INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Forecasting Bitcoin's Volatility: Exploring the Potential of Deep-Learning

Tiago Emanuel Teixeira Pratas

Master in Economics

Supervisor:
PhD, Filipe Roberto de Jesus Ramos, Invited Assistant Professor,
ISCTE Business School

MSc, Didier Rodrigues Lopes, CEO
OpenBB

October, 2022

Department of Economics
Department of Political Economy

Forecasting Bitcoin's Volatility: Exploring the Potential of Deep-Learning

Tiago Emanuel Teixeira Pratas

Master in Economics

Supervisors:
PHD, Filipe Roberto de Jesus Ramos, Invited Assistant Professor,
ISCTE Business School

MSc, Didier Rodrigues Lopes, CEO
OpenBB

Outubro, 2022

Acknowledgements

What a journey!

First, I want to thank my mum and dad for all my academic journey and allowing me to develop who I want to be, with all the needed support, love and freedom.

Secondly, I would like to thank Professor Filipe for believing in me and giving me the chance of pursuing a dissertation of my true interest, Didier for all the help with the coding portion of the dissertations part and Carolina for the help formatting it.
This work couldn’t be done without all the help and support you gave me!

Thirdly, to my friends Andrew and Nick for all the support given while I was working in Dubai. Hope I could convert all the Lobster roles, Mr Beast Burgers and Cinnabon’s in good spaghetti code and perform top notch Sharpe Ratio calculations.

Love you all and hope I made you proud.

Resumo

A importância de usar as ferramentas estatísticas, matemáticas e computacionais certas pode certamente influenciar o processo de decisão. Com os recentes avanços computacionais, as metodologias *Deep-Learning*, baseadas em Inteligência Artificial apontam para uma ferramenta promissora para o estudo de séries temporais de dados financeiros, caracterizadas por padrões que são fora do normal. As criptomoedas são uma nova classe de ativos que são caracterizados por alta volatilidade, elevado número de quebras de estrutura e outras características que podem dificultar o estudo e previsão por parte de modelos clássicos.

O objetivo deste trabalho é analisar de forma crítica as capacidades de previsão das metodologias clássicas (ARCH e GARCH) comparativamente a metodologias de *Deep-Learning* (nomeadamente arquiteturas de redes neuronais: MLP, RNN e LSTM) para a previsão da volatilidade da bitcoin. O estudo empírico deste trabalho foca-se na previsão da volatilidade da *bitcoin* com os modelos supramencionados e comparar a sua qualidade preditiva usando as medidas de erro MAE e MAPE para horizontes de previsão de um, três e sete dias.

As metodologias de *Deep-Learning* apresentam algumas vantagens no que respeita à qualidade de previsão (pela análise da métrica de erro MAPE) mas apresentam um custo computacional superior. Também foram realizados Testes de *Diebold-Mariano* para comparar as previsões, concluindo-se a superioridade das metodologias de *Deep-Learning*.

Palavras-Chave: criptomoedas; Bitcoin; volatilidade; previsão; garch; arch; deep-learning

Sistema de Classificação JEL: C01, C02, C10, C22, C45, C53, C58, C60, G17

Abstract

The importance of using the right statistical, mathematical and computational tools can highly influence the decision-making process. With the recent computational progress, Deep Learning methodologies based on Artificial Intelligence seem to be pointed out as a promising tool to study financial time series, characterised by out-of-the-ordinary patterns. Cryptocurrencies are a new asset class with several specially interesting characteristics that still lack deep study and differ from the traditional time series. Bitcoin in particular is characterised by extraordinary high volatility, high number of structural breaks and other identified characteristics that might further difficult the study and forecasting of the time series using classical models.

The goal of this study is to critically compare the forecasting properties of classic methodologies (ARCH and GARCH) with Deep Learning Techniques (with MLP, RNN and LSTM architectures) when forecasting Bitcoin's Volatility. The empirical study focuses on the forecasting of Bitcoin's Volatility using such models and comparing its forecasting quality using MAE and MAPE for one, three- and seven-day's forecasting horizons.

The Deep learning methodologies show advantages in terms of forecasting quality (when we take in consideration the MAPE) but also require huge computational costs. Diebold-Mariano tests were also performed to compare the forecasts concluding the superiority of Deep Learning Methodologies.

Keywords: cryptocurrencies; Bitcoin; volatility; predicting; garch; arch; deep-learning

Branding JEL Classification System: C01, C02, C10, C22, C45, C53, C58, C60, G17

Index

Figure Index

Table Index

Introduction

Civilisation as we know it would not exist without money.

However, the concept of money has evolved over-time not only from a physical perspective, from seashells to paper-money, but also from how we transfer money, from wagons full of gold to online transfers.

As the concept of money evolves, it will continue to transcend and transform our societies. With the recent advancements on blockchain technology, for the first time in history it was possible to transfer value, from any point of world, without the use of an intermediary like a bank or an exchange. Slowly but surely, cryptocurrencies are becoming part of the global financial and economical ecosystem. Nonetheless, this new type of asset, it's also bringing new and interesting questions that currently present research opportunities by academia from the most different fields of knowledge.

The current macro-economic conditions, with parity the of Euro and U.S Dollar in hand with worldwide high inflation, result in the right time, more than ever, to question the concepts of money, the role of central banks and to better understand what opportunities these alternative systems can actually bring to the discussion and ultimately, if these new ideas can actually help improving our societies as whole.

These opportunities surge, from a technological perspective, with blockchain technologies at their core, where cryptocurrencies are revolutionising the way humans can exchange and store value; from a philosophical perspective, on what is value, what is money, how humans can create value; but mainly from an economical perspective and how can we make use of theoretical and econometric models and better understand this phenomenon and better prepare for future events, weather its price or volatility, with the most different range of forecasting methodologies.

This forecasting models are of extreme importance on our modern societies, as forecasting financial and economic data has become a crucial tool for the decision making of economic agents, investors and governments that can help create competitive advantages.

According to the literature, there are several econometric models, like ARCH and GARCH, that can help us to model and understand the volatility of financial assets. However, the high volatility and out-of-the-ordinary patterns and behaviours of the cryptocurrency markets present a difficulty for this type of model as stated by the literature.

Several authors suggest that a possible solution for this issue can be the use of more modern techniques such as machine learning in the search for models that can better help us understand and explain the nature of these new and exciting markets as means to help businesses understanding the risks surrounding these assets or a mean to help pricing derivatives.

With this said, the aim of this work will consist in three aspects: (i) aggregate the latest literature around the topic while trying to focus on aggregate information on how Bitcoin works and what are its main out-of-ordinary volatility drivers for future researchers; (ii) study and analysis of forecasting models for Bitcoin's Volatility on short-term time horizons and find out which one might be most suited for this time series; (iii) compare the forecasting quality of classical models to the results of Deep-Learning forecasting methodologies and analyse if Deep-Learning Methodologies are best suited than classical methodologies this time series.

With goals that define the scope of the research set, this are the research questions we are looking forward to answer:

(A) How Bitcoin functions and what are the out-of-ordinary volatility drivers?
(B) From the models in study, which one is the best forecasting Model for Bitcoin's Volatility?
(C) Are Deep-Learning Methodologies an improvement of classical methodologies?

To achieve this goal, this work was divided in five parts:

On the Chapter 1, it was performed a general literature review of the topic in study, with a first section where it explored briefly what Bitcoin is and how it functions. The second section is divided in two parts: (i) where we explain how some inner protocol mechanics might be the drivers of out-of-ordinary volatility; (ii) where we explore the importance of forecasting as a tool for decision making and a general review of previous research regarding forecasting price and volatility, using classical approaches and deep-learning models, for cryptocurrencies and traditional asset classes.

On Chapter 2, we focus on general concepts of Time Series analysis and the different methodologies and test to help our analysis. This Chapter is divided in three parts. First, we cover general econometric concepts such as unit root, stationarity, structural breaks and Auto-Correlation and Auto-Covariance. Secondly, we cover Univariate Linear Models such as Moving Average and Autoregressive Models. In also on this section, when we introduce Autoregressive Conditional Heteroskedasticity models like ARCH and GARCH. On the last section, we discuss three Deep Learning Models function and how their training and evaluation works.

The Chapter 3, where we discuss the variables that are going to be used on this study and a explanation of the computational implementations that allowed for the data treatment and for the implementation of our models.

The Chapter 4 is the practical Chapter where we present the results from the forecasting of our models in study. We started by performing a statistical analysis of Bitcoin's Price and Volatility using Python. The next step was modelling and forecasting using the five models in study and comparing the prediction errors among the models also using Python. Lastly, its performed Diebold-Mariano Test to compare the forecasting accuracy of some models.

On the Conclusion, we present a critical summary of the main discoveries from our study while briefly describing our methodology. It is also proposed a few suggestions and possible improvements of the current work, that open the path for future research.

Last but not least, it's also important to mention that several important outputs of this work are on the final part of this dissertation, in the Appendix, that are crucial for a full understanding of this study as a whole, but particularly, for Chapter 4.

1 – Literature Review

The goal of this chapter is to present a review of the existing literature regarding three different areas: (1) reviewing the main concepts regarding Bitcoin[1] by briefly explaining how it works and what makes this asset unique but with the focus on what might explain its volatility and price[2]; (2) a review on what is volatility and its importance on the decision-making process for companies and policymakers; (3) and a review on prior research work regarding the forecasting of volatility and prices of traditional financial assets and cryptocurrencies, using econometric and deep-learning models with the focus of comparing the quality and accuracy of such models as this is metric might be the more interesting for investors and policy makers.

1.1 – Bitcoin: General Concepts

According to Bitcoin's creator Nakamoto (2008), Bitcoin is a peer-to-peer version of electronic cash that allows for online payments to be sent directly from one party to another without going through a financial institution, meaning you can transfer the ownership of private property without any third-party validating that transfer.

This technology was able to solve the double-spending and byzantine generals' problem for the first time in a purely decentralised way with the use of a decentralised public ledger known as the Blockchain.

Several other projects with the goal of implementing a digital currency like Bitcoin have been tried over the last decades. Examples of these can be the ''B-money'', described by Dai (1998) as cryptography-based currency, but still centralised as it would be necessary for a centralised third-party to validate transactions in order to avoid double-spending. However, Bitcoin was the only one that has done it in a purely decentralised way, preventing what is known as a single-point-of-failure, common on prior projects.

Is also important to take notice that blockchain technology is a merge of several existing technologies such as Pretty Good Privacy (PGP) with Zimmermann (1991); Merkle trees with Merkle (1987); Sha-256 encryption[3]; or Hashcash, a proof-of-work system used to limit email spam and denial-of-service attacks, developed by Back (1997), that is used in the Bitcoin mining process. The way Bitcoin can ensure the validity of the transactions in a purely decentralised way, is through a proof-of-work mechanism, generally called ''Mining'', that is a decentralised security mechanism based on computational power and energy spending that

[1] Bitcoin with capital ''*B*'' stands for the Bitcoin project as a whole, while bitcoin, without the capital letter, stands for the currency as a unit of account (like euro, dollar, etc).

[2] For a broader explanation of how Bitcoin works, I recommend reading two books: ''Mastering Bitcoin'' by (Antonopoulos, 2015) and ''Bitcoin - A moeda na era digital'' by Fernando Ulrich.

[3] The Sha-2 Hash Algorithm family was developed by a group of researchers of NSA.

allows not only for the maintenance and security of the validity of the transactions stored on the blockchain as it serves the purpose of creating monetary incentives to alight the good behaviour of its participants, the miners, while allowing for the minting of new bitcoins and therefore programmatically increasing bitcoin's monetary supply every time a block, a set of blockchain transactions, is processed (or mined).

Antonopoulos (2015), explains that bitcoin is a system of trust is based on computation as transactions are bundled into blocks that require a large amount of computational power, and therefore energy, to validate the transactions. According to the author, this serves two main goals: (1) creating new bitcoins in each block, like a central bank issuing new money and (2) creating trust on the system, by ensuring that the transactions are only confirmed if enough computational power is devoted to the particular block that contains them.

Ferreira (2018), based on the ideas of Antonopoulos (2015) and Nakamoto (2008), adds that due to this procedure, we create and alight monetary incentives for miners to participate in the validating process of new transactions of the network while assuring that the blockchain security is increased and becomes more resilient regarding attacks.

Another important characteristic that is important to mention, is the fact that there is a long-term fixed supply of BTC. The total number of BTC that is ever going to exist is fixed at 21 million units and currently, from each block that is mined, there is the creation of a new 6.25 BTC. However, for every 210,000 blocks that are mined the bitcoin reward that is given to miners is slashed in half and therefore reducing the amount of bitcoin that is produced. This happens every four years if we considered that each block is mined every 10 minutes on average. This is a very important concept for understanding the ideas discussed in Chapter 1.2. The aim is to maintain each block being processed at around 10 minutes and ensure network security, Nakamoto (2008) programmed watch is called the ''mining difficulty''.

As the proof-of-work mechanisms are simply a brute-force mechanism to find the nonce of a given block, it's easy to understand how the increasing of new miners will reduce the amount of time per block and therefore put at risk the network security as it was easier to perform a 51% attack. Imagine that we were trying to find the number 5 from a machine by continually asking for the numbers 1 to 5 until we get a valid answer. This process is called brute-force. It would take us 5 tries to find that number 5, if we start from 1. However, if several other people were asking for numbers of random numbers from 1 to 5 at the same time, the amount of time we take to find that number would reduce substantially.

To tackle this issue, Bitcoin algorithm increases the amount prime numbers the miners need to brute-force in order to increase the time needed to find the nonce that makes that transaction valid. This is known as difficulty adjustment that can be increased or decreased in order to maintain the 10-minute block time.

This adjustment happens automatically every 2,016 blocks, or around every two weeks. These adjustments have several implications since the rise in the difficulty of mining will impact the profitability of the mining operations and therefore the total amount of miners and therefore the total amount of computational power that the network has. This is usually known as hash rate. Again, this concepts are important to take in mind when we discuss the ideas of ‘’*Price-Hashrate Spirals*‘’, further discussed on Chapter 1.2.

1.2 – Volatility

1.2.1 – Price and Volatility Drivers

Since Bitcoin is not controlled by a Central Governmental Authority, many may question the economic value and price of this intangible asset. However, this discussion is not new for the literature. From what may be the fundamental value or the inerrant internal protocol mechanics that may be the cause of volatility, this chapter will focus on what might be the price and volatility determinants of bitcoin.

Hayes (2017) argues that one of the main price determinants of bitcoin is the relative cost of production and therefore the price can be described by its cost of production (electricity costs). This has interesting implications as everything that serves to reduce the cost of producing new bitcoins, like hardware energy efficiency, low electricity prices or lower mining difficulty will result in negative price pressure for bitcoin.

Garcia et al. (2014), shares similar views, introducing a lower bound model to estimate fundamental value based on the cost of electricity of mining. They also state that bitcoin dynamics are originated by two types of users: those who mine and those who influence the exchange rate. They also discussed the existence of two positive feedback loops: ‘’a reinforcement cycle between search volume, word of mouth and price (social cycle), and a second cycle between search volume, number of new users and price (user adoption cycle).’’

Hanley (2013) disagrees with such ideas, stating that the value of bitcoin doesn’t have any fundamental value and that is based on set false claims. He argues that ‘’the valuation of bitcoin will always be determined by speculation’’ and therefore, people buying and selling against other currencies. In his words, ‘’Bitcoin’s developers combine technical implementation proficiency with ignorance of currency and banking fundamentals.’’

Regarding the *Halving* events and the minting process described on the previous chapter and in *Table 1.1*, Yermack (2013) highlights their importance as ‘’all of the quantities and growth rates of Bitcoins are known with certainty by the public’’, since the rules for the minting of new bitcoins are already embedded in the computer code that makes this protocol operate and therefore will stay unchanged forever. This gives a clear and transparent understanding of how many bitcoins are going to be minted over-time and let all the participants anticipate minting changes.

Gronwald (2019), also explains that the long-term fixed supply of Bitcoin is just like other ‘’exhaustible resource commodities such as crude oil and gold’’. Just like gold or crude oil, the more you extract from nature, the less there is available to extract making it more scare and harder to extract. A similar process occurs with Bitcoin. The author mentions that the production and available quantity of bitcoin is known with certainty where, for example, the future oil production rates are controlled by OPEC and can be adjusted based on several factors. In addition, technological developments, weather conditions or strikes on oil platforms can change the amount of oil that is being produced or total quantity that is estimated to exist.

Table 1.1 - Bitcoin’s minting quantities over time

Year	Bitcoin blocks by cumulative totals	Block remuneration, bitcoin	The amount of mined bitcoins	The amount of mined bitcoins of the maximum emission, %	The amount of mined bitcoins by cumulative totals
2009	210,000	50	10,500,000.00	50%	10,500,000.00
2012	420,000	25	5,250,000.00	25%	15,750,000.00
2016	630,000	12.5	2,625,000.00	12.5%	18,375,000.00
2020	840,000	6.25	1,312,500.00	6.25%	19,687,500.00
2024	1,050,000	3.125	656,250.00	3.125%	20,343,750.00
2028	1,260,000	1.5625	328,125.00	1.5625%	20,671,875.00
2032	1,470,000	0.78125	164,062.50	0.78125%	20,835,937.50
2036	1,680,000	0.390625	82,031.25	0.390625%	20,917,968.75
2040	1,890,000	0.1953125	41,015.63	0.1953125%	20,958,984.38
2044	2,100,000	0.09765625	20,507.81	0.09765624%	20,979,492.19
2048	2,310,000	0.048828125	10,253.91	0.048828143%	20,989,746.09
2052	2,520,000	0.0244140625	5,126.95	0.0244140476%	20,994,873.05
2056	2,730,000	0.01220703125	2,563.48	0.01220704762%	20,997,436.52
2060	2,940,000	0.006103515625	1,281.74	0.00610352381%	20,998.718.26
2064	3,150,000	0.0030517578125	640.87	0.00305176191%	20,999,359.13
2068	3,360,000	0.0015258789063	320.43	0.00152585714%	20,999,679.57
2072	3,570,000	0.0007629394531	160.22	0.00076295238%	20,999,839.78
2076	3,780,000	0.0003814697266	80.11	0.00038147619%	20,999,919.89
2080	3,990,000	0.0001907348633	40.06	0.00019076191%	20,999,959.95
…	…	…	…	…	…
2140	6,930,000	0.00	≈ 0.001222534	100%	21,000,000.00

Source: https://bit2me.com/

Is also important to mention that halving’s are nothing less that programmed supply shocks of the production of bitcoin and therefore, this is expected to result on price volatility as buyers and sellers adjust for an equilibrium price. However, as shown on *Table 1.1*, these supply shocks became less relevant over time, as the supply-shock becomes smaller. On the first halving, there was a decrease of the production of bitcoins by 25 BTC per block, on the second 12.5 BTC, on the third 6.25 BTC. Therefore, is expected that the possible volatility caused by such events, should be reduced over time.

This idea has been discussed by authors, such as Chaim and Laurini (2018), that state that ''Volatility was highest in late 2013 and during 2014.'' when comparing data from May 2013 to April 2018, suggesting the validation of the previous thoughts.

Gronwald (2019), also discusses a new term entitled ''demand shocks'', as he argues that since the monetary supply expansion of bitcoin is programmatically predictable and the short-run supply of gold and oil is uncertain, he argues that the movements we can observe in bitcoin prices surges can be interpreted as demand shocks. He also argues that bitcoin has a lot of similarities to other commodities and therefore we should analyse bitcoin with similar frameworks.

Another event that might help understanding the volatility of bitcoin is an event called the *Price-Hashrate Spirals,* as shown on *Figure 1.1,* Pagnotta and Buraschi (2018) explain this process in detail, when modelling an equilibrium valuation for Bitcoin as a decentralised financial network.

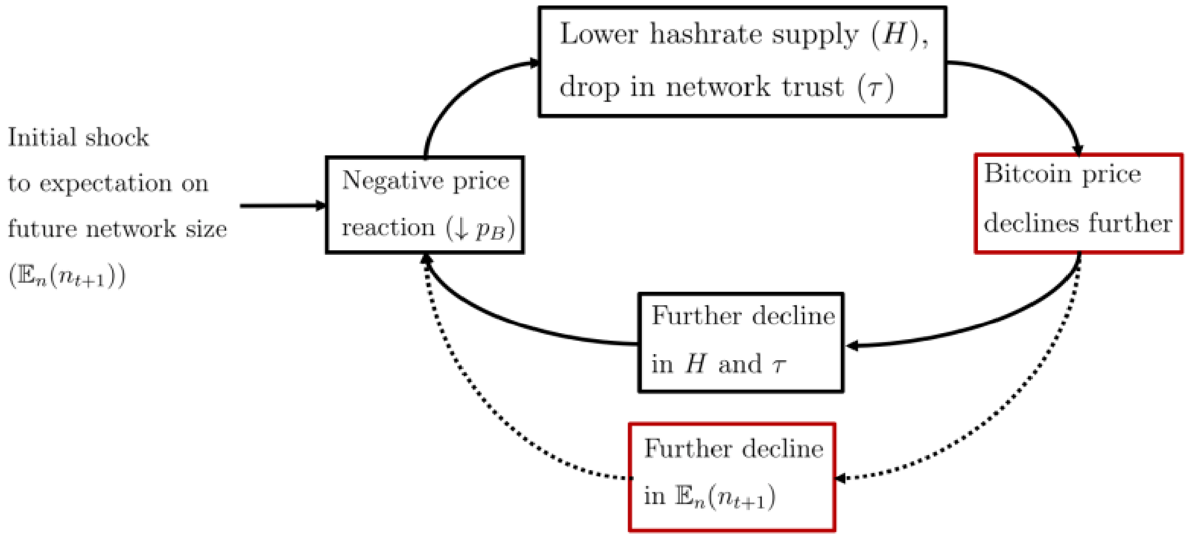


Figure 1.1 - A Price-Hashrate Spiral diagram

Source: Pagnotta and Buraschi (2018)

To sum up, if Bitcoin is a network and its price is a function of its total size and trust, a decreasing in the hashrate will translate in negative price action. This negative price action will result in further mining operations being shut down due to unprofitability, resulting in further decreasing in decreasing of the hashrate.

Mueller (2020) argues that we need to make a distinction between miners using ASIC and GPU equipment since the ASIC miners, usually bitcoin miners, ''have asymmetric reactions to price shocks'' and ''respond only to negative disequilibria (when hashrate is relatively low compared to price)''.

Another event that is important to understand, are the liquidation cascades. They occur when a huge price movement leads to the liquidation of the trader's position, which makes the trader forced to exit their position in the direction of that price movement, which contributes for the continuation of the price movement, leading to more liquidations and leveraging the overall effect of that price movement.

This is particularly important since the unregulated nature of the majority of crypto markets allows for the usage of high leverage and market manipulation, which may contribute to this issue and increase volatility.

Yu (2019), studied the effects and links of leverage when robustly forecasting bitcoin's volatility on 5-min high-frequency data using a Leverage Heterogeneous Auto-Regressive with Continuous, Volatility and Jumps (LHAR-CJ) model. It was concluded that the leverage effect significantly impacts future volatility and that the leverage effect has more power than jump components when forecasting bitcoin volatility and therefore contains predictive information.

However, Senarathne (2019), states that the Bitcoin return variance cannot be effectively explained by GARCH (1,1), Glosten, Jagannathan, and Runkle GARCH (GJR-GARCH) or Exponential GARCH models (EGARCH) given the stationarity of variance of return and that the leverage effect is not even observed on the EGARCH model.

Other final considerations about this asset we should take in consideration, were discussed by Liu and Tsyvinski (2021), where it was concluded that cryptocurrency returns have low exposures to traditional asset classes such as currencies, commodities, and stocks, and to macroeconomic factors, making it an uncorrelated and unique asset class which increases the interest for research studying.

