09	4.11e+13	0.000	3.01e+05	3.01e+05

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	146715.78
Prob(Q):	0.99	**Prob(JB):**	0.00
Heteroskedasticity (H):	4776.46	**Skew:**	1.35
Prob(H) (two-sided):	0.00	**Kurtosis:**	41.04

Dep. Variable:	returns	**No. Observations:**	2420
Model:	ARIMA(3, 0, 3)	**Log Likelihood**	-6710.047
Date:	Wed, 01 Feb 2023	**AIC**	13436.095
Time:	15:39:32	**BIC**	13482.427
Sample:	09-19-2014	**HQIC**	13452.944
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2680	0.089	3.020	0.003	0.094	0.442
ar.L1	0.3481	0.124	2.798	0.005	0.104	0.592
ar.L2	-0.4338	0.074	-5.878	0.000	-0.578	-0.289
ar.L3	0.9335	0.123	7.562	0.000	0.692	1.175
ma.L1	-0.3507	0.129	-2.719	0.007	-0.604	-0.098
ma.L2	0.4414	0.076	5.840	0.000	0.293	0.590
ma.L3	-0.9265	0.128	-7.233	0.000	-1.178	-0.675
sigma2	14.9923	0.201	74.551	0.000	14.598	15.386

Ljung-Box (L1) (Q):	0.43	**Jarque-Bera (JB):**	6416.59
Prob(Q):	0.51	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	-0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.97

Dep. Variable:	returns	**No. Observations:**	2420
Model:	ARIMA(4, 0, 4)	**Log Likelihood**	-6710.810
Date:	Wed, 01 Feb 2023	**AIC**	13441.620
Time:	15:43:14	**BIC**	13499.535
Sample:	09-19-2014	**HQIC**	13462.681
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2791	0.079	3.555	0.000	0.125	0.433
ar.L1	-0.5407	1.559	-0.347	0.729	-3.596	2.514
ar.L2	-0.5609	1.381	-0.406	0.685	-3.267	2.145
ar.L3	-0.5386	1.432	-0.376	0.707	-3.345	2.267
ar.L4	0.3307	1.319	0.251	0.802	-2.255	2.916
ma.L1	0.5229	1.554	0.336	0.737	-2.523	3.569
ma.L2	0.5716	1.357	0.421	0.674	-2.088	3.232
ma.L3	0.5357	1.450	0.370	0.712	-2.305	3.377
ma.L4	-0.3441	1.327	-0.259	0.795	-2.945	2.257
sigma2	15.0146	0.199	75.288	0.000	14.624	15.405

Ljung-Box (L1) (Q):	0.01	**Jarque-Bera (JB):**	6719.47
Prob(Q):	0.92	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.50	**Skew:**	-0.15
Prob(H) (two-sided):	0.00	**Kurtosis:**	11.16

Dep. Variable:	norm_ret	**No. Observations:**	2420
Model:	ARIMA(0, 0, 10)	**Log Likelihood**	-13073.251
Date:	Wed, 01 Feb 2023	**AIC**	26170.503
Time:	08:03:41	**BIC**	26240.001
Sample:	09-19-2014	**HQIC**	26195.775
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.8366	1.181	-3.249	0.001	-6.151	-1.522
ma.L1	-0.0161	0.013	-1.230	0.219	-0.042	0.010
ma.L2	0.0035	0.017	0.213	0.831	-0.029	0.036
ma.L3	0.0236	0.017	1.399	0.162	-0.009	0.057
ma.L4	-0.0052	0.016	-0.328	0.743	-0.036	0.026
ma.L5	0.0089	0.017	0.534	0.593	-0.024	0.041
ma.L6	0.0563	0.016	3.458	0.001	0.024	0.088
ma.L7	-0.0332	0.015	-2.264	0.024	-0.062	-0.004
ma.L8	-0.0166	0.018	-0.907	0.364	-0.053	0.019
ma.L9	-0.0273	0.018	-1.525	0.127	-0.062	0.008
ma.L10	0.0583	0.018	3.308	0.001	0.024	0.093
sigma2	2890.5087	40.296	71.732	0.000	2811.530	2969.488

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	6322.17
Prob(Q):	0.97	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.91

