MSc Thesis in Computer Science

Time series forecast in non-stationary environment with occurrence of economic bubbles

Bitcoin Price prediction

Mateusz Garbacz
2018

Time series forecast in non-stationary environment with occurrence of economic bubbles

Bitcoin Price prediction

by

Mateusz Garbacz

to obtain the degree of Master of Science
at the Delft University of Technology to be defended publicly on Wednesday August 29, 2018 at 2:00 PM.

Student number:	4571681	
Project duration:	December 1, 2017 – August 29, 2018	
Thesis committee:	Prof. dr. ir. M.J.T. Reinders,	TU Delft
	Prof. dr. M. Loog,	TU Delft, supervisor
	Dr. ir. A. Bozzon,	TU Delft

This thesis is confidential and cannot be made public until August 29, 2018.

TU Delft

Abstract

Being capable to foresee the future of a given financial asset as an investor, may lead to significant economic profits. Therefore, stock market prediction is a field that has been extensively developed by numerous researchers and companies. Recently, however, a new branch of financial assets has emerged, namely cryptocurrencies. As a representative of these tokens, we chose the largest and most popular cryptocurrency, called Bitcoin [40]. Its value is characterized by with non-stationary behaviour [63] and occurrence of speculative bubbles, which cause a rapid explosion of the price, followed by a major crisis and market panic [63].

Currently, most of the research community does not take these issues into account, while predicting its price, which may lead to wrong conclusions or unstable results. Therefore, in this thesis, we take a step back and reconsider how does the environment influence model's performance and how to use this knowledge to implement more accurate forecast in the future. Moreover, by designing an appropriate methodology and employing semantic features from online text sources, such as Twitter, Reddit and online news portals, we attempt to build a robust prediction system that offers stable performance regardless of the market fluctuations.

Executed experiments prove that non-stationarity negatively influences the results, causing the deterioration of model's performance over time. Furthermore, it appears that there may be certain properties of economic bubbles that facilitate more efficient prediction, as well as some predictors have an ability to successfully forecast the beginning of a market crisis. However, these findings are based on individual observations, which need to be confirmed by further research. In addition, by designing an appropriate methodology, we prevented performance deterioration, caused by price signal non-stationarity. Although, the semantic features based on online sources did not boost the robustness of the system significantly, combined with the suitable system's design, they lead to improvement in the overall performance of the predictor.

Preface

This document describes the study performed for the completion of the MSc degree in Computer Science, the Data Science track. The research was conducted at the Patter Recognition Laboratory (PRLab) at the Delft University of Technology. The research topic is a mixture of the fields of my interest, ranging from Machine and Deep Learning, through information retrieval and text processing, to cryptocurrencies and financial markets. It started as a side project, developed along my studies, and transformed into a large scale MSc Thesis project. Initially, the focus was put on simply predicting the Bitcoin price, yet, with time, I discovered how deep and complex some topics are, and decided to dive into them. Moreover, by observing how the price time series developed over time, I realized that the data has some interesting properties, namely non-stationarity and occurrence of economic bubbles. Furthermore, while reviewing the related literature, it occurred that these features of the signal are disregarded by most of the researchers. This lead the study to shift its focus, from an application of the predictive algorithms to a research on the importance of the mentioned properties and how to better deal with them in the future.

During the project I have gained a vast of knowledge, especially in the field of Deep Learning. The main lessons learned from the study are the importance of understanding of what a given algorithm really does, the impact of a suitable hyperparameters setting on the end result, and that, one mistake in the data processing steps, particularly in case of Deep Learning, may cost weeks of debugging.

I would like to thank several people for all the support they gave me. Firstly, I am grateful to my supervisor Marco Loog for all the constructive feedback and knowledge that he shared with me during the project. I suppose that with no other professor as a supervisor, I would get so much motivation to put my best effort in this research. I would also like to thank my family and my girlfriend, Katarzyna Garncarek, for the endless support and patience with me during my hardest times. This thesis was certainly the most demanding project in my entire life, and without them, I would most probably fail. Finally, I would like to thank my friends: Marc Juchli, Bartosz Czaszyński and Ramy Al Sharif, who went through the same journey, called the 'MSc Thesis Project'. Even though each of us conducted his own research, we have always spent time supporting and motivating each other, and providing others with creative suggestions. Thanks to you, the 9 months of hard work were much more enjoyable.

Mateusz Garbacz
Delft, August 17, 2018

Contents

1 Introduction

Recently developing cryptocurrencies, such as Bitcoin, are an interesting branch of investment assets. Not only the market is young in general, but also considerably volatile, since basically, any transaction or event related to cryptocurrencies may strongly influence their value [7]. Moreover, in 2017 the price of Bitcoin has grown from around $950 to over $15500[1]. Interestingly, the cryptocurrency has been influenced by economic bubbles, most probably, due to over-optimistic predictions of its future price and overall public flurry [63]. The phenomenon is characterized by a drastic value growth and further, a crash [34].

Such volatile environment poses several challenges, while forecasting the value of the asset. One needs to make sure that the system accommodates to the frequent market fluctuations as well as to general long-term trends, driving the price to the extreme levels that have never been reached before. Hence, it is crucial to design a system that deals with phenomena such as time series non-stationarity, economic bubbles and occasional explosive growth [1],[63]. Even though certain steps are taken to prevent the negative influence of these aspects, they may still affect the future forecast performance. Therefore, it is essential to understand how economic bubbles and data non-stationarity impact the prediction of Bitcoin price in order to make the system more robust to it. This may lead to an increase of confidence in the forecast or an acknowledgement of its limitations in case of major events, such as crisis.

This thesis considers the problem of Bitcoin price prediction and examines the short-term, 15 minutes ahead, forecast to answer its research questions. It focuses on the properties of the forecasted time series, namely non-stationarity and the presence of economic bubbles as well as deepens understanding on what is their effect on the results and how to deal with them.

1.1. Bitcoin and Cryptocurrency market

Bitcoin as a first and the largest decentralized electronic currency system is based on a peer-to-peer transactions with no centralized intermediary. Another aspect which differentiates it from standard currencies is that the transactions and liquidity within the network are based on cryptography. These characteristics have underlined Bitcoin's uniqueness and attractiveness since its creation by Satoshi Nakamoto in 2009. From that point on, it has been developed by a thriving open source community and it has appealed to a large number of users, investors and media attention [40].

Since the introduction of Bitcoin, numerous other decentralized cryptocurrencies have been introduced. These virtual coins are primarily characterized by their constant and significant price change [10]. For example, Bitcoin's price increased since the beginning of 2013, when it has costed below $14 per coin, to its value of over $14000 in December 2017[2]. Moreover, other cryptocurrencies have shown similar, unstable price behavior, illustrated by, for example, Ripple's and Litecoin's value fluctuations since the end of December 2013 [10].

Overall price volatility of cryptocurrencies is related to many factors, e.g. security breaches and introduction of new features [7]. However, the reasons can be summarized by general users' perception of a given currency, meaning whether they believe in its stability, safety and profitability. If there are certain negative issues, such as delegalisation or fraud, the general price drops, because the customers tend to overreact by

[1] Data based on https://www.coindesk.com/price/

[2] Price information based on the charts from https://www.coindesk.com/price/

converting it into standard, safer currencies. Nevertheless, whenever the positive prospect of cryptocurrencies is promoted, for example, by announcement of development of new features or approval by banks and governments, the price analogously increases [7].

Furthermore, cryptocurrency market volatility creates opportunities for income seeking investors. To maximize the financial reward, those investors employ special strategies, e.g. high-frequency, low-latency trading. This is further enhanced by the price prediction, provided by robust Machine Learning algorithms [40]. The prediction systems mostly employ the market specific features, described in detail in Chapter 3. However, enhancing the prediction using text data from news articles, online forum, and social media, gathers increased interest of the research community [10][42].

1.2. Bitcoin price prediction

Price Prediction problem is a time series task, aiming at forecasting in what way the value of given resource will change in the future [50]. It can be modeled in two ways, either as a classification [40][44] or a regression [26] problem. The former case predicts whether the price will be higher or lower after a given period of time, [50], while the latter's target is to forecast the exact future value of a given asset [26].

In order to prevent confusion with a closely related field, namely **Fluctuation Detection problem**, the distinction should be clarified first. The task is to, firstly, forecast whether at a given time there will be a major asset's price change and only then classify in what way the value will shift [10]. Therefore, the main difference is that the final worth change prediction is made only under certain circumstances, for instance, if there is a significant fluctuation in the number of positive sentiment tweets posted by Twitter users [42].

In the field of Bitcoin Price Prediction, the research community focuses on building accurate systems that use different types of data. The extracted features are mainly based on:

- Market data, which consists of currency exchange market specific features, for instance price, traded volume, number of trades [40][10][44].
- Coin specific data, gathered based on the Bitcoin network information, for example hash rate, mining difficulty [40].
- Text data, gathered from a broad range of online sources available, form news articles, through forum comments, till social media [50][42][62].

The Chapter 3.2 elaborates on the related work in the field as well as features and methodology they employ.

1.3. Time series and non-stationarity

The discussed problem belongs to the financial time series prediction class of tasks.

A time series is a sequential set of data points, typically representing measurements performed over consecutive time steps. Mathematically, it is defined as a set of vectors $x(t)$ for elapsed time $t=0,1,2...$ [2]. Features in such vector may represent a single measurement examined at a specific point or aggregated data over a time window. Therefore, a significant property of such dataset is its chronological order [2]. Thus, in time series prediction, given a data point x at time t, one tries to foresee the value of vector $x(t+k)$, where $k>0$, [2].

In general, financial time series belong to the group of the most difficult signals to forecast [1]. There are two main problems with the data, namely, noisiness and non-stationarity [1]. The considered markets are dependent on many factors, such as large investments, rumours, news events etc. Thus, the predictive models applied in the field struggle to take all the relevant information into account. Consequently, noise in such data is defined as the factors that influence the value of a given asset, but are not expressed with the time series [1]. On the other hand, non-stationarity stands for a phenomenon of data distribution changing over time. The formal definition of the property follows in section 2.1.3. Appearance of new and previously unseen situations is usual for financial markets, leading to their constant development [1]. Thus, with time, the current knowledge and rules extracted from data become outdated and such forecasting system requires continuous updates to retain its predictive power. Typically, these programs are designed to predict the future price movements and therefore, are tested on the data subsequent to the training set [1]. Since Supervised Learning approaches typically assume that the train and test samples come from same distribution, and non-stationary dataset's distribution shifts in time, the prediction task in a financial market is problematic for a Machine Learning algorithms [1]. This means that the samples shown to the classifier in the development phase are often not representative of the data on which the classifier is applied.

Moreover, as shown in chapter 6 as well as proven in the literature [63], the Bitcoin price signal contains several speculative bubbles. This phenomenon, also called an economic bubble or simply a bubble, appears when the value of an asset strongly exceeds its intrinsic value [20]. On such occasions, the price is based on implausible or inconsistent views about the future of the asset's price [20]. Speculative bubbles are especially risky for the investors, due to an unnatural growth of asset's value and subsequent panic and crash right after [20]. Therefore, they lead to disturbance of the market's stability and may cause difficulties to the prediction systems. Section 2.1.4 describes the formal definition of an economic bubble.

Consequently, predicting Bitcoin price time series is a challenging task and requires careful consideration of the applied solution and derived conclusions.

1.4. Problem Statement

As touched upon, learning a prediction system for forecasting Bitcoin price is a difficult task. Not only, the data distribution evolves over time, causing violation of the basic Machine Learning assumption [1], but also, the signal contains speculative bubbles, which may further influence the forecast [63]. Moreover, one cannot be certain whether the system would have comparable performance, for instance, next year or in case of crisis, unless one employs the features that directly model the market fluctuations.

Thus, this leads to the main Research Question (RQ) of the thesis:

RQ 1 How do the non-stationarity and speculative bubbles encountered in the Bitcoin price signal influence its prediction and how to make the forecast more robust?

As shown in the Chapter 3, most of the research community in this field simply reports the model's test set performance with no reflection on how it would act in case of major events influencing the market. When such systems suffer from non-stationarity, one needs to find methods to enhance its robustness or provide insights on circumstances, in which not to trust the prediction. The thesis employs the short-term (15 minutes) prediction of Bitcoin price, due to much lower predictability of the local fluctuations in the time series [40] [62].

In order to answer the main research question, several subquestions need to be addressed. Firstly, one needs to analyse the Bitcoin price signal, by inspecting whether the time series is non-stationary, as well as detecting the economic bubbles. The investigation of the time series properties and location of bubbles in the sequence allows for answering further stated RQs. Thus, the first research subquestion is defined as:

RQ 1.1 What are the properties of the Bitcoin signal in terms of stationarity and speculative bubbles?

Secondly, before performing the forecast, features that can be employed in prediction need to be extracted from raw data. As demonstrated, financial time series is not only non-stationary, but also very noisy [1]. Therefore, to decrease the amount of noise, one needs to extract features that express as many factors that influence the price as possible. Moreover, by considering data from different sources, one may potentially find correlations that are not influenced by non-stationarity of the price signal. Even though the data from the market changes together with the asset's value over time, similar events in press may influence the price in a more regular way. For instance, news about Bitcoin related to fraud might suggest the future price drop, while intense discussion on social media about positive aspects of the currency may be an indicator of a long-term uptrend. Thus, incorporating these variables into the forecast can reduce the noisiness of the time series and allow the prediction model to build universal rules that are, to some extend, robust to market's non-stationarity. This thesis considers the three main Bitcoin related **text data sources**:

- Social Media data streamed from Twitter that represents public perception, concerns and expectations towards Bitcoin.
- Forum comments extracted from Reddit, where users discuss upon various cryptocurrency related topics.
- Articles from a number of online news portals, which describe in detail the current events and future plans.

Unfortunately, this unstructured text may not directly be incorporated into the prediction. Therefore, the second subquestion that needs to be answered states as follows:

RQ 1.2 How to incorporate the text data into prediction and how does it influence model's robustness to non-stationarity?

Further, once the features are extracted from the raw data and further transformed into a time series, one can perform a prediction. However, as shown in the section 3.1, in the field of Bitcoin Price Prediction there is neither standard forecast methodology, nor performance metrics. What is more, used approaches do not take into consideration non-stationarity of the time series. Thus, one needs to set up a methodology that would increase robustness to that phenomenon and, so called, concept shift, described in section 3.3. Moreover, most of the systems that are tested on data from different periods of time are not comparable to each other. In other words, it is difficult to pinpoint, which methodology leads to best results, for example, a regression model trained and tested in 2017, with Mean Squared Error of 100, is not necessarily worse than the one from 2016, with value of error 10. That is due to different volatility levels of Bitcoin price for the two years. In order to answer the RQ1, one needs to take into account how to make the forecast system robust to non-stationarity and how to compare its performance. Thus, the third subquestion that needs addressing is:

RQ 1.3 How to predict Bitcoin price and measure its performance, taking into consideration non-stationarity of the data?

Once the prediction framework is set up and the models are trained with various feature sets, one can consider testing how the developed systems react to the data distribution evolution as well as speculative bubbles. It is crucial to find which factors cause performance deterioration and whether designed methodology extracted features prevent it. The analysis may provide valuable insights into dealing with non-stationarity in the environment, therefore, the fourth subquestion addressed by the thesis is:

RQ 1.4 How well does a prediction system deal with non-stationarity and economic bubbles, and can semantic features derived from the considered data sources improve the forecast?

1.5. Outline

There are 8 chapters that follow this introduction. Firstly, in chapter 2, the background information, necessary for understanding the following steps, is described. The focus is mainly put on the topics of time series analysis and Deep Learning. Then, chapter 3 presents the related work in the fields of Bitcoin price prediction as well as semantic features extraction and dealing with non-stationarity. The gathered knowledge will be considered across the thesis, while designing the research methodology and analysing the results. Further, in chapter 4, the process of data collection from multiple sources is described, together with brief analysis of the resulting dataset. Next, chapter 5 defines the data processing methodology. Therefore, samples of Bitcoin price data, tweets, comments and news are transformed into a single multivariate time series, on which the forecasting algorithms can be applied. After that, in chapter 6, the extracted sequence is analysed in terms of stationarity and economic bubbles. Additionally, an appropriate data transformation is applied to ensure successful prediction. Then, chapter 7 considers issues appearing while predicting non-stationary variable with occurrence of bubbles and formulates methodology for predicting Bitcoin price. Later, chapter 8 presents the experiments performed in order to answer the main research question of this study. Finally, in chapter 9, a discussion is conducted, regarding each of the stated RQs and a conclusion of the thesis is formulated.

2

Background Knowledge

The aim of this chapter is to introduce the background the knowledge required to fully understand the methodology applied and the discussion conducted in the thesis. Firstly, section 2.1 describes the theory of economic time series analysis. Next, section 2.2 elaborates on the complex Machine Learning methodology that the thesis exploits to answer the stated research questions.

2.1. Time series Analysis

Any time series can be represented by a mathematical model called a stochastic process [39]. The stochastic process, also called a random process, is a set of random variables, values of which are uniquely indexed based on some mathematical set, namely an index set [39]. The index set is often represented by consecutive natural numbers, usually interpreted as a sequence of distinct steps in time [39]. Values of a given variable in the stochastic process come from a certain mathematical space, called a state space [39], for instance a set of integers, or real numbers. Finally, a sample function is the mapping between the index set and the values of the stochastic variables [39].

The autoregressive–moving-average (ARMA) models provide the mathematical representation of a stochastic process X in terms of two polynomials: autoregression and the moving average [28].

The AR(p) notation, refers to the autoregressive model of order p, described with the following formula:

$$X_t = c + \sum_{i=1}^{q} \phi_i X_{t-i} + \varepsilon_t. \tag{2.1}$$

In the equation 2.1, c is a constant number, $\phi_1, ..., \phi_q$ are parameters and ε is a white noise coefficient.

The MA(q) is a moving average model of order q:

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^{q} \theta \varepsilon_{t-i}. \tag{2.2}$$

The equation 2.2 uses μ as expectation of X, $\theta_1, ..., \theta_q$ as parameters and ε as a white noise term.

Therefore ARMA(p,q) model consolidates the two polynomials into:

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \phi_i X_{t-i} + \sum_{i=1}^{q} \theta \varepsilon_{t-i}. \tag{2.3}$$

The same model can be represented using a lag operator L, which indicates the previous value of a given sample [39]. Essentially it shifts a value X_t back in time by one step:

$$LX_t = X_{t-1}. \tag{2.4}$$

Thus, the ARMA(p,q) model can be alternatively described using the lag operator:

$$(1 - \sum_{i=1}^{p} \phi L^i) X_t = (1 + \sum_{i=1}^{p} \theta L^i) \varepsilon_t. \tag{2.5}$$

This equation can be simplified to the form

$$\Phi(L)X_t = \Theta(L)\varepsilon_t \tag{2.6}$$

The ARMA model is exploited in the further subsections, while studying the time series characteristics.

2.1.1. Unit roots

Unit root is an major property of some stochastic processes [39]. It is present in a given process if 1 is a root of its characteristic equation [39].

Consider ARMA(p,q) lag representation in the formula 2.6, where $\{X_t\}_{t=0}^{\infty}$. Unit root is present in the polynomial, if $\Phi(L)$ has a single root, equal to 1. In other words, the solution of the equation below is 1:

$$1 - \phi_1 L - ... - \phi_1 L^p = 0 \tag{2.7}$$

This property has serious implications on the time series. If the unit root is present, the sequence is non-stationary, causing its mean and variance change over time [39]. Therefore, any fluctuation in such series may cause a significant shift of its mean [39]. On the other hand, absence of the unit root has a significant influence on a series as well. The solution of a characteristic equation being lower than 1 indicates time series stationarity, while being larger than 1 suggests an explosive non-stationary process [39].

2.1.2. Augmented Dickey-Fuller test

Augmented Dickey-Fuller (ADF) test is a statistical tool for detecting presence of the unit root [28]. Consequently, it can be used to identify crucial properties of a time series, such as non-stationarity or economic bubbles.

Consider AR(p) model:

$$X_t = c + \sum_{i=1}^{q} \phi_i X_{t-i} + \varepsilon_t. \tag{2.8}$$

This equation can be rearranged as follows:

$$X_t = \rho X_{t-1} + \psi_1 \Delta X t - 1 + ... + \psi_{p-1} \Delta X t - p + 1 + \epsilon_t. \tag{2.9}$$

In equation 2.9, Δ denotes the first difference operator, ρ represents $\phi_1 + ... + \phi_p$, while ψ_n is an equivalent of $-\sum_{i=n+1}^{p} \phi_i$. Hence, the unit root is present in the model represented by equation 2.9 if $\rho = 1$ [28].

The unit root test is conducted under the null hypothesis of $\rho = 1$. Its left-tail alternative is $\rho < 1$ and the right one, $\rho > 1$. Therefore, depending on the version of the test, one can use it to prove stationarity or explosive behaviour of the sequence.

2.1.3. Process Stationarity

Essentially, stationarity, also called **strict stationarity**, is a property of a stochastic process characterized by temporal stability of its unconditional joint probability distribution [28]. Hence, the parameters such as mean and variance of such series remain the same for the entire length of the signal [28].

In order to formally define strict stationarity, one needs to consider a stochastic process X_t and a cumulative distribution function $F_X(x_{t_1+\tau}, ..., x_{t_k+\tau})$ of the unconditional joint distribution of X_t at times $t_1+\tau, ..., t_k+\tau$. The X_t is strictly stationary if for all n, for all τ and for all $t_1, ..., t_k$:

$$F_X(x_{t_1+\tau}, ..., x_{t_k+\tau}) = F_X(x_{t_1}, ..., x_{t_k}). \tag{2.10}$$

However, strict stationarity is often an unrealistic assumption for a dataset to fulfill [28]. In such cases, one can employ **weak stationarity** term, also called the second order stationarity. Such setting only requires the mean and the autocovariance of the process to remain the same over time [28].

In order to formally define the assumption, consider $m_x(t)$ as mean function of a stochastic process X at time t and $C_x(t_1, t_2)$ as the covariance function at times t_1 and t_2. The process is weakly stationary if for all τ:

$$m_x(t) = m_x(t+\tau) \tag{2.11}$$

and for all t_1 and t_2:

$$C_x(t_1, t_2) = C_x(t_1 - t_2, 0) \tag{2.12}$$

If any of these requirements is violated, then the process is non-stationary.

Therefore, the properties of the non-stationary process depend on the adopted assumption regarding stationarity [39]. This thesis considers the weak stationarity of the data, due to its convenience and less restrictive nature.

Moreover, presence of the unit root is another crucial aspect. A process with this property is also called **difference stationary**. In other words, it is a non-stationary time series, which can be made stationary by applying the technique called differencing [39] (described in section 3.3), either once or multiple times. The Box-Jenkins approach [39] models such time series X using the equation:

$$\Delta^D X_t = \mu + \psi(L)\varepsilon_t, \tag{2.13}$$

where $\Delta^D = (1-L)^D$, $\psi(L)$ is a infinite-degree lag operator polynomial $1 + \psi_1(L) + \psi_2(L)^2 + ...$ and ε_t is the white noise. Therefore, if presence of the unit root cannot be disproven, we will assume that the time series is difference stationary.

2.1.4. Speculative Bubble

Speculative bubble is a phenomenon appearing in the financial market, in which the actual value of a given asset becomes higher than its fundamental value [20]. Such deviation may be caused by several causes. One option is an excessive monetary liquidity of an asset, corresponding to the excessive demand of the market compared to the available supply [20]. Otherwise, it may be caused by social factors of the market participants [20]. For instance, the greater fool theory underlines that the overly optimistic investors (the fools) purchase the overvalued item, simply to sell it to the other purchaser at a higher price (the greater fool) [20].

Kindleberger et al. [34] characterize 5 stages of an economic bubble:

1. Displacement- increase of the asset's value, caused by some sort of event, being a shock to the market.
2. Boom- increase of optimism regarding the asset's price. Its increasing value causes greater consumption, which further pushes the price to rise, fueling the positive feedback loop.
3. Euphoria- unrealistic perception of the future of the asset. Investors expect long-lasting growth of the price. Even though some of them realize that there is a bubble, they believe that they can sell their resources before the downfall.
4. Distress- decline of market participants' confidence, decrease of price trend momentum and shift of the trend.
5. Panic- realization of the unfolding crisis, causing a hurried selling of the assets by most of investors at the same time. The phase lasts until the confidence in the profitability of the investment grows or the price falls low enough for investors to start purchasing again.

There are several theories that explain the emergence of bubbles, however, the dominant approach is a **rational bubble** model [20]. This approach analyses the bubbles' evolution, given that agents act rationally. Blanchard and Watson [11] proposed a mathematical model representing a rational bubble, which decomposes the asset's value into its fundamental and bubble components.

$$P_t = P_t^{fund} + B_t \tag{2.14}$$

The fundamental component, is represented using the Discounted Cash Flows, see Equation 2.15.

$$P_t^{fund} = \sum_{t=0}^{t=\infty} \frac{C_r}{(1+r)^t} \tag{2.15}$$

The formula models the fundamental worth of an asset with infinite maturity, which at each time step t distributes dividends C_t and applies the interest rate r. Also, the bubble component at step t is formulated as its future value at $T \to \infty$, growing with the interest rate r.

$$B_t = \lim_{T\to\infty} \left[\frac{B_T}{(1+r)^{T-t}} \right] \tag{2.16}$$

Hence, the equation 2.14 can be rewritten as follows:

$$P_t = E_t\left[\sum_{\tau=t+1}^{\infty}\frac{C_\tau}{(1+r)^{\tau-1}}\right] + \lim_{T\to\infty} E_t\left[\frac{B_T}{(1+r)^{T-t}}\right] \tag{2.17}$$

However, the consequence of exploiting this model is the requirement of the bubble component to grow exponentially for the bubble to exist [20]. Moreover, it assumes the asset is infinitely lived [20]. Despite these issues, the approach form the basis for the further proposed modifications, which relax these restrictions [21] [4].

The methods widely used for rational bubbles detection are Supremum Augmented Dickey-Fuller (SADF) [54] and Generalized Supremum Augmented Dickey-Fuller (GSADF) [56]. These techniques are designed to identify the unobserved bubble component in the observed asset price, together with the date of it's occurence. They recursively apply the right-tailed ADF test [55]. Its null hypothesis is the unit root being present in the sequence, while the alternative identifies the presence of an explosive autoregressive coefficient:

$$H_0 : \rho = 1 \tag{2.18}$$

$$H_1 : \rho > 1 \tag{2.19}$$

For each point in the time series, the algorithms recursively calculate the supremum [54][56] of the ADF test statistic over the expanding window of data. However, as shown by Phillips et al. [56], the GSADF algorithm has more discriminatory power than SADF, due to an improved window size selection procedure. The formula computes the GSADF statistic at a given time step:

$$GSADF(r_0) = sup_{r_2\in[r_0,1],r_1\in[0,r_2-r_0]} ADF_{r_1}^{r_2}, \tag{2.20}$$

where r_1 is the beginning of the current window, r_2 is its ending and r_0 is its minimum size. Whenever the test statistic exceeds the critical value L_T a rational bubble is present at the given time.