De Nicola (2021) have done extensive research on traditional assets stylised facts and to compare the set common empirical properties of bitcoin's returns from March of 2015 and June of 2018. He highlights Bitcoin's high volatility, by comparing it to the EUR/USD pair, stating that bitcoin is 10 times more volatile than the pair. He states that such extreme volatility conditions exist since there is no "book" to base valuations on and therefore the price discovery is much more subject to news, events and speculation''. It is also stated that Bitcoin is the least volatile asset in the cryptocurrency class. It is also mentioned that there is less volume on the weekends and the distribution of returns follows the non-gaussian character distribution, and price jumps are present on the time series, just like on the traditional markets.

However, regarding Gain/Loss Asymmetry, the authors conclude that on shorter time frames such as the 5, 15 and 30 minutes, the negative returns are significantly larger in absolute values but that this asymmetry becomes less significant at larger timeframes. The authors explains that this might be caused by investor overreaction, loss aversion effects, and high levels of volatility. In terms on autocorrelation of the returns, traditional liquid assets do not exhibit any significant autocorrelation behaviour. However, his research shows the existence of shows that autocorrelation not only of lower time intervals but also mediumfrequency returns meaning that this partial predictability of Bitcoin’s returns can suggest the market inefficiencies.

He also mentions there is a positive correlation between jump size and absolute correlation and that despite the volatility conditions, the tendency for price after a movement is a given direction, is to revert and move to the opposite direction. This can be attributed to market microstructure effects, stating that the bigger the price movements are, the bigger the overreaction and cascading liquidations will be.

Taleb (2021) disagrees with most of the previously mentioned characteristics. The author doesn’t agree with the previously discussed cost models and that states that any non-dividend yield asset must have a present value of 0, if it has the tiniest constant probability of hitting an absorbing barrier, resulting in its value becoming 0. He also states that if from any other reason bitcoin can drop do zero, due to technological obsolescence for example, then its present value must also be zero now. He also argues that bitcoin do not have inflation hedging proprieties and that is has failed as a payment network due to high transaction costs and volatility in value, which makes it now a bad medium-of-exchange or unit-of-account that results in arbitrage opportunities.

1.2.2 - Forecasting Volatility

Whether we are exploring the field of economics or finance, volatility plays an important role in measuring and accessing potential risks resulting in policymakers making more informed decisions on the future of our economies and financial markets by giving them a better understanding of what can be expected from the future. Some examples of these are due to the fact that, if we can accurately forecast volatility, it can help us not only predict the price of financial derivatives, as the risk of this particular asset.

With this, every researcher, business owner or police maker can be aware of potential future risks and implement precautionary measures and directly help their decision-making processes.

This importance and interest are clearly observed by the vast available literature and research regarding this topic but why understanding volatility is important? How predicting volatility can help our societies?

According to Kim and Won (2018) in their works regarding forecasting the volatility of stock price indexes, volatility is described as ‘’the degree to which asset prices fluctuate’’. These authors also state that volatility ‘’plays crucial roles in financial markets, such as in derivative pricing, portfolio risk management, and hedging strategies’’.

Black and Scholes (1973) would corroborate this importance due to their work and research regarding option pricing models such as Black-Sholes.

These authors developed one of the most commonly used methodologies for option pricing and where volatility is one of the variables that is utilised in their calculations, represent by sigma, on the following formula:

$$p_t = Ke^{-rT}N(-d_2) - S_0e^{-qT}N(-d_1) \tag{1.1}$$

and

$$c_t = S_0e^{-qT}N(d_1) - Ke^{-rT}N(d_2) \tag{1.2}$$

where

$$d_1 = \frac{ln(\frac{S_0}{K}) + (r - q + \frac{\sigma^2}{2})T}{\sigma\sqrt{T}} \tag{1.3}$$

and

$$d_2 = \frac{ln(\frac{S_0}{K}) + (r - q - \frac{\sigma^2}{2})T}{\sigma\sqrt{T}} \tag{1.4}$$

Markowitz (1952) on his work on Portfolio Theory also considers that volatility is one of the key indicators to measure risk and uncertainty implying that the higher the volatility, the higher is the risk of the asset or portfolio of assets. In his works, he theorises important concepts such as Efficient Frontier or Minimum Variance Portfolio where the risk (and hence the volatility) should be taken into consideration in order to statistically optimise the risk of a given portfolio.

Hang (2019), highlighted the importance of forecasting stating that ''Forecasting is an important tool, an indispensable part in the operation of businesses to help create a competitive advantage''. This becomes clear whether you are a financial institution that wants to optimise its portfolio exposure to bitcoin and speculate on financial options or the owner of a bitcoin mining operation that has several fixed costs and wants to protect its revenue from the high volatility of the cryptocurrency markets, by applying hedging techniques.

Engle and Patton (2001), point out several stylised facts about asset price volatility that we should take in consideration: (i) volatility exhibits persistence, in the sense that large changes on the price of a given assets are usually follower by other large changes and small changes and usually followed by small changes (clustering); (ii) Volatility is mean reverting, meaning that it there is a mean level where the volatility will tend to return to; (iii) There might be asymmetric impacts, as positive and negative shocks in price might impact volatility differently; (iv) exogenous variables may influence volatility; (v) Tail probabilities, since unconditional distribution of asset returns has heavy tails.

Faced with this, several authors have applied the most different techniques to forecast the volatility of time series data, from more classical econometric models to the use of new computational techniques such as deep-learning that was made possible with the latest technological advancements.

Some of the most important models to forecast volatility across the literature include ARCH developed by Engle (1982), and later a model improved by Bollerslev (1986) entitled Generalised Autoregressive Conditional Heteroscedasticity (GARCH), allowing lagged conditional variances.

Bergsli et al. (2022) made a paper discussing which models would be most suitable for Bitcoin's Realised Volatility Forecasting. In their research, they analysed a range of GARCH models (GARCH, EGARCH, GJR-GARCH, IGARCH, MSGARCH, and APARCH) and two HAR models (regular and logarithmic). They utilise several different types of forecasting horizons on their predictions (1, 2, 5, 10, and 15 days ahead) concluding that from the GARCH models, EGARCH and APARCH performed better. However, both HAR models perform better than any GARCH models analysed.

Gronwald (2019), discussed the implications of both linear and non-linear GARCH models with the goal of analysing extreme price movements in the Bitcoin Market and utilising the same set of models for crude oil and gold, considering a GARCH(1,1) as the benchmark.

Kim and Won (2018) agree on the advantages of such models since they are able to capture the volatility clustering and heteroskedasticity (proprieties that Financial time-series seems to have) and leptokurtosis, which consists of statistical distributions that have a kurtosis that is higher than three.

However, as mentioned before, when dealing with Bitcoin's volatility, we need to take into consideration that historically, cryptocurrencies have higher volatility than other asset traditional classes and their returns have a set of anomalies which might result in predictive problems for the mentioned models, as discussed by De Nicola (2021) regarding his research on the intraday behaviour of Bitcoin.

Ramos (2021) state that although with a simple application, classic linear methodologies have some difficulties dealing with events that have out-of-the-ordinary patterns based on the thoughts of Pesaran and Timmermann (2004), that argue that structural instability can have a negative impact on several traditional predictive econometric models and Chatfield (2016), stating that constant instability regarding financial-series can result in behaviours on the historical data that can make them difficult to model and predict.

Figlewski (1997) reinforces this idea in his research work regarding Forecasting Volatility, stating that ''important implications for volatility estimation tend to be overlooked by those following traditional lines of thought''.

Although most economists try to model volatility using more classical models, the advancements of computational power are currently providing interesting alternatives to face the issues mentioned before with the help of Neural-Networks.

This situation is discussed by Wilson and Spralls (2018), it is of great importance of adapting the forecasting tools that are taught to real live scenarios, highlighting the fact that most of the forecasting methodologies currently utilised in business by professionals are the classic models, such as moving averages, linear regression and exponential smoothing.

The literature on forecasting and artificial neural networks methodologies is vast, as we can be confirmed by several literature review compendiums such as the ones compiled by Sezer et al. (2020) which covers financial time series forecasting with deep learning from 2005 to 2019, and Tealab (2018) that compiled a systematic review of time series forecasting using neural network models from 2006 to 2016.

Nevertheless, there is the need to highlight the research work after those periods forecast, with the focus on forecasting volatility and prices of cryptocurrencies and/or financial time series using classical econometric methodologies such as ARCH and GARCH, deep-learning models and hybrid models with both methodologies. Here are some examples:

Pichl and Kaizoji (2017) implemented a Heterogeneous Autoregressive (HAR) model for realised volatility of bitcoin and fed these values to a neural network with 2 hidden layers using a 10-day moving window in order to predict the next-day logarithmic return. Although the authors conclude that such a neural network model is capable of forecasting, they suggest that RNN and LSTM techniques might be useful if we aim for better prediction accuracy.

Balcilar et al. (2017) explore the relationship between returns and volatility (squared values of returns) with trading volume and try to forecast it with the causality-in-quantiles technique. They conclude that there is no evidence that volume granger causes returns or volatility but when forecasting using the causality-in-quantiles they argue that transaction volume can sometimes help predict returns but convey no information on volatility and that they detected nonlinearity and structural breaks on the two variable relations. The authors also emphasise the importance of studying the nonlinearity of the variables when analysing predictability.

Bariviera (2017) when researching about bitcoin’s inefficiency, concludes that there is empirical evidence bitcoin returns and volatility display long-memory characteristics when using the Hurst exponent technique. They also identified a regime shift for the daily returns, since from 2011 to 2014, the returns were persistent and after this period, the behaviour was more similar to white noise. However, for volatility (measured as the logarithmic difference between daily maximum and minimum price), the behaviour was consistent during all the data used for the research.

Mallqui and Fernandes (2019) also tried to forecast the direction, maximum, minimum and closing prices of daily Bitcoin Exchange Rate with machine learning algorithms, concluding that a combination of RNN and a Tree Classifier are the best model to predict the Bitcoin price direction and a Support Vector Machines (SVM) for forecasting the Bitcoin exchange rate, stating that the results obtain with the machine learning model achieved an improvement of more than 10% accuracy for price direction forecasting when compared to other state-of-the art papers that studied the same periods.

Kristjanpoller and Minutolo (2018) proposed a hybrid volatility forecasting framework that utilises the different types of GARCH models to feed to a Neural Network Model in order to forecast price volatility of bitcoin, forming what they conceptualised as an Artificial Neural Network-Generalised Autoregressive Conditional Heteroskedasticity (ANN-GARCH), showing again, how the application of Neural Networks has become a trend to improve the forecast of the classical auto-regressive models. The authors also mention that several volatility studies have demonstrated improved forecasting results when neural networks are included in the model.

Lahmiri and Bekiros (2019) applied deep learning models to forecast the price of Bitcoin, Digital Cash and Ripple, which the authors considered as chaotic financial data structures. From the nonlinearity tests performed, the digital currencies analysed exhibit fractal dynamics, long memory and self-similarity. The authors concluded that the best model to forecast the price was an LSTM model.

Lahmiri and Bekiros (2020) also made research regarding forecasting high-frequency Bitcoin price series by employing two statistical models, with Support Vector Regression (SVR) and Gaussian Regression Poisson (GRP) models; two algorithmic models: with regression trees (RT) and the k-nearest neighbours (kNN); and finally three artificial intelligence models: Feedforward Neural Networks (FFNN), Bayesian Regularisation Network (BRNN), and Radial Basis Function Networks (RBFNN). They conclude that BRNN has the best forecasting properties and highlight the overall superiority of artificial neural networks with a set of advantages when forecasting and modelling large data sets such as bitcoin.

Ramos (2021) in his works regarding modelling and forecasting the Portuguese Stock Index 20 (PSI 20) and the Standard & Poor's 500 Exchange-Traded Fund (SPY) using classical models such as ARMA and ETS with DNN. He states that nonlinear neural networks can be an alternative to model and forecasting financial time series, however, with the need for higher computational power and resources to execute them. The author also argues that LSTM models seems to underperform when forecasting stationary data series.

Ramos et al. (2022) proposed a new hybrid approach using Box&Jenkins methodology for Neural Networks. This solution resolves the issue of high computational costs of running Neural Network methodologies since it was able to reduce the implicit computational time by 20% while keeping the forecasting quality when compared to MLP and LSTM models taking in consideration the MAPE.

Nevertheless, even with this research work, authors such as Hayes (2017), Pichl and Kaizoji (2017), and Lahmiri and Bekiros (2019) point out the fact that there still is a reduced number of research work done regarding Bitcoin and cryptocurrencies which creates value to the development of the current dissertation as few works regarding these subjects have been developed since the previously cited authors. In addition to these, the mentioned research showed different models and methods to forecast, hence the necessity of this research to verify which one might be the best fit to forecast the volatility of this particular asset, that as shown before exhibits unique characteristics, as to do it with more recent data available.

2 - Time Series

2.1 - General Concepts of Time Series

In simple terms, a Time Series might be characterised as a successive sequence of data points over a certain period of time, as explained by Kitagawa (2022). It can be characterised as:

$$Y_t(discrete), t = 1,2,\dots, n, \quad t \in T \subseteq \mathbb{N} \quad \text{or} \quad Y(t)\ (continous), t \in T \subseteq R$$

2.1.1 -Time Series Components

As mentioned by Ramos 2021), in order to perform a deep analysis of the time series, it is of great importance to take in consideration its trend, cyclicity, seasonality and randomness where:

- **Trend** reflects an evolution of the directions (increasing or decreasing) of the monotony of the series on the long term. It can show linear and non-linear dynamics.
- **Cyclicity** reflects fluctuation pattern on the medium-term that can directly affect the trend of the series. They can be periodic and non-periodic but don't show a fixed frequency.
- **Seasonality** reflects regular cyclical movements that are observed in constant but smaller periods. On the opposite of cyclicity, these movements show a fixed frequency and are usually related to natural factors, like time of the year.
- **Randomness** reflects unpredictable fluctuations that cannot be modelled by any of the previously mentioned components.

2.1.2 – Stationarity

When we aim for analysing and modelling time series data, we need to take to take in consideration the assumption that the values are generated from an unknown stochastic process where the evolution of a set of variables indexed by time is a random phenomenon.

Nonetheless, in order to been able to make valid statistical inferences we still need to test for several factors such as stationarity, as the models also make such assumptions.

The definition of stationarity consists in the fact that a given time series $\{y_t\}_{t \in T}$ is weakly stationary or covariance stationary if, to $t, t-k, t-j, t-j-k \in T$ arbitrary.

1. **Average Expected** Values as:

$$E(y_t) = E(y_{t-k}) = \mu \tag{2.1}$$

2. **Variance** Values as:

$$E((y_t - \mu)^2) = E((y_{t-k} - \mu)^2) = \sigma_y^2 < \infty \tag{2.2}$$

3. **Covariance** Values as:

$$E((y_t - \mu)(y_{t-k} - \mu)) = E((y_{t-j} - \mu)(y_{t-j-k} - \mu)) = \gamma_k \quad (2.3)$$

Therefore, a given time series is weakly stationary if the average and variance are constant over time and not a function of time, just as the covariances between lagged values are equally constant. If the time series do not follow this requirement, we say that the time series is non-stationary.

Non-Stationary is a very common feature of financial time series, as they usually have trends and/or seasonality and do not show constant mean values and variance. However, regarding volatility (the aim of this study), there is a tendency to see stationary features within the series.

2.1.3 – Unit Root

The existence and analysis of unit roots on financial time series have been the objective of the study of several authors such as Libanio (2005).

Taking in consideration the following model:

$$y_t = \rho y_{t-1} + \beta_0 + \beta_1 t + u_t \quad (2.4)$$

When we are trying to infer the nature of the time series in order to understand if there is the existence of a deterministic or stochastic trend, we need to take into consideration the following four scenarios:

1. If $\rho = 0,$ we are in the presence of a stochastic process that has a linear trend, and therefore, a deterministic trend.
2. If $\rho = 1$ and β_1 = 0, there is a stochastic trend where we obtain a purely stochastic process on first differences, also known as a random walk.
3. If $\rho = 1$ and $\beta_{i=0,1} \neq 0$, we obtain a random walk with drift[4] and deterministic trend.
4. If $0 < \rho < 1,$ there is a serial correlation but not a stochastic trend.

The first three scenarios will result in the series being non-stationary. In general terms, when we test if $\rho = 1$ we are testing the non-stationary of the time series, or in other terms, test the unit root.

	Unit Root Test	
$H_O: \rho = 1$	*VS*	$H_1: \rho < 1$
(Non-Stationary Series)		*(Stationary Series)*

[4] As explained by (Ramos, 2021), one of the characteristics of these types of processes is the persistence of random shocks (processes with infinite memory). The effect of each error term does not dissipate over time and the process saves the information of all the shocks that have occurred.

Thus, in the case of not rejecting $\rho = 1$, we can assume that there is a unit root, and the series is non-stationary.

From the different Unit Root Tests on existence, the most common ones we find on the literature are Dickey-Fuller (DF) / Augmented Dickey-Fuller (ADF) and the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test.

The Dickey-Fuller (DF) test has the assumption that the errors are independent and identically distributed (*i.i.d.*) and therefore do not have any autocorrelation.

When we subtract Y_{t-1} in both sides of the equation:

$$Y_t = \rho Y_{t-1} + \beta_0 + \beta_t + \varepsilon_t \tag{2.5}$$

We obtain:

$$Y_t - Y_{t-1} = (\rho - 1)Y_{t-1} + \beta_0 + \beta_t + \varepsilon_t \tag{2.6}$$

Where:

$H_0: \rho = 1$, (and the series is non-stationary).

$H_1: \rho < 1$, (and the series is stationary).

In situations where ε_t is not white noise, there is the need to introduce a correction, known as Augmented Dickey-Fuller (ADF). Phillips and Perron (1988), suggested that the solution should include the increase of the regression with lagged components with the objective of cleaning the residues and eliminating the autocorrelation. Therefore, the ADF test is considered a new model from (2.6) that we obtain by adding the lagged values of the independent variables $Y_t - Y_{t-1}$, represented as Δy on the following equation:

$$\Delta y_t = (\rho - 1)y_{t-1} + \sum_{i=1}^{k}(\rho - 1)_i \Delta y_{t-1} + \beta_0 + \beta_t + \varepsilon_t \tag{2.7}$$

In addition to the previously mentioned tests, KPSS serves as a complement to testing the unit root and evaluating the stationarity of a time series.

This test complements the Unit Root Test results, and it tests the null hypothesis of the stationarity of a given timeseries against the hypothesis of non-stationary. This might help to clarify some cases where there is not enough information to guarantee the existence of the Unit Root.

Therefore, assuming a series with n observations is expressed as the sum of trend, random walk and a residual that is stationary, these tests test if the random walk having null variance.

Therefore, we can consider:

$$y_t = \beta_1 t + \xi_t + \varepsilon_t \tag{2.8}$$

where ξ_t represent the random walk and ε_t is stationary with:

$$\xi_t = \xi_{t-1} + \zeta_t, \qquad \zeta_t \sim WN(0, \sigma_v^2) \tag{2.9}$$

and where we can say that in the case of $\sigma_\zeta^2 = 0$, we have $\xi_t = \xi_0$ for any $t \in T$ and therefore y_t where $t \in T$ is stationary.

With this said, we can write KPSS with the following form:

Kwiatkowski-Phillips-Schmidt-Shin Test (2)

$H_O: \sigma_\zeta^2 = 0$ *vs* $H_1: \sigma_\zeta^2 > 0$

When we consider that $S_t = \sum_{i=1}^{t} \varepsilon_t$ and the $\hat{\sigma}_{\varepsilon_t}^2$ as the estimator for the variance of the error, the test statistic is given by:

$$\tau KPSS = \frac{1}{n^2} \sum_{t=1}^{n} \frac{{S_{\varepsilon_t}}^2}{\hat{\sigma}_{\varepsilon_t}^2} \tag{2.10}$$

As mentioned by Kwiatkowski et al. (1992) the KPSS has a distribution that converges to a Brownian Movement asymptotically for the critical values for the significance of (*1%, 5%* and *10%*).

2.1.4 – Structural Breaks

Although it was previously discussed the implications of modelling and analysing non-stationary time series and how the literature proposes some solutions, they make the assumption that the behaviour of parameters of the series will remand constant. However, there are several examples where we can have sudden changes on such parameters. This is what the literature calls ‘’structural breaks’’ or ’’structural changes’’, usually common on financial time series.

According to Hansen (2001) we are dealing with a structural break when at least one of the parameters used in the model suffers a change in its behaviour on a given moment – break dates. Structural Breaks in time series data is defined as an instability of the parameters of a given forecasting model or any other data generation process.

Structural breaks can be shown in several different ways: They could affect all the parameters of the model or just a few; they can be abrupt or be a gradual process they can occur on a known specific date or an unknown one; they can occur once or several times.

The author also highlights the importance of considering structural changes when researching and modelling with time series as its negligence might result in misleading results, wrong economic inferences and detriment of the stability and accuracy of the forecasting models.

When discussing this issue, is also important to mention the research of Valentinyi-Endrész (2004), as it allows to understand the reasons why such changes might occur.

According to the author, structural breaks are usually associated with meaningful economic and political shifts. The implementation on new political and economic policies, the change in interest rates by the Central Banks or implementation of new regulation are some examples of events that can result in such breaks.

With this said, the importance of understanding structural breaks when analysing or forecasting time series data is abundant and evident on the literature as by not doing so, it can lead the researcher to incorrect conclusions.

There are several different tests can help us test for structural breaks of the data series in study. The focus of this study will take in consideration the CUSUM test that was developed by Brown et al. (1975). This test analyses of the variances of the residual component over time in order to find structural breaks. This test takes the null hypothesis the stability of the parameters. This can be seen has a test to the stability of u_t, the variance of the residual component. (Ramos, 2021)

$$Var(u_t) = \begin{cases} \sigma_1^2, & t \leq T_1 \\ \sigma_2^2, & t > T_1 \end{cases} \tag{2.11}$$

Therefore, for each $1 \leq t \leq n$ moment, we consider the hypothesis test as:

Structural Breaks - CUSUM

$H_O: \forall\ i\ \neq j, \sigma_i^2 = \sigma_i^2, i, j = 1, \dots n$ *vs* $H_1: \exists\ i\ \neq j, \sigma_i^2 \neq \sigma_i^2, i, j = 1, \dots n$

(No evidence of Structural Breaks) (Evidence of Structural Breaks)

2.1.5 – Auto-Covariance and Auto-Correlation Functions

In order to fully study and analyse stationary processes, there is the need to consider auto-covariance (FACV), autocorrelation (FAC) functions.

Considering that the series $\{y_t\}_{t \in T}$ is stationary and that the stochastic processes are at least, stationary on the 2nd order and therefore weakly stationary, we define:

- The *mean and the variance* of the series as:

$$E(y_t) = \mu \quad \text{and} \quad Var(y_t) = \sigma^2 \tag{2.12}$$

- The *FACV* of the series as:

$$\gamma_k = Cov(y_t , y_{t+k}) = E[y_t , y_{t+k}] - \mu_t \mu_{t+k} , \quad k \in Z \tag{2.13}$$

- The *FAC* of the series as:

$$\rho_k = \frac{Cov(y_t , y_{t+k})}{\sqrt{Var(y_t)\, Var(y_{t+k})}} = \frac{\gamma_k}{\gamma_0}, \quad k \in Z \tag{2.14}$$

Since $Var(y_{t+k}) = Var(y_t) = \gamma_0$

With the exception of very specific cases, an increase in *k* will result in a decrease of γ_k and ρ_k and is expected for the memory capacity of the process to be limited. Nonetheless, on the majority of cases, we have $|k| \rightarrow \infty \Rightarrow \gamma_k \rightarrow 0$ and $\rho_k \rightarrow 0$.

2.2 – Univariate Linear Models

Univariate Linear Models consist in models that only take in consideration data from the time series itself.