Dep. Variable:	returns	**No. Observations:**	2420
Model:	ARIMA(0, 0, 10)	**Log Likelihood**	-6703.508
Date:	Wed, 01 Feb 2023	**AIC**	13431.016
Time:	06:21:35	**BIC**	13500.515
Sample:	09-19-2014	**HQIC**	13456.289
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2758	0.085	3.257	0.001	0.110	0.442
ma.L1	-0.0161	0.013	-1.233	0.218	-0.042	0.009
ma.L2	0.0035	0.016	0.214	0.830	-0.029	0.036
ma.L3	0.0236	0.017	1.402	0.161	-0.009	0.057
ma.L4	-0.0052	0.016	-0.328	0.743	-0.036	0.026
ma.L5	0.0089	0.017	0.535	0.593	-0.024	0.041
ma.L6	0.0563	0.016	3.468	0.001	0.024	0.088
ma.L7	-0.0332	0.015	-2.271	0.023	-0.062	-0.005
ma.L8	-0.0166	0.018	-0.910	0.363	-0.052	0.019
ma.L9	-0.0273	0.018	-1.529	0.126	-0.062	0.008
ma.L10	0.0583	0.018	3.317	0.001	0.024	0.093
sigma2	14.9123	0.207	71.930	0.000	14.506	15.319

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	6322.16
Prob(Q):	0.97	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	-0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.91

Dep. Variable:	returns	**No. Observations:**	2420
Model:	ARIMA(0, 0, 7)	**Log Likelihood**	-6708.664
Date:	Wed, 01 Feb 2023	**AIC**	13435.328
Time:	06:28:17	**BIC**	13487.452
Sample:	09-19-2014	**HQIC**	13454.283
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2759	0.083	3.329	0.001	0.113	0.438
ma.L1	-0.0161	0.013	-1.252	0.211	-0.041	0.009
ma.L2	0.0024	0.016	0.149	0.882	-0.030	0.034
ma.L3	0.0248	0.017	1.489	0.136	-0.008	0.058
ma.L4	-0.0089	0.016	-0.574	0.566	-0.039	0.022
ma.L5	0.0103	0.016	0.625	0.532	-0.022	0.042
ma.L6	0.0581	0.016	3.583	0.000	0.026	0.090
ma.L7	-0.0308	0.015	-2.118	0.034	-0.059	-0.002
sigma2	14.9762	0.207	72.339	0.000	14.570	15.382

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	6291.73
Prob(Q):	0.98	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	-0.17
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.89

Dep. Variable:	norm_ret	**No. Observations:**	2418
Model:	ARIMA(10, 0, 0)	**Log Likelihood**	-13132.818
Date:	Tue, 31 Jan 2023	**AIC**	26289.635
Time:	15:30:31	**BIC**	26359.124
Sample:	09-21-2014	**HQIC**	26314.906
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.9745	1.220	-3.257	0.001	-6.366	-1.583
ar.L1	-0.0138	0.013	-1.070	0.285	-0.039	0.011
ar.L2	0.0021	0.016	0.128	0.898	-0.030	0.034
ar.L3	0.0213	0.017	1.270	0.204	-0.012	0.054
ar.L4	-0.0035	0.016	-0.220	0.826	-0.034	0.027
ar.L5	0.0109	0.017	0.656	0.512	-0.022	0.044
ar.L6	0.0569	0.016	3.512	0.000	0.025	0.089
ar.L7	-0.0283	0.015	-1.926	0.054	-0.057	0.001
ar.L8	-0.0188	0.018	-1.031	0.303	-0.054	0.017
ar.L9	-0.0276	0.018	-1.540	0.124	-0.063	0.008
ar.L10	0.0561	0.017	3.240	0.001	0.022	0.090
sigma2	3055.6268	42.201	72.406	0.000	2972.914	3138.340

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	6365.20
Prob(Q):	0.97	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.51	**Skew:**	0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.94