2.2. Machine Learnng and Deep Learning

Machine Learning (ML) is a field of Computer Science that focuses on learning programs from data [17]. This means that, computer can solve various tasks without being specifically programmed for them, but rather, by generalization from examples. In order to do so, one trains a **model**, defined as a set of rules, determined from the data using a ML algorithm. Further, a **prediction system** is a program that applies the entire pipeline of techniques and trained models on the data to perform a specific job, in case of this research a forecast.

There are two major types of learning that this thesis deals with:

- **Supervised Learning**, in which an algorithm finds a mapping from the input to the output, based on the ground truth input-output samples.

- **Unsupervised Learning**, which infers a description of hidden structure in the provided data.

Typically, construction of an efficient ML-based solution requires careful feature engineering. Thus, experts in the field, in which such methods are applied, need to spend a considerable amount of time to extract features from the raw data that, in their opinion, would facilitate learning. [38]. However, out of the broad group of available approaches, Deep Learning (DL) overcomes the requirement of manual feature engineering, by the use of further described Artificial Neural Networks (ANN) [35]. By stacking a number of simple, non-linear modules on top of each other, and then, extensive training and optimization, these systems discover a representation needed to solve a given task efficiently, from the raw data. At each level, starting from the original input, given module receives the output of previous layer, and transforms it into more abstract representation. For the classification task, each further layer amplifies the characteristics of the input that assist class discrimination and depress irrelevant attributes. Therefore, given a sufficient number of samples and a adequately powerful architecture, these models can learn very complex relationships, without any extensive data processing [38]. Moreover, modern DL frameworks allow to parallelize the performed computations and take advantage of efficient Graphical Processing Units. Additionally, the state-of-the-art performance in multiple domains [38] make this methodology appropriate to apply in a data intensive research.

2.2.1. Machine Learning assumptions

Consideration of assumptions made for a given dataset is crucial, while selecting appropriate methodology and analysing the results.

Typically, while applying most of the ML algorithms, one needs to assume that a variable X is independent and identically distributed (I.I.D.) [2]. Hence, the assumption consists of two parts.

Firstly, any two samples should not depend on each other. This typically does not hold true for time series, since given value is, to some extent, correlated with the previous ones [2]. A market trend may be an illustration of such violation, for instance if value of an asset has continuously been increasing over past few minutes, there is a high chance that it will continue to increase. However, the algorithms designed for sequential data can handle this dependence by making prediction, given the series of past samples [2]. Therefore, violation of this assumption has, typically, no serious implications for the results, if an appropriate algorithm is applied, with proper parameters that allow to exploit the temporal dependency.

Another aspect that is expected for the dataset to fulfill is identical distribution. Therefore, all samples of variable X need to come from the same distribution [2]. This assumption is closely related to the concept of process stationarity. Unfortunately, especially for financial datasets, one needs to handle non-stationary time series [1]. Violation of this assumption may have critical effect of the model's performance, since the model might not be able to generalize to the test data [1]. Therefore, in such case, one needs to undertake appropriate steps to develop a stable and trustworthy prediction system, which are reviewed in Section 3.3.

2.2.2. Artificial Neural Networks

Artificial Neural Networks are computational systems, inspired by the structure and learning mechanism of a human brain [35]. Such systems are built of units, called **nodes** or **neurons**. The directed connections existing between them are capable of transmitting a signal, which is a real-valued, initial node's output. Furthermore, these links are typically weighted, therefore, they can decrease or increase the passed number. Neurons are often grouped in **layers**, which can perform different computations and send information to the further layers. At a given iteration, each node, sums the input, incomming via the connections, maps it to a non-linear function, called an **activation function**, and passes it via the outgoing links [35].

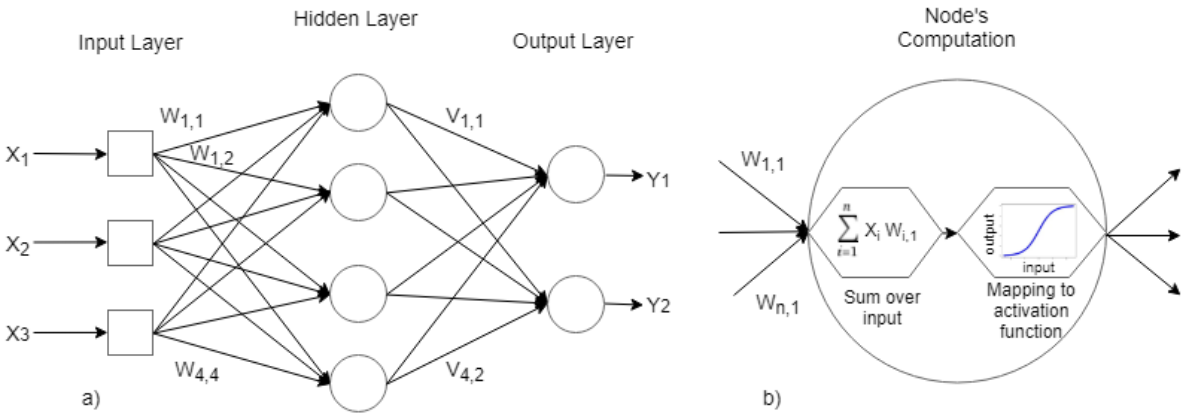

Figure 2.1: The simplified MLP architecture: a) overall structure, b) example node's computations

In order to better understand the ANNs, one should consider the architecture of a simple Multilayer Perceptron (MLP), illustrated in Figure 2.1 a). It consists of three types of layers: Input, Hidden and Output. The information flows only one way in the network, therefore it is also called feedforward. The initial set of nodes takes input X, with a specific number of features (in this case it is 3). Then, it sends the signal over the weighted connections w to the neurons of the Hidden Layer. As shown in the Figure 2.1 b), each of these units takes the sum over the incoming values, maps it to the activation function f_{act}, and passes it further through the network: Thus, the output $y_{n,j}$ of node m, in layer j, with n nodes in the previous layer is computed as follows:

$$y_{m,j} = f_{act}(\sum_{i=0}^{n} w_{i,j-1} y_{i,j-1}). \tag{2.21}$$

It is crucial to note that in DL architectures there is typically more than one hidden layer. Hence, a model is the deeper, the more hidden layers it has. Since each layer learns a more relevant representation of the

training data to solve the target problem, ANNs become increasingly powerful with size. However, the large networks have a tendency to overfit on the training data and fail to generalize to the unseen samples. Instead of extracting general patterns from data, they may start memorizing specific examples. Thus, one has to find a balance between appropriate capabilities and concurrent generalizability. Finally, the output layer presents the results, for instance, posterior class probabilities in the supervised classification problem [35].

This basic ANN has been widely applied in the past [35], yet it had certain limitations [41]. Those incapabilities leaded to the development of more sophisticated solutions, for instance, models with both: forward and backward information flows [30]. Whereas this section aims at introducing the ANNs, the further subsections describe in detail the architectures employed in this study.

2.2.3. Recurrent Neural Networks

Recurrent Neural Network (RNN) is a class of highly expressive Deep Learning architectures, which are designed to work with sequential datasets [32]. One of the first RNN models was developed by Elman [18], presented in Figure 2.2. It differs from the single hidden layer MLP architecture, by having a context layer interconnected with the hidden layer.

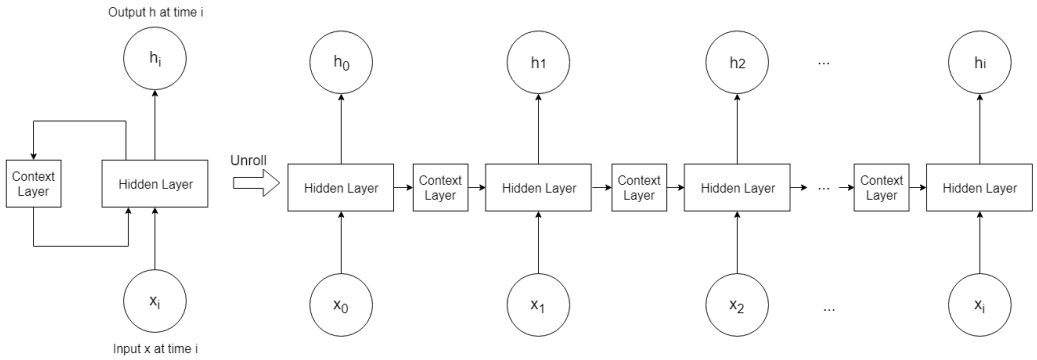

Figure 2.2: Simplified architecture of Elman's RNN before and after unrolling for a data sequence of i elements.

The context layer contains a state vector storing information about past samples. Thus, at each time step, it is merged with the input to produce the current output, and further, replaced with the current instance. The feature of RNNs that allows to store the information about past samples is called a memory.

However, one of the issues that the architecture suffers from are, so called, vanishing and exploding gradient problems [53]. The weights of the network are updated after unfolding the RNN over the training sequence. While the gradient of the loss function gets passed over the consecutive time steps, it tends to get either close to zero, or reach significant level. Therefore, the network may stop learning or change weights by very high values and consequently, never converge. Finally, it is difficult and in practice it is rare for the network to learn the long-term dependencies from the data [30], which was a motivation for developing the architecture described in the following subsection.

Long Short-Term Memory

Long Short-Term Memory (LSTM) is an architecture belonging to RNNs, developed by Hochreiter and Schmidhuber [30]. One of the main characteristics of this method is an ability to learn both, short- and long-term dependencies from the data. Moreover, it does not suffer from the previously described gradient issues, and therefore, this model is considered easier to apply [30].

The LSTMs characterize with a chain structure, where at each time step t, an LSTM module performs prediction h_t, given input x_t. A simplified architecture [51], applied on consecutive data samples is shown in Figure 2.3, and further, a legend of its building blocks is depicted by Figure 2.4.

The network's module consists of two horizontal 'conveyor belts' responsible for transporting the data, highlighted in the in Figure 2.5 a) and b). The former one (Figure 2.5 a)) takes current input vector x_t and concatenates it with the output from the previous time step h_{t-1}, both in range -1 and 1. Then, after transformations performed by the output gate (described further), it returns the sample's output, and passes it to the next iteration. This structure is responsible for managing the short-term memory in the network. In contrast, the upper horizontal belt, depicted in the Figure 2.5b), is not directly connected to the input and output of the network. It contains the **cell state**, a real-valued vector C storing the long-term memory, content of which, is carefully controlled by incoming structures called **gates** [51].

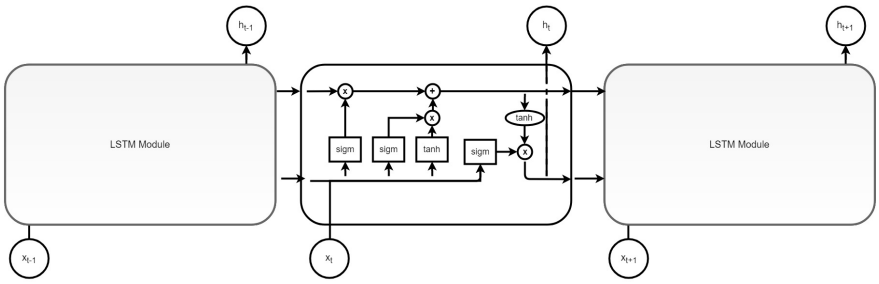

Figure 2.3: Simplified architecture of LSTM cells applied on consecutive data samples.

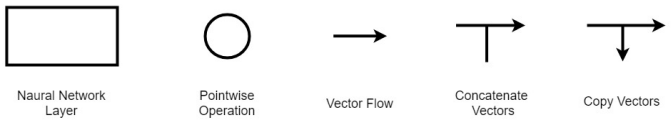

Figure 2.4: Legend showing the building blocks of the LSTM architecture diagrams.

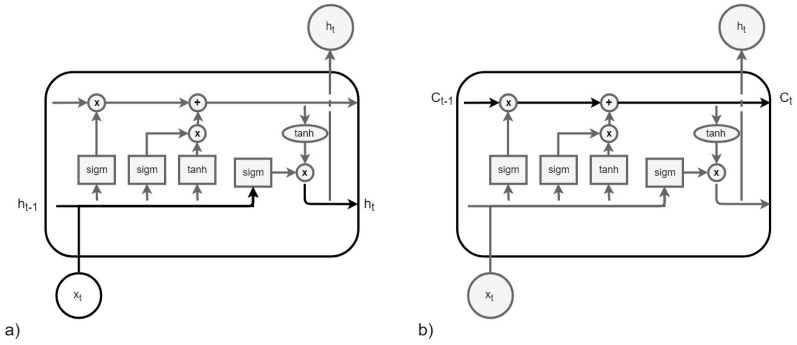

Figure 2.5: Important structures in the LSTM: a) part of the network responsible for short term dependencies, which concatenates input and the previous output b) component that contains information about long-term dependencies of the data.

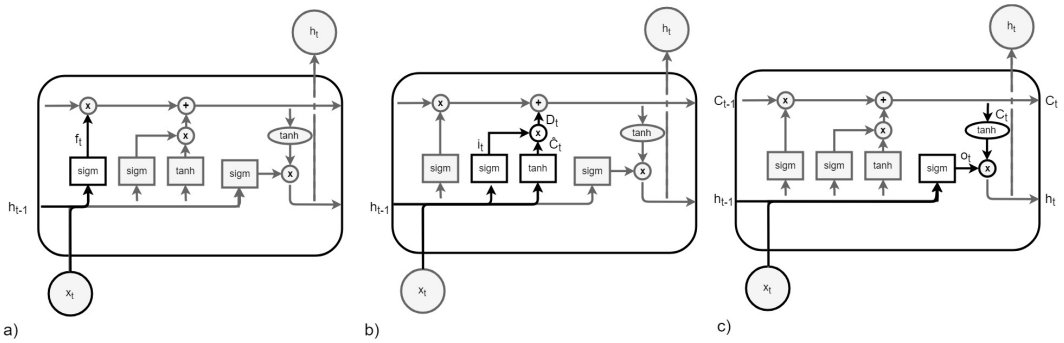

Figure 2.6: Three gates of LSTM: a) Forget gate, b) Input gate , c) Output gate.

A gate is a structure that regulates the information flow between the two belts described above. Gates make use of two types of layers of neurons: sigmoid σ and tanh, and each of these types has its specific purpose. The former, selects information to be erased from given vector V, by returning a vector of numbers in range of 0 and 1, which is further multiplied by V. The latter, transforms the data and keeps it in range between -1 and 1.

There are three gates in the LSTM network:

a) **Forget Gate**, at each iteration t determines vector f_t, namely the information that should be erased from the state vector C_t:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f). \tag{2.22}$$

Therefore, given the short-term memory h_{t-1} and current sample x_t, it determines which features of the cell state should be depressed, using weights W_f and the bias b_f of the hidden layer with sigmoid activation function.

b) **Input Gate**, weights the information to be transferred from the short- to the long-term memory and executes the transmission:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{2.23}$$

$$\hat{C}_t = \tanh(W_{\hat{C}} \cdot [h_{t-1}, x_t] + b_{\hat{C}}), \tag{2.24}$$

$$D_t = i_y \cdot \hat{C}_t. \tag{2.25}$$

The sigmoid layer with weights W_i and bias b_f calculates vector i_t, deciding which values of the vector $[h_{t-1}, x_t]$ should be passed from the bottom belt. Its multiplication by the vector $\hat{C}_t$, namely input transformed by tanh layer with weights $W_{\hat{C}}$ and the bias $b_{\hat{C}}$, results in the vector D_t, which is added to the cell state. Therefore the cell state C_t at iteration t iteration is computed as follows:

$$C_t = f_t \cdot C_{t-1} + Dt. \tag{2.26}$$

c) **Output Gate**, determines the outcome of LSTM network h_t at iteration t:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \tag{2.27}$$

$$h_t = o_t \cdot \tanh(C_t). \tag{2.28}$$

Based on the the vector $[h_{t-1}, x_t]$, the sigmoid layer with weights W_o and bias b_o decides, which information from the cell state is relevant for the current output and calculates the vector o_t. Then, it computes the output h_t by multiplying o_t by the cell state C_t, transformed by the $tanh$ activation function.

There are many variants of the LSTM, which differ from minor details [32], to major aspects e.g. Gated Recurrent Unit (GRU) [14] combines Forget and Input gate into an Update Gate. All in all, the LSTM architecture and its alternatives are applied in the state-of-the-art solutions to various supervised learning problems, such as language modeling, translation, speech synthesis and many more [27]. In this thesis, is it applied to forecast the future price of Bitcoin in chapters 7 and 8.

2.2.4. Word Embeddings

Text data may contain valuable information, however, it is not that straightforward to use in a data-driven application. Therefore, before applying supervised algorithms to learn the rules from it, one has to perform text processing to extract the relevant information for a given problem, which may be a time consuming task [57]. Luckily, there is a branch of techniques that can automatically process the unstructured text and represent it in a systematic manner, collectively named **word embedding**. The task of such algorithms is to extract a pre-specified sized vector of real numbers from text. Although there are various methods to achieve that [57], the recently developed Word2Vec [47][46] and Doc2Vec are especially interesting [37]. There are several reasons for their attractiveness. Firstly, the models can be trained on a vast amount of data in a reasonable time [47]. Secondly, the algorithms learn to encode many patterns and linguistic regularities, with no prior information about the language. Surprisingly, a well trained model maps the input according to the complex semantic and syntatic text relationships. Thus, words or sentences with similar meaning lay closer to each other in the vector space, while distinct ones further [37]. Moreover, the encoded patterns correspond to linear translations, for instance the result of the calculation *vector('Madrid')-vector('Spain')+vector('France')* is the closest to vector of word *Paris* [47]. In this case, the model recognizes that both Paris and Madrid are the capital cities, while Spain and France are the countries that they lay in.

Further subsections describe in detail the Word2Vec and Doc2Vec algorithms, which are employed to extract semantic features from text data in chapter 5.

Word2Vec

Word2Vec is a model that takes a single word or terms surrounding it in form of bag-of-words as an input, and maps it to a numerical representation in N-dimensional space [47]. Thus, by applying the model on the raw text data of t words one can extract t vectors of real numbers with N elements each.

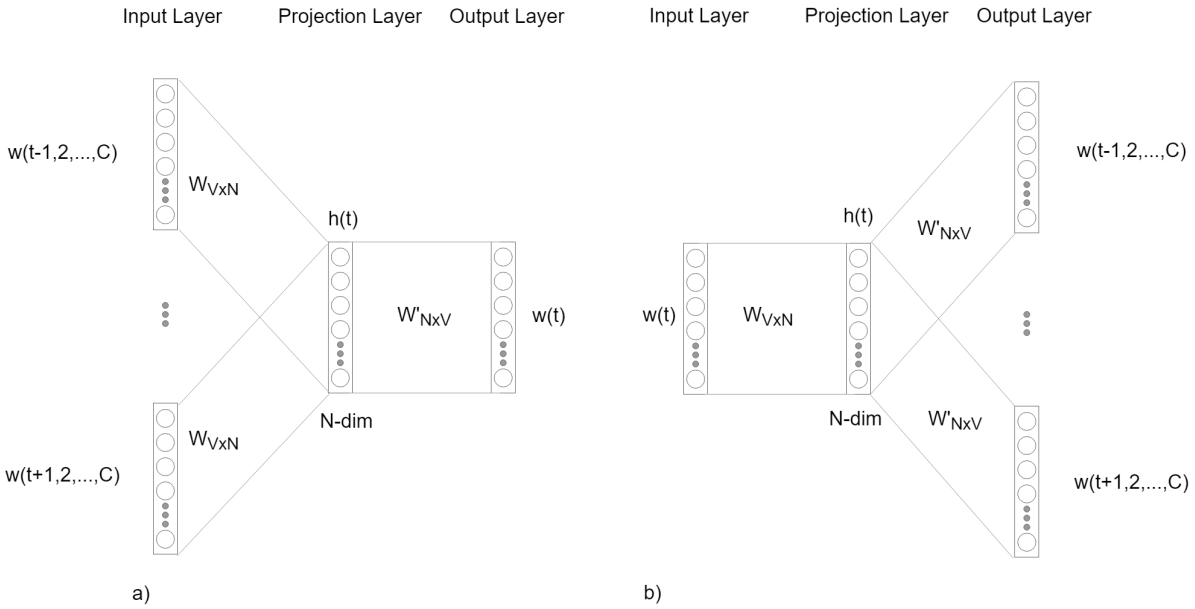

Figure 2.7: Two architectures of Word2Vec models: a) CBOW, b) Skip-gram.

There are two main architectures of Word2Vec with different properties, namely Continuous bag-of-words (CBOW) [46] and Skip-gram [47], both depicted in Figure 2.7.

The CBOW model (Fig. 2.7 a)) consists of three fully connected layers: Input, Projection, and Output, as well as the links' weights are stored in the W and W' matrices. During the training phase, the model takes a sequence of words, called a paragraph, and iterates over them, trying to predict the current word given C preceding and C following items. More formally, it maximizes the average log probability of the items in the document. The input layer takes the surrounding of the target in form a V-dimensional vectors, where V is equal to the size of the dictionary of all words available in the training corpus. These vectors consist of zeros and a single value of one at the position, corresponding to the represented item's dictionary index. Each unit in the Projection layer takes an average over the incoming weighted input, yet, unlike typical Hidden layer's elements, it does not map the result to the activation function. Finally, the output is computed by summing over the incoming data to the Output layer and mapping it to the softmax activation function [46]. Thus, the generated V-dimentional vector can be interpreted as posterior probabilities of any word in the dictionary being at position t. Moreover, the role of the Projection layer is to produce a representation of the input data that allows for a better discrimination of the target. Consequently, once the training is completed, the network is converted to calculate the word embedding, by the removal of the Output layer together with the incoming links. Therefore, the output vector h is equal to the value of Projection layer's neurons at time t.

In contrast, the Skip-gram model (Fig. 2.7 b)) predicts the C preceding and C following words, given the item at position t. Nevertheless, the maximized loss function is still the average log probability, yet of the surrounding words [47]. The rest of the architectural details remain the same as in the CBOW model. Overall, the main difference between the Skip-gram and CBOW is that the former focuses on representing the context in which a given word is used, while the latter infers the fitting word given the context. Since Word2Vec provides the representation for each word in a sequence, in order to use it as a document descriptor, one needs to combine the resulting vectors. This can be done, by for instance, averaging the word vectors for the given paragraph [13]. However, the drawback of merging these vectors is not taking into consideration the order of averaged values.

Doc2Vec

Dov2Vec model is a upgraded version on Word2Vec, in which the applied modification of the architecture allows to compute a single descriptor for the entire document, called a paragraph vector [37]. In addition to that, it takes into consideration the words' order in a sequence, which may have a crucial effect on the actual meaning of the sentence, and consequently, the output.

The architecture described in this section is Paragraph Vector-Distributed Memory (PV-DM) [37], see figure 2.8. The task performed by the network at the training time is a prediction of the next word given a

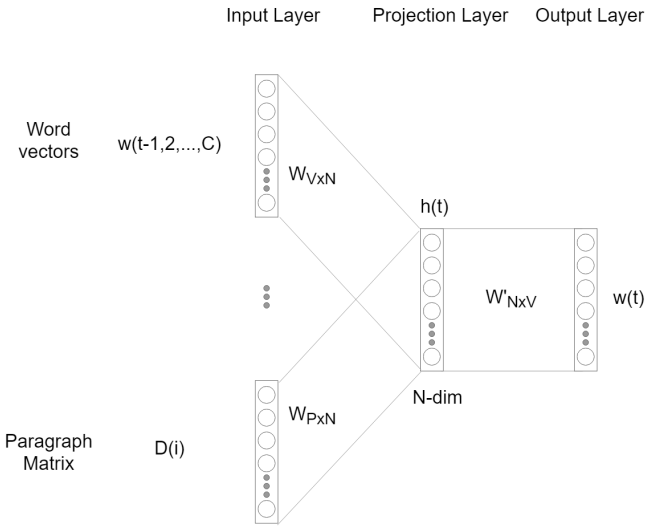

Figure 2.8: The architecture of PV-DM

sequence of C preceding words, thus, similar to the Word2Vec model. However, it also incorporates the information about the current document, as the additional context information. More formally, in the training phase the algorithm receives P documents which consist of V distinct words in total. The method iterates over each paragraph i and at each time step t, it tries to predict the next word. The input words are passed as vectors, similarly as in Word2Vec, together with the, so called, paragraph matrix D, not to be confused with paragraph vector. The matrix is a P dimensional vector of mainly zeros, with a single value of one at the document's index i that is shared for a given paragraph and differs between documents. The rest of the network remains the same as in the previously described CBOW model.

Once the training is completed, for an unseen sample, the model inserts a new element to the matrix D, representing the current paragraph. Subsequently, it initializes a neuron in the Input layer and a set of weights connecting it to the Projection layer. Therefore, the size of new D is $P+1$ and W is $VxN+(P+1)xN$. Further, the network performs a training iteration on the newly added document, yet modifying only weights of the currently joined links. Finally, the output paragraph vector is calculated by multiplying the paragraph matrix by the weights of its outgoing links[37]. Thus, the paragraph embedding can be interpreted as the context information specific for a current paragraph, that allows for an improved prediction of its words.

3

Literature Review

3.1. Cryptocurrency prediction

One of the first studies in the field of cryptocurrency forecasting has been conducted by Madan and Saluja [40]. The authors exploited over 25 market and cryptocurrency specific features, and using ML algorithms, predicted the sign of the price change for different granularity levels. The results for daily value direction of change prediction reached impressive 98.7%, while the ones for the next 10 seconds and 10 minutes were in range of 50-55%. Hence, the former setting is much simpler, since the data points are spread over longer period of time, which levels the unexpected value fluctuations. It is crucial to mention that the authors applied linear and tree-based classifiers, which do not take advantage of the sequential nature of the dataset. Moreover, the high accuracy achieved for daily forecast, may be influenced by a low size of the test set and an ongoing price uptrend in that time. However, the authors did not provide sufficient information to assess the trustworthiness of these numbers.

Subsequently, Georgoula et al. [23] studied the relationship between the price of Bitcoin and the sentiment of Twitter posts about the currency, as well as multiple variables related to economy and technology in general. The factors that are positively correlated with the value are the positive to negative sentiment ratio of tweets, number of Wikipedia search queries for Bitcoin and some other cryptocurrency specific features. In contrast, the negative association has been observed for global economic variables, e.g. USD to euro exchange rate and Standard and Poor's 500 market index. Interestingly, the research suggests that the features derived from online activity of the public, such as, Wikipedia and Google Trends queries as well as Twitter comments, are valuable predictors of Bitcoin price.