From this type of models, the most common ones would be ARMA, when we are trying to study stationary time series and ARIMA, when we are trying to study non-stationary time series.

However, when we are discussing volatility, according to Engle and Patton (2001), we can separate the volatility models in two types: the models that utilises conditional variances as a function of observables, like ARCH and GARCH models, and the second type that model volatility not as a function but purely of observable variables, like stochastic volatility models.

2.2.1 – Autoregressive Models – AR*(p)*

The autoregressive model is a linear regression model in which the regression variables are the process values at p previous times.

If we take in consideration that the values of time (*t*, *t* – 1 , *t* – 2 , …) as $(y_t, y_{t-1}, y_{t-2}, \ldots)$ we obtain:

$$y_t = c + \sum_{i=1}^{p} a_i y_{t-j} + \varepsilon_t \tag{2.15}$$

This is the basis of what we called the autoregressive model of order p , AR(p), where $(a_{1,} a_2, a_3, \ldots)$ is a vector of the coefficients (real constants or parameters) and where the white noise (a random process) is given by ε_t .

If we consider L as the lag operator, represented by $L(y_t) = y_{t-1}$, the AR(p) can also be represented as:

$$\phi_p(L)y_t = c + \varepsilon_t \tag{2.16}$$

and where $\phi_p(L) = 1 - \sum_{i=1}^{p} a_i L^i$.

Therefore, we can state that a given autoregressive model of order p, AR(p), is stationary if all the roots $z = \frac{1}{a_i}$ of the characteristic polynomial:

$$\phi(z) = 1 - a_1 z - a_2 z^2 - \cdots - a_p z^p \tag{2.17}$$

are situated in the exterior of the unitary circle, or $|a_1| < 1$.

The autocorrelation function of an stationary AR(p) process with a mean of $E(y_t) = 0$ and taking in consideration equations (2.14) and (2.15), is given by:

$$\rho_k = a_1\rho_{k-1} + a_2\rho_{k-2} - \cdots + a_p\rho_{k-p} \tag{2.18}$$

and its variance by:

$$Var(y_t) = \frac{\sigma^2}{a_1\rho_1 + a_2\rho_2 - \cdots + a_p\rho_k} \tag{2.19}$$

2.2.2 – Moving Average Models – MA*(q)*

The moving average (MA) processes of order q, use the average of a given number of previous observations to forecast new predictions.

As mentioned by Murteira et al. (2000), these models aim to represent y_t in terms of a random process ε_t. The effects produced by innovation of such processes, only last for a short period of time, on the contrary of autoregressive processes effects, that last longer.

With this said, we can describe a MA process of order q, MA(q), can be described as:

$$y_t = \varepsilon_t - \sum_{j=1}^{q} b_t \varepsilon_{t-j} \tag{2.20}$$

Where ε_t is white noise (random process) and $b_1, \ldots, b_q$ are real constants.

Therefore, a MA process of order q, results on a weighted average of the last q + 1 observations of a white noise process in each instant t. A *MA* process is always an independent stationary process of $b_1, \ldots, b_q$ values.

Because of this, since $E(\varepsilon_t)$ = 0, we can also verify that $E(y_t)$ = 0, where the ε_t is not correlated, we have:

- *Variance* as:

$$Var(y_t) = \sigma^2(1 + b_1^2 + \cdots + b_q^2) \tag{2.21}$$

- *FACV* (with $\gamma_0 = Var(y_t)$) as:

$$\gamma_k = \{ \quad -\sigma^2\,(b_k + b_{k+1}b_1 + \cdots + b_q b_{q-k}\,, \quad 0 < k \leq q\; 0\,, \qquad k > q \tag{2.22}$$

- *FAC* as:

$$\rho_k = \{ \quad -\frac{b_k + b_{k+1}b_1 + \cdots + b_q b_{q-k}}{1 + b_1^2 + \cdots + b_q^2}, \quad k \leq q\; 0\,, \qquad k > q \tag{2.23}$$

It is also important to mention that a MA(q) processes are, meaning that it is possible to transform MA(q) into AR(∞). In addition to this, the first q autocorrelation coefficients are not null and the rest being equal to zero. FAC is null from order q and FACP has an exponential decay smoothed to zero.

2.2.3 – Autoregressive Conditional Heteroskedasticity – ARCH*(q)*

According Engle and Patton (2001) the volatility models are separated in two types: the models that utilises conditional variances as a function of observables, like ARCH and GARCH models, and the second type that model volatility not as a function but purely of observables[5].
Engle (1982) introduced stochastic models in the form of:

$$y_t = \varepsilon_t {h_t}^{1/2} \tag{2.24}$$

where:

$$h_t = \alpha_0 + \alpha_1 y_{t-1}^2 + \cdots \alpha_q y_{t-q}^2, \quad \alpha_0 > 0, \alpha_i \geq 0, \sum_{i=1}^{q} \alpha_1 < 1 \tag{2.25}$$

It also important to mention that ε_t are *i.i.d.* with $E(\varepsilon_t) = 0$ and the $Var(\varepsilon_t) = 1$. This is nothing less than the ARCH model of order q.

When we add the assumption of conditional normality and Φ_{t-1} as the set of information available at $t - 1$, we can write the model as it follows:

$$y_t \mid \Phi_{t-1} \sim N(0, h_t) \tag{2.26}$$

The author also adds that the non-negativity of the $\alpha_i s$ is required for the variance to be non-negative, whereas the requirement that the $\alpha_i s$ sum to less than one is needed for y_t to be wide sense stationary.

[5] For examples of such models, consult the original paper of Engle and Patton (2001).

2.2.4 – Generalized Autoregressive Conditional Heteroskedasticity – GARCH *(p, q)*

Bollerslev (1986) proposed the incorporation of a moving average component with an autoregressive component to the ARCH(*q*) model, forming what we know as GARCH.

This difference allows for the conditional variance to be dependent of its own past values of squared errors and past conditional variances.

On this model, the conditional variance is given as:

$$h_t = \alpha_0 + \sum_{i=1}^{p} \beta_i h_{t-i} + \sum_{i=1}^{q} \alpha_i y_{t-i}^2, \quad \alpha_0, \beta_i, \alpha_i > 0, \sum_{i}^{p} \beta_i + \sum_{i}^{q} \alpha_i < 1 \tag{2.27}$$

The non-negative conditions also imply a non-negative variance, while the condition that the sum of $\alpha_i s$ and $\beta_i s$ is smaller than on required for wide sense stationarity of y_t.

2.2.5 – Comparison: Models and Forecasting Evaluation Metrics

As discussed by Campbell et al. (1997), GARCH models show advantages to ARCH models since the model uses fewer parameters, results in less probably of violating the non-negativity constraints and that they incorporate the previous forecasted variance to forecast the following estimations.

As mentioned by Zivot (2008), the GARCH(*p , q*) seems to be a better model to study and forecast time series data that exhibits heteroskedasticity and volatility clustering. However, this model seems to have difficulties capturing ''leverage effects''.

Several authors agree that a GARCH(1,1) is often sufficient to forecast volatility (such as Brooks (2014) and the other works cited on the literature review). However, both models are not able to capture asymmetric volatility, and to cope with this problem, several GARCH based models have been proposed, such as EGARCH by Nelson (1991).

Nonetheless, there are several criteria that can be used to empirically select the models with most know methodologies be Akaike Information Criterion (AIC) developed by Akaike (1974) and Bayesian information Criterion (BIC) by Schwarz (1978). The AIC values precision with a function that penalises models with a high number of parameters whereas the BIC, incorporates the likelihood function with a Bayesian formalism that penalises models with a higher number of parameters more rigorously.

In a strict sense, if *k* is the number of parameters of the model and $\mathcal{L}$ the value of the correspondent maximum likelihood function, the statistic value of AIC is given by:

$$S_{AIC} = -2\ log(\mathcal{L}) + 2k \tag{2.28}$$

The statistic value of BIC, where k is the number of parameters of the model and $\mathcal{L}$ the value of the correspondent maximum likelihood function is given by:

$$S_{BIC} = -2\ log(\mathcal{L}) + k\ log(n) \tag{2.29}$$

In addition to this, we can complement our analyses with the use Diebold-Mariano Test proposed by Diebold and Mariano (1995). If you take in consideration the forecasts results of two models (e.g Model A and Model B), the test evaluates the null hypothesis H_O of the mean of the loss differential on Model A being lower or equal than of the model B. Therefore, by rejecting H_O, is means that the forecast of Model B is significantly more accurate than the forecasts of Model A.

If we define the residuals of two forecasts as:

$$e_i = \hat{y}_{it} - y_t, \quad i = 1{,}2 \tag{2.30}$$

Where y_t are the real values and $\hat{y}_{it}$ is the two forecast values and let the loss-differential function be defined as[6]:

$$d_i = |e_1| - |e_2| \quad \text{or} \quad d_i = {e_1}^2 - {e_2}^2 \quad \text{or ...} \tag{2.31}$$

With this, we can define:

$$\underline{d} = \frac{1}{n}\sum_{i=1}^{n} d_i \qquad \mu = E[d_i] \tag{2.32}$$

And define the Autocorrelation Function γ_k as the autocovariance of lag k for $n > k \geq 1$ as:

$$\gamma_k = \frac{1}{n}\sum_{i=k+1}^{n} (d_i - \underline{d})(d_{i-k} - \underline{d}) \tag{2.33}$$

We can define the Diebold-Mariano statistic for $h \geq 1$ as[7] :

$$DM = \frac{\underline{d}}{\sqrt{\left[\gamma_0 + 2\sum_{k=1}^{h-1}\gamma_k\right]/n}} \tag{2.34}$$

Regarding the statistical hypothesis testing, we could describe it as it follows:

[6] Note that this formula will be based on the error statistic we decide to define. On this case, the first formula is for MAE and the second formula for MSE, but others may be applied to the test.

[7] Where is generally sufficient to use the value $\mathrm{h} = n^{\frac{1}{3}} + 1$

Diebold-Mariano Test		
$H_O: E(d_i) = 0 \ \ \forall t$	*VS*	$H_1: E(d_i) \neq 0$
(Forecasts have the same accuracy)		*(Forecasts have different levels of accuracy)*

Where under the assumption of the null hypothesis H_O it follows a normal distribution $DM{\sim}N(0,1)$.

2.3 – Deep Neural Network Models

The interest on Artificial Intelligence, Machine Learning and Neural Networks to solve economic problems has increased clearly increased on the last couple of years. This interested is justified as the several researchers using this type of approaches and techniques seem to have improved results over the more ''classical'' methodologies in various cases and therefore, help us better understand the non-linear part of our economic reality.

The main components of neural network are neurones (or perceptron's) and layers. Artificial neurons behave in similar way to a human neuron, where it receives a set of inputs, processes its inputs based on its function and produces an output signal that is transmitted to other neurones.

The most common version of neurone model is the ones described by Rosenblatt (1958) and Minsky & Papert (1969) where generically we can describe a perceptron k, p_k, by the sum of *m* inputs, $X = (x_1, x_2, \ldots x_m)$, each one with a weight w_{kj} with $j = 1, \ldots, m$, with an output y_k, that is defined by a given activation function φ an external bias, b_k, as shown in on the *Figure 2.1* .

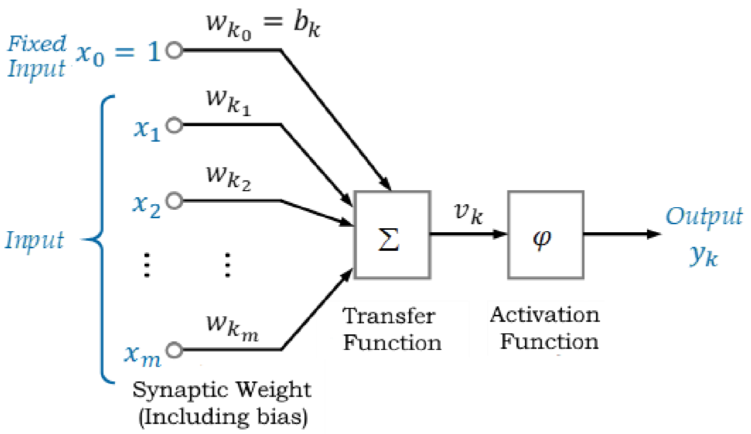


Figure 2.1 – Non-Linear Model of *perceptron*

Source: Adapted from Ramos (2021)

This can be represented as:

$$p_k = \sum_{j=1}^{m} w_{kj}x_j = W^T X \quad (2.35)$$

$$y_k = \varphi\,(v_k) \quad (2.36)$$

where $W = (w_{k1}, w_{k2}, \dots, w_{km})$ represents the vector of synaptic weights associated with perceptron k, that is known as parameterization vector and v_k is the activation potential of the bias b_k, that is equal to $p_k + b_k$.

Another important feature is that each neurone can have an activation function φ, that acts as a filter that defines if a given neurons should be activated or not, allowing it to give a neural network non-linear propriety. A few examples of activation functions can be seen on *Figure 2.2.*

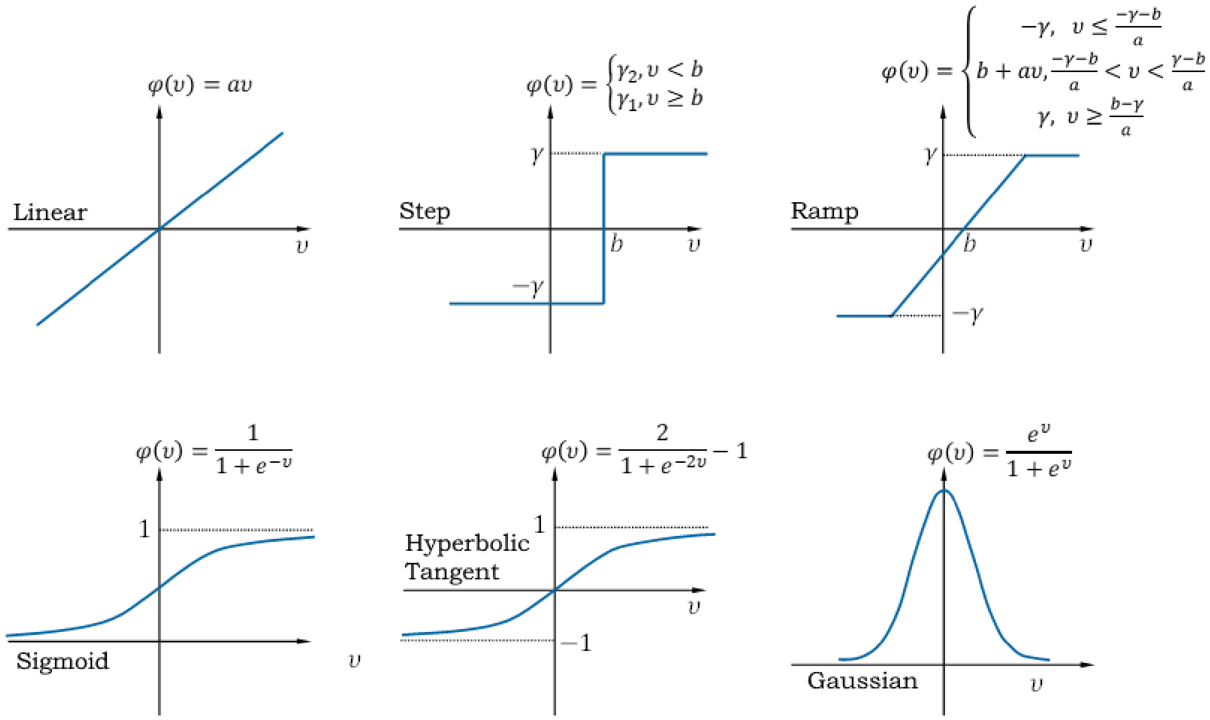


Figure 2.2 – Examples of Activation Functions (linear and non-linear)

Source: Adapted from Ramos (2021)

The other component of a neural network are the Layers. Layers are an organised set of neurons that have the same activation function and a given number of neurons.

Their connections of the neurons are related to its weight, as the signal outputted by a given neuron is multiplied by the weight of its own weight, before being passed to the other neurons of the layer.

There are three common layers, *shown on Figure 2.3*, that should be highlighted:

- **Input Layer:** This layer has the goal of receiving the inputs of the explanatory variables of each observation and therefore will have as many neurons as inputs that we wish our model to have. It will pass act as a pass through to the hidden layers without altering any data of the input variables.
- **Hidden Layer:** This layer is characterised by having an arbitrary number of neurons and a single activation function. There can be many hidden layers that can transform the inputs into new information to pass as output to the next layers. You can have several Hidden Layers where l, denotes the number of it.
- **Output Layer:** This layer is the last layer of the neural network, and it will have as many neurons as the desired output number. It will have an activation function according to nature of our problem.

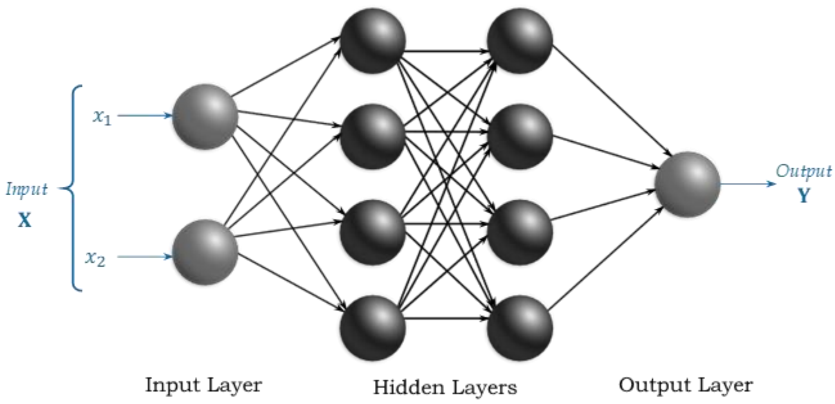


Figure 2.3 – Neural Network with two Hidden Layers

Source: Adapted from Ramos (2021)

The synaptic weights, *W*, are calculated in a methodology called ''Learning Process''. This training process depends on the type of network, the learning algorithm, learning paradigm and the training set. After this training, we can evaluate the performance of the model with a given independent subset.

Regarding the learning paradigms, as shown on *Figure 2.4,* there are three[8] main classes that should be highlighted:

- **Supervised Learning:** where the parameters are optimized trough iteration having in consideration the input patterns and the error signal to reach our output goal. It is used for regression and classification problems and generally uses Backpropagation algorithm.
- **Unsupervised Learning:** where is given to the neural network a set of weight adaptation rules and the network iterates in order to understand the patterns of the

[8] There are also authors that use a fourth option called ''semi-supervised learning'' that is a mix of supervised and unsupervised learning.

data set. It is mainly used on clustering, association, and dimensionality reduction problems. (Becker and Plumbley, 1996)

- **Reinforcement Learning:** where the network is rewarded or punished based on the desired goal and it will try to maximise the reward through trial and error with a given environment. There is only a qualitative evaluation, and the network receives a heuristic reinforcement signal. (Barto et al., 1983)

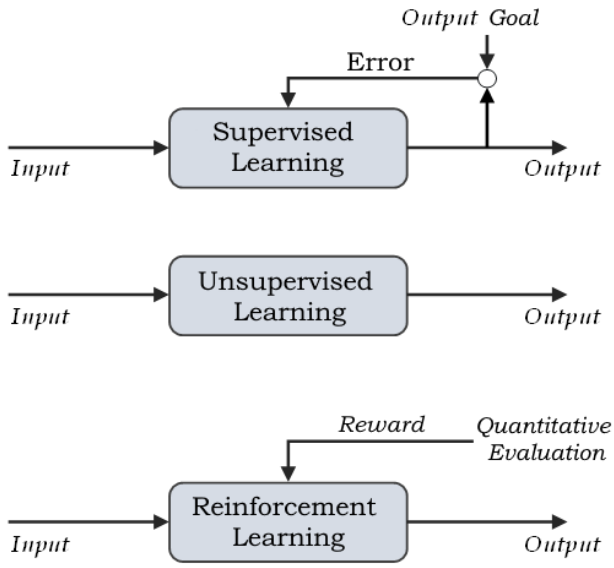


Figure 2.4 – Three main Learning Paradigms

Source: Adapted from Ramos (2021)

Focusing on the supervised learning paradigm, when we consider $d_i(t)$ and $y_i(t)$ as the desired output signal and the one obtained by the neurone i of the output layer on instant t, we can define the error signal as:

$$e_i(t) = d_i(t) - y_i(t) \tag{2.37}$$

The error signal and converted into real numbers by an error function, $C(\boldsymbol{W})$, and the weights and bias of the network will be optimized based on the minimization of such function. According to Bishop (1995), there are the most common cost functions for supervised learning:

- *Error Sum of Squares:*

$$\sum_{i=1}^{m} e_i^2 \tag{2.38}$$

- *Minkowski Error:*

$$\sum_{i=1}^{m} |e_i|^R, \ with\, R \in N \tag{2.39}$$

- *Mean Squared Error:*

$$\frac{1}{m}\sum_{i=1}^{m} e_i^2 \tag{2.40}$$

- *Cross-entropy:*

$$-\sum_{i=1}^{m}[t_i \, log \, log \, y_i + (1 - t_i) \, log \, log \, y_i \ ] \tag{2.41}$$

The minimization problem of $C(\boldsymbol{W})$ solution can be found with the error surface gradient[9] where the hypersurface is the cost function in a space with dimesons n, with n the dimension of $\boldsymbol{W}$:

$$\nabla C(W) = \frac{\partial C(W)}{\partial W} \tag{2.42}$$

And the minimization problem is reach trough iteration, by a applying a learning rate, η, in each iteration given by:

$$W(t + 1) = W(t) - \eta \, \nabla C(W) \, |_{w=w(t)} \tag{2.43}$$

The most common learning method for supervised learning was published by (Rumelhart et al., 1986) and is called Backpropagation. On simple terms, the goal of this algorithm is to minimize the error function by using a gradient technique.

However, it's important to notice that the training process can become extremely slow, regardless of the learning rate, due to the non-linearity of the activation function or the high number of parameters and it has the potential of converging to local minimums on the error function that will be different from the global minimums. [10]

2.3.1 – Multilayer Perception

The Multilayer Perceptron (MLP) architecture (previously shown on *Figure 2.3)* are a simple form of Neural Network that have the goal of training the model using a set of input data in order to reach a generalisation. The training of this model is done using the Backpropagation[11] algorithm where the network parameters, weights, and bias, are adjusted by minimising the error. However, is important to mention that on the majority of the cases, the learning process duration (training epochs) is set by the researcher as an hyper parameter of the network. In

[9] In cases where the cost function is the Error Sum of Squares and the activation function is linear, the solution can be calculated analytically.

[10] For a deep explanation on this algorithm, consult Ramos (2021)

[11] In cases where is impossible to escape to the converge of the local minimum, the literature suggests the use of some adaptations like *Mini-Bach Gradient Descendent* or ADAM, introduced by Kingma and Ba (2014)as mentioned by Haykin (2009).

addition, this model tends to be less robust over other methodologies has it doesn't take in consideration a time factor, that might be a limitation when dealing with time series data.

2.3.2 – Recurrent Neural Networks

The Recurrent Neural Network Architecture (RNN) is characterised by having a double learning process. This model has a first training cycle (following the methodologies of MLP networks) and an auto-learning process.

Although it can have an arbitrary number of hidden layers, its characterised by having a feedback loop where the neuron uses its last output as an input on a following future (that is, following training epoch), sharing the weights matrices over time as shown on *Figure 2.5.* Because of this, they tend to perform better when modelling and forecasting time series data.