Dep. Variable:	norm_ret	**No. Observations:**	2418
Model:	ARIMA(6, 0, 0)	**Log Likelihood**	-13138.849
Date:	Tue, 31 Jan 2023	**AIC**	26293.697
Time:	15:30:23	**BIC**	26340.023
Sample:	09-21-2014	**HQIC**	26310.544
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.9744	1.228	-3.238	0.001	-6.380	-1.568
ar.L1	-0.0159	0.013	-1.251	0.211	-0.041	0.009
ar.L2	0.0006	0.016	0.038	0.970	-0.031	0.032
ar.L3	0.0178	0.017	1.073	0.283	-0.015	0.050
ar.L4	-0.0011	0.016	-0.068	0.946	-0.032	0.030
ar.L5	0.0114	0.016	0.695	0.487	-0.021	0.043
ar.L6	0.0567	0.016	3.539	0.000	0.025	0.088
sigma2	3070.1127	41.484	74.006	0.000	2988.805	3151.421

Ljung-Box (L1) (Q):	0.01	**Jarque-Bera (JB):**	6448.55
Prob(Q):	0.94	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.52	**Skew:**	0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	10.99

Dep. Variable:	returns	**No. Observations:**	2417
Model:	ARIMA(1, 0, 0)	**Log Likelihood**	-6705.236
Date:	Tue, 31 Jan 2023	**AIC**	13416.471
Time:	14:32:42	**BIC**	13433.842
Sample:	09-22-2014	**HQIC**	13422.789
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2787	0.078	3.568	0.000	0.126	0.432
ar.L1	-0.0153	0.013	-1.227	0.220	-0.040	0.009
sigma2	15.0368	0.194	77.352	0.000	14.656	15.418

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	6613.33
Prob(Q):	1.00	**Prob(JB):**	0.00
Heteroskedasticity (H):	1.50	**Skew:**	-0.16
Prob(H) (two-sided):	0.00	**Kurtosis:**	11.10

Dep. Variable:	returns	No. Observations:	2417
Model:	ARIMA(7, 0, 0)	Log Likelihood	-6699.913
Date:	Tue, 31 Jan 2023	AIC	13417.826
Time:	14:37:51	BIC	13469.939
Sample:	09-22-2014	HQIC	13436.778
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2787	0.083	3.340	0.001	0.115	0.442
ar.L1	-0.0144	0.013	-1.117	0.264	-0.040	0.011
ar.L2	0.0015	0.016	0.095	0.924	-0.030	0.033
ar.L3	0.0176	0.017	1.058	0.290	-0.015	0.050
ar.L4	-0.0008	0.016	-0.051	0.959	-0.032	0.030
ar.L5	0.0112	0.016	0.684	0.494	-0.021	0.043
ar.L6	0.0561	0.016	3.465	0.001	0.024	0.088
ar.L7	-0.0275	0.015	-1.872	0.061	-0.056	0.001
sigma2	14.9702	0.207	72.344	0.000	14.565	15.376

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	6308.91
Prob(Q):	0.98	Prob(JB):	0.00
Heteroskedasticity (H):	1.51	Skew:	-0.16
Prob(H) (two-sided):	0.00	Kurtosis:	10.91

Dep. Variable:	Close	No. Observations:	2418
Model:	ARIMA(3, 0, 0)	Log Likelihood	-18688.728
Date:	Tue, 31 Jan 2023	AIC	37387.457
Time:	07:48:14	BIC	37416.410
Sample:	09-21-2014	HQIC	37397.986
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	7178.9363	5.791	1239.567	0.000	7167.585	7190.287
ar.L1	1.0128	0.008	125.997	0.000	0.997	1.029
ar.L2	0.0139	0.010	1.330	0.184	-0.007	0.034
ar.L3	-0.0270	0.007	-3.758	0.000	-0.041	-0.013
sigma2	3.02e+05	1981.120	152.450	0.000	2.98e+05	3.06e+05

Ljung-Box (L1) (Q):	0.02	Jarque-Bera (JB):	144742.42
Prob(Q):	0.89	Prob(JB):	0.00
Heteroskedasticity (H):	4293.11	Skew:	1.38
Prob(H) (two-sided):	0.00	Kurtosis:	40.80