This observation is further confirmed in the research of Matta et al. [42], who tested cross-correlation between volumes of tweets or Web Search media results, and the value of the currency. The most powerful predictor found is based on the number of Google queries for phrase 'Bitcoin'.

Greaves and Au [26] exploited features specific to Bitcoin network topology, while forecasting the next hour price. Not only variables such as net flow per hour and mining speed was used, but also the three most influential owners of the currency were identified and their trades were tracked to improve the system. Then, the authors trained regression and classification ML models, with best results of 1.94 mean squared error (MSE), achieved by the Linear Regression, and 55.1% accuracy, achieved by the feed-forward Neural Network. As in the previously conducted research by Madan and Saluja [40], the applied algorithms do not take into consideration the sequential nature of the data. Moreover, the regression error measure employed is not be comparable to other works in the field, since the range in which the price fluctuates has significantly changed over time.

Further, McNally [44] investigated the use of algorithms, designed for sequential data, to predict Bitcoin price. The author employed the data from multiple exchanges and Blockchain specific features. Then, RNN, LSTM and ARIMA models were applied to forecast the exact future value of Bitcoin. In addition, the model's output was converted into one of three classes, namely, price up, price down or no change, to allow for investigation of both: regression error and classification accuracy. The lowest root mean squared error (RMSE) of 0.0545 was achieved by the RNN and the highest accuracy of 52.78% by the LSTM. When it comes to the ARIMA, it achieved significantly worse results, especially when it comes to the RMSE, which reached 0.5374. It is neccessary to note that the results are not comparable with the related work in the field, due to the sensitivity of the regression metric to the change of fluctuation range over time. Even if authors of other works

would employ the RMSE measure to test their systems on data from different time period, it would be difficult to distinguish whether they managed to reach superior results.

Bin Kim et al. [10] analysed how shifts in overall sentiment of users' comments from online forums indicate the fluctuations of three major cryptocurrencies. Therefore, in case of a significant drop or increase in positive or negative polarity classes of comments, the price prediction was made accordingly to the change. The results indicate high accuracy in binary classification, namely 79.57% for Bitcoin and around 72% for the other currencies. Thus, at the cost of prediction frequency, one may perform much more reliable forecast, by using user generated data from online forums only.

Stenqvist and Lonno [62] investigated the correlation between the positive or negative sentiment Bitcoin related tweets fluctuations and its price change change. The authors tested how the volumes of these messages changed over consecutive time steps and further, in case of the difference exceeding a threshold, they make a forecast according to the sentiment. Interestingly, by employing this simple methodology, they achieved a reliable 2 hours forward prediction with 79% accuracy. For higher values of the threshold, the accuracy increased, yet the frequency of forecasts strongly decreased. Therefore, the paper presents a straightforward methodology for high confidence prediction, however, at the cost of the amount of returned results.

Strati [63] analysed the Bitcoin price time series in terms of presence of the economic bubbles. Using the SADF test, he detected several rational bubbles between January 2010 and May 2017. Interestingly, the largest one started at the beginning of 2017 and continued until the end of the test period. Moreover, the author has shown that the detected events coincide with the bursts of Bitcoin related Google search volumes, indicating increased public interest accompanying the explosive growth of the price.

To sum up, the field of Bitcoin price prediction is young, yet much work has been devoted to studying the indicators of changes of the cryptocurrency value. There are many research challenges, such as, non-stationarity of the market data or a large variety of features that can be employed for more effective price behaviour modelling. However, the text data extracted from online sources has been processed only using Sentiment Analysis [23] [42] [10] [62]. As shown by [50] in their review of the text mining methodology for stock market prediction, there are several different feature extraction techniques successfully employed in stock market prediction. The authors underline that extraction of semantic variables and understanding of topics discussed in text becomes more popular recently and brings advancements to the field. Thus, applying these techniques in the context of Bitcoin price prediction may, not only boost the results, but also, provide more insightful analysis of factors that influence the value. This thesis employs these variables, therefore, subsection 3.2 reviews methodology for extracting semantic features.

In addition, most of the studies focus on building and testing a prediction system [40] [26] [44] [62] [10] or finding correlation between variables and the Bitcoin value [23] [42] [62]. There is a research gap in terms of comparison of systems with different feature sets and investigation of the robustness of built models to events such as economic bubbles. Not only most authors use different evaluation metrics, for instance: MSE [26] or RMSE [44], but also these results are incomparable since the Bitcoin value experienced a major growth over the past years. Therefore, consideration of the time series non-stationarity may lead to finding performance metrics that would enable comparison of these systems, even if they are tested on data from different periods of time. Finally, Strati [63] detected a major economic bubble driving Bitcoin price in 2017. One should research how well do the prediction systems deal with this phenomenon. Hence, section 3.3 considers the means of dealing with time series non-stationarity and the explosive bubbles.

3.2. Semantic feature extraction

Semantic features describe the basic conceptual components of the meaning of given text, namely, they represent its message. Such features may have different forms, from a collection of words that are most representative for the context, through vectors of probabilities of the text describing each of the defined topics, to a single semantic topic assigned to the sample. The variables are extracted based on a collection of documents, which is referred to as a corpus.

One of the dominant approaches to deal with semantic feature extraction is supervised **text classification** [3]. In the setting, one requires a single or multiple labels per document, which indicate the category or multiple categories to which the item belongs [3]. The interpretation of these classes may be, for instance, the type of the document or the topic it describes. Therefore, depending on the annotation, one can describe different semantic or non-semantic context. Agarwal and Mittal [3] describe the steps performed in the text classification process. Firstly, the samples need to be pre-processed, by, for example, tokenization, stemming and stop-words removal. Then, one extracts features from text, in the form of vectors, which can

be processed by a ML algorithm. Finally, based on the corpus and the labels, one trains a classifier. However, the annotated datasets are rarely available for some tasks, especially when the data is streamed from online portals or scraped from websites. Moreover, the annotation process is demanding in terms of recognition of the classes to be discriminated and time required for humans to assign these classes to each sample in the corpus. Consequently, a large number of approaches exploit, further described, automatic unsupervised techniques to extract semantic features, which overcome the requirement of labels for each sample in the corpus.

The field of **semantic modelling** aims at transforming the input text into a condensed form, representing the most important information in terms of its meaning [13]. A crucial characteristics of the domain is the use of **language models**, namely, unsupervised or semi-supervised techniques, which learn the relations between words in the documents [13]. The remainder of the section describes the most important algorithms for semantic modelling and how one can evaluate such approaches.

Term frequency - Inverse document frequency (tfidf) is a statistical measure, which weights the importance of each term in a document [13]. It is based on the term frequency $tf(t,d)$ of given word t in the document d, multiplied by the inverse document frequency $idf(t,D)$, namely a statistic, which penalizes terms that are common for the entire corpus D [13]. Therefore, the words with highest $tfidf(t,d,D)$, appear multiple times in the document d and rarely across the corpus D. The statistic is often used to select the most important words that are specific to a given document [13].

Scott et al. [58] introduced the **Latent Semantic Analysis (LSA)**, also called Latent Semantic Indexing (LSI), which automatically extracts n latent topics from the corpus. The method exploits the assumption that there is an underlying latent structure in the word usage [58]. This means that, terms used in the same way across sentences, have a related meaning as well as documents having associated semantic use similar words. The technique uses Singular Value Decomposition (SVD) to decompose the term document matrix A into three matrices $A = U\Sigma V^T$. U represents location of each term in the topical vector space, V^T maps each document to the latent vector space, while Σ indicates the importance of each extracted topic. Therefore, the method describes each document as a vector, which lays close in the latent space to the documents with similar meaning. However, the SVD is computationally intensive and extracted topics are often difficult to interpret, since the vectors lay in the artificial space [13].

Blei et al. [12] formulated the **Latent Dirichlet Allocation (LDA)** that automatically extracts a number of topics from the corpus and represents each document as a probabilistic mixture of the derived subjects. The topics consist of words and their weights, indicating their importance for given subject. Thanks to the representation, one can easily analyse and interpret the results. However, the LDA model training process is slow for large corpus and requires specification of the number of topics to be extracted beforehand, which limits its flexibility [13].

Mikolov et al. [47] introduced the **Word2Vec** model (described in detail in section 2.2.4.1, which computes the word embeddings. Thus, from each word in the document one can generate a multidimensional numerical vector, which represents its semantics [47]. Even though the representation is difficult to interpret, the cosine similarity between terms or documents express the similarity of their meaning [13]. In addition, the model can learn, based on a vast amount of documents in a reasonable time [47].

Next, Le and Mikolov [37] formulated the **Doc2Vec** (described in section 2.2.4.2), which generates the document embeddings. Unlike the Word2Vec, it computes a meaningful vector representation of the entire paragraph (document). Furthermore, it shares properties regarding results interpretability and computational efficiency with Word2Vec.

Campr and Ježek [13] provides an overview of the semantic modelling methodology and evaluates the techniques using a restaurant reviews dataset [13]. The authors had 150 pairs of text summaries annotated, by three annotators, in terms of the semantic similarity between texts. The assigned scores ranged from 4 (completely equivalent), to 0 (different topics). Further, they tested whether the representation, computed by the described above models, correlated with human intuition. Therefore, they calculated the Pearson's correlation coefficient between the cosine similarity of vectors generated from the reviews of each pair and their annotated similarity. The results indicate that Word2Vec and Doc2Vec reached the highest value of the statistic, 0.46 and 0.66 respectively, while tfidf, LSA and LDA scored below 0.17. The research suggest that the words and document embeddings provide he automatic document semantic representation, which successfully imitates the human estimates.

3.3. Prediction in a non-stationary environment

The section reviews the methodology for dealing with non-stationarity of dataset. Once the property is detected, one should consider converting it to a form that the statistical properties of each variable remain over time. Ostashchuk [52] elaborates on such methodology that can be applied on a difference stationary time series, namely differencing. The technique transforms each element of a variable X, by taking a difference between the current value and the past one with the lag l. One can also apply the relative differencing, which divides the outcome of the previous computation by the subtracted number. The former approach is efficient in case of linear uptrend or downtrend, while, the latter, is more appropriate if the trend is dependent on the overall value of an asset, for instance, the time series grows between 1% and 5% at each step. Finally, when the logarithmic return rates are experienced, one should perform the differencing of logarithm of the X variable. In general, the main drawback of the transformation is the requirement of understanding the nature of non-stationarity and assume that given behaviour remains over time, which does not always hold.

The previously described technique allows to transform each variable to stationary form. However, another crucial aspect of non-stationarity is, so called, **Concept shift**. It refers to a situation, in which the relationship between the predicted variable and the extracted features change over time [45]. In classification tasks the phenomenon can be illustrated by shifting optimal class decision boundary. Thus, even though one ensured stationarity of each feature in non-stationary environment, it is still necessary to limit the negative impact of concept shift.

The evolution of relation between variables is often caused by certain context factors. If these factors are not available while making the prediction, one has to do with the hidden causes [64]. In such cases, it is often possible to extract data describing the change determinants from the external sources. Turney [64] reviews the five heuristic strategies for handling contextual features that make the model more robust to the concept shift. The described solutions range from simple ones, for example, expansion of the primary variables with the context by merging, to more complex, for instance, making adjustment of the initial model's forecast based on the context.

However, additional variables are often not available. In such cases, one can detect the abrupt changes in the data and actively prevent the accuracy deterioration. Hence, a model is trained on the initial dataset and employed to forecast the future, until a shift in the target data distribution is detected. Basseville and Nikiforov [8] analyse the statistical tests that allow for the discimination. Then, one may replace the current model with the one trained on the most recent data. Gama et al. [22] applied such methodology, by using several ML algorithms, on eight datasets with different properties. It is shown that the approach works well independent from the type of ML algorithm used and allow to maintain its accuracy, in case of concept shift.

On the contrary, one may act passively, by continuously retraining the model with the most recent data. Widmer and Kubat [66] introduce multiple frameworks exploiting the procedure. In general, a model is trained on a portion of the most recent data, employed on a fixed number of further samples. The authors highlight that such constantly revised decision making model may, in general, be robust to the concept shift. Moreover,, in this approach the balance between time efficiency and robustness depends on the frequency of updates.

A similar framework called an online learning [5], does not require reintializing the model at each step. In this setup, a ML algorithm trains the model on the initial part of the dataset, and then, for each next sample, it performs the prediction and a number of training iterations on the new object. Thanks to the continuous updates with the most recent data, the program replaces unnecessary historical knowledge. Anava et al. [5] applied ARMA algorithm in an online learning setup, to forecast several time series with different properties, for instance, non-stationarity or containing a correlated noise. It appears that the learning framework deals with the concept shift efficiently, yielding high robustness compared to other settings. However, an issue of online learning is the normalization and scaling of the samples streamed 'online'. Since, some algorithms require input in range between -1 and 1, for instance LSTM [30], new samples cannot be simply fed to the model, because they may exceed the maximum and minimum values in the dataset. Especially, the vast increase of Bitcoin price causes online learning problematic to apply.

Another group of approaches, dealing with concept shift, is based on embeddings, namely a set of models that find consensus of the prediction by voting. Elwell and Polikar [19] introduce a method that trains a new classifier on each consecutive batch of the data, and adds it to the collection. Then, it performs a dynamically weighted majority voting among the developed models to perform the forecast. This way, the old classifiers have minor influence on the results, while the ones based on the new samples, major one. The advantage of this approach is the balance between tracking the currently relevant knowledge and maintaining the most stable data dependencies, by keeping older models.

Finally, Mendoza et al. [45] proposed a method for prediction under the concept shift, based on segmentation of the historical data. The main assumption is that the novel events to some extent resemble the already registered ones in the past. Thus, concise data segments are detected in the training set and compared with the ones in the target data. By means of dynamic time warping alignment function, the nearest historical unit is found and used to make the prediction. Hence, under the assumption that history gets repeated in the future, one is able to deal with non-stationary time series. Unfortunately, in case of Bitcoin time series, there are major events that are unique, for instance, significant and explosive price growth in November 2017, which may pose a difficulty while applying the described method.

To sum up, there is a variety of techniques for dealing with non-stationarity of the variables and concept shift. However, it is crucial to set up methodology that meets the requirements of the application and takes into consideration the properties of available features. Therefore, the Chapters 6 and 7 elaborate on the techniques applied in the thesis, to deal with non-stationarity of the dataset.

4

Data Collection

Having introduced the crucial background knowledge and reviewed related literature, one should perform tasks related to this study. This chapter describes the program that collects the data required to answer the stated research questions. It will be used to continuously stream and scrape the data available online. Thus, following sections define the high-level architecture of the program, describe some of the important implementation details and analyse the raw input gathered from each source.

4.1. System setup

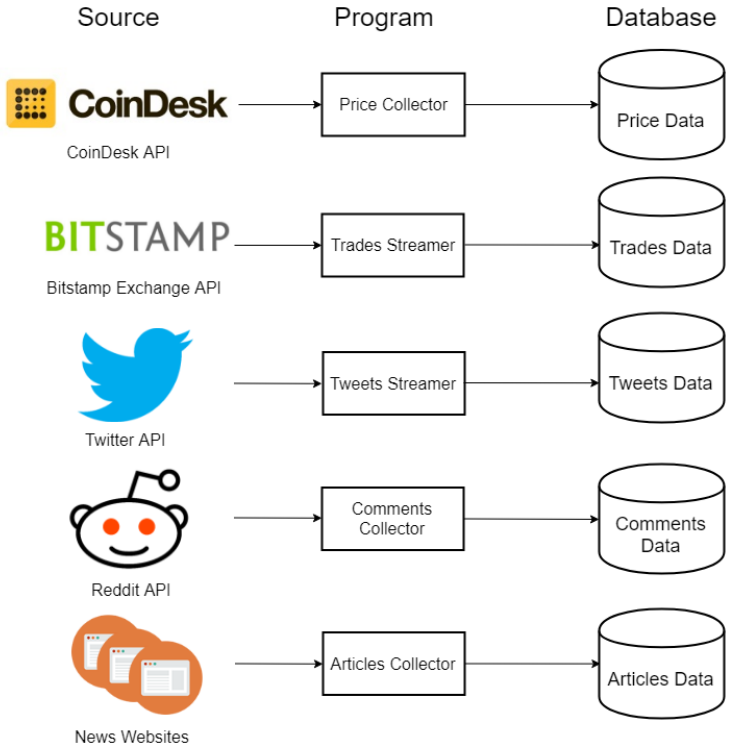

Figure 4.1: Simplified architecture of data collection system

This section presents the high-level design of a data collection program. Figure 4.1 illustrates the architecture of the program, which consists of 5 parallel processes. The left column represents the data sources, the middle one depicts the program's name and the right one the database, to which the raw and unprocessed output is stored. Most of the data is collected using Application Programming Interfaces (API), which are responsible for data exchange between a given platform and the implemented applications. They define a number of protocols and tools that a programmer can use to extract structured content. The collection

process is handled by the implemented software that is divided into two types: streamers and scrapers. The former run continuously over the period of data collection, fetch the current records and store them in the database. The latter programs are launched periodically to collect historical data from a given source. This distinction is made due to the API's restrictions of some platforms, which do not allow, or limit scraping of the content.

This research is based on the data collected between 28.09.2017 and 07.02.18. The following sections elaborate on the implementation details of the illustrated pipelines and analyse the raw data collected.

4.2. Averaged price data

CoinDesk is a company that provides digital media and information services, as well as organizes events related to cryptocurrencies and blockchain technology [62]. The firm hosts a website that enables use of the CoinDesk Bitcoin Price Index API[1]. This program allows to programmatically extract closing price of Bitcoin in USD, gathered with minutely frequency and averaged over several major exchanges. Thanks to the usage of the mean value, the measure is more reliable and robust to the volatility of a given market. Therefore, a simple script is implemented that executes an API request for data of each minute in a specified time range. The minutely records of price and exact date are concatenated and added to the Price Database.

4.2.1. Data analysis

The collected price data consists of over 205 thousand minutely records with the second decimal place precision. In order to ensure the data quality, a program, which tests whether there are any records missing over the considered period of time, has been implemented. Interestingly, there are several gaps over the time line, which range from minor issues, for instance, one or two consecutive records missing, to a single, one hour gap. The latter has been identified as a problem with Daylight Saving Time backward change on 28th of October. Even though the data is extracted directly in Coordinated Universal Time (UTC) format from Coindesk, the problem appears. The issue may have a critical effect on the prediction due to time misalignment of the further concatenated research variables, and therefore, it needs to be addressed in the data preparation process.

4.3. Market data

Cryptocurrency exchange markets are platforms, in which the users can trade financial assets. According to Spooner et al. [59], at any time, the value of an asset depends entirely on the actions of investors, namely bids and asks. The bid is a purchase offer of a given amount of an asset at a certain price and the ask is a sales proposal, defining the amount available and the price for it. All the currently available asks and bids are listed in the order book. A trade (or transaction) occurs after the buyer and seller agree on the price and the volume. Finally, the space between the lowest ask and highest bid is called a spread and it represent the current value of the asset and its liquidity [59].

In order to access the transaction data as well as the order book states, one can use the Bitcoin Exchange Market Data Feed Handler (BitcoinExchangeFH)[2]. The application records the price depth and trades of a given exchange. It streams in real time every trade (price and volume) and the state of the order book at the time of the transaction, namely 5 lowest asks and 5 highest bids (price and volume). The thesis employs the program to stream the data from the Bitstamp [3] Bitcoin to USD exchange.

4.3.1. Data analysis

In the streaming process approximately 130 million trades and order book states have been recorded. Figure 4.2 illustrates the daily volumes of the collected data. As shown, the program generates vast amount of data, ranging from 600 thousand to 1.3 million per day.

Each sample has 22 features, 2 related to the trades and 20 describing the order book state. The data has high quality, however, requires extraction of more condensed features, which is addressed in the next chapter.

[1] Powered by CoinDesk: https://www.coindesk.com/api/
[2] https://github.com/Aurora-Team/BitcoinExchangeFH
[3] https://www.bitstamp.net/

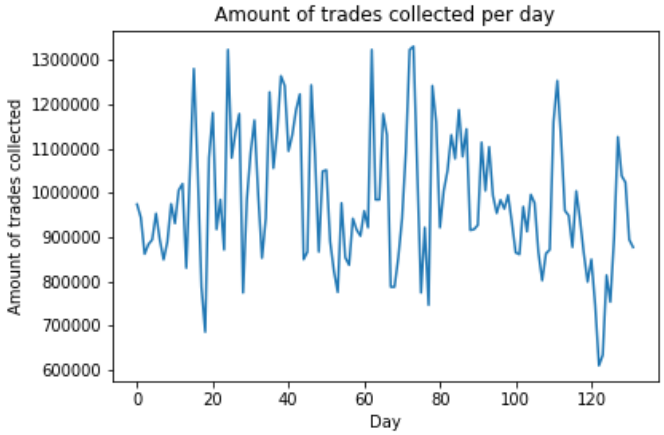

Figure 4.2: Daily amounts of raw market samples collected for the thesis.

4.4. Tweets data

Twitter is an on-line micro-blogging platform that enables its users to publish short messages, called tweets [62]. Their content can be discussed, liked and reposted (retweeted) by other users. Moreover, tweets contain up to 140 characters and may include metadata, which indicates either the context of the message, starting with the '#' ('hashtag') operator, or reference to other user, specified by '@' followed by the username [62].

The platform facilitates data exchange, by development of the Twitter API[4]. The interface provides users with many tools, such as, quering or streaming data functionality. In order to collect tweets related to Bitcoin, the 'Tweets streamer' program is implemented It continuously fetches the currently published tweets in English, which contain at least one of the specified filters. These keywords include equivalents of the cryptocurrency's name, namely 'bitcoin', 'btc' and 'xbt'. Lastly, the program stores the textual content and the timestamp of the recovered tweets in the database.

4.4.1. Data analysis

The 'Tweets Streamer' has gathered over 48 million tweets over the period of 133 days, on average 363 thousand per 24 hours. However, by inspection of the figure 4.3, one can observe that daily amount of messages is highly volatile. Moreover, the peak of the social media activity is accumulated during the period of time, when Bitcoin had the highest price, which indicates the flurry of the community.

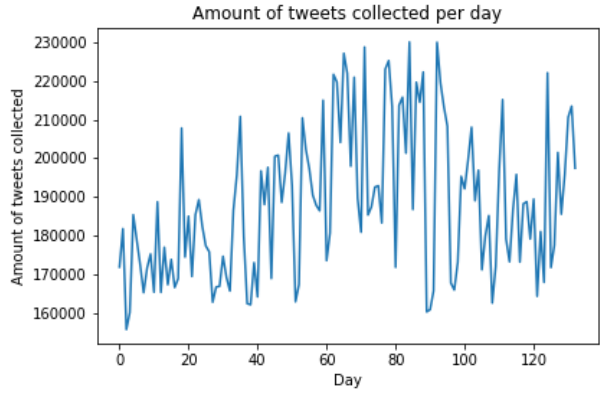

Figure 4.3: Amount of raw tweets collected for the thesis.

Investigation of the tweets' text indicates several issues that need to be handled in the processing step. Automatically generated advertisements are a large part of the content. They are mostly posted in groups by bots, thus, they have similar structure. However, the programs generating such tweets publish them from various accounts and randomly replace certain words, making it harder to filter the noise out. Table 4.1 depicts an example of such advertisement.

[4]https://developer.twitter.com/en/docs

Noisy tweets generated by an automatic bot
Crowdsales DOT online #ICOs #ITOs. #BitCoin #tech #mexico #wavesplatform #tucson #token
Crowdsales DOT online #ICOs #ITOs. #privateequity #saltlakecity #FinTech #gamecoin #btc
Crowdsales DOT online #ICOs #ITOs. #minneapolis #btc #zloadr #saintpaul #baltimore #waves #uk
Crowdsales DOT online #ICOs #ITOs. #btc #sydney #singapore #kickstarter #honolulu #Ethereum
Crowdsales DOT online #ICOs #ITOs. #abraaj #losangels #funding #cleveland #AVCJKorea #btc
Crowdsales DOT online #ICOs #ITOs. #ITOs #london #venturecapital #icotracker #bitcoin #bonus
Crowdsales DOT online #ICOs #ITOs. #platform #softbank #austin #blockchain #BitCoin #tampa

Table 4.1: Example of noisy tweets generated by an automatic bot

When it comes to the content of the collected tweets, there are several elements included in the text that require removal or processing:

- 'RT'- the term is automatically added at the beginning of the message and it indicates that the item is a retweet. Yet, it does not contribute to the meaning of the tweet, so it should be removed.
- Username references- indicate the links between a tweet and a user. However, this thesis disregards the interaction of the community and focuses on the subject of the text.
- Hashtags- underline the context of the message and may play a crucial role in analysing the topic of a tweet. Thus, these tags need to be kept and exploited.
- Links- associate the message with its Twitter address or an external website. Potentially, scraping the information from the outside resources would extend the semantic information, yet such opportunity does not appear often. Therefore, only the body of the tweet is considered and the links can be removed.
- Emoticons and emojis- express the emotions using textual signs or small images. However, they do not contribute to the description of the discussed topic.

All in all, the further applied Twitter data processing has to implement noise reduction as well as text processing to remove the unnecessary content.

4.5. Comment data

Reddit is a platform which allows its users to share, rate and discuss content. Posts are organized by topic into forums called subreddits. Members of a given forum can generate submissions, comment on them and vote on their importance. Therefore, the subreddits can be visualized as a tree with initial branches being posts, and further, the conversation-trees made of comments [61].

The Reddit API[5] allows to programmatically extract the content of the platform. Moreover, PRAW [6] is a Python package that simplifies the collection, by providing a vast array of high-level functions, building on top of low-level API's protocols. The 'Comment collector' makes use of the library to traverse the posts of several subreddits related to Bitcoin, namely 'r/Bitcoin', 'r/Btc' and 'r/BitcoinMarkets'. From each of the submissions published on the forums, the program extracts and adds to the database all the comments, issuers' usernames and timestamps. However, the information about the discussed post is not included, because of its varying structure, being either a link, a text field or an image. Hence, only the users' discussion on given topic can be extracted in unified way from all the submissions.

4.5.1. Data analysis

The 'Comment Collector' has accumulated over 1.1 million samples, published in the specified period of time on the considered forums. Based on figure 4.4, which represents the daily gathered amounts of comments, one can observe that there is a high variability of forum activity between consecutive days. The highest value on the graph is around 400% higher than the lowest one, and, as before, the peak of the discussion lays close to the date of top Bitcoin price. Therefore, the variables derived from comments' volumes may require transformation that would prevent the negative effect of non-stationarity, for instance, differencing. The issue is addressed in section 6.1.