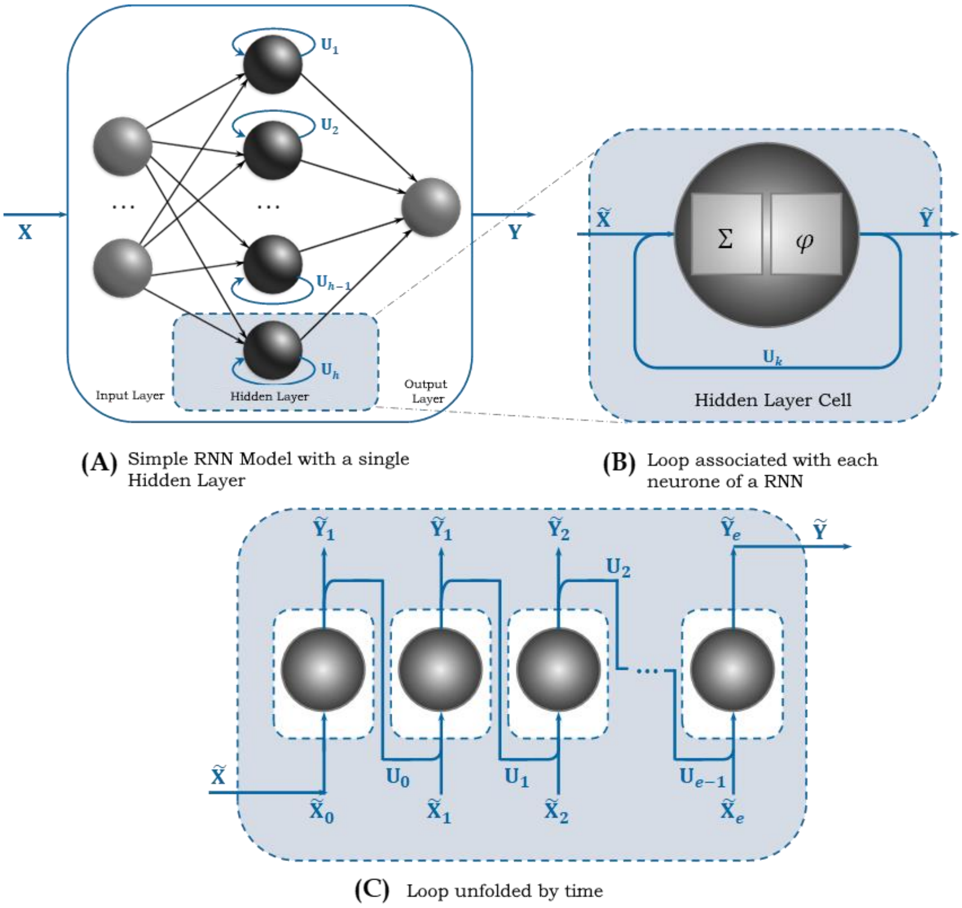


Figure 2.5 – RNN Architecture

Source: Adapted from Ramos (2021)

2.3.3 – Long Short-Term Memory

The Long Short-Term Memory (LSTM) networks are a type of RNN that allows the model to ''learn'' long term dependencies. This results in having a model that has memory of old information that it considers to be important and to eliminate short term noise that might not be important for the learning process, resulting in a reduction of the errors. This type of neural network seems to be performing better when modelling time series data compared to RNN and MLP.

The way this network functions is by having a chain structure that contains several different types of memory blocks, called cells. Each moment of the training, t, carries out the important information through the sequence, from the input received, C_{t-1}, to the output sent, C_t, as shown in *Figure 2.6*. This information flux can be re-fed and re-refreshed several times on each training iteration using three different types of gates (Forget Gate, Input Gate and Output Gate)[12].

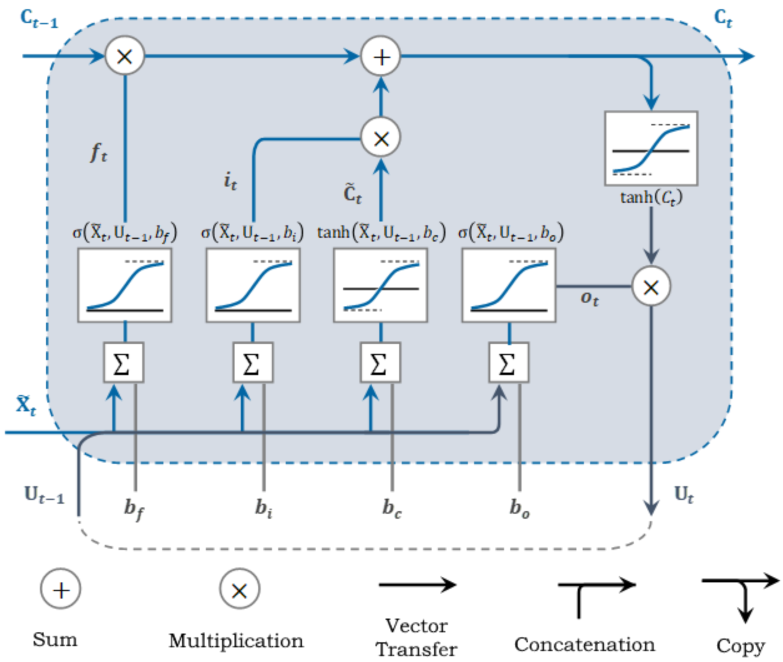


Figure 2.6 – LSTM Architecture

Source: Adapted from Ramos (2021)

2.3.4 – Model Comparison

Based on the three discussed models, the MLP has an element that can give Deep Learning a lot of potential since there might be an arbitrary number of hidden layers between the input and output layers. On the RNN, the information persists on the short term, meaning the model has memory capabilities. Therefore, while the MLP only produce a static model of the data, RNN produce a dynamic model with information storage. This might be useful has the previously stored data can be combined with the new data inputs.

On the LSTM models, with the help of information gates, we can have long term and short-term memory on the model that results a continuous learning process for the model. With this,

[12] For a better understanding of how such gates work, consult chapter 3.5 of Ramos (2021).

the model can filter irrelevant data from the pass that was once useful from past events and store them on different gates (Forget Gate and Input Gate). This selective mechanism of memory storage allows the model to learn in a recurring way as it passes to each epoch.

2.3.5 – Training and Evaluating Networks

As mentioned before, the training of a Neural Networks is done by executing given a learning algorithm that will adjust its synaptic weights by receiving a given dataset as input allowing to find patterns on the data and made forecasting.

After this training, we need to evaluate the performance of such a model. A common situation on this type of techniques is a problem called *overfitting* where the model has a high predictive performance on a particular set of data, and has ''learn'' them well, but doesn't have good generalisation and forecasting performance. According to Brownlee (2017) overfitting can have two origins: the selection of the hyperparameters of the network and/or the determination of the stopping criteria of the training.

In order to solve this issue, authors such as Hastie et al. (2009) use a method called cross-validation in order to evaluate the generalisation capacity of the network during its own training. This technique divides the data into subsets: training sample and testing sample. The training sample will be divided into two subsets: one to optimise the parameters of the model (training set) and the other to evaluate the quality of network generalisation by the end of each training epoch. There are also different cross-chain methodologies such as Forward Chaining, k-Fold and Group K-fold.

Is also important to mention the fact that Training and Evaluating a Neural Network is a fundamental step on the construction of the model whose mathematical foundation is extensive. Due to this fact, a detailed explanation of the implicit mathematical mechanisms of this models would not be feasible in this work. Nonetheless, it's recommended to consult Ramos (2021) and Haykin (2009).

3 – Methodology

On this Chapter, it is going to be explore what variables are going to be used on this study. This section will be followed the explanation of the computational implementations that allowed for the data treatment and for the implementation of our models.

3.1 – Data

The data used on this study was obtained from the *Yahoo Finance* Public API by calling the ticker ''BTC-USD'' and it refers to all the available data for the Bitcoin Daily Prices (from 07-09-2014 to 01-05-2022) expressed in U.S. Dollars. Using this data, we computed the volatility by squaring the daily returns of this variable. This data was then feed to the codes previously created in Ramos (2021) and Ramos and Lopes (2021).

With the goal of achieving the research purpose of this study, we take in consideration two time-series variables:

1. *BTC-USD:* Bitcoin's Daily Closing Prices
2. *BTC-USD-VOL*: Bitcoin's Daily Volatility

3.2 – Computational Implementation

Inspired by the methodologies developed by Ramos (2021), it was utilised a *Jupyter Notebook* environment that allows for the coding of *Python* programming language[13].

In order to make the code development faster and more efficient, it was used the following Python Open-Source Libraries:

1. **NumPy and Pandas**: In order to structure, process and analyse data and perform basic mathematical calculations;
2. **Matplotlib**: In order to implement visualisation tools;
3. **StatsModels**: In order to perform multiple statistical tests;
4. **Arch*:*** In order to implement Arch and Garch models;
5. **TensorFlow**: In order to implement neural networks;
6. **DM_Test***:* In order to perform Diebold-Mariano test.

In order to achieve this, four notebooks were implemented[14]:

1. ***ExploratoryDataAnalysis.ipynb***: where it was performed an exploratory data analyses of each Time Series evolving graphical representations, statistical metrics and other statistical tests;

[13] During this study, it was utilised the 3.7.3 version of Python.

[14] The notebooks *1* and *3* are available in an open-source repository at Lopes (2019) and Lopes (2020) whereas the notebooks *2* and *4* were implemented by the author using the previously mentioned open-source Python libraries.

2. ***Arch_Garch.ipynb***: for the implementation of the Arch and Garch models;
3. ***DeepNeuralNetwork.ipynb***: for the implementation of the Neural Networks;
4. ***DMTest.ipynb***: where the Diebold-Mariano Test was implemented.

3.2.1 – Exploratory Time Series Data Analyses

According to Ramos (2021), when we are trying to implement forecasting methodologies is useful to understand a few behaviours of the data such as trend or seasonality. The methodology proposed by Hodrick and Prescott (1997) as used in order to remove cycle and trend components using an *Hodrick-Prescott* filter on the raw data, using the *hpfilter* of the *statsmodels* python library.

In order to study structural breaks of the timeseries in question, it was used the CUSUM algorithm. The implementation was done using *detect_cusum* function on Python that was developed by Duarte and Wantanbe (2018).

For inferential analysis, it was applied several hypothesis tests in order to study normality (Jarque-Bera, Skewness and Kurtosis Tests) using *scipy* Python's Library. In order to study independence to the data, it was implemented the BDS Test using *statsmodels* Library.

3.2.2 – Model Implementation: ARCH, GARCH and Neural Networks

Regarding the computational implementation of the ARCH and GARCH models (*Arch_Garch.ipynb*), they were implemented using Python's *Arch* Library. ACF and PACF plots were calculated using *plot_acf* and *plot_pacf* functions from *statsmodels.* After this, it was made an iteration on possibilities for each model in order to find the parameters that would have the lower AIC and BIC. It was analysed ARCH(p) with $p \in \{1,2,3,4,5,6,7\}$ and GARCH(p,q) with $p,q \in \{1,2,3,4,5,6,7\}$.

When it comes to the Neural Networks, it was implemented the research procedures suggested by Ramos (2021).

With Python, the implementation of the MLP, RNN and LSTM models[15] (*DeepNeuralNetwork.ipynb)* was possible with the *tensorflow* library using *Dense, SimpleRNN* and *LSTM* functions. The methodology is described on *Figure 3.1.*

[15] ARMA models were also tested but didn't seem to present better forecasting quality.

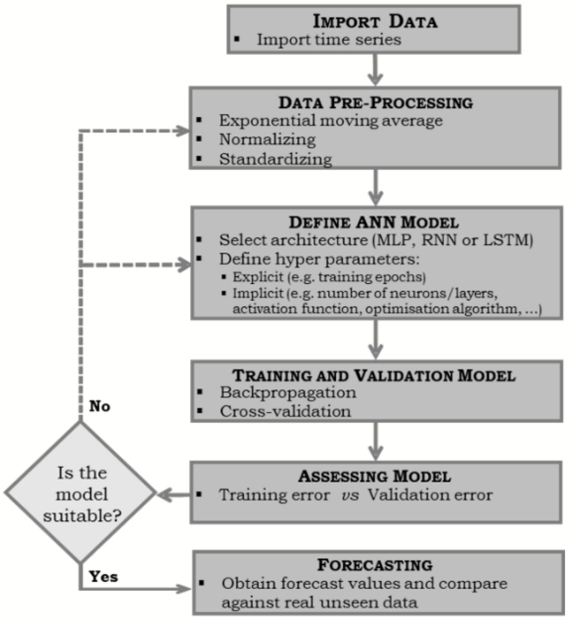


Figure 3.1 – Computational Implementation of Neural Networks.

Source: Adapted from Ramos (2021)

3.2.3 – Forecasting and Error Evaluation

In order to systematically evaluate the quality of our forecasting models, it's necessary to create a set of methodologies to evaluate the performance of the data that was extrapolated into the future using our first data sample and compared it with the real values. This is where the concept of Forecasting Error. As Ramos (2021) explores, there are two types of errors: (i) Random, due to the lack of knowledge of future variations whose factors are not included in the model; and (ii) systematic, committed consistently due, for example, to the selection of incorrect mathematical relationships between variables, or to differences between the true parameters and their estimates. According to Hamilton (1993) both of them contribute to the forecasting error and the best forecasting model will be the one that can minimize the systematic error.

Therefore, if we consider y_{t+h} the unknow value in the future and $\hat{y}_{t+h}$ the forecasting values obtained by our models based on the available information until time *t*, the forecasting error is given by:

$$e_{t+h} = y_{t+h} - \hat{y}_{t+h} \tag{3.1}$$

It is also important to establish a set of indicators that can evaluate the Forecasting Errors in numerical terms. Although there are several different error metrics and methodologies mentioned and used throughout the literature, we would like to highlight the ones that were used in this study.

If we consider *s,* as the number of forecasting values to perfume (forecasting window), we have:

- *Mean Error (ME)*

$$ME = \frac{\sum_{i=1}^{S} e^i}{s} \tag{3.2}$$

- *Mean Absolute Error (MAE)*

$$MAE = \frac{\sum_{i=1}^{S} \left| e^i \right|}{s} \tag{3.3}$$

- *Mean Absolute Percentage Error (MAPE)*

$$MAPE = \frac{\sum_{i=1}^{S} \left| \frac{y_{t+i} - \hat{y}_{t+i}}{y_{t+i}} \right|}{s} \times 100 \tag{3.4}$$

According to Ramos (2021), when you compare MAE and MAPE, it can be verified that there is a ‘’dimensional quantity’’ on MAE, since it is express in the unit of measure of the data, whereas MAPE is a ‘’dimensionless quantity’’ since we evaluate the error dimension in percentual terms. This is a characteristic that represents advantages in its analysis since it is not only a simple interpretation of the results as it allows to compare errors from data series with different units of measure.[16]

Lastly, was performed a Diebold-Mariano Test (*DMTest.ipynb*) with the best performing model of each category (*Arch/Garch* vs *Neural Networks*). This was implemented using *dm_test* library on Python with the default values.

[16] It is important to mention that MAPE cannot be used when we have null values for y_{t+i} or when we have a small data sample. (Ramos, 2021)

4 – Empirical Study

After such theoretical chapters, this chapter aims to explore and describe the methodological perspective of this research work. It will be presented a description of the financial time series in study, after a brief exploratory analysis of Bitcoin and Bitcoin's Volatility, followed by the modelling and forecasting of Bitcoin's Volatility with the respective analyses, comparison, and discussion of the results.

4.1 - Exploratory Analysis of the Time Series

4.1.1 - BTC-USD

The first time series in analysis is Bitcoin's Price. This series was obtained from *Yahoo Finance Public* API with the ticker ''BTC-USD'' and represents daily closing prices from 07-09-2014 to 01-05-2022, resulting in a total of 2754 observations, that are represented on *Figure 4.1* with complementary plots available on *Appendix A*.

Figure 4.1 - BTC-USD: Graphic Representation

The main statistics are shown on *Table 4.1.* Other complementary information available on the appendix shows that we the data series doesn't show seasonality, but it has a positive trend.

Table 4.1 – BTC-USD: Main Statistics

Count	Mean	Std	Min	Q1	Q2	Q3	Max	Kurtosis	Skew
2754	11771.1	164002.9	178.1	609.8	6385.7	10783.3	67566.8	1.786	1.741

On *Table 4.2*, we have the results of hypothesis tests regarding normality, stationarity, and independence.

Table 4.2 – BTC-USD: Normality, Stationarity and Independence Tests

	Normality Tests			Unit Root Tests		Independency Tests
	Kurtosis	**Skewness**	**Jarque- Bera**	**ADF**	**KPPS**	**BDS**
Statistic	10.7195	26.3739	1756.9660	-0.8417	5.9440	9.0845 - 18.8436
p-value*	0.0000*	0.0000*	0.0000*	0.8066	---	0.0000*

**We reject H_0 for the significance levels of 1%,5% and 10%.*

Based on those values, we reject normality for any significance level of all the tests we performed. In addition, we have statistically significant evidence that led just to the conclusion that the series is non-stationary, as expected, since we do not reject the null hypothesis for the *ADF Test* and by having a statistic value that is superior to all the critical values for the *KPSS Test*. Regarding *Independency* with *BDS Test*, by rejecting the null hypothesis for any significance level, we reject the null hypothesis and conclude there is some sort of linear dependency of the time series.

4.1.2 - BTC-USD-VOL

This is the second time series in analysis and where we will be focusing our modelling and forecasting efforts.

This series, plotted on *Figure 4.2*, was obtained by squaring the daily returns of BTC-USD time series resulting in 2753 observations from 08-09-2014 to 01-05-2022. Complementary Plots are available on *Appendix B.*

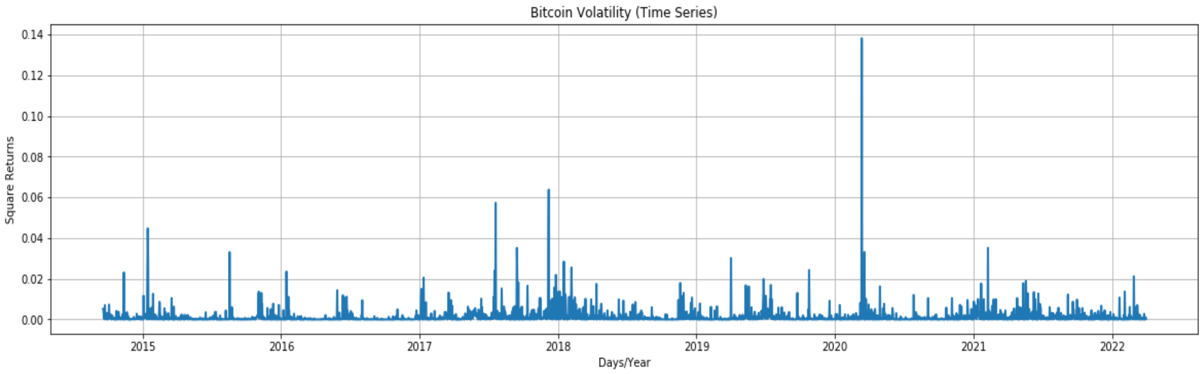


Figure 4.2 - BTC-USD-VOL: Graphic Representation

On the equation 4.1, is shown the formula in order to achieve such calculation:

$$Daily\ Squared\ Returns = \left(\frac{y_{t+1}}{y_t} - 1\right)^2 \tag{4.1}$$

As observed in *Figure 4.3* and complemented on *Appendix B*, the series shows some signs of heavy tailed distribution with several outlier values that occur when we have volatility spikes (high volatility) which is a characteristic of this particular asset.

However, this needs to be further confirmed with Normality tests, (that will be represented on *Table 4.5*).

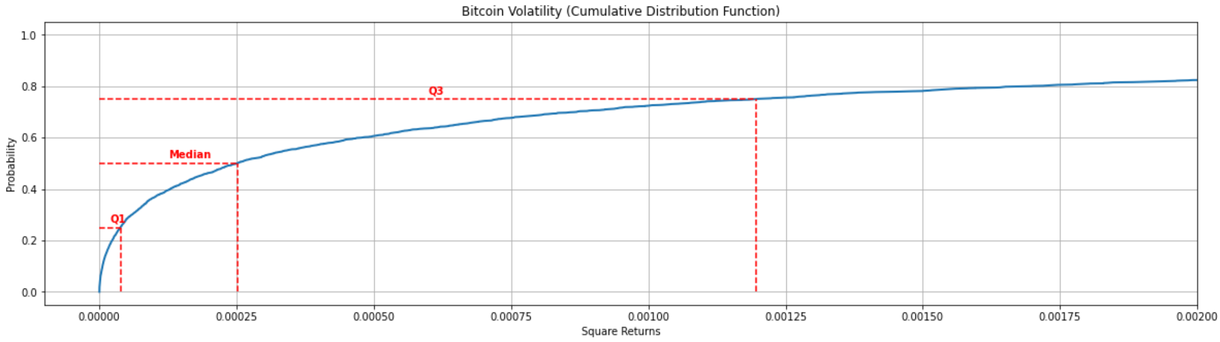


Figure 4.3 - BTC-USD-VOL: Cumulative Distribution Function with zoomed view

On *Table 4.3* and *Table 4.4*, we have a summary of BTC-USD-VOL statistics. Other complementary information on *Appendix B* where we can observe the lack of any trend and seasonality.[17]

Table 4.3 - BTC-USD-VOL: Main Statistics

Count	Mean	Std	Min
2753	0.001513	0.004576	0.000000

Table 4.4 - BTC-USD-VOL: Main Statistics

Q1	Q2	Q3	Max	Kurtosis	Skew
0.000038	0.000251	0.001196	0.138157	319.950	13.469

Based on those values, we observe a very high positive *Leptokurtic Kurtosis* and a very high *Skew* meaning that the timeseries is strongly positively skewed to the right with a right tail. Nonetheless, we need to further validate these ideas, using the *Normality Tests*.

With the *Normality Tests*, we reject all the null hypothesis for normality, and we can conclude that our timeseries does not have a normal distribution. In addition, both *ADF* and *KPPS* tests confirm that the series is stationary, as we expected.

When testing for *Independency*, we conclude that by not rejecting the null hypothesis for any significance level and therefore the data is *i.i.d.*. This can be observed on *Table 4.5.*

[17] Note that is not possible to perform multiplicative decomposition to the series since we have zero values on the timeseries.

Table 4.5 - BTC-USD-VOL: Normality, Stationarity and Independence Tests

	Normality Tests			Unit Root Tests		Independency Tests
	Kurtosis	**Skewness**	**Jarque-Bera**	**ADF**	**KPPS**	**BDS (Dim.2 – Dim. 6)**
Statistic	36.4657	61.5091	11825693.6472	-15.1772	0.2025	0.2417 - 0.4654
p-value*	*0.0000**	*0.0000**	*0.0000**	*0.0000**	---	0.8090 - 0.6416

**We reject H_0 for the significance levels of 1%,5% and 10%.*

As previously explored, on *Chapter 2*, this highlights the importance of implementing non-linear models for forecasting timeseries data.

Lastly, by applying a *CUSUM* algorithm, we identified several structural break situations, as shown on *Appendix B,* where we observe several regimes shifts throughout our series (highlighted with red circles). These shifts occur when the cumulative sum of the variations reach a value that is higher than our threshold (marked with a red line, on the lower chart on *Appendix B*) regardless of the variations being positive or negative. Is important to reenforce the fact that the series in study represents volatilities and therefore have a high number of structural breaks.

As pointed out on *Chapter 2*, the high number of structural breaks might represent forecasting difficulties for the classical econometric models and an advantage for the deep learning methodologies.

4.2 - Modelling and Forecasting

Following the models previously described in *Chapter 2*, this section will present the results from the implementation, modelling and forecasting using such models and techniques. This will be separated into two groups: The *Classical models* and the *Neural Networks models*.