Dep. Variable:	Close	No. Observations:	2418
Model:	ARIMA(4, 0, 0)	Log Likelihood	-18683.763
Date:	Tue, 31 Jan 2023	AIC	37379.527
Time:	07:48:13	BIC	37414.271
Sample:	09-21-2014	HQIC	37392.162
	- 05-04-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	7178.9367	6.904	1039.814	0.000	7165.405	7192.468
ar.L1	1.0116	0.008	124.820	0.000	0.996	1.027
ar.L2	0.0141	0.011	1.332	0.183	-0.007	0.035
ar.L3	0.0395	0.012	3.409	0.001	0.017	0.062
ar.L4	-0.0654	0.008	-8.575	0.000	-0.080	-0.050
sigma2	3.041e+05	2016.363	150.792	0.000	3e+05	3.08e+05

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	143806.34
Prob(Q):	0.98	Prob(JB):	0.00
Heteroskedasticity (H):	4083.98	Skew:	1.28
Prob(H) (two-sided):	0.00	Kurtosis:	40.69

8.3 List of Code Chunks

```
def LLR_test(mod_1, mod_2, DF = 1):
    L1 = mod_1.fit().llf
    L2 = mod_2.fit().llf
    LR = (2*(L2-L1))
    p = chi2.sf(LR, DF).round(3)
    return p
```

```
eth_train['delta_prices']=eth_train.Close.diff(1)
```

```
sts.adfuller(eth_train.delta_prices[1:])
```

```
(-10.144321399562491,
 8.262347504369866e-18,
 16,
 1483,
 {'1%': -3.4347671645756304,
  '5%': -2.86349089226533,
  '10%': -2.5678086339403325},
 16914.969030305292)
```

```
print("\nLLR test p-value = " + str(LLR_test(model_arima_1, model_arima_3, DF = 2)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_2, model_arima_3)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_4, model_arima_3)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_5, model_arima_3)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_6, model_arima_3)))
```

```
LLR test p-value = 0.004

LLR test p-value = 0.002

LLR test p-value = 0.048

LLR test p-value = 0.002

LLR test p-value = 0.048
```

```
print("ARIMA(1,1,1):  \t LL = ", results_model_arima_1.llf, "\t AIC = ", results_model_arima_1.aic)
print("ARIMA(1,1,2):  \t LL = ", results_model_arima_2.llf, "\t AIC = ", results_model_arima_2.aic)
print("ARIMA(1,1,3):  \t LL = ", results_model_arima_3.llf, "\t AIC = ", results_model_arima_3.aic)
print("ARIMA(2,1,1):  \t LL = ", results_model_arima_4.llf, "\t AIC = ", results_model_arima_4.aic)
print("ARIMA(3,1,1):  \t LL = ", results_model_arima_5.llf, "\t AIC = ", results_model_arima_5.aic)
print("ARIMA(3,1,2):  \t LL = ", results_model_arima_6.llf, "\t AIC = ", results_model_arima_6.aic)
```

```
ARIMA(1,1,1):    LL =  -8610.795256554851      AIC =  17227.590513109702
ARIMA(1,1,2):    LL =  -8610.12597071617       AIC =  17228.25194143234
ARIMA(1,1,3):    LL =  -8605.290078714881      AIC =  17220.580157429762
ARIMA(2,1,1):    LL =  -8607.247994566285      AIC =  17222.49598913257
ARIMA(3,1,1):    LL =  -8610.15517355717       AIC =  17230.31034711434
ARIMA(3,1,2):    LL =  -8607.247371915331      AIC =  17226.494743830663
```

```
# 1
model_arima_1 = ARIMA(eth_train.Close[1:], order = (1,1,1))
results_model_arima_1 = model_arima_1.fit()
results_model_arima1.summary()
# 2
model_arima_2 = ARIMA(eth_train.Close[1:], order = (1,1,2))
results_model_arima_2 = model_arima_2.fit()
results_model_arima_2.summary()
# 3
model_arima_3 = ARIMA(eth_train.Close[1:], order = (1,1,3))
results_model_arima_3 = model_arima_3.fit()
results_model_arima_3.summary()
# 4
model_arima_4 = ARIMA(eth_train.Close[1:], order = (2,1,1))
results_model_arima_4 = model_arima_4.fit()
# 5
model_arima_5 = ARIMA(eth_train.Close[1:], order = (3,1,1))
results_model_arima_5 = model_arima_5.fit()

# 6
model_arima_6 = ARIMA(eth_train.Close[1:], order = (3,1,2))
results_model_arima_6 = model_arima_6.fit()
```

```
sts.adfuller(eth_train.res_ret_ma_2[2:])
```

```
(-38.70434351882141,
 0.0,
 0,
 1498,
 {'1%': -3.4347228578139943,
  '5%': -2.863471337969528,
  '10%': -2.5677982210726897},
 9027.645156416089)
```

```
eth_train['res_ret_ma_2'] = results_ret_ma_2.resid[1:]
```

```
print("mean is " + str(round(eth_train.res_ret_ma_2.mean(),3)))
print("variance is " + str(round(eth_train.res_ret_ma_2.var(),3)))
print("Standard deviation is " + str(round(sqrt(eth_train.res_ret_ma_2.var()), 3)))
```

```
mean is 0.002
variance is 26.675
Standard deviation is 5.165
```

```
eth_train['res_price_ret'] = results_ar_model_ret_2.resid
eth_test['res_price_ret'] = results_ar_model_ret_2.resid
```

```
eth_train.res_price_ret.mean()
```

```
-1.857860593810645e-05
```

```
eth_train.res_price_ret.var()
```

```
26.63373497151356
```

```
sts.adfuller(eth_train.res_price_ret[1:])