[5] https://www.reddit.com/dev/api/
[6] https://praw.readthedocs.io/en/latest/

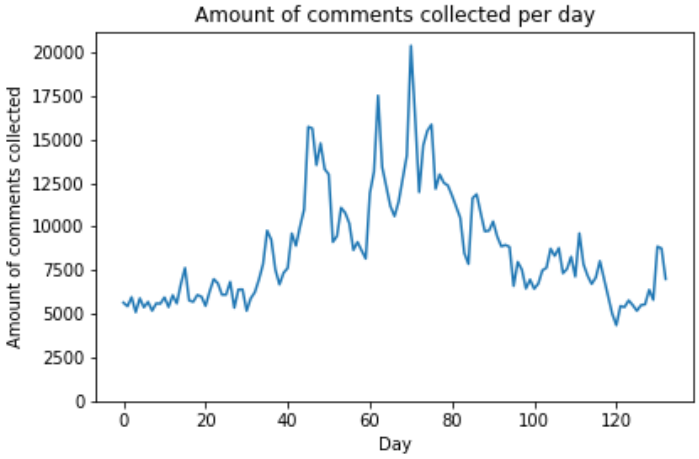

Figure 4.4: Amount of raw comments collected for the thesis.

Considering noise in the data, it is difficult to filter out the comments generated by bots, because they are mostly not focused on advertising, but rather, on extending the discussion. Such messages can be recognized by occasionally included 'I'm a bot, bleep, bloop', embedded in the text, or word 'bot' enclosed in the issuer's username. However, administrators of the subreddits regularly review the content, ban users who generate spam and delete their posts.

Furthermore, the collected comments are mostly written in informal language, with many profanities, grammar, spelling and language mistakes, and even elongated words, for instance, 'okaaaay', instead of 'okay'. Hence, it is necessary to apply text processing approach, which will extract the context of these comments, regardless of these issues. Other, easier to tackle, issues are links in the comments and emoticons and emojis, which need to be discarded.

4.6. Article data

The last type of data considered in the thesis are online articles, published by major news websites. However, only a narrow set of portals analyse the Bitcoin related issues, mainly those that focus on financial and technological topics. Moreover, in general, websites embed the text into its structure, making it more difficult to extract. The paragraphs of the text are most often built into the HTML code of the page, interchanged with advertisements and unrelated content. Hence, to acquire the data, one needs to perform web scraping, namely, inspecting the page's source code and programmatically obtaining the appropriate content from the structured item [48].

The 'Article Collector' uses the Scrapy[7] package to perform scraping of news from websites. Firstly, the program finds direct links, source portals, titles and timestamps of the most recent articles related to Bitcoin from a number of the cryptocurrency dedicated RSS feeds [8] [9] [10] [11]. This step allows to access structured information about the most up-to-date as well as older texts without the requirement of crawling the news websites. Furthermore, the software exploits a number of, so called, 'spiders', namely bots programmed to scrape the suitable content from given portal's HTML page structure. Table 4.2 illustrates the portals, for which dedicated spiders have been developed, as well as the main topics discussed by the websites. Thus, for the gathered links, the 'Article Collector' checks whether it can scrape its content, based on available spiders, and if so, it performs the action. Finally, the title and the content of the articles are merged into a single text field and the data is stored in the database, together with the previously gathered features.

4.6.1. Data analysis

There are approximately 11 thousand articles collected in the specified period of time, much fewer than for any other data type. Figure 4.5 depicts the daily amounts of gathered samples, which deviates significantly

[7] https://scrapy.org/
[8] http://feed.informer.com/digests/I2GGLAVR70/feeder.rss
[9] http://bitcoin.worldnewsoffice.com/rss/category/1/
[10] http://bitcoin.worldnewsoffice.com/rss/category/2/
[11] http://www.bitnewz.net/Articles/

Website with dedicated spider	Description of portal's content
https://www.bitcoinmagazine.com/	News related to blockchain and cryptocurrencies
https://www.businessinsider.com	Business and finance related topics
https://www.coindesk.com	Articles about blockchain, cryptocurrency and their price analysis
https://www.cointelegraph.com/	News related to cryptocurrencies and markets dealing them
https://www.ccn.com/	Topics associated with cryptocurrencies and their price analysis
http://www.livebitcoinnews.com/	News about Bitcoin and other cryptocurrencies
https://www.marketwatch.com/	Stock market news
https://www.newsbtc.com/	Articles related to cryptocurrencies and their price
https://www.themerkle.com/	News associated with blockchain and cryptocurrencies
https://www.zerohedge.com/	Economy and company related topics

Table 4.2: Portals crawled and overall type of news that they present.

from day to day, however, over time the signal remains in a range between 20 and 120. Moreover, the content of the samples is not always directly related to Bitcoin, but rather to companies that invest in blockchain-based solutions or other tokens. Even though these news may have influence on how public perceives cryptocurrencies in general, the thesis focuses on a single coin. Therefore, the items, in which Bitcoin is not mentioned, should be removed from the dataset.

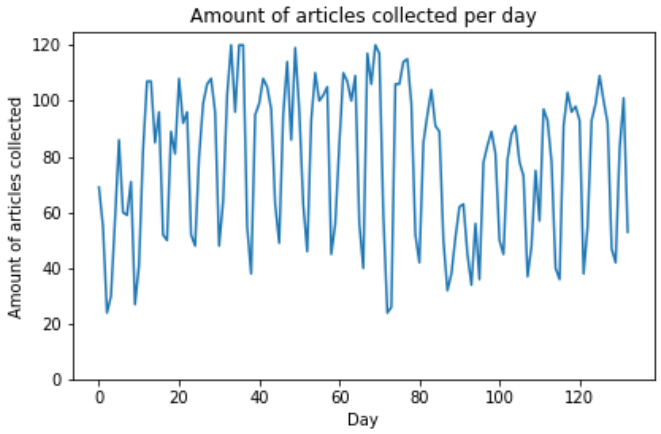

Figure 4.5: Daily amounts of raw articles collected for the thesis.

As far as the quality of the content is concerned, the articles are mostly long and written in a business language. Hence, processing to be applied on this data source consists mainly of noise filtering and feature extraction.

5

Data Preparation

The previous chapter has described the steps performed in the data collection process. For each type of data, we analysed the accumulated samples in terms of their content, quality and the issues that need to be handled before performing a prediction.

This chapter describes the data processing steps, applied in order to convert the collected raw data from several sources and of various types, into a multivariate time series dataset. Thus, we take the aggregated event data, filter out the noise, preprocess the samples, extract the adequate features and finalize the process with generation of samples that summarize all the events appearing in a given time interval. Based on the representation, the predictive algorithms selected in chapter 7 will be used to forecast the Bitcoin price.

Another vital aspect that this chapter addresses is the initial part of the RQ 1.2, namely how to extract the semantic features from the gathered text data. Later in the chapter, we will propose the suitable methodology that assigns each text sample to a group of texts discussing similar topics. Thanks to extraction of these features, one can analyse in chapter 8 whether the trained prediction system copes with drastic market changes more efficiently using data from online sources.

5.1. System setup

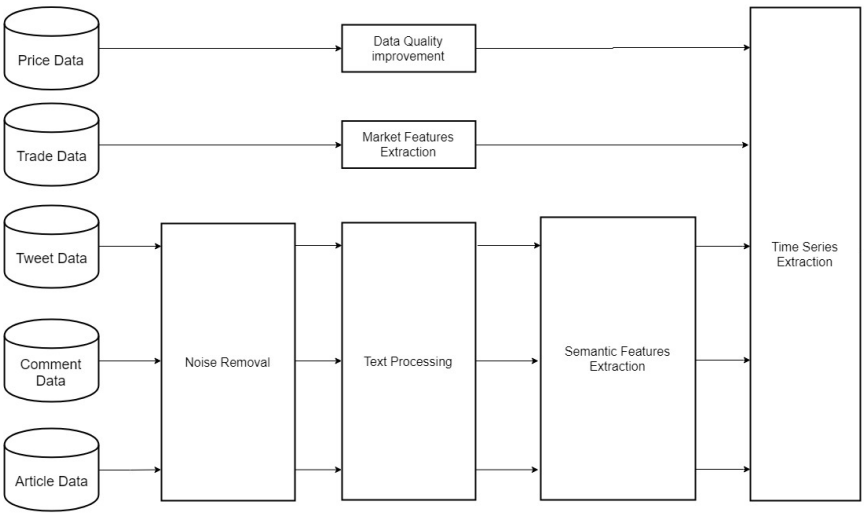

Figure 5.1: Simplified architecture of Data Preparation

The Figure 5.1 depicts the architecture of the data preparation system implemented in the thesis. At first, the program applies processing to each data type individually, to handle already identified issues with the collected samples. Further, the routine for extracting semantic features from text is defined and applied on tweets, comments and news. The final steps consist of extracting the time series from the gathered samples.

5.2. Averaged price data

Even though there are missing items in the Bitcoin price time series, they contribute to only 0.05% of the total number of samples and mainly occur in the initial part of the dataset. Since the way that these gaps are filled has a negligible effect on the performance of the prediction system, it is not necessary to use a sophisticated methodology for the imputation. Thus, the 'Data Quality Improvement' segment of the pipeline resolves the issue using the linear interpolation [49]. Given two points (x_1,t_1) and (x_2,t_2), namely the last sample before the gap and the first one after, the technique finds the coordinates of the line connecting them and generates the missing samples based on the missing timestamps.

5.3. Market data representation

One of the oldest and most popular representations of the market data is candlestick charting [16], see an example in Figure 5.2a [1]. Its goal is to illustrate the market changes that appeared over a specific period of time, for instance an hour or a day, using a so called candlestick. We make use of this representation to summarize the events encountered on the exchange market for a given time interval. Therefore, the following description defines what the candlestick charting is and how to compute it from the collected order book states. The section also discusses how to extract features summarizing the gathered trades over a specific period of time. Hence, this section aims at defining methodology specific to the market data, used in the process of the window extraction, in section 5.5.

Figure 5.2: The visualization of a) candlestick price chart and b) elements of a candlestick

Figure 5.2b depicts the structure of the candlestick. Its main part is the body, which illustrates the spread between the price at the beginning of the period (open price) and the one at the end (close price). It can have either a green or a red color, if the close price is higher or lower than the open value respectively. The remaining components of the candlestick are the shadows, which depict the highest and the lowest value of the asset over the period. In this thesis, the price of Bitcoin is defined as the mean of the lowest ask and the highest bid, representing the middle of the spread in the order book.

Feature Name	Description
Close Price	The price of the asset at the end of the window.
Open Price	The price of the asset at the beginning of the window.
Low Price	Lowest price of Bitcoin over the window.
High Price	Highest value of the cryptocurrency over the window.
Num. of Transactions	Total number of transactions recorded over the window.
Volume of Transactions	Aggregate amount of Bitcoin traded over the window

Table 5.1: Overview of features extracted from the raw market data

Therefore, in the 'Market Features Extraction' step of the system setup in the figure 5.1, one can derive several variables for a given window of time. Table 5.1 presents an overview of the features extracted from order book states and trades and their computation procedure for a given window of time. The first four variables are components of the candlestick, while the other two aggregate the trade specific data. We will employ this

[1] The Figure is based images from http://www.chart-formations.com/stock-charts/candlestick-charts.aspx , accessed on 17.05.2018.

Hashtags	#freebitcoin, #livescore, #mpgvip, #makeyourownlane, #footballcoin, #sexservice, #bitindia , #mytracknet, #antminer, #win
Wods	{antminer}, {subscribe}, {entertaining}, {free}, {android}, {tokensale}
Bigrams	{dot online}, {dot com}, {ready shipping}, {current price}, {earn bitcoin}, {free trading}, {android app}, {join moneypot}, {join our}
Trigrams	{start trading bitcoin}, {satoshis best kept}, {join daily signals}, {hash rush update}, {in real estate}, {invest in our}, {we are accepting}, {join the ico}, {create your own}, {mined at height}

Table 5.2: Hashtags and n-grams that indicate noisy tweets

methodology in the section 5.5 to summarize the market events for windows of time, while generating the time series.

5.4. Text data processing

As already shown in sections 4.4, 4.5 and 4.6, there are several processing steps that need to be performed on the text data from each source before the extraction of semantic features. At first, we need to filter out the noisy samples from each dataset, and then, we can transform each text item into an ordered list of words that represent the context of a given sample. Moreover, text data from each source has its certain properties and issues that need to be handled individually. Finally, once the samples from Twitter, Reddit and news websites are converted to a suitable form, we will be able to extract the semantic features from them in the next section.

The following subsections describe pipeline of steps applied to the text data and highlight the implementation details customized for each text data source. We will refer to the tweets, comments and articles in this section as data types.

5.4.1. Noise Reduction

The aim of this step is the removal of noisy samples. It is a crucial process, since as proven by Colianni et al. [15], by discarding the irrelevant items, one can get a significant error rates reduction. However, the definition of such objects and the methods used to detect them, differs for each data type. Thus, they are considered separately below.

For Twitter data, noise can be defined as bot generated content or posts that are not related to Bitcoin. In order to get rid of these samples, this thesis applies the methodology exploited by Stenqvist and Lonno [62]. Using this approach, one manually identifies text features of the noisy data and apply filtering in order to remove these samples.

Thus, 5000 randomly sampled tweets from the first two months of the data are analysed. The goal is to identify the n-grams and hashtags, which repeatedly appear in the noisy samples. Table 5.2 displays the identified tokens.

Number of tweets	Before Filtering	After Filtering	Percentage Removed
Over first two months	20666660	9382663	54.6%
Overall	48263395	**23156777**	**52.02%**

Table 5.3: Reduction of the noisy tweets

Further, the 'Noise removal' segment of Twitter data pipeline discards the tweets with the identified hashtags and n-grams. Table 5.3 illustrates the amount of samples remaining after the reduction. Clearly, a major part of the dataset is irrelevant and overall the program deletes 52.02% of the data. As shown, the removed percentage for the initial two months is slightly higher, due to the manual analysis of the data. However, the defined filters maintain high performance over time, even though the bot generated content evolves.

When it comes to the Reddit data, noise is defined as any comment from the target subreddits that is generated by a bot. In general, the selected subreddits concern Bitcoin and moderators make sure that any topic discussed there relates to the cryptocurrency. The issue of advertisements appearing in the data is also addressed by Reddit users, who can report spam or delete such content. However, some of the noisy comments remain in the dataset and most of them are not obvious to detect, even by a human reader. There are two straightforward indicators of a bot generated content:

- Presence of a 'bot' string in the username
- Addition of the 'I'm a bot, bleep, bloop' string into the comment.

Thus, the noise removal module discards the items having any of these markers. This leads to 3.5% reduction of the number of comments, from 1105869 to 1067161.

In case of articles scraped, the database contains data on several topics related to blockchain, cryptocurrencies and economy. However, only a fraction of them is directly related to Bitcoin, the rest is treated as noise. This thesis considers any article that discusses the cryptocurrency, even if it is not as the main topic. In order to filter out the noise, the text samples that do not contain the word 'Bitcoin' are discarded. This way, the program removes 37% of the samples, leaving 6943 articles.

5.4.2. Text Processing

The aim of the 'Text Processing' step of the pipeline is to prepare the unstructured text samples for the semantic features extraction. Therefore, the program needs to discard the redundant information, namely any part of the text that does not facilitate learning the context of the sample. Only the most informative words should be kept that allow extraction of the text's meaning. Thus, the method takes as input the raw text data and outputs the sequence of words that represent the subject of the sample. The program needs to follow the steps listed in the algorithm 1.

Algorithm 1 General text processing scheme

1: Convertion of text to lower case to unify the words written with and without capital letters, yet having the same meaning, for instance, Bitcoin and bitcoin are consolidated.
2: Handling of the source specific issues in case of twitter and reddit data, described further in this section.
3: Substitution of each digit with a placeholder '_'. This way, the numbers consisting of the same amount of digits are unified, for instance, records discussing Bitcoin price contain '_____', '____','__k', '_k'.
4: Replacement of any irrelevant non-alphabetical signs, with a space, so that the sentences are merged into a single textual body. While doing this, most of the emoticones are discarded, unless they contain letters.
5: Tokenization of text, namely converting a sentence into a list of tokens representing the words in their original order [60].
6: Stop words removal, the program discards the tokens representing the most common terms in the English language, which do not contribute to the context of the text, such as, 'and', 'or', 'nor' [60].
7: Removal of the tokens consisting of a single letter, which mostly are formed due to appearance of characters in the emoticones.
8: Deletion of the sample, in case it does not contain sufficient context representation. In order to extract semantic features later in the section 5.4.3, we need a sufficient number of tokens present in each analysed sample. Thus, we set a threshold for the minimum number of tokens per item equal to 5, which slightly reduces the overall dataset size, but improves the quality of further extracted features.

Due to high variability of data collected between sources, there are specific issues that need to be handled differently for tweets and comments. News on the other hand, characterize with high quality of the text and do not require any modification of the general procedure.

The Section 4.4.1 describes the issues specific to Twitter data that need to be addressed while in this section. To achieve that, the general text processing procedure is modified. Algorithm 2 presents the steps taken and highlights the customizations with regards to the algorithm 1.

The examples of Twitter posts in their raw form and after performing the transformations are depicted in table 5.4. The identified issues with text are handled by the text processing step, which makes the the resulting list of tokens contain only terms that represent contextual information. Similarly, the general procedure is modified for data from Reddit and presented in the algorithm 3.

The comments often contain profanities, grammatical errors and elongated words. There are techniques that allow us to deal with these issues [60], however, much simpler approach is taken. The tokens that do not appear often enough in the entire dataset, namely at least 10 times are discarded. This way, only the relevant words remain in the dataset, for which the further applied feature extraction algorithm has enough samples to learn from. The sample of Reddit comments before and after the current processing steps are illustrated in table 5.4.

Algorithm 2 Text processing steps performed on Twitter data

1: Convertion of text to lower case.
2: **Removal of emojis**, which have a characteristic format in the utf-8 encoding, for example '\xF0\x9F\x98\x81' represents an icon with smiling face.
3: **Removal of usernames** following the '' sign, since it does not contribute to the contextual information of the tweet.
4: **Deletion of the 'rt' string** from the beginning of some tweets.
5: **Removal of links**, by filtering the words containing 'http' or 'www.'.
6: Substitution of each digit with a placeholder '_'.
7: Replacement of any non-alphabetical sign, **other than '#' or '_'**, with a space.
8: Tokenization of text.
9: Stop words removal.
10: Removal of the tokens consisting of a single letter.
11: Deletion of the sample, in case it does not contain sufficient context representation.

Original Text	Extracted Tokens
Bitcoin eliminates bullshit from the world. It starts by exposing it, and the sight is not pretty. Hence all the butthurt.	{bitcoin}, {eliminates}, {bullshit}, {world}, {starts}, {exposing}, {sight}, {pretty}, {hence}, {butthurt}
RT BitcoinWrld: Russia Officially Legalizes Bitcoin! Finally https://t.co/OABAFzTMDn	{russia}, {officially}, {legalizes}, {bitcoin}, {finally}
#bitcoin SegWit2x/NYA support overview (https://t.co/8PZcgnsHSh alternative) https://t.co/JPyPE3iPbW	{#bitcoin}, {segwit_x}, {nya}, {support}, {overview}, {alternative}
Bitcoin Climbs Over $4,000/Ethereum/Bitcoin Cash/Neo/China: https://t.co/Ox8rxMUsyn via YouTube	{bitcoin}, {climbs}, {over}, {_}, {___}, {ethereum}, {bitcoin}, {cash}, {neo}, {china}
RT PlatformNotary: The role of NTRY Token. Proof of Stake? https://t.co/81RMhe1Iwi #blockchain #erc20 #ProofOfStake #ethereum #bitcoin #I...	{role}, {ntry}, {token}, {proof}, {stake}, {#blockchain}, {#erc20}, {#proofofstake}, {#ethereum}, {#bitcoin}, {#i}

Table 5.4: Examples of raw tweets and the tokens extracted by the text processing procedure

Algorithm 3 Text processing steps performed on Reddit data

1: Convertion of text to lower case.
2: **Removal of emojis**, which have a characteristic format in the utf-8 encoding, for example '\xF0\x9F\x98\x81' represents an icon with smiling face.
3: **Removal of references to other subreddits** following the ' r/' sign.
4: **Removal of usernames** following the 'u/' sign.
5: **Removal of links**, by filtering the words containing 'http' or 'www.'.
6: Substitution of each digit with a placeholder '_'.
7: Replacement of any non-alphabetical sign with a space.
8: Tokenization of text.
9: Stop words removal.
10: Removal of the tokens consisting of a single letter.
11: Dropping of the sample, in case it does not contain sufficient context representation.

Original Text	Extracted Tokens
i am interested to know the cypherpunks referenced in satoshi's whitepaper who oppose segwit/core.	{interested}, {know}, {cyberpunks}, {referenced}, {satoshi}, {whitepaper}, {oppose}, {segwit}, {core}
i don't see this momentum halting at 4k imhoedit: korean and japanese markets dumping	{see}, {momentum}, {halting}, {_k}, {imhoedit}, {korean}, {japanese}, {markets}, {dumping}
it's interesting how emergency situations are so standard for this software that users not only expect them, they are actually praised.	{interesting}, {emergency}, {situations}, {standard}, {software}, {users}, {expect}, {actually}, {praised}
how significant is the europe market on bitcoin price actually? i assume the big players are american and asian markets?	{significant}, {europe}, {market}, {bitcoin}, {price}, {actually}, {assume}, {big}, {players}, {american}, {asian}, {markets}
a) many view this as good news. b) it's all relative, who says people are selling because of south korea?	{many}, {view}, {good}, {news}, {relative}, {says}, {people}, {selling}, {because}, {south}, {korea}

Table 5.5: Examples of raw comments and the tokens extracted by the text processing procedure

5.4.3. Semantic Features Extraction

In this section we propose the methodology to extract semantic features from the three text datasets of tweets, comments and news, preprocessed by the previous sections. At the same time we address the first part of the **RQ 1.2**, which states as follows: 'how to incorporate the text data into prediction and how does it influence model's robustness to non-stationarity?'. Therefore, we try to transform the raw text data into a concise semantic representation, by for instance, classification of the topic, which can be further used in the window extraction process in section 5.5.

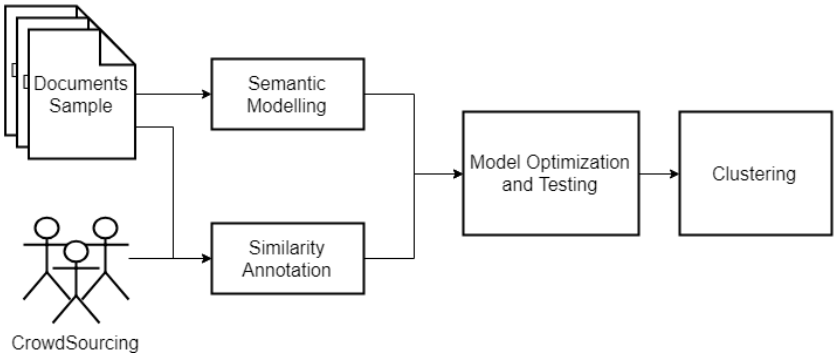

Figure 5.3: Pipeline for Semantic Features Extraction

The overview of the proposed pipeline for dealing with the task is depicted in figure 5.3.

First of all, the collected text data does not include labels indicating its subject, which eliminates the use of the supervised learning approaches for topic classification discussed in section 3.2. Therefore, one needs to discover the subjects that appear in the data either qualitatively, namely by manual inspection, or automatically, using the semantic modelling algorithms such as Doc2Vec or Word2Vec. This thesis applies semantic modelling techniques to generate a meaningful representation of the data. In general, the methodology analyses raw text documents and transforms them into output, which depicts the underlying relations between samples in terms of their subject [13]. The detailed discussion of the taken approach follows in section 5.4.3.1.

In order to evaluate which approach works best and optimize the parameters, one can perform the procedure applied by Campr and Ježek [13]. This process starts with a few annotators rating the topical similarity of randomly sampled pairs of text samples. Then, by measuring the correlation between the averaged similarity of pairs and the cosine similarity of their generated semantic representation, one can analyse which configuration imitates the human intuition the most successfully. Hence, the procedure is divided into the following steps: annotation of the data using Crowdsourcing, described in section 5.4.3.2, as well as Optimization of the model by maximizing the correlation, presented in section 5.4.3.3.

Finally, it is desirable to extract a concise topical representation of the data, due to the further applied window extraction. However, to make it possible to count how many, for example, tweets were posted on a given subject in the last hour, each text sample has to belong to a single group of topics. Kumar Rangarajan Sridhar [36] applies a clustering algorithm to find a natural grouping of word embeddings. This approach suits this framework well, since the performed optimization step ensures that documents with similar meaning have high cosine similarity. Therefore, a clustering algorithm that minimizes the cosine distance measure is chosen and applied on the data in section 5.4.3.4 in order to split the vector space into several groups, referred to as semantic clusters. Figure 5.4 illustrates how the text data is converted by the applied pipeline.

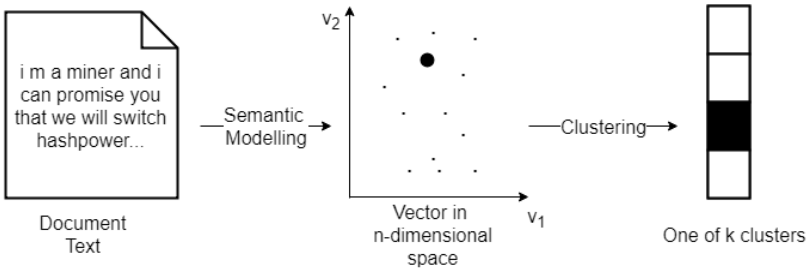

Figure 5.4: Text Data conversion steps during Semantic Features Extraction

The following subsections discuss the applied methodology for semantic features extraction, which transforms the raw text data into the concise topical representation. Firstly, the suitable algorithms for semantic modelling are considered. Secondly, the annotation process, using crowdsourcing as well as the optimization steps to select the best configurations are described. Lastly, the problem of assignment of each text to a single semantic group is addressed.

Semantic modelling

Section 3.2 presents an overview of the methodology used in the field of semantic modelling. Based on this review, this section considers the algorithms applicable in the thesis for each type of the collected text data.

There are 4 main approaches, currently applied in the field, to map the text to a the numerical vector space, namely LSA, LDA, Word2Vec and Doc2Vec. Unfortunately, the initial two are computationally intensive, which makes them problematic to apply on millions of records. Therefore, the techniques are not considered in the further experiments. On the other hand the Word2Vec and Doc2vec are efficient for large datasets and are reported as the most successful for the considered task [13]. Even though the mapping is not easily interpretable, the further applied clustering algorithm groups the related samples into a meaningful representation. Moreover, Word2Vec computes the embedding per each word, so for each document it generates a list of vectors, which need to be combined into a single one. This can be achieved by simply averaging the vectors [13]. One drawback of this approach is that it disregards the order of the terms in the document. However, the CBOW architecture of Word2Vec takes the arrangement of words into consideration while generating the embeddings, which to some extent prevents the negative effect of further applied averaging.