It's important to mention that we considered all the values above 0.06 as outliers and were excluded from the dataset. This resulted in an exclusion of only 2 data points out of 2753 observations but resulted in an overall decrease in *MAPE* across all models, especially with the *neural network models* (with some models doubling their forecasting accuracy). On *GARCH* and *ARCH*, this was less noticeable.

4.2.1 - Autoregressive Conditional Heteroskedastic Models

Although several variations on such models have been discussed and presented, this research will be focusing on the use of two models: *ARCH* and *GARCH*. As the time series in question is stationary, the following step would be to perform the *Autocorrelation* and *Partial Autocorrelation* test functions to find the number of lags for our models.

As shown in *Appendix B*, the time series has a slight correlation in the first few lags. After iteration with different lags[18], it was confirmed that ARCH(4) and GARCH(4,2) have the best expected generalisation properties with both AIC and BIC showing the lower values for the given parameters.

After this model selection, we plotted the forecast estimations and its respective for seven days, as observed in *Figure 4.4* and *Figure 4.5.*

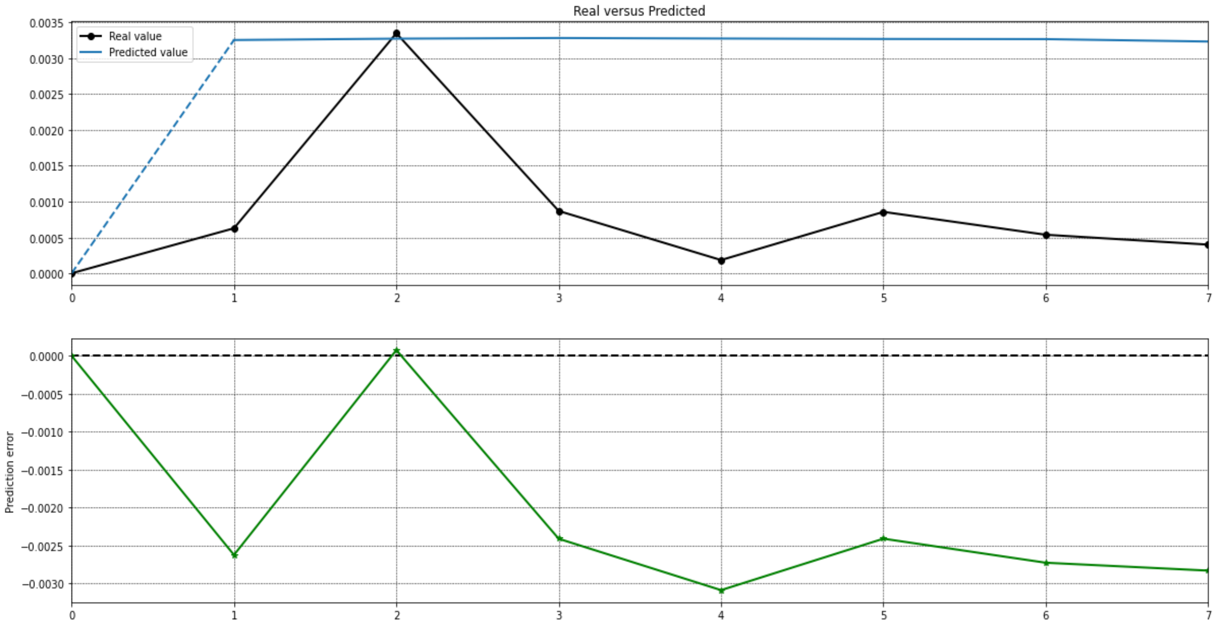


Figure 4.4 – ARCH(4) Forecasting Values and Prediction Error comparison

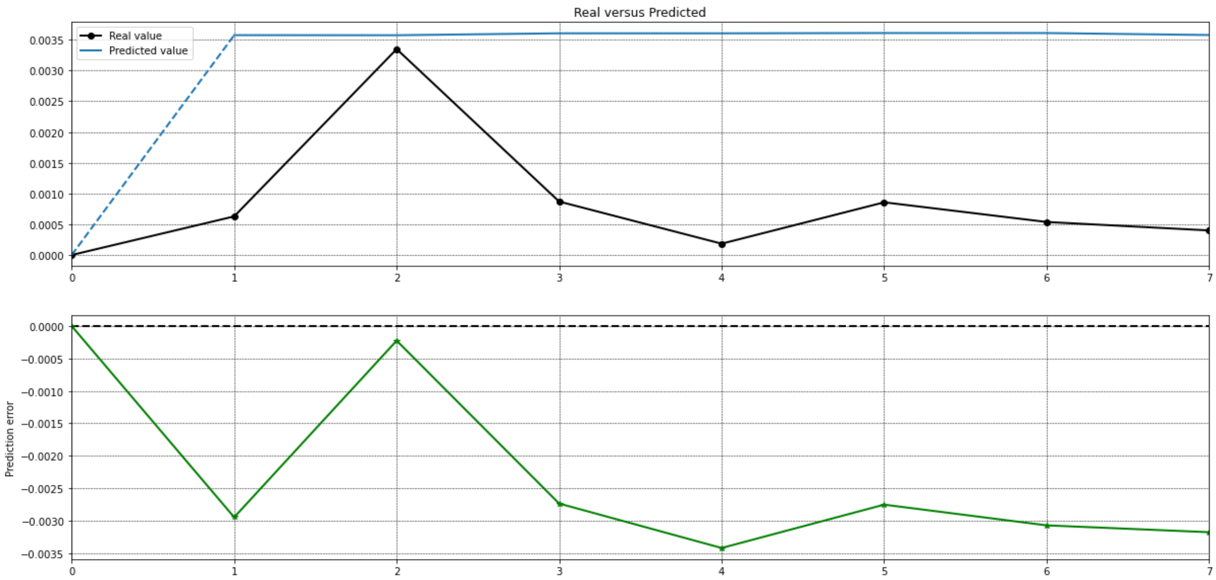


Figure 4.5 – GARCH(4,2) Forecasting Values and Prediction Error comparison

On the *Table 4.6*, we computed the values for MAE and MAPE for each error metric (MAE and MAPE) for each forecast horizons in study.

[18] As mentioned in *Chapter 3*, ARCH(p) with $p \in \{1,2,3,4,5,6,7\}$ and GARCH(p,q) with $p,q \in \{1,2,3,4,5,6,7\}$.

Table 4.6 – ARCH(4) and GARCH(4,2) Forecasting Errors comparison

	Days	MAE	MAPE
ARCH(4)	1	0.00216	343.672 %
	3	0.00155	194.452 %
	7	0.00198	463.812 %
GARCH(4,2)	1	0.00244	388.740 %
	3	0.00164	217.583 %
	7	0.00219	519.654 %

4.2.2 - Neural Network Model

The *neural network* study will be focusing on the modelling and forecasting of using three types of architecture: MLP, RNN, and LSTM.

There was a pre-processing procedure of the entire BTC-USD-VOL data set, where it performed an exponential smoothing of the time series as this results in better performance of the errors of the cross-validation tests.

Regarding the types of architectures and their respective hyperparameters optimization, it is important to mention that our study was limited by the amount of the computational time we had available, since the training of these models can go from a few seconds to several days, for the more complex models. In addition, the choice of the hyperparameters can be subjective. From our research, choosing between three to five hidden layers will optimise the learning of our models without compromising its generalisation properties through overfitting.

The other hyperparameters were adjusted based on the architecture type and suggestions made Brownlee (2017). With the hyper parameters set, we perform the training of our model. As suggested by Kingma & Ba (2014) we used ADAM optimization algorithm and ran 200 epochs for each model. Regarding the cross-validation, it was tested using Forward Chaining methodology, as suggested by Ramos (2021). In order to evaluate the networks, we will use MAE and MAPE calculated based on a one-, three- and seven-day prediction.

Based on these assumptions, after training, validation and evaluation of our models, we plotted our forecasting estimations for each model. On the *Appendix E,* we can observe: (1) the real volatility (plotted in black); (2) the data window used to train our models (orange area); (3) our in sample-predictions (plotted in orange) and (4) our out-of-sample forecast (plotted in blue).

On *Table 4.7*, it is presented the values for MAE and MAPE computed from the out-of-sample predictions where we have each error metric for forecast horizons in study.

Table 4.7 – Neural Networks Out-of-Sample Forecasting Errors Comparison

	Days	MAE	MAPE
MLP	1	0.00003	4.911 %
	3	0.00123	56.738 %
	7	0.00074	61.633 %
RNN	1	0.0002	31.852 %
	3	0.00128	61.909 %
	7	0.00067	49.017 %
LSTM	1	0.00043	69.427 %
	3	0.001517	96.774 %
	7	0.00089	149.801 %

On *Figures 4.6, 4.7* and *4.8,* it is plotted, in black, the real value and, in blue, the predicted value of each model in analysis followed by the prediction error.

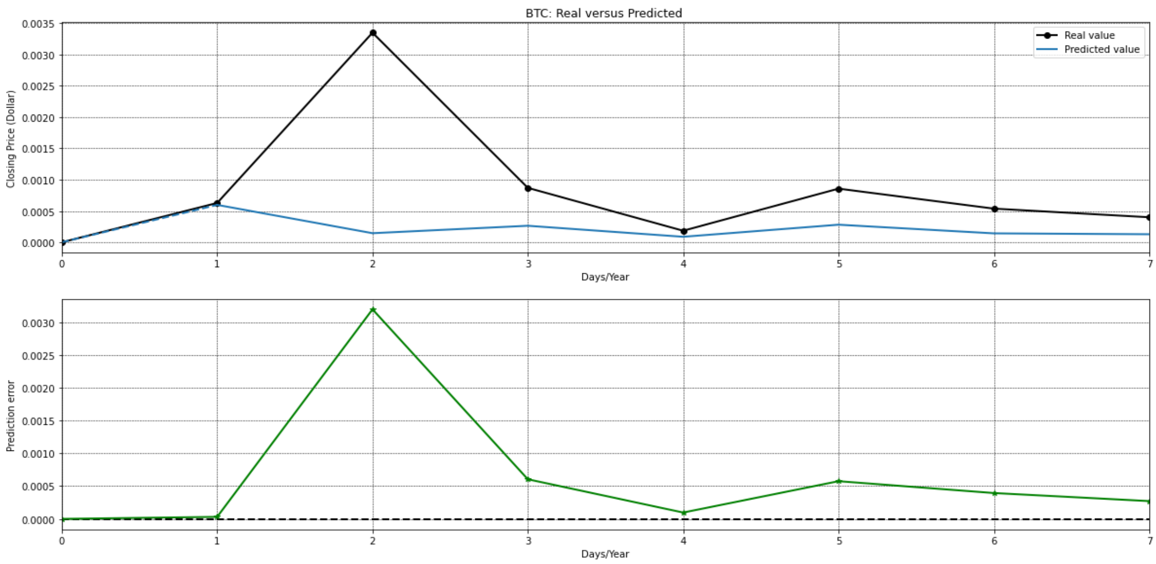


Figure 4.6 – MLP Forecasting Values and Prediction Error comparison

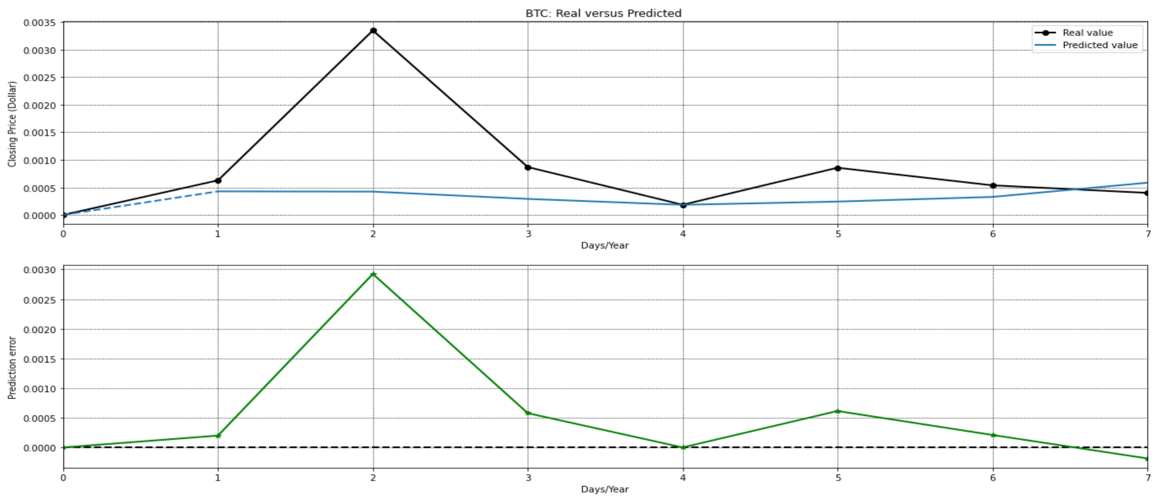


Figure 4.7 – RNN Forecasting Values and Prediction Error comparison

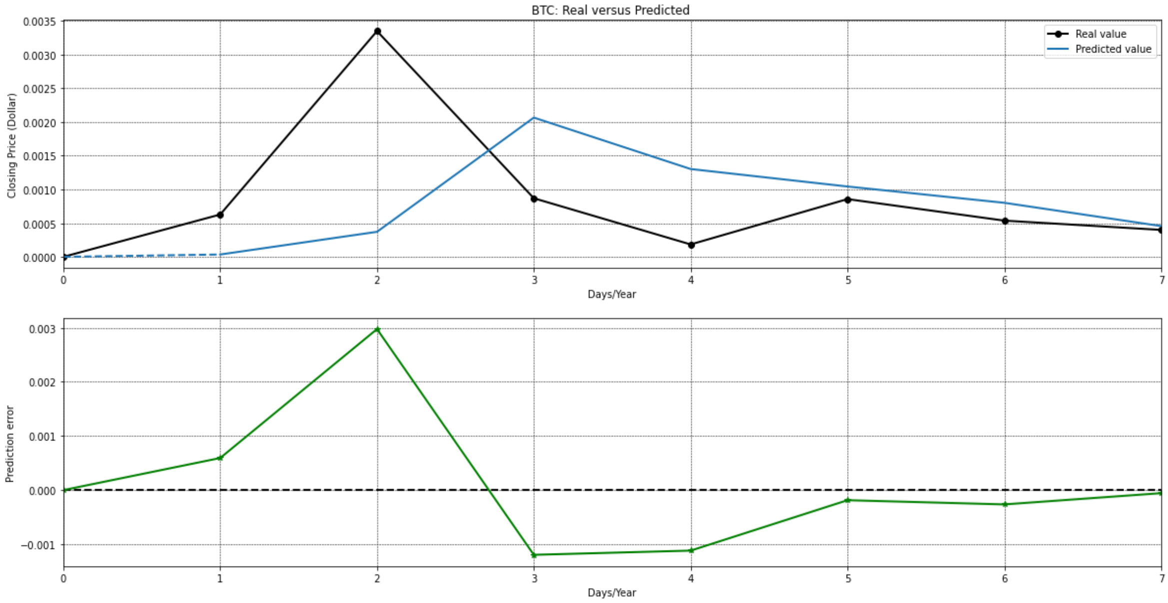


Figure 4.8 – LSTM Forecasting Values and Prediction Error comparison

4.2.3 – Model Comparison Analyses

With all the models computed and results shown, it is of great importance to finish with a comparison between all the five models (*Figure 4.9* that is complemented on *Appendix F*) that we studied throughout this research and conclude which one is the most accurate to forecast Bitcoin's Volatility – one of the research questions of this work.

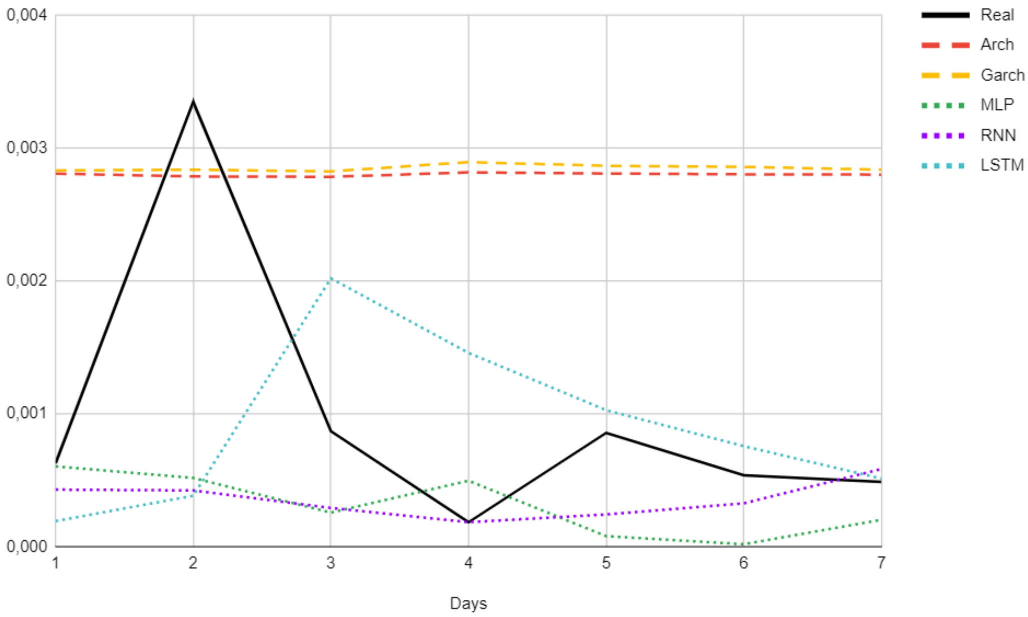


Figure 4.9 – Forecasting Comparison of all Models

From the classical approach, it was confirmed that ARCH(4) and GARCH(4,2) have the best expected generalisation properties with both AIC and BIC showing the lower values for the given parameters. Based on the *Forecasting Error Metrics* of *Table 4.6,* ARCH(4) has the best forecasting quality. Although the literature tend to prefer the use the GARCH(1,1) for volatility forecasting, in the case of BTC-USD-VOL, this are not the best GARCH parameters (in conformity with Senarathne (2019)).

Regarding the *Deep-Learning* models with the graphic representation of *Figure 4.6, Figure 4.7* and *Figure 4.8 (*with complements on *Appendix E)*, we can observe that all the models show some degree of forecasting properties, however MLP model had performed the better on the shorter time horizons (one-day and three-days) while RNN was the model with lower forecasting errors on the seven-day horizon *(*as shown on *Table 4.7)*. Although LSTM was the most complex model it had performed the worst. This was an expected phenomena since LSTM models tend to underperform when forecasting stationary time series as highlighted by Ramos *et al*. (2022). In addition to this, this type of neural network is the one that requires the most computational cost and time, so is highly inefficient to use it on our forecast. Nonetheless, both MLP and RNN seem to have a smoother prediction that fluctuate less but fails to capture the big volatility spikes, like the one that occur on day two, while seems to LSTM react strongly to such movements and tries to adjust their forecast accordingly. This happens due to its long-term memory proprieties that allows the model to ''remember'' that such volatility spikes in the past might result on high volatility spikes on the following days known as volatility clustering (as explored on *Chapter 2*).

In addition to this, we will also implement the Diebold-Mariano Test proposed by Diebold and Mariano (1995) with modification suggested by Harvey et al. (1997) to statistically identify forecast accuracy equivalent for two sets of forecasting results and therefore also evaluate with the difficulties to model a neutral network really represent significant advantages compared to the classical models.

Based on the metrics in study, we have chosen the models with lower MAE and MAPE and decided to compare the ARCH(4) model with the MLP neural network and perform the Diebold-Mariano Test for the seven-day horizon and presented the results on *Table 4.8.*

Table 4.8 – Diebold-Mariano Test

	Diebold-Mariano
Statistic	-3.2724894
p-value	0.03345

Based on Diebold-Mariano test, for the significance level of 5%, we reject H_0. With this, there is statistically significant evidence to assume that the forecasts don't have equal predictive accuracy and that one is significantly more accurate than the other.

5 - Conclusion

On the last years, Bitcoin has sparked the interest of researchers for a variety of reasons. The fact that this new asset class has several patterns and characteristics that fall out of the ordinary such as high volatility, high number of structural breaks and unusual probability distributions result in several interesting research subjects. Nonetheless, as mentioned by the literature, the amount of academic research that we have about the topic remains particularly low. From the current study, here is a synthesis of some contributions to topic, despite the fact that we also acknowledge its limitations that should be considered as future research ideas.

I. Contributions

Aligned with the initial motivation of this study, the research goals of this work where achieved, from where highlight the following: (i) systematic literature review on how Bitcoin operates and identification of several inner protocol mechanics that generate price volatility and that can be used has a base for future research; (ii) compilation on computational methodologies that can automate parts of the model construction process, making it more efficient; (iii) compared the forecasting results of classical models vs Deep-Learning methodologies and discussed the ones that have more forecasting quality.

These contributions were baselined on the research questions mentioned on the Introduction. On this section, we aim to synthesize the main facts and Ideas that were developed on this work.

(A) How Bitcoin functions and what are the out-of-ordinary volatility drivers?

This study was able to aggregate and organise several works on the history of Bitcoin, how it functions, it's main characteristics as a new asset class discussed on *Section 1.1*.

Secondly, a revision on what the literature discussed on what can be the main reasons behind its price and the extraordinary volatility of this asset. According to the literature, this includes the price of production (*electricity costs*), programmed scarcity, programmed supply shocks (*halving's*), Demand Shocks (*Price-Hash Rate Spirals*), hash rate, Network Trust and Liquidation Cascades. It also discussed some stylised facts about bitcoin such has the fact that the returns have low exposure to traditional asset classes, it has a positive trend, it is characterised by high volatility, that is has less trading volumes on weekends, and doesn't have inflation hedge proprieties. However, is important to mention that some authors also suggest that the price and volatility of bitcoin happens only because of the speculative nature of its investors or that the cost models cannot explain price since a non-dividend yield asset that has the probability of going to zero (e.g., due to technical obsolescence), should have a present value of zero.

(B) From the models in study, which one is the best forecasting Model for Bitcoin's Volatility?

In order to discuss this research question, we compared the five different types of models where the approach was to separated them in two sections: Classical vs Deep Learning Methodologies.

According to the literature, one of the most common models to study and forecast volatility of financial timeseries is GARCH(1,1), however, as mentioned by Senarathne (2019), this is not the case for bitcoin. The results of this work are in line with this fact, has for the forecasting time horizons and data in study, the model that best fit the data, when we took in consideration ARCH and GARCH models, were ARCH(4) and GARCH(4,2), with ARCH(4) being the one with best forecasting quality when you consider MAPE.

From the Deep Learning approach, MLP model had performed the better on the shorter time horizons (one-day and three-days) while RNN was the model with lower forecasting errors on the seven-day horizon. As mentioned on *Chapter 2,* this might be explained by the fact that the RNN model has basic memory capabilities, producing a dynamic model with information storage while the MLP only produce a static model of the data.

Although LSTM was the most complex model it had performed the worst of the Deep-Learning methodologies. These results corroborate with the literature since this type of architecture seems to underperform when the series is stationary. Despite of that, the neural network long-term memory properties seem have ''learned'' the concept of volatility clustering'', a well-known and studied characteristic of volatility, where model seems to ''remember'' that a volatility spikes in the past might result on high volatility spikes in the future.

Nonetheless, the MLP neural network model that has built was the model that had more forecasting quality and lower forecasting error from the five models explored on this work while having the lowest computational costs of the deep-learning Methodologies.

(C) Are Deep-Learning Methodologies an improvement of classical methodologies?

The Deep-Learning Methodologies seem to show advantages to classical methodologies when it comes to forecasting quality mainly due to the fact that they are a more complex methodology that can capture non-linear dependencies of the data in a more robust way. Nonetheless, is important to highlight that Deep-Learning models have much higher computational costs and are much harder to implement than the classical techniques.