```

```
(-38.746625407896836,
 0.0,
 0,
 1499,
 {'1%': -3.4347199356122493,
  '5%': -2.86347004827819,
  '10%': -2.567797534300163},
 9031.09628608121)
```

```
sts.adfuller(eth_train.res_price[1:])
```

```
(-9.90433891325914,
 3.293878938828608e-17,
 16,
 1483,
 {'1%': -3.4347671645756304,
  '5%': -2.86349089226533,
  '10%': -2.5678086339403325},
 16916.052118069485)
```

```
eth_train['res_price'] = results_ar_model_1.resid
eth_test['res_price'] = results_ar_model_1.resid
```

```
eth_train.res_price.mean()
```

```
2.245538426615259
```

```
eth_train.res_price.var()
```

```
5947.85755665728
```

```
benchmark_ret = eth_train.returns.iloc[0]
eth_train['norm_ret'] = eth_train.returns.div(benchmark_ret).mul(100)
eth_test['norm_ret'] = eth_test.returns.div(benchmark_ret).mul(100)
```

```
sts.adfuller(eth_train.norm_ret)
```

```
(-11.416317801866343,
 7.048091371372087e-21,
 9,
 1491,
 {'1%': -3.434743423170358,
  '5%': -2.8634804142964025,
  '10%': -2.567803054306163},
 17802.319193813113)
```

```
benchmark = eth_train.Close.iloc[0]
```

```
eth_train['norm'] = eth_train.Close.div(benchmark).mul(100)
eth_test['norm'] = eth_test.Close.div(benchmark).mul(100)
```

```
LLR_test(ar_model_ret_1, ar_model_ret_2)
```

```
0.003
```

```
LLR_test(ar_model_norm_ret_1, ar_model_norm_ret_2)
```

```
0.0
```

```
eth_train['returns'] = eth_train.Close.pct_change(1).mul(100)
eth_test['returns'] = eth_test.Close.pct_change(1).mul(100)
eth_train = eth_train.iloc[1:]
```

```
sts.adfuller(eth_train.returns)
```

```
(-11.412390521316967,
 7.197384243781305e-21,
 9,
 1492,
 {'1%': -3.434740473427213,
  '5%': -2.863479112458789,
  '10%': -2.5678023610641922},
 9054.48520801563)
```

```
sts.adfuller(ethdata.Close)
```

```
(-1.4089708365742828,
 0.5779679105330212,
 17,
 1861,
 {'1%': -3.4338687226315336,
  '5%': -2.863094318475046,
  '10%': -2.5675974634086765},
 21446.440104463112)
```

```
btc_train['delta_prices']=btc_train.Close.diff(1)
```

```
sts.adfuller(btc_train.delta_prices[1:])
```

```
(-8.177641199656662,
 8.293887243613951e-13,
 27,
 2392,
 {'1%': -3.4330867606360274,
  '5%': -2.862749062318083,
  '10%': -2.5674136347538057},
 36839.442886560086)
```

```
print(wn.mean())
```

```
1111.7392034831769
```

```
LLR_test(ar_model, ar_model_1)
```

```
0.851
```

```
ARIMA(1,1,1):    LL =  -18690.930198249596      AIC =  37387.86039649919
ARIMA(1,1,2):    LL =  -18689.748010232594      AIC =  37387.49602046519
ARIMA(1,1,3):    LL =  -18686.541816054236      AIC =  37383.08363210847
ARIMA(2,1,1):    LL =  -18689.894258525324      AIC =  37387.78851705065
ARIMA(3,1,1):    LL =  -18686.098516218015      AIC =  37382.19703243603
ARIMA(3,1,2):    LL =  -18655.94414688421       AIC =  37323.88829376842
```

```
print("\nLLR test p-value = " + str(LLR_test(model_arima_5, model_arima_6)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_4, model_arima_6, DF = 2)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_2, model_arima_6, DF = 2)))
print("\nLLR test p-value = " + str(LLR_test(model_arima_1, model_arima_6, DF = 3)))
```

```
LLR test p-value = 0.0

LLR test p-value = 0.0

LLR test p-value = 0.0

LLR test p-value = 0.0
```

```
# 1
model_arima_1 = ARIMA(btc_train.Close[1:], order = (1,1,1))
results_model_arima1 = model_arima1.fit()
results_model_arima1.summary()
# 2
model_arima_2 = ARIMA(btc_train.Close[1:], order = (1,1,2))
results_model_arima_2 = model_arima_2.fit()
results_model_arima_2.summary()
# 3
model_arima_3 = ARIMA(btc_train.Close[1:], order = (1,1,3))
results_model_arima_3 = model_arima_3.fit()
results_model_arima_3.summary()
# 4
model_arima_4 = ARIMA(btc_train.Close[1:], order = (2,1,1))
results_model_arima_4 = model_arima_4.fit()
# 5
model_arima_5 = ARIMA(btc_train.Close[1:], order = (3,1,1))
results_model_arima_5 = model_arima_5.fit()

# 6
model_arima_6 = ARIMA(btc_train.Close[1:], order = (3,1,2))
results_model_arima_6 = model_arima_6.fit()
```

```
LLR_test(model_arma_1, model_arma_2, DF = 3)
```

```
0.0
```

```
btc_train['res_ret_ma_2'] = results_ret_ma_2.resid[1:]
```

```
print("mean is " + str(round(btc_train.res_ret_ma_2.mean(),3)))
print("variance is " + str(round(btc_train.res_ret_ma_2.var(),3)))
print("Standard deviation is " + str(round(sqrt(btc_train.res_ret_ma_2.var()), 3)))
```

```
mean is 0.003
variance is 14.903
Standard deviation is 3.86
```