Therefore, the CBOW Word2Vec and PV-DM Doc2Vec models are used to extract the semantic representation from text. However, the variability of samples between the data types is high, for instance, tweets are short and contain numerous hashtags, while news are much longer and well structured. Hence the selected algorithms are applied on text data from each source separately. The sections 5.4.3.2 and 5.4.3.3 describe the process of data annotation and model optimization, namely selection of the parameters that work best for the domains of tweets, comments and articles.

Crowdsourcing

Crowdsourcing is the process of realizing given services, ideas, or generating content by requesting a group of people for contributions, particularly an online community, rather than traditional suppliers [67]. Currently, crowdsourcing has emerged as a popular and standard way to prepare and annotate datasets [67]. Online platforms like Crowdflower, Amazon Mechanical Turk etc. facilitate interaction between the employer (Researcher) and potential annotators (workers).

In this work, the process of annotating pairs of tweets, comments and articles is realized via Crowdflower [2], and is performed similarly to Campr and Ježek [13]. Three separate jobs are created, one per the type of the

[2]https://crowdflower.com

data, in which the worker gets a pair of randomly selected samples and rates the topical similarity between 0 and 4 as follows:

0 : Completely unrelated topics.

1 : There is some semantic relation e.g. both discuss Bitcoin or they both talk about an exchange market.

2 : They discuss similar topic, but there is a mismatch in an important element, the authors talk about price change of Bitcoin, one about its crash, while the other about its increase.

3 : They discuss the same topic, but different aspects of it, for instance, both authors comment on the upcoming major software update, but have different opinion on it.

4 : The samples are semantically identical.

There are 400 pairs of samples per data type, annotated by 3 users each. To get more reliable measure, the final similarity score is derived by taking the average of three available labels.

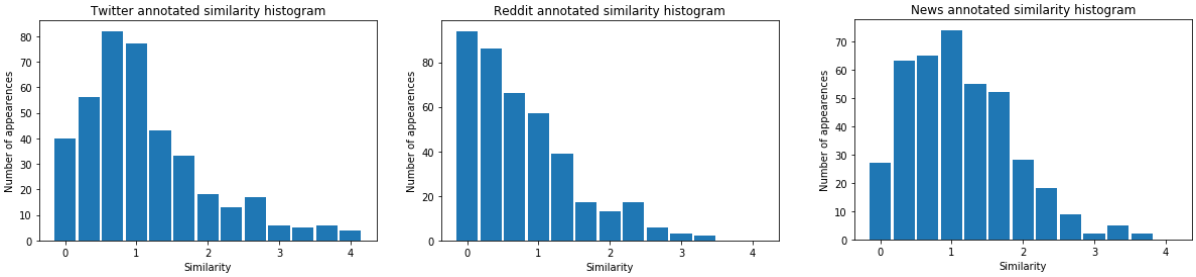

Figure 5.5: The histograms of the annotated similarity for datasets of Twitter, Reddit and news websites

Figure 5.5 illustrates the histograms of the resulting annotation. Clearly, most of the pairs of samples are unrelated or have weak relation, yet for tweets and articles there is mostly some topical similarity. This may be a consequence of the applied filtering step, in which, the samples that do not directly mention Bitcoin are discarded. On the other hand, Reddit comments are scraped based on selected subreddits, where users discuss a wide range of topics. This leads to higher number of pairs of unrelated samples.

Optimization and analysis

Once the pairs of samples are annotated, one can optimize the semantic models' training process. The aim of this step is to find, which algorithm and which parameter configuration works best for each type of data, namely correlates the most with the human intuition. In order to do that, each set of 400 annotated pairs is split into validation and test subsets, with 200 samples each. Stratified sampling [31] ensures that both collections have similar statistical properties. In general, validation sets are used to find the best performing models, while test sets confirm their results and ensure that there is no overfitting.

The optimization procedure starts with the training of the Word2Vec and Doc2Vec models with different values of the hyperparameters, on the data from each source separately. By employing grid search technique [9] one can effectively find the suitable setting. Table 5.6 lists the hyperparameters that are optimized, de-

Name	Description	Tested values
OutputDim	The size of the output vector	{10, 25, 50, 75, 100, 125, 150, 175, 200, 225, 250}
Windowsize	The number of terms used to predict the next word	{3, 4, 5}

Table 5.6: Hyperparameters considered in the optimization process and their values tested in the grid search procedure

scribes them and presents their considered values. Therefore, Word2Vec and Doc2Vec models are trained on 10 million tweets, 5 million comments or 10 thousand, articles for each combination of these parameters. Then, these models generate the embeddings for each document from the validation set and the program finds the cosine similarity of these vectors for the target pairs.

For two vectors a and b the cosine similarity is defined as:

$$similarity(a,b) = cos(\theta) = \frac{\sum_i^n a_i b_i}{\sqrt{\sum_i^n a_i^2}\sqrt{\sum_i^n b_i^2}}, \tag{5.1}$$

Data Type	Model	Parameters {Windowsize,OutputDim}	Val. Corr.	Test Corr.
Tweets	W2V	{3,10}	**0.50**	**0.49**
	D2V	{3,200}	0.19	0.19
Comments	W2V	{5,225}	**0.46**	**0.46**
	D2V	{4,10}	-0.08	-0.09
News	W2V	{5,150}	**0,48**	**0,50**
	D2V	{4,10}	0,36	0,35

Table 5.7: The results of the semantic feature extraction optimization process. The highest correlations for each data type are in bold print.

and the cosine distance is calculated by the formula:

$$distance(a,b) = 1 - similarity(a,b). \tag{5.2}$$

The resulting similarity ranges from -1, through 0, to 1, meaning exactly opposite, orthogonal and parallel vectors. Thus, it is desired that vectors of samples discussing related topics point in a similar direction, while others, the opposite.

In order to find the correlation between the model's output and human annotation, this thesis employs the Spearman's rank correlation coefficient [29]. In other work [13], the authors measure the Pearson's correlation, yet the method finds the linear relationship between two variables. However, in practice the assumption of linearity does not need to hold for a given dataset. Moreover, the Pearson's correlation is not as robust to outliers, which are evident in the figure 5.5. Therefore, the non-linear measure is used instead. For two variables, it computes a statistic between -1 and 1, where -1 represents perfect negative correlation, 0 stands for no correlation and 1 means the perfect positive correlation. In the optimization procedure, the target is finding the model that has the highest value of this coefficient on the validation set. The procedure is executed for tweets, comments and news separately. Finally, the tuned models are tested on the test sets to confirm their performance.

Table 5.7 illustrates the results of the optimization process. For each data type the Word2Vec model works significantly better than Doc2Vec. The tuned models correlate with human annotation between 0.46 and 0.5 on both validation and test sets. The results on the latter set confirm stable performance and deny the possible overfitting. When it comes to Doc2Vec the outcomes range from -0.08 and 0.36. The negative correlation for the comments may be caused by low quality of text, for instance, grammar and syntax mistakes, as well as, very high variability of topics discussed. In contrast, for high quality data such as articles, the method performs much better. However, Word2Vec deals with all the types of considered text effectively, therefore, it is used for the semantic features extraction.

Clustering

As shown in the previous subsection, the optimized semantic modelling algorithm transforms text input into a vector of numbers with the dimensionality specific to the given data type. A crucial property of this mapping is that vectors representing texts that share a given topic have, in general, higher cosine similarity, while those that are unrelated, have it lower. Therefore, clustering the transformed data based on cosine distance leads to finding groups of samples that are semantically similar.

K-Means algorithm is an efficient way to divide the vector space into k groups. [6]. The technique minimizes the within-cluster sum of squares, namely the sum of squared euclidean distances of each point in each cluster to the mean of points that belong to the cluster. The main advantages of the approach are its simplicity and efficiency, which make it applicable even for large datasets [6]. However, using different distance function, while minimizing the loss function may stop the algorithm from converging.

The Spherical K-means [6] is a modification of the K-Means algorithms, which minimizes the within-cluster sum of squared cosine distances. It takes advantage of the fact that cosine distance is equivalent to euclidean distance for two vectors normalized to the unit circle, so that their second norm is equal to 1 [6]. Therefore, the only difference between the algorithm and the K-Means is the normalization of the input data at the beginning, and the cluster centers at the end of each maximization step. The thesis employs the technique due to its efficiency and the use cosine distance metric.

The only parameter that has to be selected while applying the algorithm is the k, which defines the number of clusters extracted. In this work, it also determines the number of semantic groups, into which the data

Cluster	Top tf-idf terms	Interpretation
1	'username', '__', 'check', '#bitcoin', '___', 'btc','bitcoin', '____', 'price','set', 'ltc', '#blockchain', 'major', 'eth', 'hurdle', 'bitmain', 'trend', 'shift', '#bitmain', 'l_'	Bitcoin price analysis and
2	'username', '#bitcoin', 'ico', 'bitcoin', 'exchange', '#blockchain', 'btc', '#crypto', '#ico', '#btc', 'google', 'daily', 'payment', 'buy', 'earn', 'website', 'feels', 'gbp', 'basis', 'withdrawals'	Consideration of various tokens
3	'username', 'mining', '#bitcoin', 'bitcoin', '#blockchain', 'user', 'adding', 'consensus', 'disapproval', 'business', 'btc', '#btc', 'change', 'no__', '#its', 'rubbing', 'rich', 'china', 'miners', 'think'	Recent events regarding Bitcoin
4	'#bitcoin', 'username', 'bitcoin', 'government','come', 'blockchain','#india', '#blockchain', 'new', 'banks', 'india', '#bitindia', 'wave', '#btc', 'demands', 'succumbs', 'ransomware', 'county', 'alabama', 'payments', 'blocks', 'size', 'think', 'coins', 'use'	Banks' and governments' stance towards cryptocurrencies and blockchain technology
5	'bitcoin', 'username', 'company', '#bitcoin', 'morgan', 'fad', 'stanley', 'says', 'ceo', 'political', 'hates', 'loves', 'blockchain', 'china', 'just', '#ethereum', '#btc', '#blockchain', 'banks', 'finance'	Companies' stance towards cryptocurrencies and blockchain technology

Table 5.8: Analysis of the generated semantic clusters from tweets

Cluster	Top tf-idf terms	Interpretation
1	'__', '___', '____', 'bitcoin', 'btc', 'usd', 'reward', 'powered', 'stake', '_k', 'pool__', 'months', 'tippr', 'received', 'just', 'bearish', 'eth', 'time', 've', 'buy'	Discussion of the price level of Bitcoin
2	'bitcoin', 'just', 'use', 'wallet', 'like', 'don', 'coinbase','btc', 'key', 'bank', 'keys', 'people', '__', 'private', 'need', 'money', 'cash', 'make', 'want', 'fiat'	Consideration of means of safe storage of tokens
3	'don', 'people', 'bitcoin', 'just', 'like', 'know', 'think', 'really', 'btc', 'said', 'read', 'posts', 'crypto', 've', 'doesn', 'want', 'thread', 'way', 'didn', 'sub'	Expression of opinion and commenting on some events
4	'bitcoin', 'blockchain', '__', 'block', 'people', 'chain', 'btc', 'like', 'transactions', 'transaction', 'cash', 'miners', 'just', 'time', '_mb', 'blocks', 'size', 'think', 'coins', 'use'	Discussion of technical details of Bitcoin, such as block size, mining difficulty
5	'bitcoin', 'cash', 'core', 'segwit', 'people', 'satoshi', 'community', 'btc', 'isn', 'just', 'segwit__', 'development', 'chain', 's__', 'capacity', 'fork', 'project', 'support', 'time', '__'	Software development of Bitcoin

Table 5.9: Analysis of the generated semantic clusters from comments

is divided. Consequently, if the parameter is too low, the clusters may have too general semantic interpretation. On the other hand, setting it too high makes the further training of the prediction system more difficult, due to higher risk of overfitting. However, there is no quantitative statistic, which allows to measure how well the method works for different values of the parameter. Therefore, k is tuned qualitatively, by investigating the highest tf-idf valued terms in each produced cluster, interpreting the most often discussed topics for each clusters. The lowest value of k is selected, such that the groupings for each data segments the topics discussed.

The algorithm is applied on randomly sampled texts from each source separately, namely 50 thousand tweets, 50 thousand comments and 10 thousand news. After cluster analysis of the results for different values of k, the parameter is set to 5. Tables 5.8, 5.9, 5.10 illustrate the properties of produced clustering. Even though there is a high overlap between interpretation of the clusters, the grouping provides a general idea what kind of subject is described. The method works especially well for news, for which there are distinct and regularly published articles, for instance, technical price analyses. The clusters of Twitter data have similar properties and the grouping is straightforward to analyse thanks to the repeatedly appearing hashtags indicating tweet's context. On the other hand, the amount of topics discussed by Reddit users is very high, which makes the groups more general and negatively influences their quality.

All in all, the section partially answers the second research subquestion, by proposing a methodology for semantic features extraction, that fits to the requirements of the thesis. It automatically detects the semantic

Cluster	Top tf-idf terms	Interpretation
1	'bitcoin', 'cryptocurrency', 'people', 'new', '__', 'time', 'cash', 'blockchain', 'segwit__', '____', 'fork', 'money', 'price', 'currency', 'users', 'said', 'network', 'right', 'mining', 'miners'	Development of software behind Bitcoin and cryptocurrency mining related topics
2	'bitcoin', 'blockchain', 'new', 'bank', 'said', 'digital', 'currency', 'cryptocurrency', 'financial', 'exchange', '____', 'cryptocurrencies', 'central', 'referer', 'government', 'russian', 'market', 'currencies', '__', 'trading'	Banks' and governments' stance towards cryptocurrencies and blockchain technology
3	'bitcoin', 'price', '__', '____', '_____', '___', 'market', 'share', 'trading', 'cryptocurrency', 'time', 'new', 'bull', 'peak', 'bear', 'cash', 'high', 'mid', 'ethereum', 'news'	Bitcoin price analysis and trends description
4	'blockchain', 'platform', '__', 'token', 'tokens', 'new', 'ico', 'technology', '____', 'cryptocurrency', 'users', 'industry', 'based', 'company', 'use', 'bitcoin', 'market', 'ethereum', 'sale', 'exchange'	Other tokens and platforms based on blockchain
5	'___', 'price', '____', '__', 'bitcoin', 'support', 'resistance', 'usd', 'hourly', '_____', 'retracement', 'chart', 'rsi', 'fib', 'trend', 'break', 'high', 'macd','market', 'bullish'	Bitcoin price analysis, trends description.

Table 5.10: Analysis of the generated semantic clusters from news

groups, which correlate with human intuition, in three different text datasets, and further, allows for insightful analysis of the data. The resulting features extracted from Reddit are less descriptive, due to the low quality of text and broad discussion. However, for news and tweets, the method manages to find the underlying groupings of topics.

5.5. Window extraction

Window extraction is the process of generating a time series from a number of samples occurring irregularly over time [28]. A window is defined as a time interval, for which a given algorithm generates the summary of the time-stamped data. It is shifted through the dataset with equal distances, whereas a time step refers to a single forward shift [28]. As a result, this methodology computes a multivariate sequential time series from various types of data points (for instance tweets and trades), such that the predictive algorithms described in section 2.2.3 can be applied on the data.

Type of data	No. of features	Representation
Averaged price	1	Closing price at the end of the window
Order book states	4	Candlestick extraction
Trades	2	Moving sum over amount and volume of trades
Tweets	5	Moving sum over number of samples per cluster
Comments	5	Moving sum over number of samples per cluster
News	5	Moving sum over number of samples per cluster

Table 5.11: Feature extraction for each window from different data types

There are various techniques that summarize the data points encountered in a given time interval. As already discussed, a popular representation of the market data is a candlestick. For other types of data, one can employ techniques like moving average or moving sum [28]. The former takes average of the numerical samples that appear in a given window, while, the latter sums the elements. Table 5.11 describes the methodology used to extract the time series dataset from samples of each type.

A crucial aspect of window extraction is the choice of the suitable parameters, namely window size and the shift distance. With larger values of these parameters, one computes a time series with smaller amount of more general samples. On the other hand, for shorter window and shift, the dataset becomes larger and more specific. Moreover, one needs to take into consideration the type of prediction that is made using the generated dataset. For short-term prediction, a dataset of higher time granularity is preferred [40], while for long-term prediction, the samples mostly represent longer intervals of time [62][26].

In order to answer the stated research question, the thesis focuses on 15 minutes prediction of the averaged Bitcoin price. However, the window size is set to 5 minutes, to increase the size of the dataset and

enable training of more complex prediction models. Similarly, the shift distance is fixed to 5 minutes, due to negative effect of windows' overlap on the candlestick creation [28]. Thus, the resulting time series consists of approximately 38 thousand samples, with 22 features each, and training labels being equal to the value of the averaged price variable, 3 time steps ahead.

6

Time series analysis

In previous chapters, we have collected data from various sources, preprocessed it and extracted a multivariate time series. Once these steps are completed, one can analyse the sequential dataset and address the **RQ 1.1**, namely what are the properties of the Bitcoin price signal in terms of stationarity and speculative bubbles.

In order to do so, in section 6.1 statistical tests will be conducted to determine whether the data is stationary. It is a crucial property, which is assumed by most of the predictive algorithms. If the stationarity cannot be proven for some variables, an appropriate transformation will be applied to enforce their identical distribution over time. Further, in section 6.1.1, we will convert the non-stationarity features of time series using differencing and test again whether stationarity can be proven for the output of the technique.

Moreover, in section 6.2 the Bitcoin price signal will be analysed, in terms of the occurrence of economic bubbles, with special attention paid to detecting the time intervals of their occurence. These findings will allow for investigation in chapter 8 of how the prediction model reacts to the explosive price growth and the consequent crisis.

6.1. Stationarity

The section aims at verifying stationarity of the extracted multivariate time series. To achieve that, the Augmented Dickey-Fuller test for unit root with 95% confidence level is applied on each feature of the dataset. The detailed description of this test can be found in section 2.1.2. The significance level is set to 95%, because it serves a balance between high quality of results and difficulty of disproving the null hypothesis. The null hypothesis H_0 of the test states that there is a unit root present in the characteristic equation of the stochastic process generating a given variable, while the alternative hypothesis rejects the property and proves stationarity of the signal. Based on a sequence of samples, the method calculates the ADF_τ statistic and p-value it corresponds to. If the test statistic is below the critical value corresponding to the selected confidence level, then the null hypothesis is rejected. Thus, one can reject H_0 and confirm stability of the signal, if the derived p-value below 5%. Otherwise, non-stationarity of the time series cannot be disproven and one can assume that the data is non-stationary. The lag term, which defines the order autoregressive model used when conducting the test, is selected by minimizing Akaike's Information Criterion (AIC).

Table 6.1 presents the results of the test for each feature in the extracted time series. Clearly, all the variables representing trades and text data from Twitter, Reddit and news websites have p-value below 5%, which causes rejection of H_0 and proves their stationarity. On the other hand, the null hypothesis cannot be rejected for any of the variables related to Bitcoin price and therefore, one can assume that Bitcoin price signal is non-stationary. However, the further applied predictive algorithms assume stationarity of the time series [2]. Hence, it is necessary to transform these features to the stationary form, which is addressed in the following subsection.

6.1.1. Differencing

One of possible ways for dealing with non-stationary dataset is applying differencing, described in detail in section 3.3. Even though this transformation does not guarantee stationarity of the output, it often stabilizes temporal data properties [52]. Therefore, we employ this technique on the variables, for which H_0 of the left-

Feature	Lag	Test statistic	p-value
Averaged close price	39	-1.422	**p=0.572**
Market close price	53	-1.449	**p=0.559**
Market open price	53	-1.455	**p=0.556**
Market high price	53	-1.443	**p=0.561**
Market low price	53	-1.486	**p=0.541**
Num. of trades	44	-3.06	**p=0.03**
Volume traded	28	-4.686	**p<0.001**
Twitter cluster 1	53	-8.076	**p<0.001**
Twitter cluster 2	53	-7.005	**p<0.001**
Twitter cluster 3	53	-8.129	**p<0.001**
Twitter cluster 4	53	-9.397	**p<0.001**
Twitter cluster 5	53	-9.399	**p<0.001**
Reddit cluster 1	52	-9.410	**p<0.001**
Reddit cluster 2	53	-10.001	**p<0.001**
Reddit cluster 3	53	-10.531	**p<0.001**
Reddit cluster 4	53	-11.000	**p<0.001**
Reddit cluster 5	53	-11.627	**p<0.001**
News cluster 1	53	-21.768	**p<0.001**
News cluster 2	53	-19.192	**p<0.001**
News cluster 3	53	-20.084	**p<0.001**
News cluster 4	53	-22.737	**p<0.001**
News cluster 5	53	-23.372	**p<0.001**

Table 6.1: Results of the left tailed ADF test applied on the extracted time series

tailed ADF test has not been rejected. The target features directly relate to Bitcoin price, which suggests the use of relative differencing, widely adapted for visualising and transforming financial data [52]. Moreover, the sequence clearly involves non-linear growth and decline, which further indicates applicability of this approach. Therefore, for variable X, the technique applies the following transformation:

$$X_{diff} = \frac{X - L^d X}{L^d X}, \tag{6.1}$$

where L is the lag operator and d is the order of differencing. The order is set as 3 for convenience in data handling, due to label shift distance being 15 minutes (3 times steps). Thanks to this, while predicting the Bitcoin price 15 minutes into the future, one can simply forecast the relative difference between the future price and the current one.

Feature	Lag used	Test statistic	p-value
Averaged close price	53	-27.608	**p<0.001**
Market close price	53	-27.481	**p<0.001**
Market open price	53	-27.491	**p<0.001**
Market high price	53	-26.767	**p<0.001**
Market low price	53	-27.877	**p<0.001**

Table 6.2: Results of the left tailed ADF test after relative differencing

The method is applied on the target variables of the time series as well as the labels. The figure 6.1 illustrates the Averaged close price sequence before and after the transformation. Further, the ADF test for unit root is applied on the resulting differenced features and the results of this experiment are presented in table 6.2. Clearly, the H_0 is rejected for each variable, which proves the stationarity of the converted signal. Therefore, the further chapters apply the predictive algorithms on the extracted multivariate time series, with the 5 features related to price of Bitcoin and the labels transformed using relative differencing.

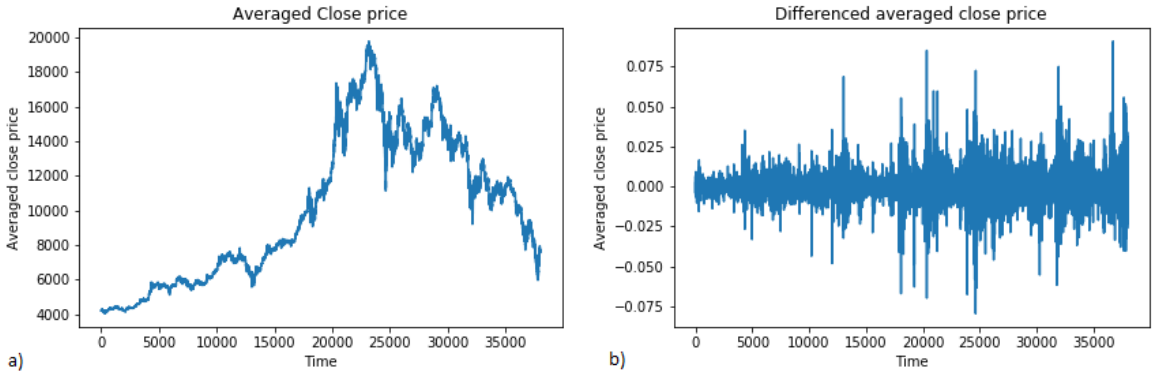

Figure 6.1: The averaged close price time series a) before and b) after differencing

6.2. Economic bubbles

This section analyses the properties of the Bitcoin price, in terms of economic bubbles, based on averaged close price feature. This thesis focuses on long-term bubbles, since they allow for more comprehensive analysis of their influence on the prediction in chapter 8. In that chapter, we will measure the effectiveness of the forecast over multiple segments of that bubble with prespecified length (further set to 200 samples, 16.7 hours). Moreover, larger number of such slices will increase the confidence of the applied statistical test. Thus, in this section we take into consideration bubbles that last for at least one day.

Consequently, we employ the SADF approach [56], due to its effectiveness in detecting long-term bubbles at a relatively low computational expense [63]. The algorithm recursively applies the right-tailed version of the ADF test. The significance level of the critical values is again set at 95%. The test obtains the t-statistic at each point of the time series after 1000 replications. This coefficient represents the likelihood of the explosive bubble component present in the signal at given index. Therefore, the intervals of the time series, for which the t-statistic exceeds the 95% critical value indicate when the bubble component is present.