However, the implementation of such models resulted in a very significant improvement of the forecasting errors allowing us to reach the conclusion that the increase on the computational costs justify the implementation of such models especially if it's taken in consideration that MLP is not the most complex model to implement or has the biggest computational costs when we compare it to RNN or LSTM.

Lastly, when it was compared the forecasting accuracy of ARCH(4) an MLP models with a Diebold-Mariano test for the seven-day time horizon, it was concluded that the accuracy of the models is not the same, and one is better for the significance, where it can be inferred that MLP is the one that performs better.

These facts are in line with the literature and previous research work. It also highlights the importance of these new methodologies and how researchers should be equipped with the knowledge on how these models work in order to help them better explain the world by better understanding the economic reality.

II. Limitations and Future Research Perspectives

Although it is believed that this work contains valuable contributions to the scientific debate of the topic in question, it is also important to highlight the limitations associated with our study and its conclusions with the goal of opening room for future research.

The first limitation of the volatility drivers resides in the fact that this study tried to focus on inner protocol mechanisms that can result in volatility drivers. Although we briefly touched on the subject, there are more room to explore market dynamics that only exist on the cryptocurrency markets such as the low liquidity and its effects on slippage, the market microstructure, high availability of leverage or even market manipulation.

Secondly, is important to mention that we used a limited range of ARCH/GARCH models. With this, we want to highlight the fact there are more advanced models on this category that might be able to compete with Deep-learning Methodologies and even have similar forecasting capabilities without the necessary of the difficulties in the implementation or the computational power.

Thirdly, our models only have one variable, and we didn't introduce external factors. This type of univariate analysis is the simplest form of analysing data and when we are dealing with complex financial timeseries with non-linear proprieties, there is room for improvement with the implementation of more variables in the model.

With this said and with the perspective of continue improvement of this work and human knowledge, we leave the suggestion of trying to approach volatility with a multi-variable perspective. This might result in more accurate forecasting models although it could also out-turn in more complex model that requires more computation time and power to ''learn'.

Nonetheless, we would suggest researching models with three types of variables: (1) Derivatives Data - such as futures funding rates, volumes, open-interest or options realized, unrealized and implied volatility, premiums and volumes; (2) On-chain Data - such as big transactions, networks hash-rate and difficulty or number of network transactions; (3) Market Sentiment Data - using Fear and Greed Indexes or number of mentions of the ticker on social media. Nonetheless, it's important to mention that these variables only serve as suggestions and need to be further analysed as Bitcoin's volatility may help explain them and not the other way around.

In addition to this, there is also suggestion of approaching the volatility of Bitcoin with the usage of Hybrid models. These new models can not only combine the potential of several methodologies has they can improve the current ones by reducing the computational time while keeping similar forecasting quality. (e.g., Ramos *et al.* (2022))

To sum up, it is hoped that the work developed on this study could be a valuable contribution to better understand Bitcoin's volatility and the potential of deep-learning methodologies. Most researchers seem to be using a more classical approach to volatility forecasting with Autoregressive Conditional Heteroskedastic models but the recent advancements of the computational power and easy access pre-build open-source python libraries, show deep-learning methodologies as a promising option to improve the forecasting quality and research should embrace it in order to better these new types of excited asset classes that are becoming part of our economic reality.

6 – References

Akaike, H. (1974). A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, *19*(6), 716–723. https://doi.org/10.1109/TAC.1974.1100705

Antonopoulos, A. (2015). *Mastering Bitcoin* (1st ed.). O'Reilly Media Inc.

Back, A. (1997). *Hash Cash Postage Implementation*. http://www.hashcash.org/papers/announce.txt

Balcilar, M., Bouri, E., Gupta, R., & Roubaud, D. (2017). Can volume predict Bitcoin returns and volatility? A quantiles-based approach. *Economic Modelling*, *64*, 74–81. https://doi.org/10.1016/J.ECONMOD.2017.03.019

Bariviera, A. F. (2017). *The inefficiency of Bitcoin revisited: a dynamic approach*.

Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems. *IEEE Transactions on Systems, Man and Cybernetics*, *SMC-13*(5), 834–846. https://doi.org/10.1109/TSMC.1983.6313077

Becker, S., & Plumbley, M. (1996). Unsupervised neural network learning procedures for feature extraction and classification. *Applied Intelligence*, *6*(3), 185–203. https://doi.org/10.1007/BF00126625

Bergsli, L. Ø., Lind, A. F., Molnár, P., & Polasik, M. (2022). Forecasting volatility of Bitcoin. *Research in International Business and Finance*, *59*, 101540. https://doi.org/10.1016/J.RIBAF.2021.101540

Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Oxford.

Black F., S. M. (1973). The Pricing of Options and Corporate Liabilities on JSTOR. *Journal of Political Economy*, *81*(3), 637–654. https://www.jstor.org/stable/1831029

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307–327. https://doi.org/10.1016/0304-4076(86)90063-1

Brooks, C. (2014). *Introductory Econometrics for Finance* (C. Brooks, Ed.; 3rd Edition). Cambridge University Press. https://www.akademika.no/introductory-econometrics-finance/brooks-chris/9781107661455

Brown, R. L., Durbin, J., & Evans, J. M. (1975). Techniques for Testing the Constancy of Regression Relationships Over Time. *Journal of the Royal Statistical Society: Series B (Methodological)*, *37*(2), 149–163. https://doi.org/10.1111/J.2517-6161.1975.TB01532.X

Brownlee, J. (2017). *Introduction to Time Series Forecasting With Python*. Machine Learning Master. https://machinelearningmastery.com/introduction-to-time-series-forecasting-with-python/

Campbell, J. Y., Lo, A. W. (Andrew W.-C., & MacKinlay, A. C. (1997). *The econometrics of financial markets*. 611.

Chaim, P., & Laurini, M. P. (2018). Volatility and return jumps in bitcoin. *Economics Letters*, *173*, 158–163. https://doi.org/10.1016/J.ECONLET.2018.10.011

Chatfield, C. (2016). The Analysis of Time Series: an introduction (6th ed.). Chapman and Hall/CRC. (n.d.).

Dai, W. (1998). *b-money*. https://nakamotoinstitute.org/literature/b-money/

de Nicola, G. (2021). On the Intraday Behavior of Bitcoin. *Ledger*, *6*.

Diebold, F., Mariano, R., Diebold, F., & Mariano, R. (1995). Comparing Predictive Accuracy. *Journal of Business & Economic Statistics*, *13*(3), 253–263. https://EconPapers.repec.org/RePEc:bes:jnlbes:v:13:y:1995:i:3:p:253-63

Duarte, M., & Wantanbe, R. N. (2018). *Notes on Scientific Computing for Biomechanics and Motor Control. GitHub. Retrieved from https://github.com/BMClab/BMC*.

Engle, R. F. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*, *50*(4), 987. https://doi.org/10.2307/1912773

Engle, R. F., & Patton, A. J. (2001). *What good is a volatility model?*

Figlewski, S. (1997). Forecasting Volatility. *Financial Markets, Institutions & Instruments*, *6*(1), 1–88. https://doi.org/10.1111/1468-0416.00009

G, S. (1978). “Estimating the Dimension of a Model.” *The Annals of Statistics*, *6*(2), 461–464. https://doi.org/10.2307/2958889

Garcia, D., Tessone, C. J., Mavrodiev, P., & Perony, N. (2014). The digital traces of bubbles: feedback cycles between socio-economic signals in the Bitcoin economy. *Journal of The Royal Society Interface*, *11*(99). https://doi.org/10.1098/RSIF.2014.0623

Gronwald, M. (2019). Is Bitcoin a Commodity? On price jumps, demand shocks, and certainty of supply. *Journal of International Money and Finance*, *97*, 86–92. https://doi.org/10.1016/J.JIMONFIN.2019.06.006

Hamilton, J. D. (1993). Time Series Analysis . In *Time series analysis*. Princeton University Press.

Hang, N. T. (2019). Research on a number of applicable forecasting techniques in economic analysis, supporting enterprises to decide management. *World Scientific News*, *119*, 52–67.

Hanley, B. P. (2013). *The False Premises and Promises of Bitcoin*. https://arxiv.org/abs/1312.2048v8

Hansen, B. E. (2001). The New Econometrics of Structural Change: Dating Breaks in U.S. Labour Productivity. *Journal of Economic Perspectives*, *15*(4), 117–128. https://doi.org/10.1257/JEP.15.4.117

Harvey, D., Leybourne, S., & Newbold, P. (1997). Testing the equality of prediction mean squared errors. *International Journal of Forecasting*, *13*(2), 281–291. https://doi.org/10.1016/S0169-2070(96)00719-4

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning - Data Mining, Inference, and Prediction*. *Second Edition*. https://doi.org/10.1007/978-0-387-84858-7

Hayes, A. S. (2017). Cryptocurrency value formation: An empirical study leading to a cost of production model for valuing bitcoin. *Telematics and Informatics*, *34*(7), 1308–1321. https://doi.org/10.1016/J.TELE.2016.05.005

Haykin, S. (2009). *Neural Networks and Learning Machines,*. 3rd Ed | Pearson. https://www.pearson.com/us/higher-education/program/Haykin-Neural-Networks-and-Learning-Machines-3rd-Edition/PGM320370.html

Hodrick, R., & Prescott, E. (1997). Postwar U.S. Business Cycles: An Empirical Investigation. *Journal of Money, Credit and Banking*, *29*(1), 1–16. https://EconPapers.repec.org/RePEc:mcb:jmoncb:v:29:y:1997:i:1:p:1-16

Kim, H. Y., & Won, C. H. (2018). Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Systems with Applications*, *103*, 25–37. https://doi.org/10.1016/J.ESWA.2018.03.002

Kingma, D. P., & Ba, J. L. (2014). Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. https://doi.org/10.48550/arxiv.1412.6980

Kitagawa, G. (Genshiro). (2022). *Introduction to time series modeling with applications in R* (2nd ed.). Chapman and Hall/CRC.

Kristjanpoller, W., & Minutolo, M. C. (2018). A hybrid volatility forecasting framework integrating GARCH, artificial neural network, technical analysis and principal components analysis. *Expert Systems with Applications*, *109*, 1–11. https://doi.org/10.1016/J.ESWA.2018.05.011

Kwiatkowski, D., Phillips, P. C. B., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, *54*(1–3), 159–178. https://doi.org/10.1016/0304-4076(92)90104-Y

Lahmiri, S., & Bekiros, S. (2019). Cryptocurrency forecasting with deep learning chaotic neural networks. *Chaos, Solitons and Fractals*, *118*, 35–40. https://doi.org/10.1016/J.CHAOS.2018.11.014

Lahmiri, S., & Bekiros, S. (2020). Intelligent forecasting with machine learning trading systems in chaotic intraday Bitcoin market. *Chaos, Solitons and Fractals*, *133*. https://doi.org/10.1016/J.CHAOS.2020.109641

Libanio, G. (2005). Unit roots in macroeconomic time series: theory, implications, and evidence. *Nova Economia, 15(3), Pp 145-176*, 145–176. https://econpapers.repec.org/article/novartigo/v_3a15_3ay_3a2005_3ai_3a3_3ap_3a145-176.htm

Liu, Y., & Tsyvinski, A. (2021). Risks and Returns of Cryptocurrency. *The Review of Financial Studies*, *34*(6), 2689–2727. https://doi.org/10.1093/RFS/HHAA113

Lopes, D. & Ramos, F. (2019). Time Series Cross-Validation. Retrieved from https://github.com/DidierRLopes/TimeSeriesCrossValidation

Lopes, D. & Ramos, F. (2020). Univariate Time Series Forecast. Retrieved from https://github.com/DidierRLopes/UnivariateTimeSeriesForecast

Mallqui, D. C. A., & Fernandes, R. A. S. (2019). Predicting the direction, maximum, minimum and closing prices of daily Bitcoin exchange rate using machine learning techniques. *Applied Soft Computing Journal*, *75*, 596–606. https://doi.org/10.1016/J.ASOC.2018.11.038

Markowitz, H. (1952). PORTFOLIO SELECTION*. *The Journal of Finance*, *7*(1), 77–91. https://doi.org/10.1111/J.1540-6261.1952.TB01525.X

Merkle, R. C. (1987). A Digital Signature Based on a Conventional Encryption Function. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *293 LNCS*, 369–378. https://doi.org/10.1007/3-540-48184-2_32

Minsky, M., & Papert, S. (1969). *Perceptrons; an introduction to computational geometry*. 258.

Mueller, P. (2020). Cryptocurrency Mining: Asymmetric Response to Price Movement. *SSRN Electronic Journal*. https://doi.org/10.2139/SSRN.3733026

Murteira, B. J. Ferreira., Müller, D. A., & Turkman, K. Feridun. (2000). *Análise de sucessões cronológicas*.

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

Nelson, D. B. (1991). Conditional Heteroskedasticity in Asset Returns: A New Approach. *Econometrica*, *59*(2), 347. https://doi.org/10.2307/2938260

Nicholas Taleb, N. (2021). Bitcoin, Currencies, and Fragility. *Quantitative Finance*, *21*(8), 1249–1255. https://doi.org/10.1080/14697688.2021.1952702

Pagnotta, E., & Buraschi, A. (2018). An Equilibrium Valuation of Bitcoin and Decentralized Network Assets. *SSRN Electronic Journal*. https://doi.org/10.2139/SSRN.3142022

Pesaran, M. H., & Timmermann, A. (2004). How costly is it to ignore breaks when forecasting the direction of a time series? International Journal of Forecasting, 20(3), 411–425. (n.d.).

Phillips, P. C. B., & Perron, P. (1988). Testing for a Unit Root in Time Series Regression Testing for a unit root in time series regression. *Source: Biometrika Trust*, *75*(2), 335–381.

Pichl, L., & Kaizoji, T. (2017). Volatility Analysis of Bitcoin Price Time Series. *Quantitative Finance and Economics 2017 4:474*, *1*(4), 474–485. https://doi.org/10.3934/QFE.2017.4.474

Ramos, F. R. (2021). *Data Science na Modelação e Previsão de Séries Económico-financeiras: das Metodologias Clássicas ao Deep Learning*. Phd Thesis, Instituto Universitário de Lisboa - ISCTE Business School. https://repositorio.iscte-iul.pt/handle/10071/22964

Ramos, F. R., Costa, A., Mendes, D. & Lopes, D. R. (2021). *Explorando o poder da memória das redes neuronais LSTM na modelação e previsão do PSI 20*. SPE 2021 – XXV Congresso da Sociedade Portuguesa de Estatística. Évora.

Ramos, F. R., Lopes, D. R. & Pratas, T. E. (2022). *Deep Neural Networks: A Hybrid Approach Using Box&Jenkins Methodology*. In: Innovations in Mechatronics Engineering II. icieng 2022. Lecture Notes in Mechanical Engineering. Springer, Cham.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature 1986 323:6088*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Santos Ferreira, F., & Manuel de Sousa Leite Inácio, P. (2018). *Bitcoin: an asset or a currency?* https://repositorio.iscte-iul.pt/handle/10071/18373

Senarathne, C. W. (2019). The Leverage Effect and Information Flow Interpretation for Speculative Bitcoin Prices: Bitcoin Volume vs ARCH Effect. *European Journal of Economic Studies*, *8*(1). https://doi.org/10.13187/es.2019.1.77

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning : A systematic literature review: 2005–2019. *Applied Soft Computing*, *90*, 106181. https://doi.org/10.1016/J.ASOC.2020.106181

Tealab, A. (2018). Time series forecasting using artificial neural networks methodologies: A systematic review. *Future Computing and Informatics Journal*, *3*(2), 334–340. https://doi.org/10.1016/J.FCIJ.2018.10.003

Valentinyi-Endrész, M. (2004). Structural breaks and financial risk management. In *Magyar Nemzeti (Bank Working Paper)*. https://ideas.repec.org/p/mnb/wpaper/2004-11.html

Wilson, J. H., & Samuel A. Spralls, I. (2018). WHAT DO BUSINESS PROFESSIONALS SAY ABOUT FORECASTING IN THE MARKETING CURRICULUM? *International Journal of Business, Marketing, and Decision Sciences*, *11*(1), 1–21. https://go.gale.com/ps/i.do?p=AONE&sw=w&issn=19428162&v=2.1&it=r&id=GALE%7CA573558930&sid=googleScholar&linkaccess=fulltext

Yermack, D. (2013). *Is Bitcoin a Real Currency? An economic appraisal*. National Bureau of Economic Research, Inc. https://EconPapers.repec.org/RePEc:nbr:nberwo:19747

Yu, M. (2019). Forecasting Bitcoin volatility: The role of leverage effect and uncertainty. *Physica A: Statistical Mechanics and Its Applications*, *533*, 120707. https://doi.org/10.1016/J.PHYSA.2019.03.072

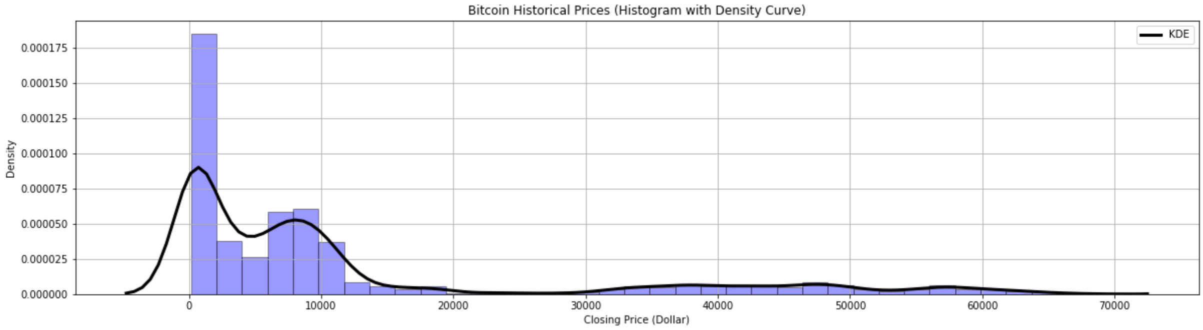
Zimmermann, P. (1991). *PGP User's Guide*.

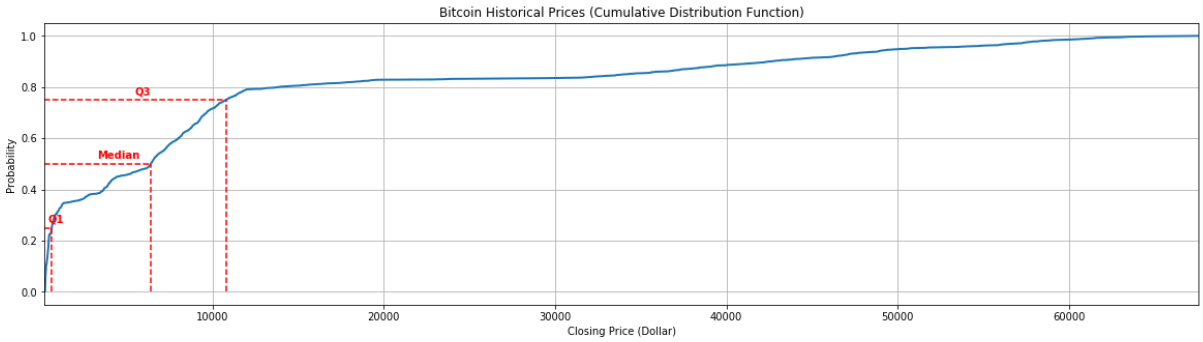
Zivot, E. (2008). Practical Issues in the Analysis of Univariate GARCH Models. In *EconPapers*. https://econpapers.repec.org/paper/udbwpaper/uwec-2008-03-fc.htm