8.4 List of Equations

$$Yt = \beta_1 * y_{-1} + \beta_2 * y_{t-2} + \beta_3 * y_{t-3} + \ldots\ldots\ldots\ldots + \beta_k * y_{t-k}$$

$$Yt = \alpha_1 * \varepsilon_{t-1} + \alpha_2 * \varepsilon_{t-2} + \alpha_3 * \varepsilon_{t-3} + \ldots\ldots\ldots\ldots + \alpha_k * \varepsilon_{t-k}$$

$$Yt = \beta_1 * y_{t-1} + \alpha_1 * \varepsilon_{t-1} + \beta_2 * y_{t-2} + \alpha_2 * \varepsilon_{t-2} + \beta_3 * y_{t-3} + \alpha_3 * \varepsilon_{t-3} + \ldots\ldots\ldots\ldots + \beta_k * y_{t-k} + \alpha_k * \varepsilon_{t-k}$$

$$LR = 2*(\ln L1 - \ln L2)$$

8.5 List of abbreviations

BTC – Bitcoin

ETH – Ethereum

ACF – Auto Correlation Function

PACF – Partial Autocorrelation Function

AR – Auto Regression

MA – Moving Average

ARMA – Auto Regression Moving Average

ARIMA – Auto Regression Integrated Moving Average

ARIMAX – Auto Regression Integrated Moving Average Exogenous

WN – White noise

LLR– Log Likelihood Ratio

Appendix

BTC_ETH_TIMESERIES.pdf - PDF form of the Python notebook which contains all the code, different models and graphs