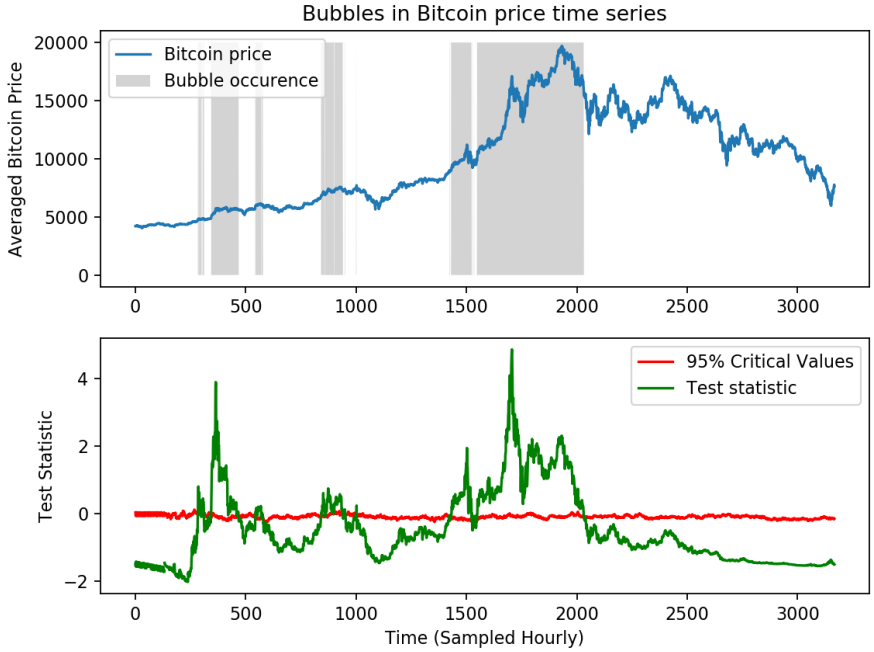

Figure 6.2: The results of the SADF test

In order to reduce the run-time of the SADF test execution the averaged close price signal is undersampled with hourly frequency. The results of the test execution are illustrated in figure 6.2. The top graph shows the undersampled time series with highlighted intervals, for which the test statistic exceeds the 95% critical values. The bottom one, depicts these two coefficients calculated for each time step of the sequence. Clearly,

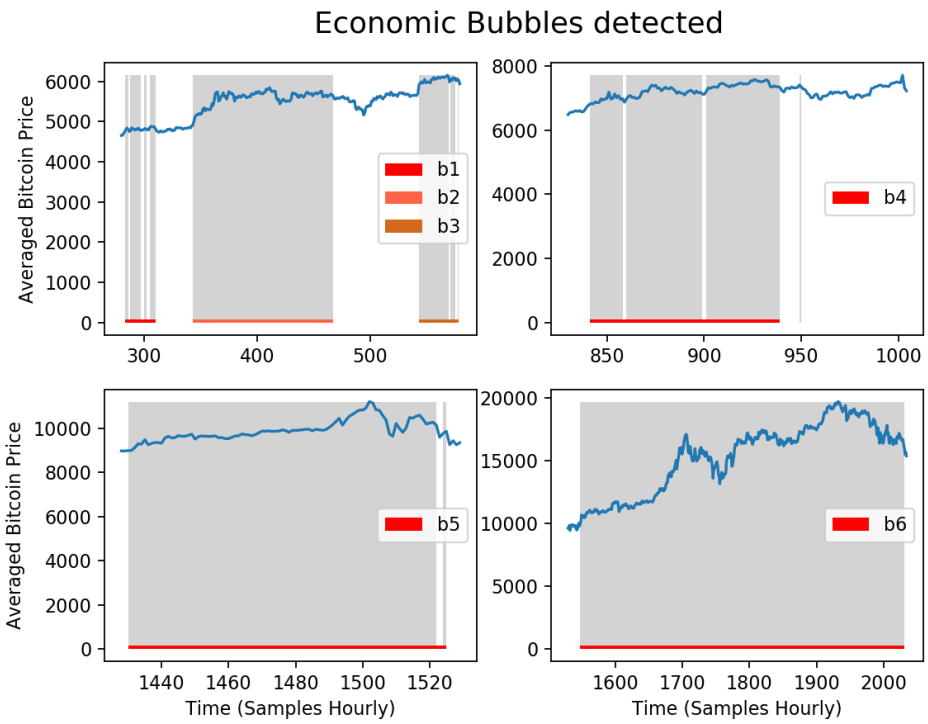

Figure 6.3: Economic Bubbles detected in the hourly sampled averaged close price time series

Bubble name	Start index	End index
b1	3396	3732
b2	4116	5616
b3	6516	6948
b4	10092	11280
b5	17160	18312
b6	18576	24384

Table 6.3: Detected economic bubbles and their respective position in the original time series

multiple bubbles are present in the time series. However, short breaks in long-term bubbles may hinder the analysis of the prediction results in chapter 8, because they cause a division of bubbles into granular segments and therefore, increases the difficulty of their investigation. Consequently, this thesis focuses on the bubbles lasting over 24 hours and disregards their brief interruptions (up to two hours).

As a result, there are six major bubbles in the time series, illustrated in the figure 6.3, annotated with letter 'b' and the respective index. For further reference, we map their start and end coordinates to their respective position in the original time series, see table 6.3. It is crucial to note that the number of detected bubbles is low and most of them are situated in the initial part of the time series, which will be used as the training set. Therefore, the analysis of the prediction in time of these bubbles, in chapter 8 will be highly limited and aiming at finding interesting patterns or anomalies, rather than their true influence on the forecast.

7

Prediction in a non-stationary environment

Having extracted the multivariate time series in chapter 5 and analysed its statistical properties in chapter 6, one can define how to perform the forecast on such dataset. Even though we have transformed the non-stationary variables into the stationary form, a problem of a relationship between this group of features and the target variable changing over time. An example that illustrates this phenomenon is when the same configuration of feature values indicates different true result of the target variable, in the time of up and down going long-term trend. Such situation is referred to as a concept shift, described in detail in section 3.3, and one needs to set up an appropriate methodology to deal with this issue. Moreover, one needs to choose a suitable ML predictive algorithm to model the problem and a relevant evaluation metric. Therefore, this chapter addresses the **RQ 1.3**, namely how to predict Bitcoin price and measure performance of the forecast system, while taking into consideration non-stationarity and occurrence of multiple economic bubbles in the time series. We answer the question in two steps.

At first, in section 7.1 we reflect on how to deal with concept shift that may appear in non-stationary environment, how to evaluate the models, and finally, which ML algorithm to employ. Thus, initially, we make a general consideration of the difficulties which need to be addressed and propose methodology to tackle them.

Having studied these matters, one can use this knowledge to design a prediction framework, namely a program, which uses the available dataset and the selected methodology to train and optimize models, and perform the forecast on new samples. Section 7.2 describes the architecture of such system and presents the implementation details.

While making design choices in this chapter, one has to consider the prerequisites for answering the **RQ 1.4** . The question states as follows: how well does a prediction system deal with non-stationarity and economic bubbles, and can semantic features derived from the considered data sources improve the forecast? Next chapter will perform experiments, based on the prediction framework that we will create in this chapter. The main requirements for these tests are:

- One needs to be able to evaluate the prediction system on the entire test set, and compare how well these systems perform for different features sets.
- It needs to be possible to objectively evaluate and compare performance of the system locally, for instance during the presence of the bubble component and right after that.
- A portion of the data containing at least 2 economic bubbles should be held out from training and optimization processes, to enable investigation of how prediction systems react to economic bubbles. It is a crucial requirement, because most of the detected bubbles are situated in the initial part of the time series, hence, we can only use them for training the models. However, it is vital that at least 'b5' and 'b6' belong to the test set, allowing experimentation in chapter 8.
- Prediction framework should enable optimization and experimentation on data with and without semantic features.

We will refer to these prerequisites in the following sections, while proposing the suitable methodology.

7.1. Problem Description

In this thesis, we model the Bitcoin price prediction problem as a regression task, in which one tries to forecast the exact value of the asset, 15 minutes into the future. However, prediction of a non-stationary variable with the occurrence of speculative bubbles requires taking additional measures to make sure it is successful. This section discusses the difficulties of forecasting such variable and how to deal with these issues, which partially addresses the RQ 1.3.

Concept shift, described in detail in section 3.3, characterizes a situation, in which the statistical properties of the predicted variable and its relation to the input features change over time [45]. The condition causes a deterioration of model's performance, because its train and test sets are not identically distributed, which violates the assumption made by most ML algorithms, described in section 2.2.1. Moreover, rules learned from the past data may become deprecated over time and ineffective, while predicting the current values of the target variable. In the following subsection 7.1.1 we will propose the methodology that efficiently deals with this issue.

Further, we will analyse how the quality of the forecast can be evaluated, taking into consideration the non-stationary price behaviour. It is a crucial aspect of this research, because in order to answer the RQ 1.4, one needs to objectively compare how well a given predictive model performs, both globally and locally, for different segments of the time series. Subsection 7.1.2 illustrates why one cannot use standard evaluation metrics for this task and proposes a solution that enables unbiased analysis of the results.

Finally, it is essential to find an appropriate algorithm to model the complex dataset. The technique should be suitable for sequential datasets and enable employment of several variables, while predicting the price. Moreover, it should optimize a loss function that resembles the selected evaluation metric. Subsection 7.1.3 focuses on these topics.

7.1.1. Dealing with concept shift

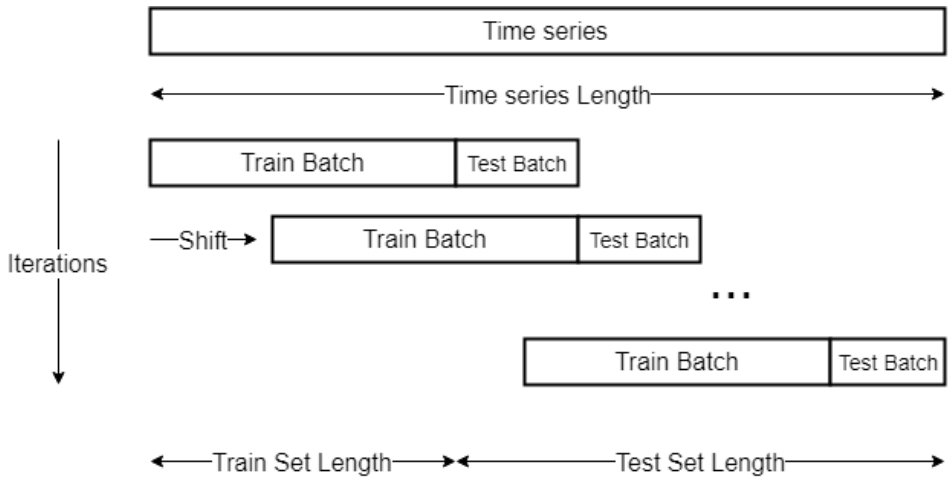

Figure 7.1: General Framework for dealing with non-stationarity

This subsection considers how to deal with the concept shift, caused by non-stationarity of the Bitcoin price signal. In the previous chapter we have applied the relative differencing on the time series and managed to reject the null hypothesis of the left-tailed ADF test, which has proven stationarity of the extracted variable. However, there is another issue related to non-stationarity, which needs to be addressed, namely the concept shift. Even though the variables have a stable mean and standard deviation over time, their relationship with the target variable may change over time. It may appear that the rules learned from the past data are irrelevant when predicting the current samples. Therefore, it is crucial to take additional measures to prevent the negative impact of concept shift on the results.

In order to obtain reliable prediction and allow systems to quickly accommodate to the changing environment, we employ the approach implemented by Giles et al. [24] to forecast a financial asset's price. Figure 7.1 illustrates the procedure. The scheme consists of iterative training and testing of the models on consecutive segments of the data. Therefore, one trains a prediction system on a window of the data, having a pre-specified length, and uses it to make the forecast on the most recent test set of a relatively small, fixed size.

Then, the considered train and test data windows are shifted by the length of the test batch and the process is repeated until the end of the time series. In the following sections, we will refer to these sets as train and test batches and to the program that iteratively trains and tests models as a prediction system. It is crucial to discuss that there is no shuffling applied to samples before assigning them to a given window. Consequently, the train batch consists of, for instance 1000 consecutive time series samples, and the test batch contains 200 samples that follow right after the former. Thus, at each step there is no overlap between the train and test windows.

Thanks to frequent retraining with the most recent data, the created models are up to date to forecast the most up-to-date test batch. One evaluates a given model on a single test batch, and further uses this data to train a new model. This way, it is possible to exploit the test set in the training process once the forecast has been made on a given test batch. Moreover, with the iterative shifting of the data windows one evaluates the prediction system on the entire test set. Another reason for selecting this scheme, is the fact that answering the RQ 1.4 requires evaluation of the model for granular segments of the data, and this approach allows for that. However, the main difficulty with its application is the selection of train and test batches' sizes. If the train window is too small, the learning algorithm has too few samples to learn from, while having too large window , causes the model to exploit outdated samples. Similarly, if the test batch is too short, the number of required iterations to complete evaluation process grows, while too large one, negatively influences the results, because of larger distances of some test samples from the train set. However, the issue of parameter optimization that comes with using the approach is addressed by the parameter optimization procedure introduced in section 7.2.2.

7.1.2. Evaluation Metrics

In this subsection, we focus on the selection of an appropriate evaluation metric for this thesis. Two of the requirements for addressing RQ 1.4, stated in the introduction of this chapter, relate to the statistic. They require it to enable a global evaluation of the prediction systems and a fair comparison of local results, regardless of slight changes in data distribution. Therefore, this section starts with an analysis of the properties of the differenced non-stationary Bitcoin price time series. Then, we investigate how these properties influence the evaluation metrics that are often used in regression tasks, and we illustrate why these statistics are not applicable in this thesis. Finally, we propose a novel quantitative measure, which fulfills the stated requirements.

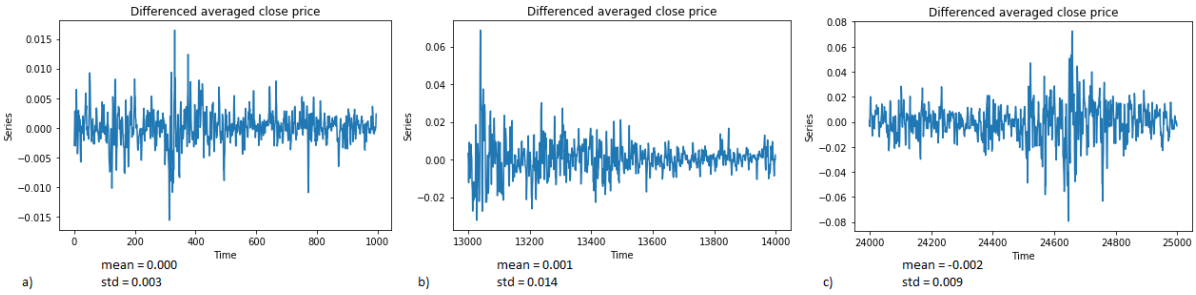

Figure 7.2: Example parts of differenced Bitcoin price time series with different mean and standard deviation

As already mentioned in the previous subsection, even though the null hypothesis of the left-tailed ADF test has been rejected for differenced Bitcoin price signal, there may be some slight local fluctuations of the mean and standard deviation. Especially in the time of the speculative bubbles, the original time series experiences a major increase (positive mean in the differenced dataset), and right after that, a downfall (negative mean in the differenced dataset). Moreover, there may be times when the market is highly unstable, resulting in an increased standard deviation of the differenced time series. In order to answer the RQ 1.4, we need a reliable measure of predictor's performance, which is insensitive to these trends in the data.

Figure 7.2 illustrates three examplary parts of the differenced Bitcoin price signal, for which the mean and standard deviation varies marginally. Even though the discrepancies of the mean and the standard deviation are low, we will show further in this subsection that they may cause significant variations of the obtained results.

An often used baseline method for the prediction model is the Persistence Algorithm (PA) [2]. The technique computes a naive prediction, by forecasting at time step t the value of the time series at $t - lag$. Therefore, in the context of Bitcoin price prediction, it estimates that the value of the asset in 15 minutes from a

Evaluation Metric	Formula
Mean Absolute Error (MAE)	$MAE = \frac{1}{n}\sum \lvert y_{pred} - y_{true} \rvert$
Mean Squared Error (MSE)	$MSE = \frac{1}{n}\sum (y_{pred} - y_{true})^2$
Root Mean Squared Error (RMSE)	$RMSE = \sqrt{\frac{1}{n}\sum (y_{pred} - y_{true})^2}$
Mean Absolute Percentage Error (MAPE)	$MAPE = \frac{100\%}{n}\sum_{i=0}^{n} \lvert \frac{y_{i,pred} - y_{i,true}}{y_{i,true}} \rvert$

Table 7.1: Evaluation metrics often used in the regression problems

given point in time, will be the same as the current one. In case of a differenced variable, the signal represents how much its value changed with regards to the lagged one. Hence, the 15 minutes into the future forecast of PA on the differenced target variable with 15 minutes lag, is always equal to 0. In this subsection, we will use the technique to illustrate how some evaluation metrics react to varying from zero mean and standard deviation of the selected parts of differenced target variable.

We have tested the prediction given by the PA on the three selected parts of the time series, based on the following statistics: Mean Absolute Error (MAE), Mean Squared Error (MSE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE). Table 7.1 presents the formulas of these metrics. The evaluation of the naive forecast for the three segments is illustrated in Table 7.2. When it comes to the three initial metrics, the measurement is significantly higher, even by a factor of 25 for MSE, when the mean and standard deviation are further from 0. Therefore, for some parts of the time series it is much harder to achieve low error using those metrics, due to the distribution of the target variable. This property makes it difficult to compare how well a prediction system works for different segments of the data, because a well working system tested on part c), may achieve a higher value of the error, than the naive forecast on evaluated part a). It is caused by the fact that MAE, MSE and RMSE do not normalize the achieved score with respect to the amount of fluctuations in the differenced ground truth. The results represent the amount of error that the predictor made, instead of the amount of error relative to the environment. When it comes to the last analysed metric, MAPE is insensitive to the data distribution change, due to the scaling of the local absolute error using the ground truth. The value of this measure for differenced dataset is above 1 if the predictor wrongly forecasted the direction of change or overshoots in the correct direction. On the other hand, its value approaches 0, if the model forecasts value in the correct direction of price change and gets close to the ground truth. Thus, MAPE provides a useful measure of efficiency of the model, relative to the distribution of the target variable and allows for fair comparison of performance of the prediction system for different segments of the data. However, there is a possibility of division by 0, if the open and close price in a 15 minutes window are equal. This issue is especially evident for the differenced target variable, which makes it not applicable in this research.

As shown, MAPE is insensitive to deviating mean and standard deviation, due to the division by $y_{i,true}$ term, yet it has a major limitation, which disqualifies MAPE from use in this study. We introduce a similar evaluation metric called Normalized Mean Absolute Error (NMAE) that deals with the problem of possible division by 0 for differenced datasets:

$$NMAE(y_{pred}, y_{true}) = \frac{\frac{1}{n}\sum \lvert y_{pred} - y_{true} \rvert}{\frac{1}{n}\sum \lvert y_{true} \rvert} \tag{7.1}$$

The statistic is calculated for a batch of samples and it is computed by dividing the MAE of a given forecast by the MAE of the prediction baseline computed by Persistence Algorithm. The normalization of the prediction

Properties	Time series part a)	Time series part b)	Time series part c)
Mean	0.000	0.001	-0.002
Std	0.003	0.009	0.014
Results of the Persistence Algorithm			
MAE	0.002	0.006	0.010
MSE	8e-6	8e-5	2e-4
RMSE	0.003	0.009	0.014
MAPE	Error-division by 0	1	1

Table 7.2: Results of the Persistence Algorithm on the three analyzed parts of the differenced Bitcoin price time series

MAE is very similar as for the MAPE, yet, instead of division by the absolute $y_{i,true}$, we divide it by the mean absolute $y_{i,true}$ for the current batch of data. Similarly to the MAPE, the lower the NMAE score, the better the forecast is. Moreover, if the value of prediction MAE, divided by PA MAE, is above 1, it indicates that given set of estimations is worse than predicting no change of price at every time step.

The main requirement when employing the approach, is that one needs to split the time series used for evaluation into a number of small batches. This way, the MAE of the prediction is normalized, based on the local properties of the data. In order to measure the global score of a given prediction system, one can calculate the mean NMAE over all the batches:

$$meanNMAE(y_{pred}, y_{true}) = \frac{1}{n\_batches} \sum_{i=1}^{n\_batches} NMAE(y_{batch\_i,pred}, y_{batch\_i,true}) \tag{7.2}$$

Thanks to the computation of NMAE for each window of the time series, one makes sure that the error is normalized, based on the local properties of the data. Therefore, taking the mean of these local measurements provides a stable estimation of the model's overall performance, even if the characteristics of the time series change over time. It is crucial to underline that if the number of batches is equal to the number of test samples, the mean NMAE is an equivalent of MAPE.

In this work we make use of the introduced NMAE statistic to measure the performance of predictive models applied on the non-stationary dataset with occurence of bubbles. The measurement is especially convenient to employ in the thesis, due to the use of iterative training and testing procedure introduced in section 7.1.1, which ensures granularity of the evaluation. The final mean NMAE can be computed based on the calculated NMAE statistic from each test batch in a given time series. Thus, NMAE estimates the effectiveness of the model for a specific data segment, while mean NMAE determines its overall performance. Therefore, the composition of the iterative training procedure with the appropriate evaluation metric allows for more robust prediction and fulfillment of the requirements of the RQ 1.4.

7.1.3. Model selection

Once the methodology for dealing with non-stationarity of the data and the evaluation metric is chosen, one can pick an appropriate ML technique to model the available dataset. There are several requirements for the predictive algorithm:

- It needs to be suitable for multivariate sequential time series, which violates the sample independence in i.i.d. assumption, described in 2.2.1. This means that current sample may be related to the previous ones. Therefore, one needs to select an algorithm that exploits sequential nature of the dataset.
- The technique should be flexible in terms of the amount of past samples, which are used to perform the prediction at a given time step. In some cases it is better to take into consideration only the most up-to-date samples, while in others, it would be beneficial to focus on long-term dependencies in the data.
- The algorithm should train a model by minimizing the Mean Absolute Error on the training samples, which is comparable to minimizing NMAE in the setup of this thesis. This statement can be justified by the fact that for each test batch one divides the prediction MAE by a positive constant, which is independent from the forecast. Furthermore, we have applied the relative differencing, which has transformed the target variable into the stationary form. This leads to a stable behaviour of both: prediction MAE and the PA MAE, by which the former is divided to compute NMAE. Therefore, the prediction MAE is divided by a similar value for each test batch of the data. We acknowledge the fact, that minimization of MAE is not equivalent to minimization of NMAE, yet in this setting it mimics the desired behaviour.

In order to address these requirements one can use Recurrent Neural Networks, which fulfill the criteria, especially the LSTM network, described in detail in section 2.2.3.1. First of all, the algorithm is designed for sequential input with multiple features. It exploits both short-term and long-term memory, which allows the model to base the prediction on the current state, but also long-lasting dependencies [51]. Moreover, it learns during the training process which information should be stored in the cell state (memory vector) and how to combine it with the current input [51]. Thanks to this property, it is flexible when it comes to number of samples that the current prediction is based on. According to Karpathy et al. [33], the LSTM model can store crucial information in the long-term memory even for 60 time steps. When it comes to LSTM's predictive abilities, it can model a highly complex non-linear decision boundary [33]. Thanks to the fact that the model

consists of layers of neurons and exploits a deep architecture, it learns useful features from raw data, without the need of feature engineering [51]. Apart from that, there are multiple straightforward to apply methods for regularizing Neural Networks [38], which prevent overfitting of such complex systems. Finally, one can apply custom loss function, which ensures optimization of the target evaluation metric. The main drawback of this approach is the need of large amount of data to train a LSTM network that generalizes well to unseen samples. However, by using various regularization techniques one can remedy this problem.

This thesis employs the the Keras implementation [1] of the LSTM algorithm to predict Bitcoin price signal. In order to ensure that LSTM mimics minimization the mean NMAE, we use the MAE loss function, available in Keras. Moreover, several regularization techniques are applied to ensure that the model does not overfit, namely Dropout, L_1 and L_2 regularization [25]. The hyperparameters, determining the impact of these techniques on the model, are optimized in section 7.2.2.

7.2. Prediction framework

The previous section analysed issues that appear, while forecasting the differenced non-stationary time series, and proposed solutions for these problems. Having considered these matters, one can use this knowledge to build a prediction framework. By the term prediction framework we refer to a system that takes as input the multivariate time series, extracted in the previous chapters, and performs three main tasks: data curation, hyperparameter optimization and experiments. The overall goal of using such system is training models with optimized parameters and applying them on the test samples to compute the forecast. Moreover, in this section we answer the remaining part of the RQ 1.3, by putting all the proposed solutions together into a single forecasting program.

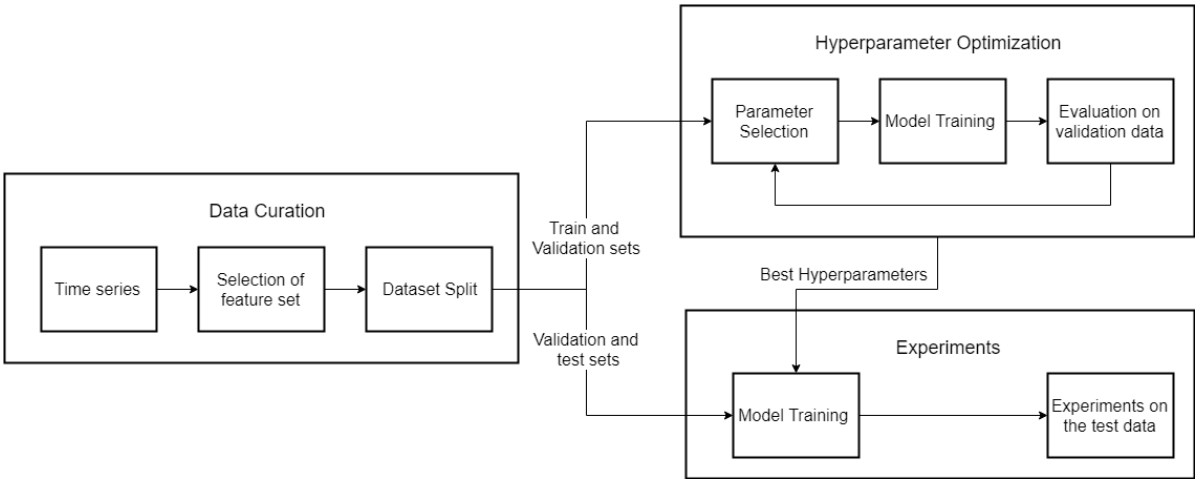

Figure 7.3: Overview of prediction framework implemented in this thesis

Figure 7.3 presents an overview of the prediction framework. As already mentioned, there are three main segments in the system.

Firstly, the data curation module manages the input time series, namely selects a set of features that are to be used in the prediction and splits the data into the train, validation and test sets. Subsection 7.2.1 describes the implementation details of this part of the program.

Secondly, the hyperparameter optimization segment takes train and validation sets as input and performs a number of iterations of random search [9] procedure to find the suitable parameters for data. In each run it applies the iterative approach of training and testing the LSTM networks on the shifting windows of data, and also evaluates given setting using mean NMAE. The best hyperparameters setup is returned and used to forecast the Bitcoin price in further steps. Subsection 7.2.2 elaborates on the optimization procedure module.

Finally, the last segment in the prediction framework is responsible for forecasting on the basis of the test data. Similarly to the previous module, it applies the iterative training and testing procedure on the test data, using the most successful hyperparameters setting. Depending on the performed experiment some implementation details may differ, yet the subsection 7.2.3 outlines the crucial aspects of the program's setup.