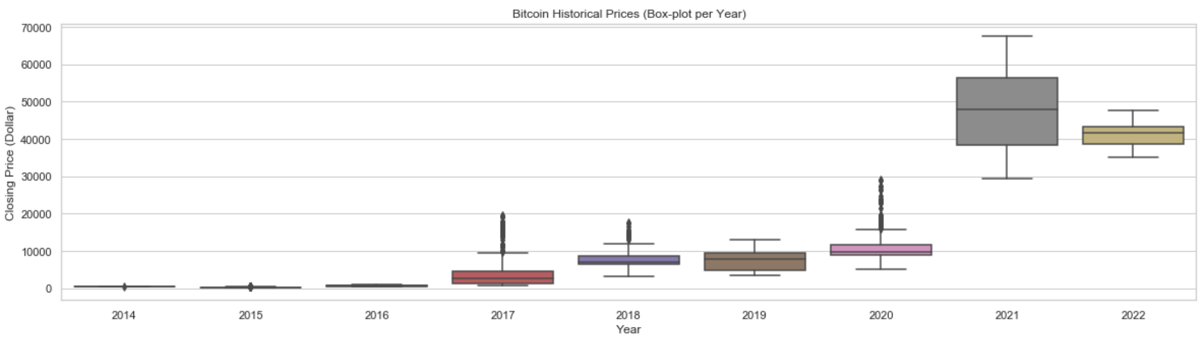
7 – Appendix

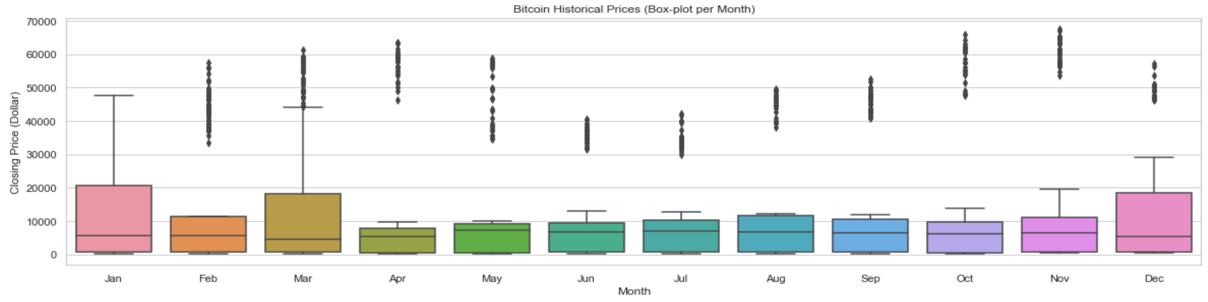
Appendix A

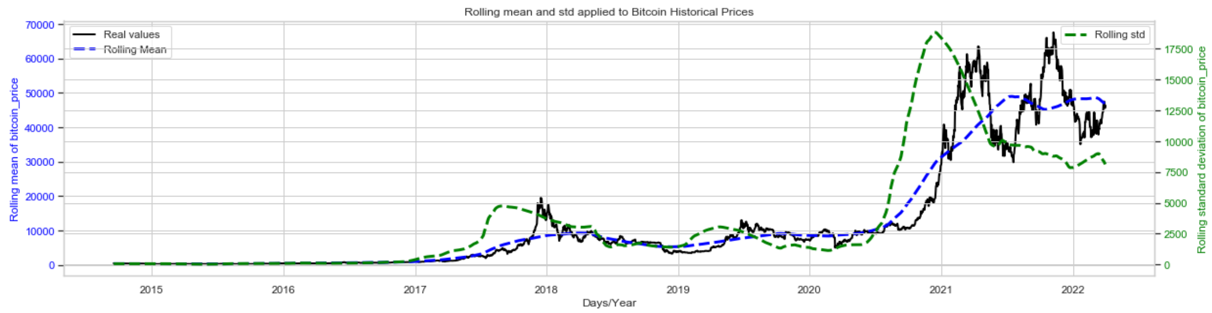
Time Series Analysis Plots for Bitcoin

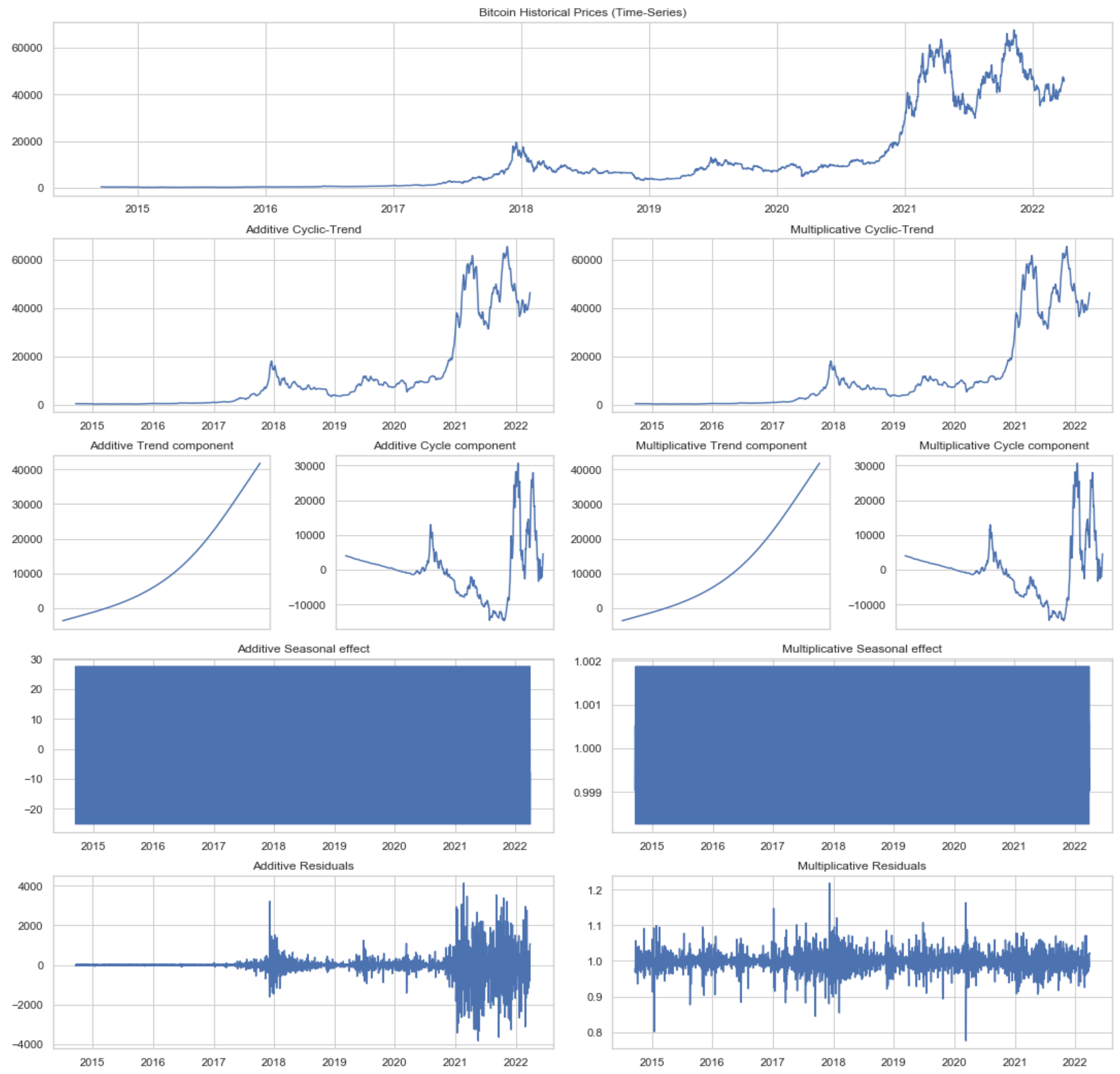




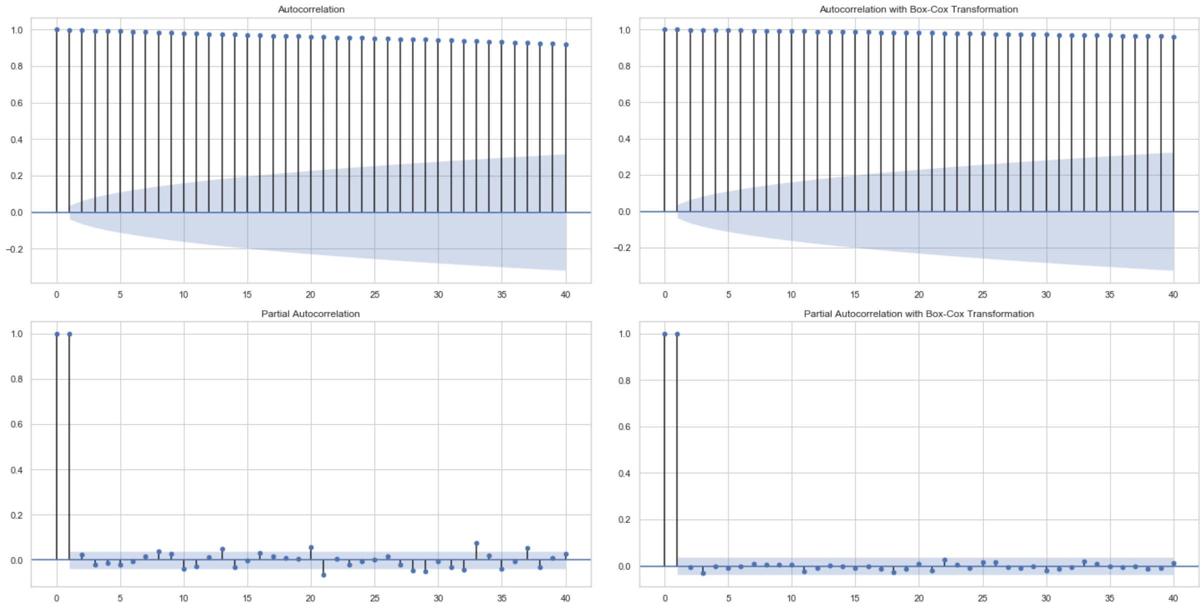


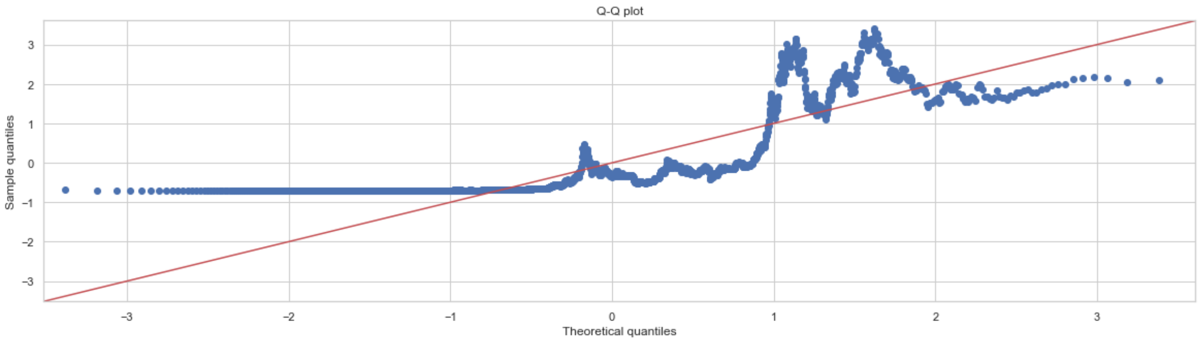

Bitcoin Historical Prices (Box-plot per Month)
Closing Price (Dollar)
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month


Rolling mean and std applied to Bitcoin Historical Prices
Real values
Rolling Mean
Rolling std
Rolling mean of bitcoin_price
Rolling standard deviation of bitcoin_price
Days/Year


Bitcoin Historical Prices (Time-Series)
Additive Cyclic-Trend
Multiplicative Cyclic-Trend
Additive Trend component
Additive Cycle component
Multiplicative Trend component
Multiplicative Cycle component
Additive Seasonal effect
Multiplicative Seasonal effect
Additive Residuals
Multiplicative Residuals

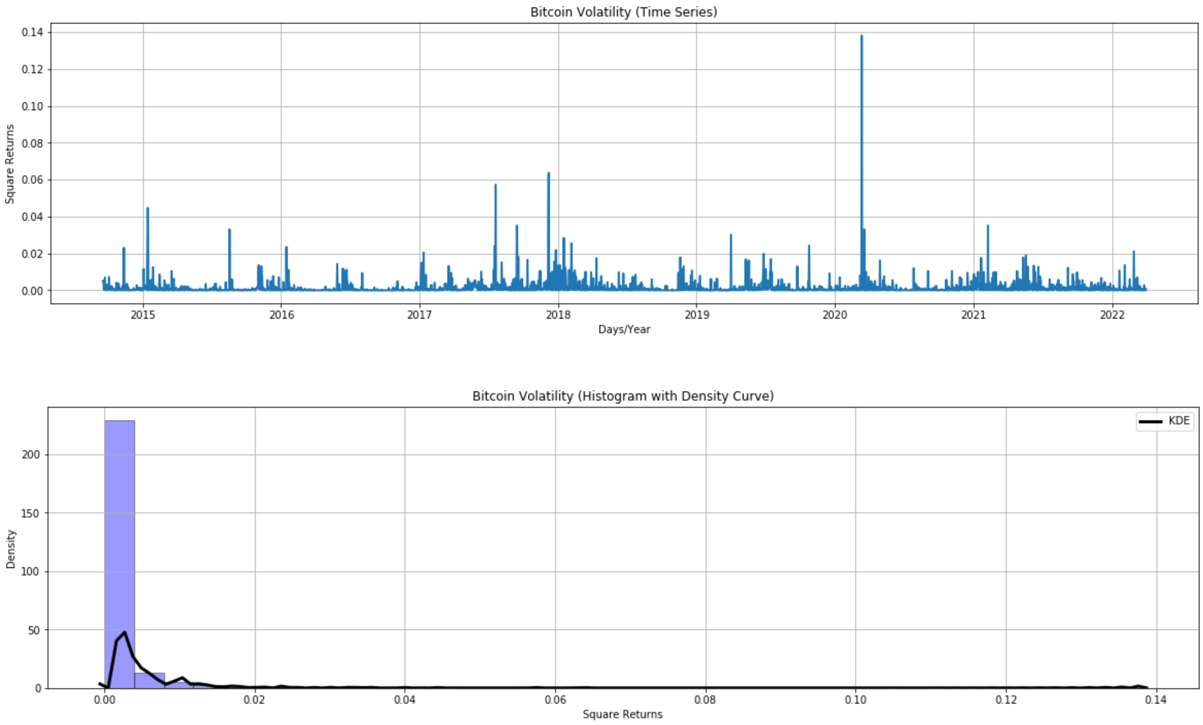

Bitcoin Historical Prices (Transformations study)
Box Cox Transformation (λ = 0.02)
Log Transformation = Box Cox Transformation (λ = 0)
Arcsine Transformation
Transformed Closing Price (Dollar)
Days/Year
2015
2016
2017
2018
2019
2020
2021
2022

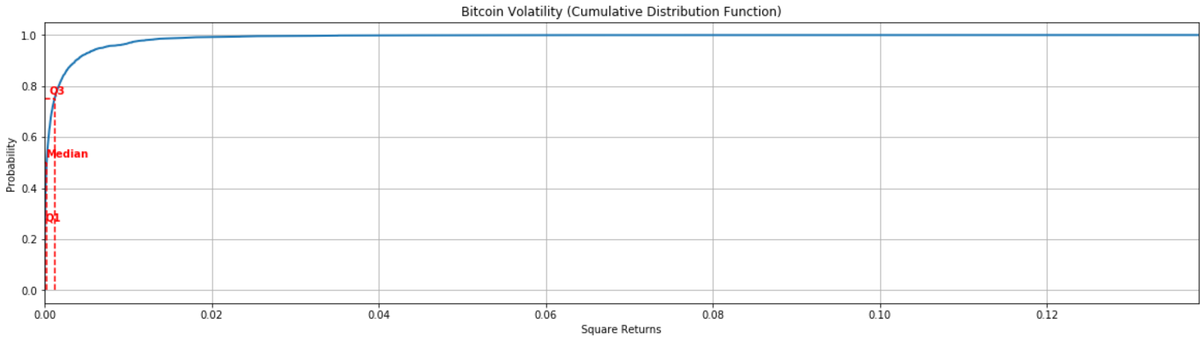

Autocorrelation
Autocorrelation with Box-Cox Transformation
Partial Autocorrelation
Partial Autocorrelation with Box-Cox Transformation

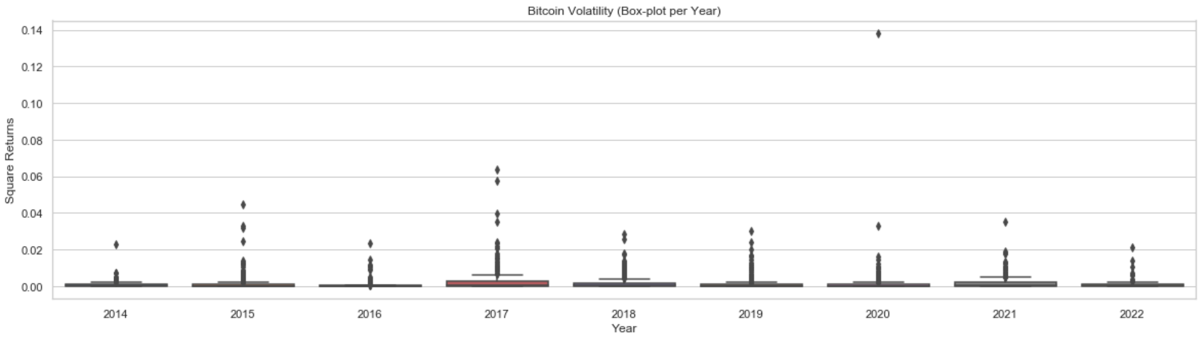

Q-Q plot
Sample quantiles
Theoretical quantiles

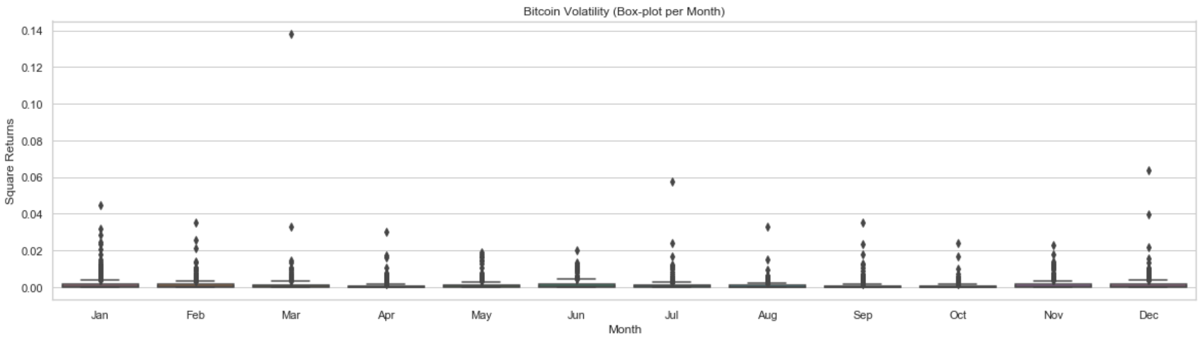
Appendix B

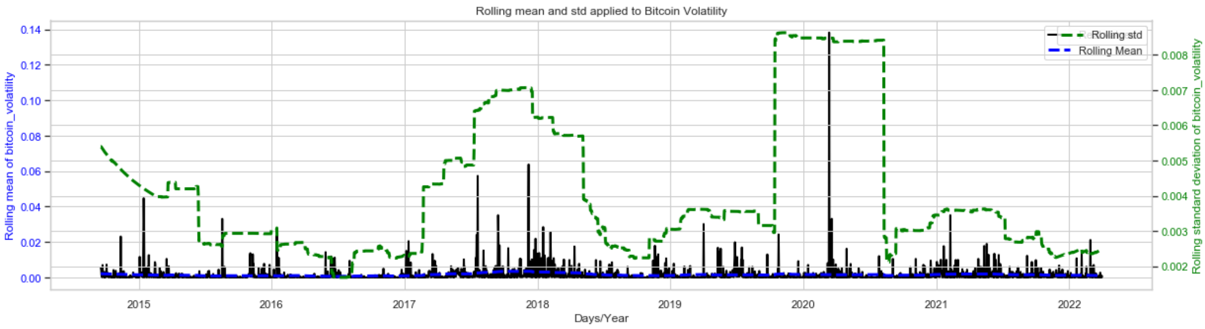
Time Series Analysis Plots for Bitcoin Volatility

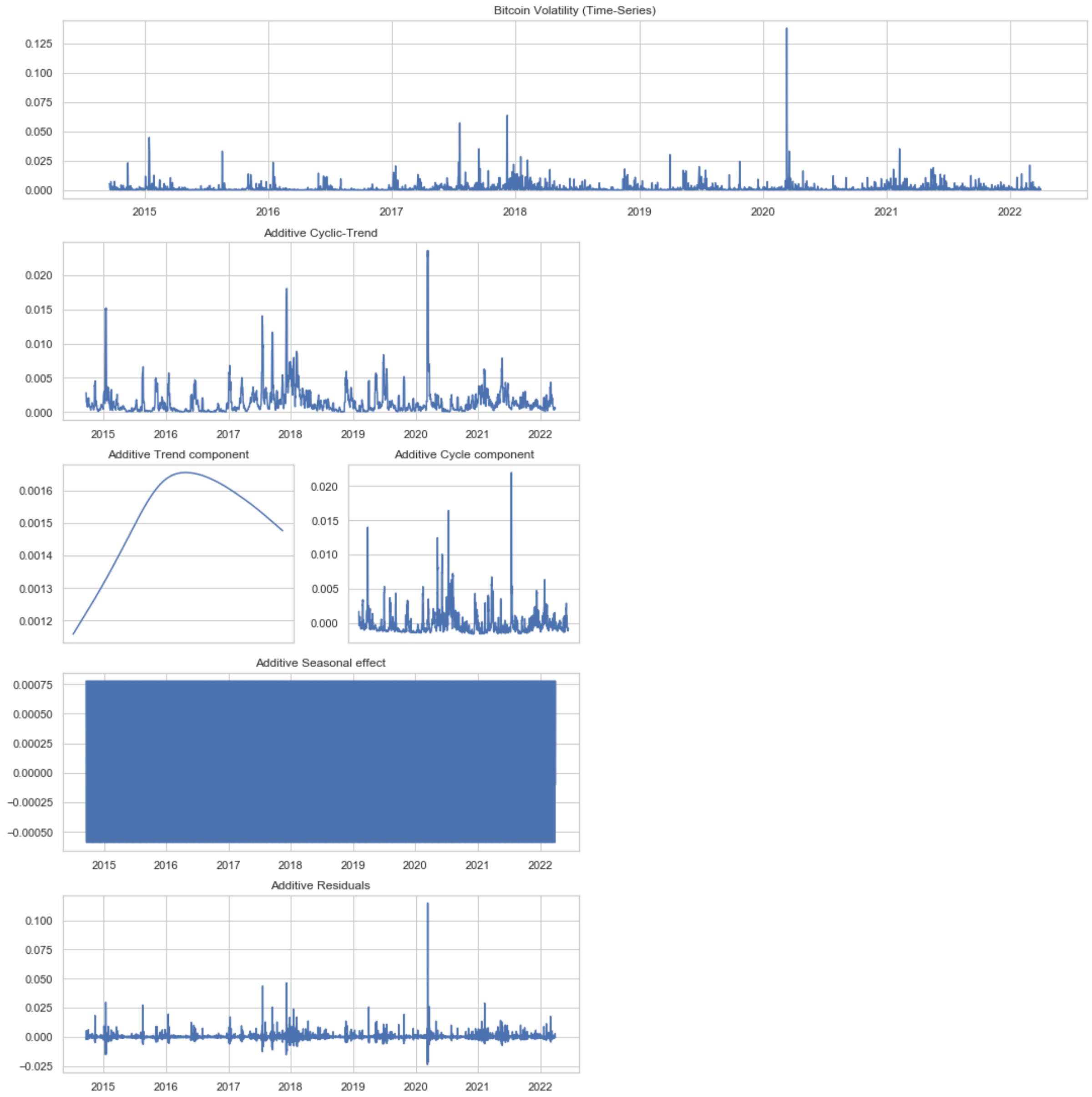


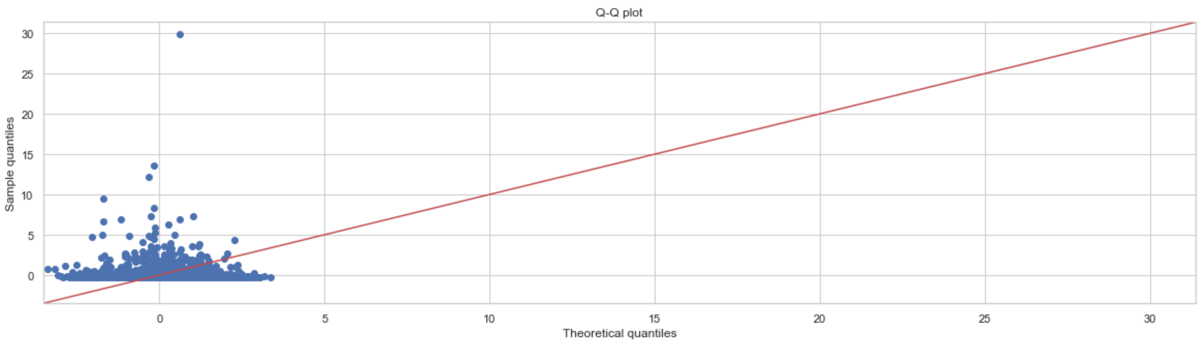

Bitcoin Volatility (Cumulative Distribution Function)
Probability
Square Returns
Q3
Median
Q1
0.0
0.2
0.4
0.6
0.8
1.0
0.00
0.02
0.04
0.06
0.08
0.10
0.12


Bitcoin Volatility (Box-plot per Year)
Square Returns
Year
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
2014
2015
2016
2017
2018
2019
2020
2021
2022


Bitcoin Volatility (Box-plot per Month)
Square Returns
Month
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec

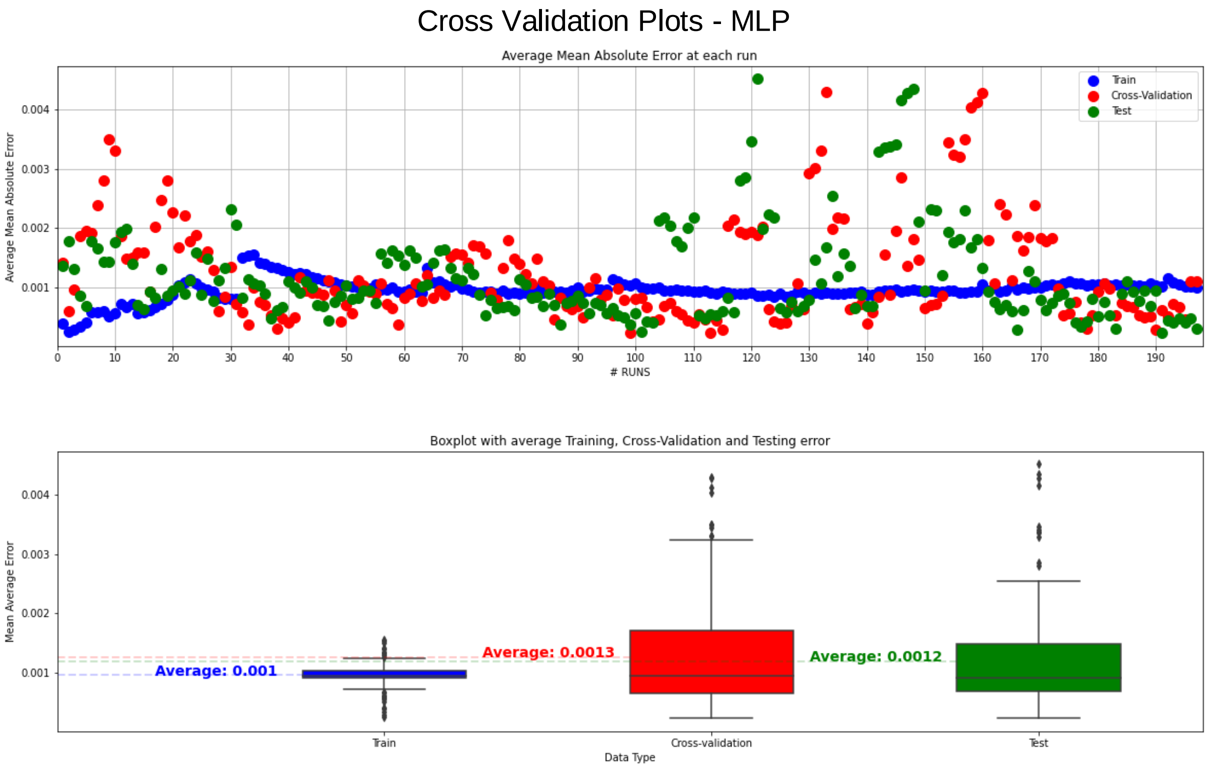

Rolling mean and std applied to Bitcoin Volatility
Rolling mean of bitcoin_volatility
Rolling standard deviation of bitcoin_volatility
Days/Year
Rolling std
Rolling Mean
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.002
0.003
0.004
0.005
0.006
0.007
0.008
2015
2016
2017
2018
2019
2020
2021
2022