[1] https://keras.io/layers/recurrent/

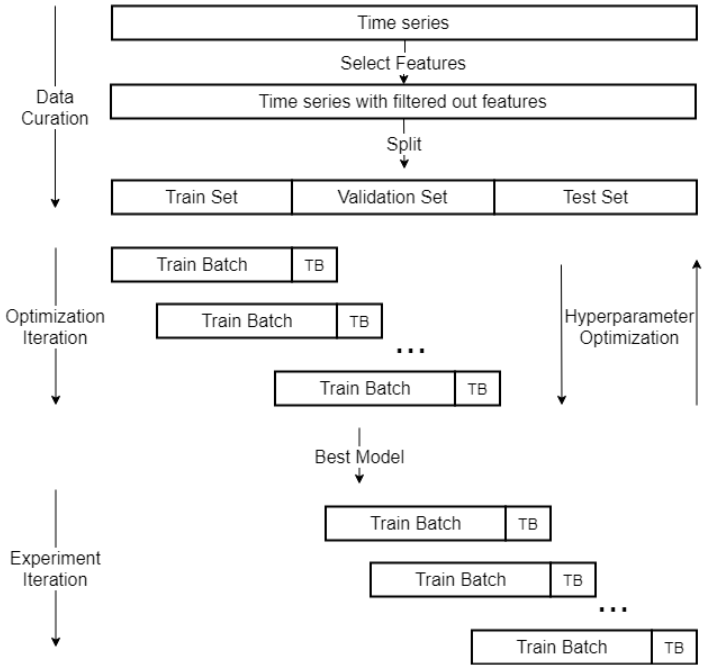

Figure 7.4: Steps performed within the prediction framework

Putting all the pieces together, the Figure 7.4 illustrates the steps performed by the prediction framework on the input time series.

7.2.1. Data curation

The initial module in the prediction framework is performing the data curation. Its main responsibility is management of the input time series and its preparation for the further applied methodology.

Initially, the dataset consists of approximately 37 thousand samples with 22 variables:

- 5 differenced price features,
- 2 columns representing trades,
- 5 semantic Twitter variables,
- 5 semantic Reddit variables,
- 5 semantic News variables.

Apart from that, each sample has a ground truth label, which represents the differenced averaged close price of Bitcoin in 15 minutes. The first step in the data curation module selects the feature set, based on which the further prediction is performed. It is a crucial step, since the RQ 1.4 investigates whether the semantic features from online sources improve prediction. Therefore, we have composed 8 different feature sets that will enable examination of this issue and presented them in table 7.3. Depending on the current examined setting, the data curation module filters out the irrelevant features.

Further, the time series is split into train, validation and test sets. The first collection is used only for model's training, the second one for optimization purposes and the final one for performing the experiments. We split the dataset of 37000 samples into 3 sets of distinct consecutive samples, with 7000, 10000 and 20000 samples respectively. While making the decision on how to make this split, we have taken into consideration that there is a large number of parameters to be optimized. Moreover, the test set needs to contain at least two economic bubbles (B5 and B6), to be able to investigate how prediction model behaves in presence of these events (RQ 1.4). The train and validation sets are further passed to the hyperparameter optimization module, while validation and test sets are used during experiments.

Name of setting	Num. of features	Types of features
M	7	5- Price, 2- Trades (Market features)
MT	12	5- Price, 2- Trades, 5- Twitter
MR	12	5- Price, 2- Trades, 5- Reddit
MN	12	5- Price, 2- Trades, 5- News
MTR	17	5- Price, 2- Trades, 5- Twitter, 5- Reddit
MTN	17	5- Price, 2- Trades, 5- Twitter, 5- News
MRN	17	5- Price, 2- Trades, 5- Reddit, 5- News
MTRN	22	5- Price, 2- Trades, 5- Twitter, 5- Reddit, 5- News

Table 7.3: Different feature sets used in the thesis

7.2.2. Hyperparameter optimization

Once the time series is pre-processed, one can begin the process of hyperparameter optimization. The aim of this process is finding a setting, in which a given system performs most effectively on the validation set, with an expectation that it will function similarly well on the test set. There are many choices one has to make before training the predictive models. The following list presents the parameters that need to be determined:

- Size of the train batch- the parameter defines the length of the window used in the iterative training and testing procedure introduced in section 7.1.1. With larger size of the collection, the model has more data to learn from and it is less likely to overfit, however, some data may be already outdated in the non-stationary environment.
- Size of the test batch- parameter of the iterative training and testing procedure. The parameter influences granularity of the evaluation. It is set it to a fixed value of 200, to get consistent results for every feature setting. The choice leads to a relatively high granularity of the results and an acceptable execution time of the iterative scheme.
- Batchsize- hyperparamer of the LSTM, which defines how large batches of the data are used for training the network. For high values of batchsize, training iterations are quicker, yet the model requires more of them for convergence.
- Epoch- number of training iterations of LSTM network. In general, with more epochs it is easier to overfit, while too little iterations cause the undertraining of the model.
- Neurons- number of units in the last hidden layer of the network. Higher values increase complexity of the model and the training time.
- Lookback- number of past samples used, while predicting the current one.
- Dropout- percentage of units turned off randomly at each training iteration. With higher values of this parameter the impact of regularization on the LSTM is stronger.
- L_1 Kernel and Recurrent connections- the strength of L_1 regularization applied on the input and recurrent connections of LSTM. The technique works as a feature selection mechanism, turning off some links in the network
- L_2 Kernel and Recurrent connections- the strength of L_2 regularization applied on the input and recurrent connections of LSTM.

In order to optimize these parameters we employ the Random Search procedure [9]. It applies a number of optimization iterations, in search for the setting that is the most effective on the validation set. In each iteration, the hyperparameters are initiated randomly, based on a pre-specified range or set of values. Then, models are trained and tested on the consecutive parts of the train and validation sets, using the scheme introduced in section 7.1.1. During each step of this procedure, we first rescale the data to a range from -1 to 1, train the LSTM network and calculate NMAE on the current test batch. Finally, once each segment of the validation data is covered, the program stores the resulting mean NMAE and the respective hyperparameters. By repeating the process 100 times, the procedure finds the most suitable setting for each feature set, defined in section 7.2.1.

Param. Name	Description	Considered Values
TrainSize	The size of the train batch	[6000, 7000]
BatchSize	The size of the data batches and size of the test batch	[200, 500]
Epoch	Number of training iterations to train a single model	[150, 500]
Neurons	Number of neurons in the LSTM network	[20, 35]
Lookback	Lookback window length of the LSTM	[20, 50]
Dropout	Dropout regularization parameter	[0, 0.5]
L_1 Kernel	L_1 regularization parameter applied on the input connections	{0, 0.00001, 0.00005}
L_2 Kernel	L_2 regularization parameter applied on the input connections	{0, 0.00001, 0.00005}
L_1 Recu	L_1 regularization parameter applied on the recurrent connections	{0, 0.000001, 0.000005 ,0.00001}
L_2 Recu	L_2 regularization parameter applied on the recurrent connections	{0, 0.000001, 0.000005, 0.00001}

Table 7.4: Parameters optimized using validation set and their considered values

In this setup we prevent overfitting on the validation set by setting the size of this set to a large value, namely 10000. Therefore, for the TestBatch length being equal 200, there are 50 tests for a given hyperparameter setting. Even though the parameter search space is large and it is likely that some setups will overfit to some regions of the time series, we facilitate finding a generalizable setting, by repeating the test for 50 different segments of the data. This kind of overfitting can be recognized by a low mean NMAE on the validation set and a higher one on the test set.

Another way that the prediction systems can fail to generalize is by overfitting on the train batches in the iterative training procedure. It results in a poor performance on the test batches of the validation set and hence, high mean NMAE for that set, and not being further taken into consideration in the optimization process. The second type of overfitting is prevented by applying several regularization methods to the model, namely Dropout, L_1 and L_2. The results of the optimization process are illustrated in the next chapter.

7.2.3. Experiments

Having set the suitable hyperparameters for each feature setting, one can progress to the experiments. This section presents a general procedure of evaluating a given setup on the test set. However, some details of the experiments designed to answer RQ 1.4 in the next chapter may differ, it is crucial to understand the concept behind them.

The program takes validation and test sets with pre-specified feature set and the optimized parameter setting for the data as input. Then, the iterative procedure of training and testing models, introduced in section 7.1.1, is applied on these datasets. This scheme is visualized in Figure 7.4. At each iteration, the data is rescaled to range from -1 to 1. Next, the LSTM model is trained on a train batch and evaluated on a test batch using NMAE. Once the process is completed for the entire test set, one can compute the mean NMAE or investigate model's performance locally, for instance during presence of an economic bubble. Therefore, this experimental procedure is adjusted to address specific requirements of the RQ 1.4, in the next chapter.

8

Experiments and Results

The previous chapter has designed the methodology for forecasting in non-stationary environment. A framework has been developed, which allows for optimization and testing of prediction systems with various feature sets. Thanks to this, one can turn to the **RQ 1.4**, namely 'how well does a prediction system deal with non-stationarity and economic bubbles, and can semantic features derived from the considered data sources improve the forecast?'. Moreover, a remaining part of RQ 1.2 needs to be answered, specifically, how incorporating the features extracted from text influence model's robustness to non-stationarity In this chapter we will address these question in within three main sections of this chapter.

First of all, in section 8.1 we will perform an analysis of hyperparameter optimization of the prediction systems with different features sets and test their performance on the entire test set. This way, one can discover whether semantic features derived from text data improve the forecast and if there is overfitting evident.

Secondly, we will focus on the prediction results in presence of economic bubbles in section 8.2. Even though the number of bubbles detected does not allow to find the influence of the phenomenon on the prediction, we will analyse the forecast in time of 'b5' and 'b6', to find interesting patterns or anomalies. Special attention will be paid to the structure of the bubbles, namely long-term trends and price peaks, during which the signal switches from up to down going trends. We hope that the study will lead to crucial observations, which can be confirmed in the future work, using a more representative dataset.

Thirdly, we will investigate how does non-stationarity influence the forecast in section 8.3. The negative effect of the property is prevented in two ways: using relative differencing, which ensures stable temporal distribution of each feature and by employment of iterative procedure introduced in section 7.1.1, which deals with concept shift. We have shown in section 6.1.1 that the transformed variables have a stationary form, yet their relation with the target variable may still evolve over time. Therefore, it is crucial to study how a given model can generalize to the samples that are further into the future. This objective is achieved, by setting up an experiment that studies how results change when each model forecasts several test batches into the future.

The following sections start with introduction of the goal of the experimentation conducted. Further, an appropriate test is designed and performed to address each problem and its results are discussed at the end of each section.

8.1. Optimization and Evaluation

The main goal of this section is the analysis of how well the developed prediction systems work in general, and whether the introduced semantic features make an improvement to the forecast. This can be achieved by simply measuring the test set performance of the systems using different input variables. Thus, in the following experiment we use the general experimental procedure introduced in section 7.2.3.

Firstly, for each feature set, the 100 iterations of the optimization scheme are employed, to find the appropriate hyperparameter setting. Next, the test set mean NMAE is measured, in order to assess given system's performance on the hold-out dataset. Moreover, in this research it is crucial to determine whether the results substantially differ, when additional features from online media are employed. Thus, we will use a statistical test to estimate how significant performance disparity of a given prediction system is, from the one, employing only the market variables. To achieve this goal, we will firstly confirm the normality of test set

results distributions for each system using D'Agostino-Pearson test with 95% confidence level [65]. Then, we will apply a two-tailed Paired Sample T-Test [65] on the NMAE scores for all test batches of the 'M' and any other prediction system. The main reason for selecting this test is its ability to take into consideration that NMAE is computed in an organized manner by the prediction framework, namely the results are computed for consecutive, ordered batches of test data. This test determines whether there is a major disparity between means of two distributions, with null hypothesis suggesting that the means are equal and alternative hypothesis proving their inequality. Therefore, in case of rejection of H_0 with 95% confidence level, one can confirm that employing additional features significantly increases or decreases effectiveness, depending on the sign of test statistic, compared to using only market variables. However, if the null hypothesis cannot be rejected, there is no major variation in the results between two systems. As in previous sections, the significance level has been set to 95%, in order to provide both: high quality of the findings as well as reasonable difficulty of rejecting the null hypothesis.

Setup		Optimized hyperparameters									
Name	Feat. num.	TrainSize	Epoch	BatchSize	Neurons	Lookback	Dropout	L1Kernel	L2Kernel	L1Recu	L2Recu
M	7	6864	227	264	22	31	0.37	0	5E-05	1E-06	0
MT	12	6330	446	422	27	38	0.50	1E-05	0	1E-05	0
MR	12	6315	295	421	30	40	0.45	1E-05	0	0	1E-05
MN	12	6630	236	390	30	23	0.40	1E-05	0	5E-06	1E-06
MTR	17	6555	308	345	29	27	0.46	1E-05	0	1E-06	0
MTN	17	6495	227	433	28	29	0.50	0	5E-05	1E-05	5E-06
MRN	17	6228	297	346	21	40	0.20	0	5E-05	1E-05	0
MTRN	22	6540	468	218	29	49	0.50	0	5E-05	1E-05	1E-05

Table 8.1: Optimized hyperparameters for each feature setting

The first step in this experiment is the hyperparameter optimization for each features setup. Table 8.1 illustrates the parameter settings of the most successful prediction systems, found over 100 iterations of the optimization process.

Name	Feat. num.	Mean NMAE on validation set	Mean NMAE on test set	Test Statistic	P-value
M	7	97.85%	97.90%	-	-
MT	12	97.71%	97.92%	-0.119	0.905
MR	12	**97.52%**	97.69%	1.994	**0.049**
MN	12	97.65%	**97.67%**	2.033	**0.045**
MTR	17	98.42%	98.96%	-15.291	**0.001**
MTN	17	98.31%	98.42%	-7.705	**0.001**
MRN	17	98.16%	99.67%	-9.222	**0.001**
MTRN	22	98.93%	99.86%	-13.773	**0.001**

Table 8.2: Results of the optimized prediction systems on validation and test sets. The last two columns present the results of Paired Sample T-Test between NMAE test scores of 'M' and a given prediction system

Further, these systems are evaluated on both validation and test sets. Table 8.2 presents the mean NMAE results achieved on each of these sets. First of all, in the optimization process, all the systems have comparable results. In all cases, in which a set of semantic features from a single online text source is employed, there is a slight improvement, compared to using only the market features. On the other hand, with more variables, the performance on the validation set deteriorates, which may be related to overfitting on the train batches in the iterative training process. All in all, the three most effective systems on the validation set are 'MR', 'MN' and 'MT'.

Secondly, evaluation of the optimized prediction systems on the test set indicates that a single set of semantic features from Reddit comments or from articles improves the results. However, it does not apply for Twitter, for which, the performance has deteriorated on the test set, most probably, due to overfitting on the validation set, while optimizing the hyperparameters. This is indicated by the difference between the mean NMAE results on validation and test sets for that settings. Similarly, with more features included into the forecast, the phenomenon of overfitting becomes more evident, and the systems lose their generalization ability. In order to apply a parametric statistical test that would validate the significance of this findings, one has to

	Region		
Setup name	B5	B6	Outside
M	94.00%	97.49%	98.31%
MT	93.51%	97.24%	98.95%
MR	94.46%	96.24%	98.72%
MN	96.09%	96.86%	98.35%

Table 8.3: Mean NMAE of the four selected prediction systems on different segments of the test time series

firstly, prove normality of the NMAE results distribution. By applying D'Agostino-Pearson, we have proven the normality of the results for each prediction system, with all the p-values being below 0.001. Then, we have applied the Paired Sample T-Test between NMAE test scores of 'M' and each further prediction system, see table 8.2. The results indicate that employing a single set of semantic features, based on comments or articles, significantly improves the forecast. On the other hand, null hypothesis of the test cannot be rejected for 'MT', which suggests no major difference introduced by these variables. Finally, addition of more sets of features significantly deteriorates the forecast, since the p-value for these systems is below 0.05 and the test statistic is negative.

To sum up, the best performing feature setups overall are 'MN', 'MR' and 'M'. We have shown that addition of a single set of semantic features, based on articles or Reddit, boosts the performance of the prediction system and does not cause overfiting. The latter finding is especially crucial, because it indicates that these systems can be used to forecast future samples, with lower risk of performance deterioration over time. When it comes to addition of more feature sets, it causes a major performance drop as well as overfitting on the validation set.

8.2. Prediction in presence of economic bubbles

Having analysed how the prediction systems perform in general, one can investigate how do they act in time of economic bubbles. As already described in section 2.1.4 and illustrated in section 6.2, the time series has specific properties during the occurrence of bubbles. Therefore, in this research, we study whether the prediction systems can employ these patterns into the forecast. The test set contains two periods with presence of speculative bubbles named 'b5' and 'b6'. Even though the analysis is limited by the number of these events, we hope to find interesting patterns, that can be further confirmed in the future work. In order to find how the predictive capabilities change over these data segments, we will execute two experiments. The following tests make use of the general experiments framework, described in section 7.2.3. However, in order to simplify the analysis, we take into consideration only four prediction systems, namely 'M', 'MT', 'MN', 'MR' configurations. Not only these systems were the most successful in the previous experiment, but also, they allow to analyse how employment of a specific set of semantic features influences the forecast.

First of all, it is crucial to investigate whether the forecast results significantly differ for the periods with and without occurrence of bubbles. Such study may indicate if prediction in time of economic bubbles has its specific properties. The evaluation is performed by calculating the mean NMAE achieved by the selected prediction systems, over duration of 'b5' and 'b6', and outside of these regions. If the results achieved by the systems are superior in time of the bubbles, one can consider which factors of these events enable such an improvement. Secondly, in order to statistically validate the significance of the difference between the results, we apply a non-parametric Mann-Whitney U test [65] on the distributions of NMAE results for regions with and without economic bubbles. Thanks to making use of non-parametric equivalent of the Two Sample T-test [65], one does not have to assume normality of the results distribution, which does not hold for 'b5'. This test determines whether there is a major disparity between two distributions, with a null hypothesis suggesting that their median scores are equal and an alternative hypothesis proving their inequality. Therefore, in case of rejection of H_0 with 95% confidence level, one can analyse of the test statistic and confirm that a given prediction system is more or less effective during a given bubble, than outside this region. However, if the null hypothesis cannot be rejected, there is no significant variation when forecasting in presence of a bubble, compared to the rest of the test set.

Mean NMAE achieved by the selected prediction systems has been measured for the regions 'b5', 'b6' and for the rest of the time series, referred to as 'outside'. Table 8.3 presents the evaluation and table 8.4 illustrates the results of the Mann-Whitney U test, applied on the distributions of results of each bubble and the outside region. For each prediction system, the recorded results are better in time of the two bubbles than over the

	Comparison			
	B5 and outside		B6 and outside	
Setup Name	Test Statistic	P-value	Test Statistic	P-value
M	140.0	**0.0262**	899.0	0.3520
MT	105.0	**0.0065**	766.0	0.1270
MR	148.0	**0.0464**	716.0	0.0580
MN	145.0	**0.0400**	800.0	0.1665

Table 8.4: Results of the Mann-Whitney U test

remaining test samples. The difference is especially visible in case of 'b5', being in range between 5.44% and 2.26%. Furthermore, the statistical test proves the significance of the observation, indicated by all the p-values being below 0.05. Moreover, in time of 'b6' bubble, all the systems achieve slightly better performance than for 'outside', with the difference varying between 2.48% and 0.82%. However, one cannot reject the null hypothesis of the Mann-Whitney U test, which indicates that the difference is not significant and might have happened by chance. Consequently, the outcomes suggest that the forecast results are significantly superior in time of 'b5' and slightly better in case of 'b6'. Thus, There may be a relationship between the occurrence of the bubbles and the prediction results, however, it has to be confirmed on a more representative dataset.

The second experiment in this section aims to find interesting patterns in the forecast results, over duration of bubbles' development. Therefore, we qualitatively investigate the results of three prediction systems over different segments of 'b5' and 'b6'. Moreover, the experiment may uncover which parts of the economic bubbles are more or less difficult to forecast, and further, justify the results of the previous experiment. The analysis is performed by visual inspection of the NMAE results achieved on the consecutive segments of averaged close price time series for 'b5' and 'b6' regions.

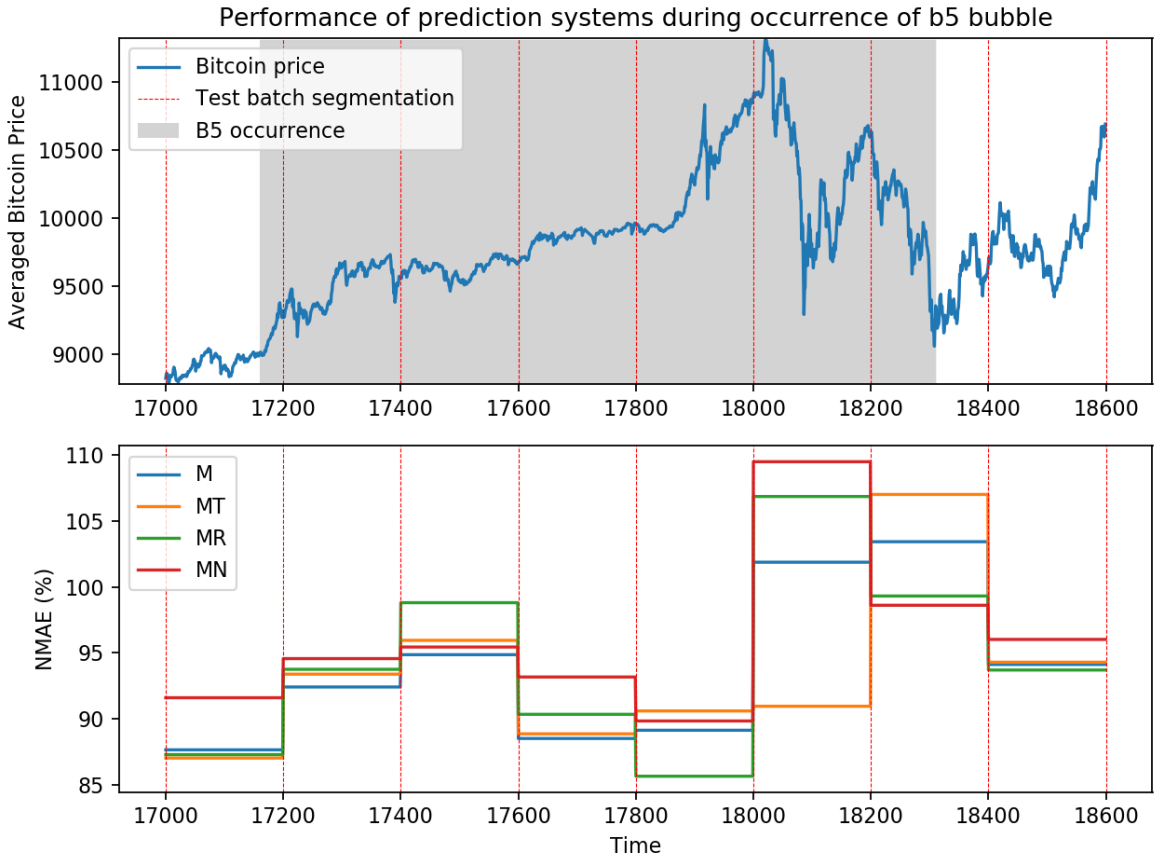

Figure 8.1: Performance of prediction systems during occurrence of b5 bubble

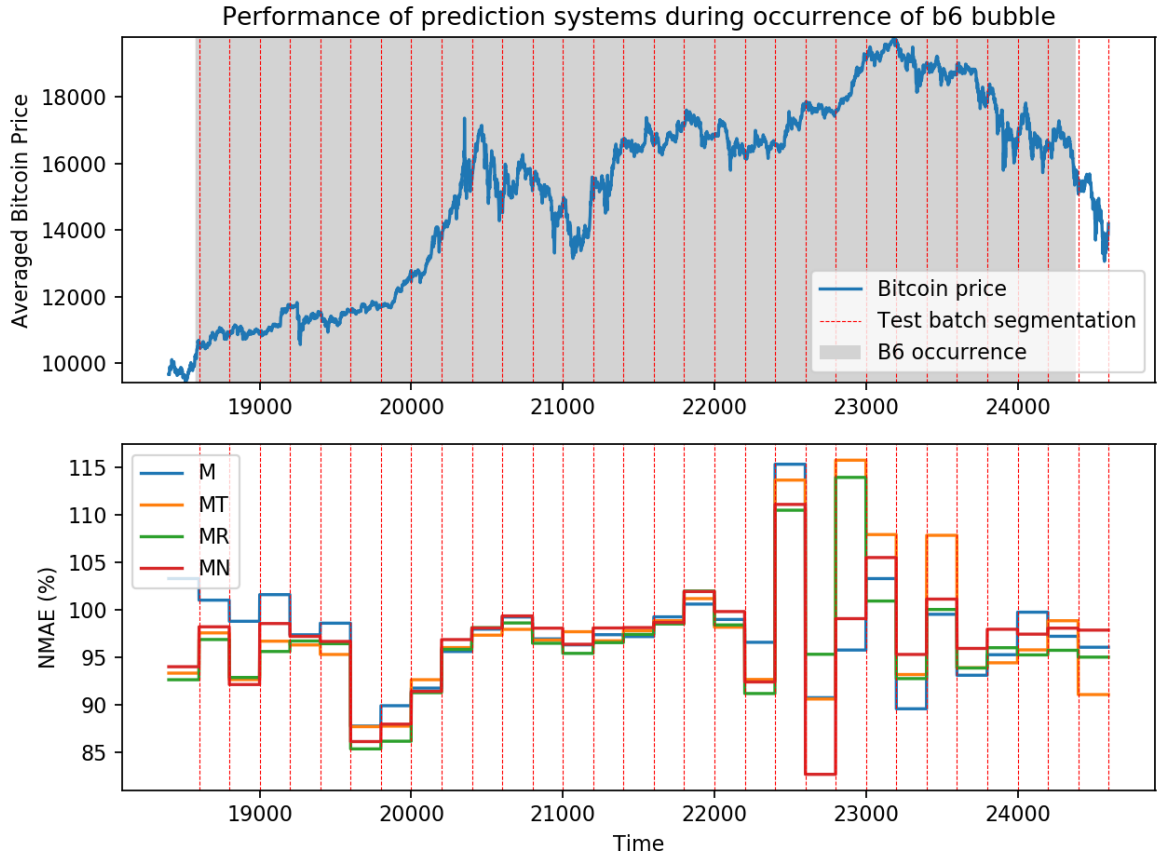

Figure 8.2: Performance of prediction systems during occurrence of b6 bubble

Figures 8.1 and 8.2 present the NMAE results achieved by the four prediction systems, for different segments 'b5' and 'b6' respectively.

By analysing the top graph of figure 8.1, one can observe that 'b5' follows a standard scheme defined by Kindleberger et al. [34] and described in section 2.1.4. The price displacement occurs from the beginning of the bubble till around 17400th index, and is followed by a boom, lasting until around 17850th sample. Further, the euphoria starts, driving the price to the level of $11000, and right after that comes the distress and the panic on the market. The last two episodes cause a significant price drop and highly unstable market situation, starting from around 18000th sample and continuing till the end of the bubble's duration.