Bitcoin Volatility (Time-Series)
Additive Cyclic-Trend
Additive Trend component
Additive Cycle component
Additive Seasonal effect
Additive Residuals

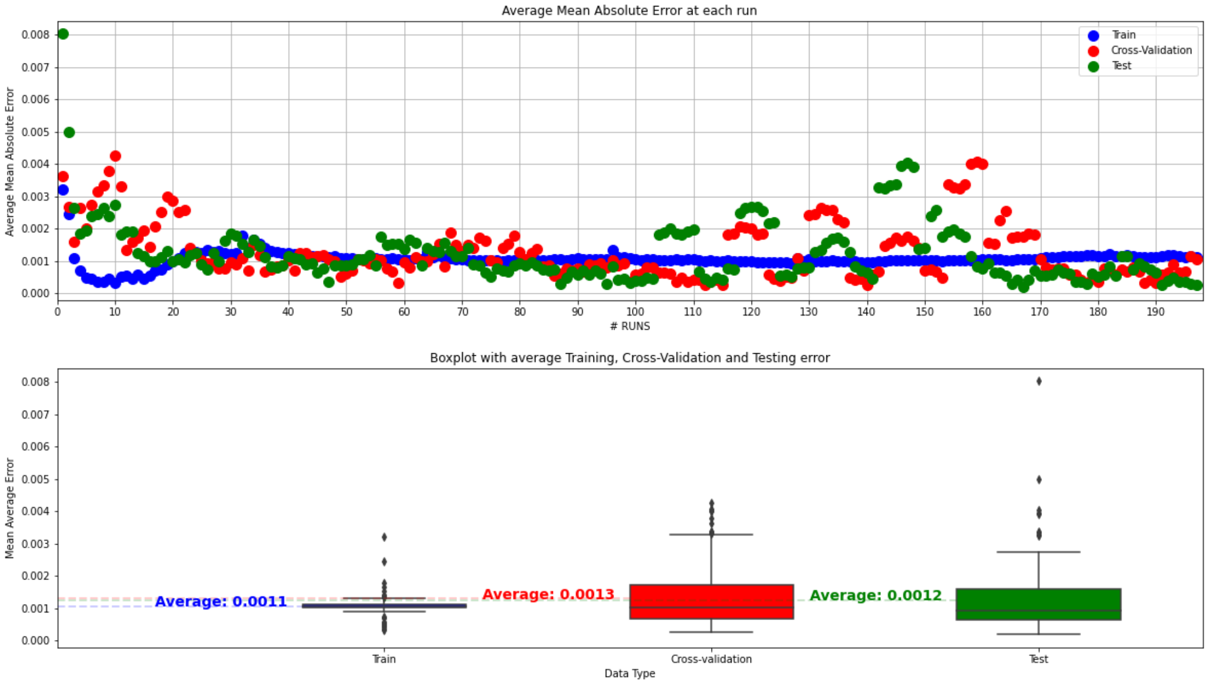

Q-Q plot
Sample quantiles
Theoretical quantiles


Time series and detected changes (threshold= 0.00345, drift= 0.00173): N changes = 283
0.15
0.10
0.05
0.00
Start
Ending
Alarm
0
500
1000
1500
2000
2500
Data #
Time series of the cumulative sums of positive and negative changes
+
-
).003
).002
).001
).000
0
500
1000
1500
2000
2500
Data #

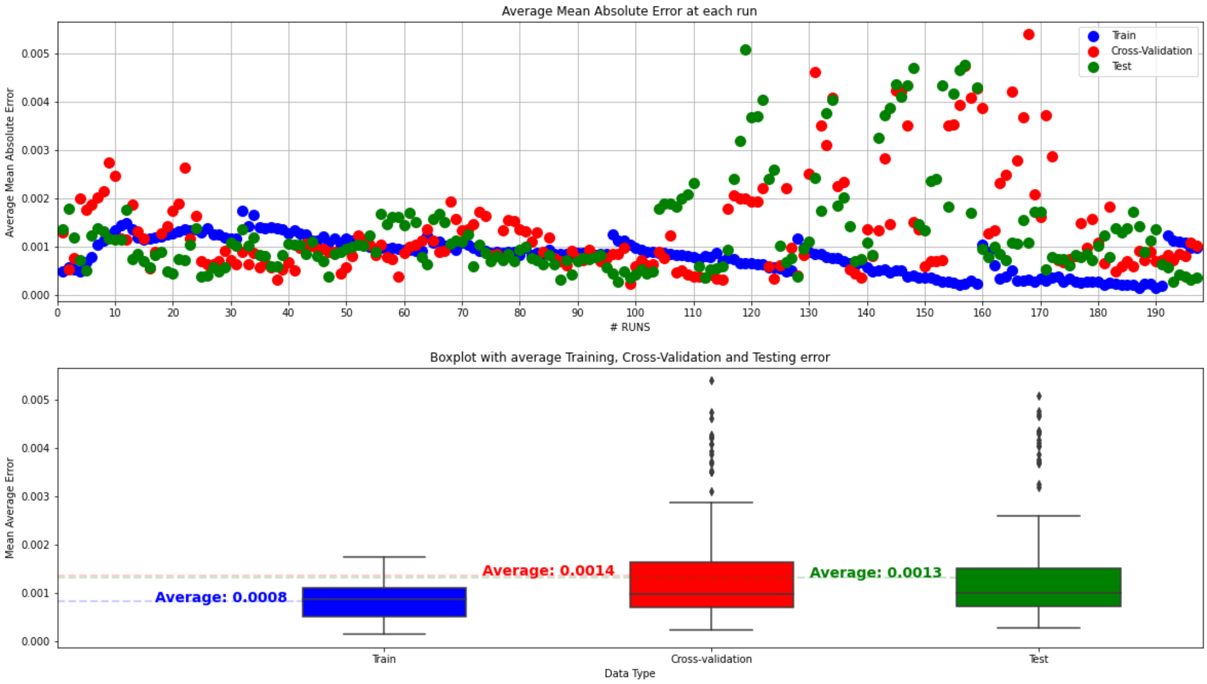
Appendix C


Cross Validation Plots - MLP
Average Mean Absolute Error at each run
Average Mean Absolute Error
0.004
0.003
0.002
0.001
Train
Cross-Validation
Test
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150
160
170
180
190
RUNS
Boxplot with average Training, Cross-Validation and Testing error
Mean Average Error
0.004
0.003
0.002
0.001
Average: 0.001
Average: 0.0013
Average: 0.0012
Train
Cross-validation
Test
Data Type

Cross Validation Plots – RNN



Cross Validation Plots – LSTM



Appendix D

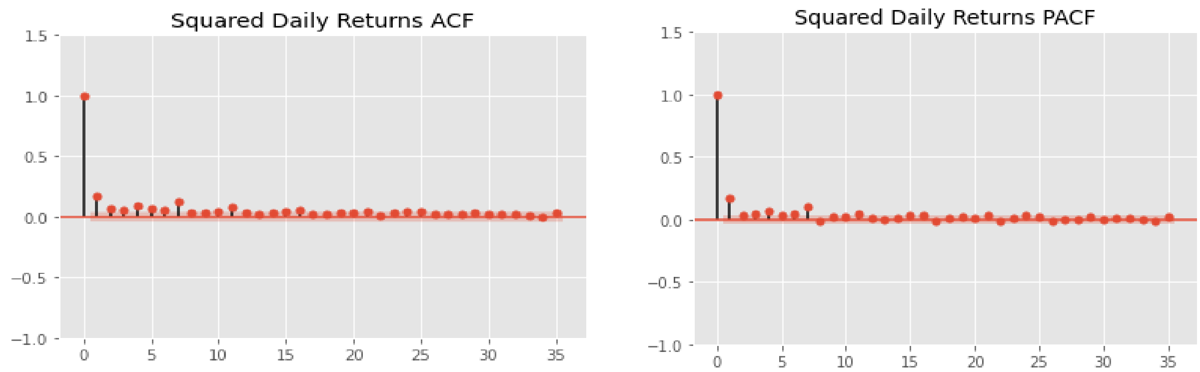
Arch Model

```
                      Constant Mean - ARCH Model Results
==============================================================================
Dep. Variable:                  Close   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                       ARCH   Log-Likelihood:                11746.0
Distribution:                  Normal   AIC:                          -23479.9
Method:            Maximum Likelihood   BIC:                          -23444.3
                                        No. Observations:                 2783
Date:                Sun, Jul 10 2022   Df Residuals:                     2782
Time:                        18:10:16   Df Model:                            1
                               Mean Model
============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
----------------------------------------------------------------------------
mu         1.1988e-03  2.615e-07   4583.900      0.000 [1.198e-03,1.199e-03]
                            Volatility Model
============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
----------------------------------------------------------------------------
omega      7.5389e-06  5.203e-12  1.449e+06      0.000 [7.539e-06,7.539e-06]
alpha[1]       0.0500  1.720e-02      2.908  3.642e-03 [1.630e-02,8.370e-02]
alpha[2]       0.0500  6.497e-02      0.770      0.442  [-7.733e-02,  0.177]
alpha[3]       0.0500  3.451e-02      1.449      0.147  [-1.765e-02,  0.118]
alpha[4]       0.0500  6.399e-02      0.781      0.435  [-7.542e-02,  0.175]
============================================================================

Covariance estimator: robust
```

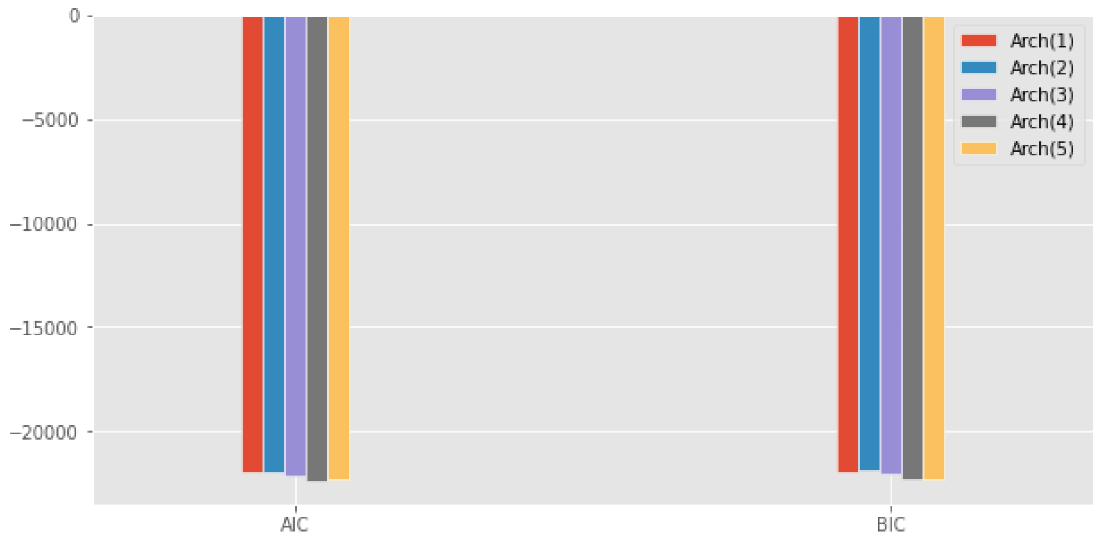
Garch Model

```
                     Constant Mean - GARCH Model Results
==============================================================================
Dep. Variable:                  Close   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                      GARCH   Log-Likelihood:                11863.9
Distribution:                  Normal   AIC:                          -23711.9
Method:            Maximum Likelihood   BIC:                          -23664.4
                                        No. Observations:                 2783
Date:                Sun, Jul 10 2022   Df Residuals:                     2782
Time:                        18:10:17   Df Model:                            1
                               Mean Model
============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
----------------------------------------------------------------------------
mu         1.1517e-03  1.349e-06    853.861      0.000 [1.149e-03,1.154e-03]
                            Volatility Model
============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
----------------------------------------------------------------------------
omega      4.5233e-06  9.947e-12  4.547e+05      0.000 [4.523e-06,4.523e-06]
alpha[1]       0.0500  1.899e-02      2.633  8.456e-03 [1.278e-02,8.722e-02]
alpha[2]       0.0500      0.161      0.310      0.757     [ -0.266,  0.366]
alpha[3]       0.0500  7.294e-02      0.685      0.493 [-9.296e-02,  0.193]
alpha[4]       0.0500  9.200e-02      0.543      0.587     [ -0.130,  0.230]
beta[1]        0.2500      0.713      0.351      0.726     [ -1.147,  1.647]
beta[2]        0.2500      0.677      0.369      0.712     [ -1.077,  1.577]
============================================================================

Covariance estimator: robust
```

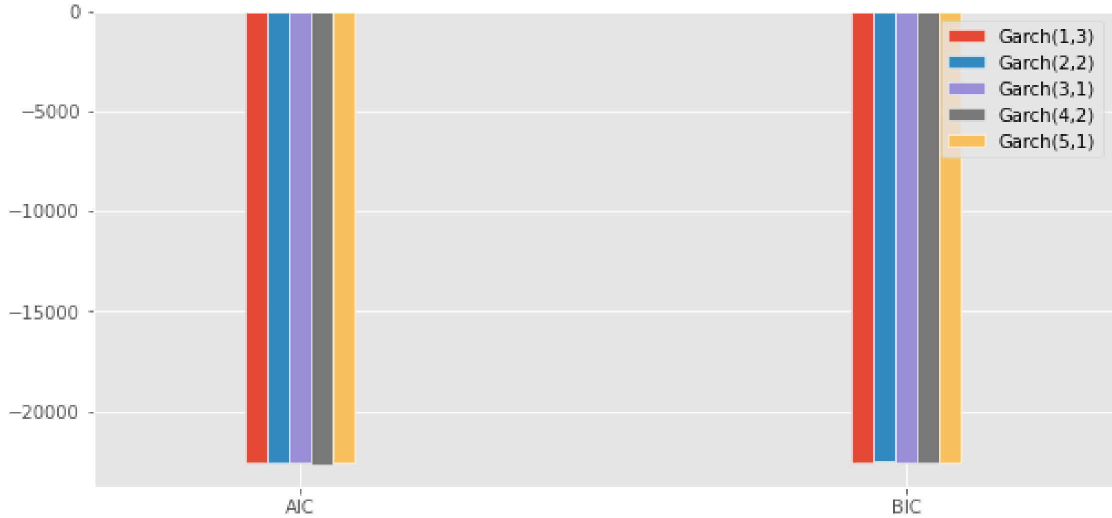
ACF and PACF Plots



Examples of AIC and BIC Values for ARCH models

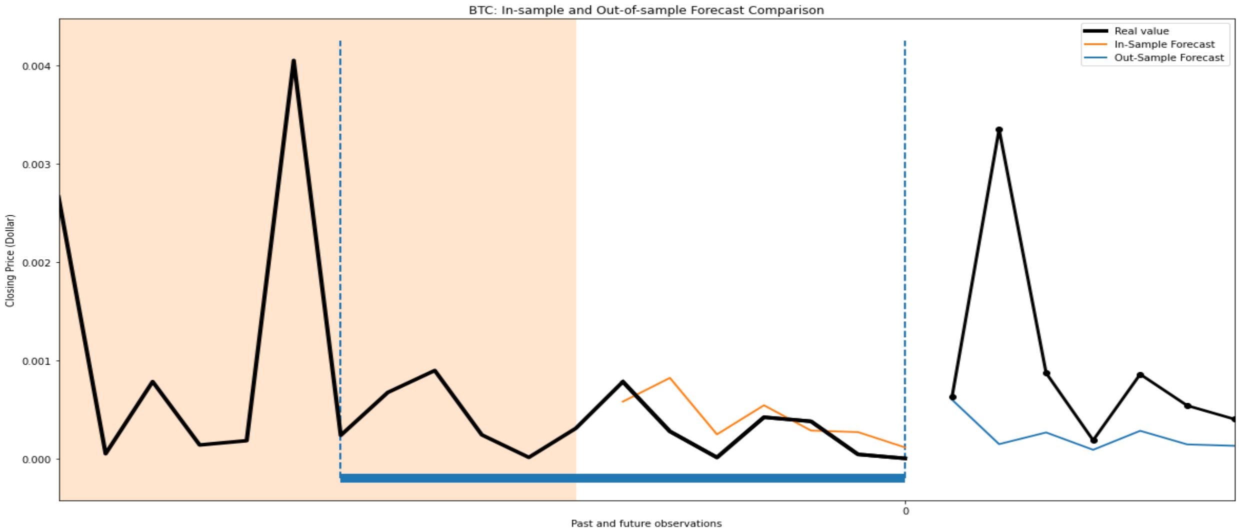


Examples of AIC and BIC Values for GARCH models

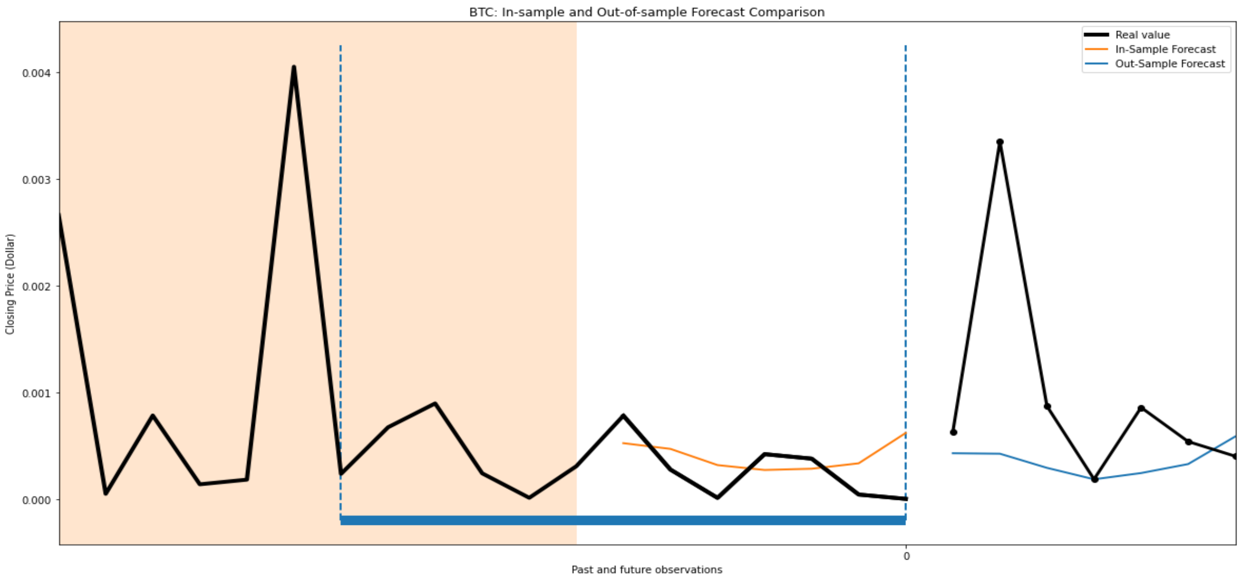


Appendix E

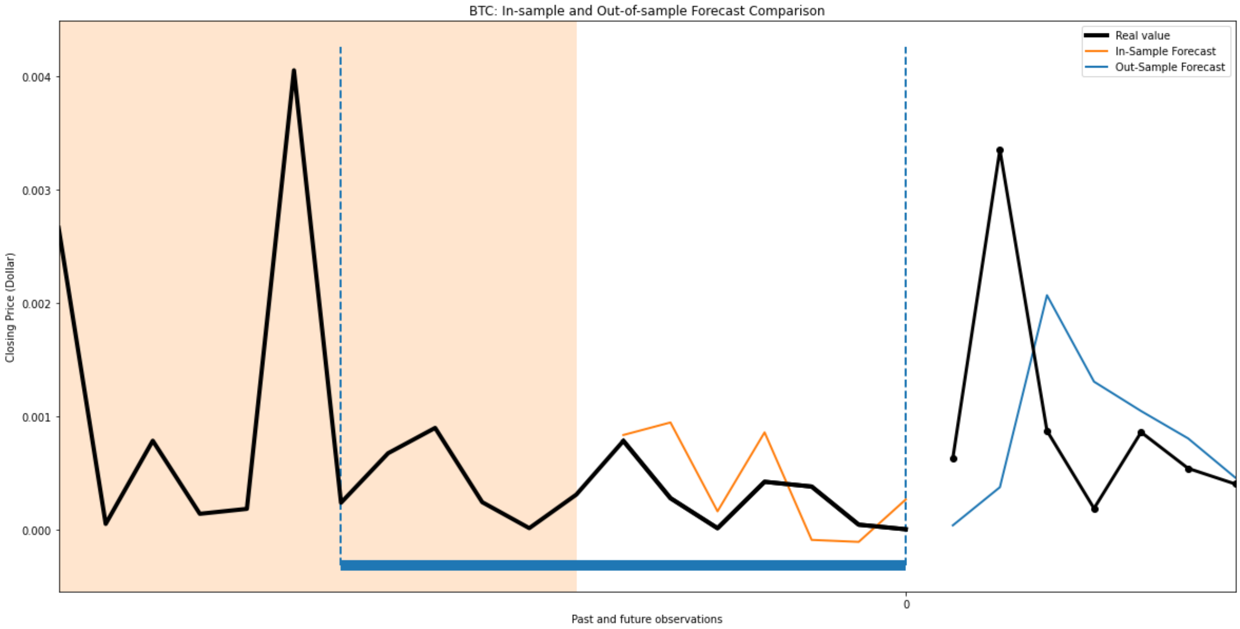
MLP Model: Forecasting and Sample Fitting



RNN Model: Forecasting and Sample Fitting



LSTM Model: Forecasting and Sample Fitting



Appendix F

Table with Real Values in Comparison to Model Forecast

Days	Real	Arch	Garch	MLP	RNN	LSTM
1	0,000628	0,002808	0,002831	0,000604	0,000429	0,000192
2	0,003348	0,002786	0,002837	0,000517	0,000423	0,000383
3	0,000869	0,002783	0,002824	0,000256	0,000291	0,002019
4	0,000184	0,002817	0,002894	0,000496	0,000184	0,001458
5	0,000856	0,002808	0,002866	0,000079	0,000242	0,001027
6	0,000537	0,002802	0,002858	0,000018	0,000326	0,000756
7	0,000487	0,002801	0,002837	0,000202	0,000584	0,000512