When it comes to the bottom graph, interestingly, one can observe that performance of the prediction systems follows some patters for some of the bubble's stage. During the displacement phase, it deteriorates, most probably, due to start of an unnaturally fast price growth above its fundamental price. Then, most of these systems accommodate to the long-term upgoing trend and we witness an improvement of their NMAE during the boom and euphoria phases. Further, at the price peak and during the market distress and panic, the graph depicts a major increase of the NMAE, even up to 110%. Therefore, at this time, most of the prediction systems fail at computing accurate forecast. This may be caused by the shift of the long-term trend's direction, which leads to the concept shift. Thus, it is evident that the prediction systems require some time to accommodate to the novel situation and most of them did not manage to generalize to these events. Interestingly, the 'MT' performed surprisingly well in time, where the first major price drop occurred. It suggests that it is possible to compute a high quality forecast, even in face of unexpected time series behaviour. However, in this case, most of the models overfit on the long-term uptrend and failed to accurately foresee the market shift.

As far as figure 8.2 is concerned, the top graph depicts the sixth detected bubble, which is much longer and more complex than 'b5'. It partially follows the standard scheme discussed in section 2.1.4. At first, the price displacement and boom appear one after another, between 18600th and 20000th indices. Further, the euphoria rapidly inflates the price till around $17500. Then, the distress appears and causes significant

price drop and highly unstable market situation, ranging from 20400th to 21200th samples. However, instead of being followed by the panic, we witness another significant price growth till around $19000. Finally, the distress and panic appear after 23200th index, driving the asset's price down to its fundamental value.

The bottom graph illustrates the performance of the selected prediction systems over the course of 'b6'. As before, during the market displacement the evaluated systems score around their average NMAE. Then, there is a significant drop of the error, caused by the boom of the price and its stable growth. Further, there are no specific patterns between 20200th and 22400th indices and the prediction systems achieve their average results. Interestingly, the performance starts fluctuating strongly between 22400th and 23800th indices. The error exceeds 100% whenever the price increases, and drops below that number, if the value decreases. This indicates, that all the prediction systems expected the upcomming distress. Most probably, thanks to appearance of multiple of such trend shifts in their training batches, they were able to forecast the price downfall. Lastly, over the time of the recession of Bitcoin price, the performance of the models is close to their average error.

There are several crucial observations made during this experiment. In both cases, the error has dropped even below 90% during the boom period, when the price has grown steadily. Another observation is that, overall it is difficult to forecast the bubble's burst, namely the beginning of the distress phase, because if follows right after a long-term uptrend. However, the second experiment shows that it is possible, especially when the training data contains several major price peaks. In case of 'b5', the 'MT' prediction system has achieved low error in that phase. Moreover, regarding 'b6', all the prediction systems expected the final price drop, mainly thanks to presence of two of such events in close past.

To sum up, in this section we have performed two experiments. It has been shown that over the time of the 'b5', the mean NMAE of the forecast is significantly lower than outside of these periods. It suggests that some bubbles may have specific structure, which enable more efficient prediction. In the other experiment, we have analysed how the forecast's performance changes over different segments of the bubbles. We have observed interesting patterns in the prediction systems' results, especially in time of the market boom and while forecasting the distress. However, it is crucial to confirm these findings in the future work, on a more representative dataset, containing multiple economic bubbles in the test set.

8.3. Influence of non-stationarity

In the final experiment we aim to find out how non-stationarity influences the results of the prediction systems as well as whether introduced feature sets impact robustness to its effects. The former question is part of the RQ 1.4, while the latter allows to finalize the answer of the RQ 1.2. In the current setting of iterative training and testing procedure, a model trained on a train batch is applied only on a single test batch. However, in order to answer the stated questions, one can make a modification of the current setup, such that, in each iteration, the model is trained on the train batch and evaluated on the 1st, 2nd, 3rd, ... consecutive test batch. If the environment was stationary and no concept shift would appear, the trained model would have a comparable performance on each of these batches. However, loss of its predictive abilities over time would indicate that learned patterns and rules become outdated over time. Moreover, employment of the semantic features extracted from the online sources may allow the LSTMs to extract knowledge from additional data, which is more robust to the evolution of the target variable. This phenomenon would be indicated by slower deterioration of the model's performance, when used to forecast further into the future. On the other hand, if addition of these variables stimulates degradation of the results over time, one can conclude that from a given set of features, the model extracted less stable rules.

The designed experiment uses the general experimental framework with the described modification. For a given feature set with optimized hyperparameters, we train and test the LSTM network on the consecutive batches of the data. The shift of the data windows is performed as in the original framework, therefore, by the length of a single test batch. However, the testing procedure is extended to measuring NMAE of multiple consecutive test batches into the future. Hence, in each iteration of the scheme, for a single train batch, one evaluates how well the trained model predicts further into the future than just one step. Finally, the mean NMAE is calculated over the entire test set for each consecutive test batch index. Therefore, one measures for a given prediction system, for instance, the average error when predicting the 10th test batch, counting from the most recent one. The experiment is performed for the feature setups: 'M', 'MT', 'MR', 'MN', with 15 test batches forecasted at each iteration of the procedure.

In order to statistically validate whether non-stationarity influences the results, additional steps are added to the scheme. At first, we apply differencing of order 1 on the the achieved mean NMAE scores for the

System name	Test statistic	P-value
M	20	**0.041**
MT	37	0.330
MR	15	**0.0185**
MN	19	**0.037**

Table 8.5: Results of the One Sample Wilcoxon Signed-Rank test on the differenced results

increasing test batch's index. The resulting vectors represent how much the mean NMAE changes with each next step forward predicted into the future. Then, we apply the 95% confidence level, One Sample Wilcoxon Signed-Rank test [65], to indicate whether the results increase, decrease, or remain stable over time. As in the previous section, the non-parametric statistical test is used, due to not being able to prove normality of these results. The null hypothesis of the test states that the median of a given distibution is equal to 0, while the alternative indicates its difference from 0. If the null hypothesis will be rejected, we will prove that non-stationarity positively or negatively influences the results, depending on whether the concluded median is negative or positive respectively. Otherwise, one cannot prove any effect of non-stationarity.

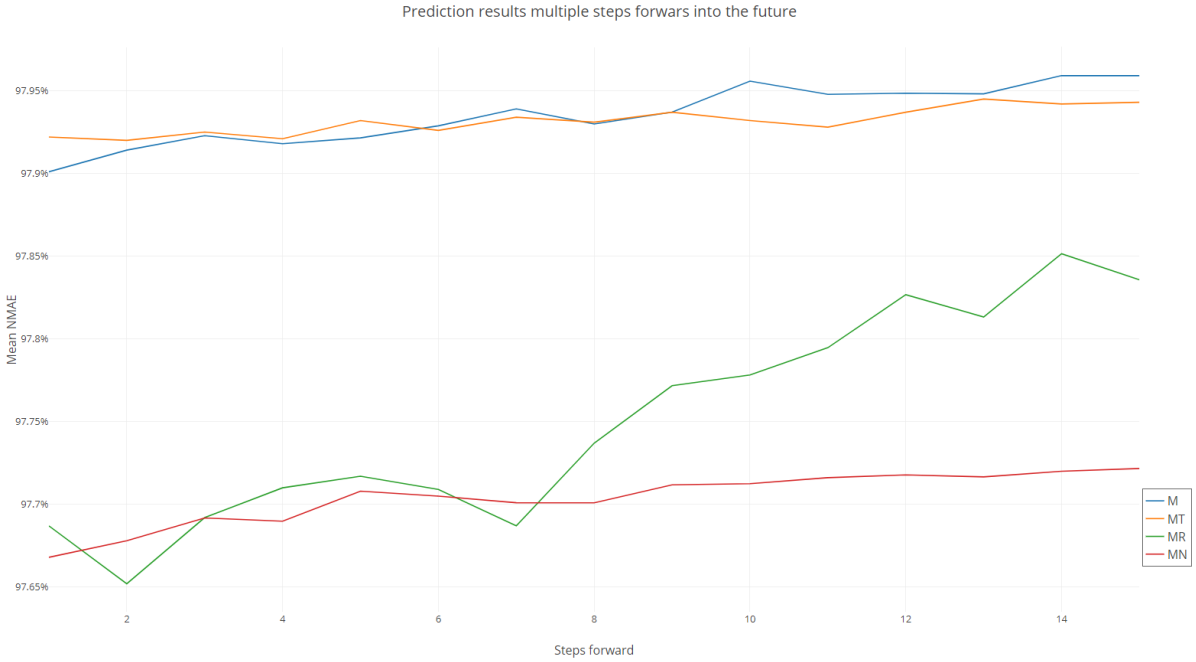

Figure 8.3: Results of the experiment illustrating the influence of non-stationarity on the results

The results of the test are presented in figure 8.3. From the graph, one can observe that the performance of the systems deteriorates if they used to forecast further into the future. The 'M' has 0.07% higher mean NMAE, while forecasting 15th test batch forward, in comparison to the 1st one. When it comes to addition of features extracted from tweets, the results are highly comparable to the previous ones, yet slightly more stable. Further, the variables extracted from Reddit, improve overall score of the system, yet they also cause a faster deterioration of the results. The difference between the mean NMAE results achieved during the 15th step of and the 1st one reaches almost 0.15%. Finally, features from articles, in general, boost performance of the prediction system. However, one cannot observe any considerable boost of robustness to non-stationarity of this system, compared to 'M'.

Then, to statistically validate the observed effect of non-stationarity on the results we have applied differencing and the One Sample Wilcoxon Signed-Rank test on the samples, see table 8.5. The null hypothesis is rejected in case of 'M', 'MR' and 'MN' and the positive test statistic indicates that the mean NMAE increases, when these systems are used to predict further into the future. Therefore, one can conclude that non-stationarity negatively influences their results. On the other hand, the null hypothesis has not been rejected for 'MT', thus, one needs to assume that non-stationarity has no effect on that system. Even though the

test would most probably result with rejection of the null hypothesis, if the sample size was larger, one can assume that features extracted from Twitter, to some extend, prevent the negative effect of the phenomenon.

To sum up, non-stationarity negatively influences the results of most of prediction systems, namely 'M', 'MR', 'MN', by causing deterioration of their performance, if they are used to forecast samples further into the future. However, semantic features based on Twitter slightly reduce this effect, which has shown to introduce stability for the tested number of steps forward.

9

Discussion and Conclusion

The previous chapters have applied various methodology to extract, process and analyse data as well as, to predict Bitcoin price. Moreover, experiments performed across the thesis allow answering all the research questions stated at the beginning of this study. Therefore, we are ready to discuss these results and their contribution to the main objective of this thesis. This chapter revisits the research questions, with respect to work and findings of this paper.

Section 9.1 considers each research question and outlines the approach taken to answer it. The methodology is discussed and its pros, cons and limitations are presented. Furthermore, the results of the experiments are reviewed and the conclusions are extracted. Finally, the section also underlines future work that can be conducted with regards to given topic.

Then, section 9.2 concludes this thesis. It outlines the main steps performed in this study and present the key points of the research.

9.1. Discussion

This study considers the problem of Bitcoin price prediction. The main research question of this thesis is:

RQ 1 How do the non-stationarity and speculative bubbles encountered in the Bitcoin price signal influence its prediction and how to make the forecast more robust?

Therefore, we progress in understanding the properties of the Bitcoin price variable and how they influence the prediction. Since the related work mainly focuses on how to make the forecast more effective, we take a step back and try to find explanation of why the topic is so challenging, as well as how to better and more knowledgeably approach it in the future. Moreover, we also consider how to build a robust prediction system. However, before fully answering the main research question, we need to reconsider the subquestions that RQ 1 is divided into. Therefore, the discussion process is split into subsections, corresponding to each of these components, and finally, one that addresses the objective of this study.

9.1.1. RQ 1.1: What are the properties of the Bitcoin signal in terms of stationarity and speculative bubbles?

Once the data of Bitcoin price was collected and processed into a time series, we could perform analysis of the signal in chapter 6. Its goal was to discover the properties of the variable, in terms of stationarity and economic bubbles.

In order to find out whether the signal is stationary, the left-tailed ADF test with 95% confidence level was applied on each variable of the time series. This test proves stationarity of a sequence, in case of rejecting the null hypothesis. Otherwise, one cannot disprove that the dataset is non-stationary and has to assume that property. The results have shown that all the features related to Bitcoin price have a high p-value, ranging from 0.541 to 0.572. This indicates that, not only, one cannot disprove the null hypothesis, but also, it is very unlikely that the data is stationary, due to high value of the statistic. The main limitation of the applied ADF test is that it cannot be used to directly prove non-stationarity. However, by analysis of the resulting test statistic and p-value, one can determine the confidence of the assumption. If the p-value would be close to 0.05, one needs to be more careful with such conclusions, yet, in this case, the certainty is high that the signal

is non-stationary. It is also confirmed by the visualization of Bitcoin price time series in figure 6.1, since the value was initially at the level of 4000 and then peaked to almost 20000. Therefore, we can safely assume non-stationarity of the data.

Further, we transformed the data using relative differencing and applied the same test again. The resulting p-value was below 0.001 for every feature representing the differenced bitcoin price. This proves with high confidence that the method has effectively converted these variables into a stationary form. Thanks to this step, we were further able to build a prediction framework, which can efficiently predict Bitcoin price without violation of assumption regarding identical distribution of the samples, described in section 2.2.1. We also tested how well differencing or logarithmic differencing works with the data, but in both cases, the p-value was higher. Thus, we suppose that relative differencing is an essential step, while processing the dataset. The main difficulty when applying the method is setting order of differencing. For this research, the parameter was set to 3, for convenience reasons. This way, when predicting the future price in 15 minutes (3 steps of 5 minutes), we could simply forecast how much the value will deviate from the current one. Even though the choice did not guarantee optimal results, it appeared successful in ensuring stationarity, as well as practical, due to simplified analysis of the results.

The final step to fully answer the first research subquestion, was the detection of economic bubbles. The focus was put on the long-term bubbles, to allow for more comprehensive analysis of prediction results over different segments of the discovered events. In order to detect bubbles and their duration, we applied the SADF test with 95% confidence level. The method is known for its high efficiency in detecting long-term bubbles [54]. The technique identified 6 distinct bubbles, which lasted from 20 hours to 480 hours. For comparison, we also ran a more complex GSADF test, which lead to 10 times longer execution time, while the results mainly varied in terms of events that lasted up to 2 hours. Unfortunately, the number of detected bubbles is low, and they are located mainly at the beginning of the time series. Therefore, the analysis of prediction in time of these events was highly limited.

The parameter that needed to be set in this step of the analysis was the confidence level of the SADF test. It specified how certain the method needed to be to label given sample as a bubble. After analysis of the bottom graph of figure 6.2, we observed that the 95% critical values of the test statistic were very close to the computed values of test statistics of multiple regions market as bubbles. This means that in case of bubbles 'b1', 'b3' and 'b4', the test was less confident about the results, while the measure exceeded the baseline by far for majority of 'b2' ,'b5' and 'b6'. Thus, setting the significance level to higher value, would result in a shorter and more scattered bubble periods and even more limited analysis. In the end, the experiments in section 8.2 were performed on the 'b5' and 'b6', due to other regions being used to train and optimize the prediction systems. Therefore, one can be highly confident that the inspected segments of the time series in the chapter 8, are actual economic bubbles.

9.1.2. RQ 1.2: How to incorporate the text data into prediction and how does it influence model's robustness to non-stationarity?

In the second subquestion we aimed to employ text data from Twitter, Reddit and online news into the forecast. At first, the data was collected and pre-processed, then, the semantic features were extracted from text, and finally, the samples were incorporated into the time series using the moving sum. The most crucial step in this process and novelty in the field of Bitcoin price prediction was extraction of the semantic variables.

The extraction is performed by, firstly, training an unsupervised word2vec model on the data from a given source. Then, the model transforms each input text into a numerical vector, which is further assigned to one of several clusters. We enhanced the vector extraction from text, by hyperparameter optimization of the unsupervised word embeddings, using data labeled using crowdsourcing. Thanks to this step, we ensured that samples that discuss related topics are placed closer to each other, in terms of cosine distance between them. Thanks to this property, further applied spherical clustering found natural semantic grouping of the samples from a given source.

The results of the optimization process illustrate that there is a high correlation between human intuition, regarding relatedness of two documents, and the cosine similarity of generated vectors, based on these samples. The correlation ranges from 0.46 to 0.5, with the lesser value obtained on data extracted from Reddit. These values are comparable to the results of a similar experiment performed by Campr and Ježek [13], in which the top models reached the correlation of 0.45 for Word2Vec and 0.55 for Doc2Vec. The supervised approaches for topic classification, described in section 3.2, would most probably achieve a higher correlation with human intuition. In the mentioned experiment these techniques obtained over 0.2 higher correlation than the best performing unsupervised model. However, the unsupervised methodology was much more

convenient to apply in this research, and to our mind it achieved a sufficient performance. Thus, we used the optimized models to extract vectors for the text corpus of each source.

Then, based on the computed vectors, we have extracted 5 clusters per text data source using spherical k-means clustering. The method minimizes the intra-cluster cosine distance, which fits well with the optimization performed earlier, due to the same distance metric being minimized. The main issue with the method is the requirement of setting the number of clusters to be computed. If the parameter was set too high, the resulting grouping would discuss more distinct topics, yet, in the later stages, it would be harder to train a predictive model that does not overfit. On the other hand, too few clusters would make the grouping too general and it would cause a difficulty, when analysing the clusters' content. We have tested several values of this parameter, and setting it to 5 provided the lowest number of clusters with high analysability of the grouping. Furthermore, the evaluation of prediction systems in section 8.1 has shown, that adding one set of semantic features improves the forecast, while using more of them causes deterioration of the results, due to overfitting. Thus, we suppose, that the chosen number of groups was close to optimal.

When it comes to quality of semantic clusters, we observed that detected topics for each cluster were more concise for Twitter and news. We came to this conclusion by inspection of the highest tf-idf terms appearing per group. This may be caused by more centered discussion around topics directly related to Bitcoin on these portals, while in case of Reddit, its users tend to deviate from the original discussion. Moreover, based on the figure 5.5, we observed that most pairs of comments, labelled using crowdsourcing, have the value of this label below 1, and thus, are semantically unrelated to each other. Since, there is a low number of leading topics of the discussion on reddit, it is harder to group the samples in a meaningful way. This indicates that the resulting semantic features extracted from that source have lower quality than Twitter and news.

When it comes to pros and cons of the employed approach, it was especially convenient to use in this thesis, because it did not require a large labeled dataset for supervised model's training, obtaining of which would be very costly and time consuming. Instead, the unsupervised methodology extracts natural semantic cluster for each source, which we further improved using crowdsourcing. Moreover, thanks to employing semantic features, we could analyse which topics are mainly discussed online. However, the main drawback is the need to set the number of clusters to be extracted in a way that the resulting variables represent concise topics, and does not cause overfitting of the prediction systems. Furthermore, we acknowledge that over time the distribution of topics may change and new ones can appear, which are not directly considered by the current approach. Thus, if one applies this methodology to forecast Bitcoin price, it would be reasonable to retrain the Word2Vec and spherical k-means models as well as even reoptimize the parameters.

The main improvements that can be introduced to the current approach is the use of more text in unsupervised training of Word2Vec or labelling of more data, using crowdsourcing, to improve the optimization procedure. Finally, looking back at the approach, we suppose that employing sentiment-based features instead of semantic ones, would simplify the study. Not only we could use a pre-trained model to directly generate the output without the optimization process, but also, we would end up with 3 analysable variables per text source, which retain their properties over time. Therefore, in the future work, we may focus on sentiment-based variables extracted from the analysed sources. Furthermore, having the semantic features employed, one can research on how discussion of a given topic correlates with the price movements.

Lastly, the research question also concerns whether the extracted semantic features increase robustness of the model to non-stationarity. Our hypothesis at the beginning of the research stated that adding features from online sources can allow the model to form more stable rules for predicting the target variable. We investigated whether it is true, by performing and experiment in section 8.3. This test visualized how the performance of prediction system change, when it is used to forecast further into the future. Based on visual inspection of figure 8.3, we have observed that results of the 'MR' deteriorate faster than for other systems. Even though this observation is not statistically confirmed, this effect can be explained by a lower quality of semantic features extracted from that source. Since each of the clusters represent much broader range of available topics, rules learned by the model may get outdated quickly. On the other hand, the 'MT' system experienced slower deterioration. In the same graph we observed, that its error increased with the number of steps predicted forward, yet even less significantly than for 'M'. Confirming this observation would indicate that features extracted from Twitter have increased robustness to non-stationarity of the models. Another crucial finding comes from the results of the Wilcoxon signed-rank test that was further applied on the differenced results of the experiment. By employing this test, we managed to prove that median change of the system's error is positive for 'M' ($p=.0453$), 'MN' ($p=0.0420$) and 'MR' ($p=0.013$), which indicates deterioration of their results over time. However, the null hypothesis was not rejected for 'MT' ($p=.1098$) and thus, one has to assume that the variable does not significantly change over time. By analysing the p-values one can

support the earlier made observations regarding the error growth of Reddit and Twitter features. However, we suppose that if we performed the same test on a larger sample, namely with higher maximum number of steps forecasted forward, the p-value would be below 0.05 for every feature set, even for 'MT'. The achieved high p-value, indicates that the results may deteriorate at a slower pace than for other systems, which implies increased robustness to non-stationarity. Still, this observation needs to be confirmed on a larger sample of the data. Moreover, we also applied the paired Wilcoxon signed rank test, between each of the samples, to check if it is possible to indicate. Rejection of the null hypothesis would indicate that the results deteriorate significantly faster or slower, for a given feature set. Yet with this sample size we were not able to reject the null hypothesis with high confidence.

9.1.3. RQ 1.3: How to predict Bitcoin price and measure its performance, taking into consideration non-stationarity of the data?

This research question aimed at studying issues related to predicting a non-stationary variable with occurrence of economic bubbles. By proposing solutions to the highlighted issues, we also made sure that the system is more robust to that environment.

As already described in section 9.1.1, stationarity of the variables is ensured using relative differencing. However, there was another issue that needed to be addressed, while predicting in non-stationary environment, namely the concept shift. In order to deal with it, we have introduced the iterative training and testing procedure. The technique repeatedly retrains the LSTM models on the current data, to predict the most recent test cases. The experiment conducted in section 8.3, also mentioned in the previous subsection, proves the efficiency of the approach. In that test, we have observed that using a single model to predict further into the future than a single test batch, causes deterioration of the performance. According to the experiment's results, if a single model was built to forecast the entire test set, one would achieve worse results. Therefore, the method prevents the negative influence of non-stationarity by applying each trained model on only one test batch. The main limitations of this approach is the time required to complete the procedure as well as the need to set the length of train and test batches. As already described in section 7.1.1, the parameters play a crucial role in the learning process. Length of the training batch determines how much data a given model exploits during its training phase, and whether it exploits general tendencies or focuses on the most current events. The size of the test batch influences the computation time, the amount of future samples that each model forecasts and the granularity of the experiments' results. The former parameter has been optimized in the hyperparameter optimization process, while, the latter has been set to 200. Considering the results inspected in section 7.2, namely the performance of prediction systems applied on different segments of the bubbles, we suppose that setting the test batch length to a lower value would enable more granular analysis, especially in case of 'b5'. However, it would significantly increase the time required for completion of the optimization process, which already took over 3 months on a single machine. Thus, it appeared that setting the value of the parameter to 200 was the right choice, finding a balance between the research constraints.

Another crucial issue encountered in this study was choosing a metric to evaluate the systems in an unbiased way with regards to the the situation on the market. In section 7.1.2, we reviewed the available evaluation metrics and indicated why they do not fit to the scope of this study. Further, a novel metric was proposed that enables the analysis, required by the research questions. The statistic suited well with the earlier introduced iterative procedure, because it assumes that the test set is segmented. The advantage it introduces in this work is the fact that we were able to evaluate the systems with regards to the current amount of fluctuations. This means that, first, the MAE of the prediction is calculated, and then, normalized according to the amount of change of the price. Thanks to this feature, whenever the value of NMAE is below 1, one can conclude that the forecast was successful, namely superior to the naive prediction. The introduced measure is strongly based on MAPE measure and can be interpreted in a similar way, yet NMAE addresses MAPE's main limitation, of possible division by 0. Moreover, using NMAE, we were able to indicate locally which system works better, while mean NMAE determined the global properties of the forecast. Furthermore, the measure was convenient to analyse, since it indicated successful prediction, if it was between 0 and 1, and unsuccessful, if it was above 1. Finally, if the researchers in the field of Bitcoin price prediction would use this metric to evaluate their systems, one would be able to point, which approach works best, regardless of segment of price sequence, on which the model was tested. Currently, every study in the field exploits a different test metrics, most of which are strongly influenced by the mean and the standard deviation of the time series, for instance, MSE [26] or RMSE[44]. Therefore, there is a need for further research on how to measure the performance in a more comparable way, in case of Bitcoin, but also, more generally, financial assets. Our solution partially addresses the problem, yet it has a couple of limitations. Firstly, one needs to set up the size of the test batch,

which may have an influence on the end results, since it normalizes the MAE result with regards to the MAE of naive forecast in that time. However, thanks to transforming the target variable to the stationary form, at each step the error is divided by a similar value. Thus, we have minimized the influence of that issue. Another open question is how to train a model that optimizes the mean NMAE. There are no available implementations of the loss function and, in section 7.1.2, we argued that by minimizing MAE using relatively small training and test batches, we can achieve close to optimal results, assuming that target variable is stationary. The main argument backing the hypothesis is the fact that for each step of the iterative testing procedure, one divides the MAE achieved for a given batch by a constant that is similar across batches. To sum up, we have made an attempt to find a measure allowing for unbiased comparison of performance of models tested on different periods of non-stationary datasets. Yet we suppose that it is an interesting issue to research in the future.

In section 7.1.3 we have selected the ML algorithm to model the dataset, namely the LSTM network. The choice fit well with the requirements of the study and had many advantages. However, by reviewing table 8.2, presenting the results of the prediction systems on validation and test sets, one can conclude that the technique may have been too complex for the problem. Even though it achieves results of mean NMAE below 1 for prediction systems with up to 12 features, including more variables causes overfitting of the model, indicated by the increased error on the test set. Since using more training data may not be effective in the non-stationary environment, because the older samples may be outdated, the alternative to prevent overfitting would be increasing the amount of regularization applied to it. Moreover, it would be insightful to compare the performance of the LSTM with some simple baseline model, for example, a linear regression. Thus, a limitation of this study was also lack of a baseline for performance comparison.

Further we have combined the proposed methodology into the prediction framework, which consisted of three steps: data curation, hyperparameter optimization and experimentation. The initial phase was responsible for filtering the unnecessary features and spliting the data into train, validation and test sets. The sizes of these sets were set to 7000, 10000, and 20000. That choice had a major effect on the performed analysis and the quality of the results. The size of train set specified the maximum amount of samples each model could use for training. If the amount of available train samples would be too low, the models would overfit in some iterations of the training and testing procedure, causing worse results overall. The validation set length, specified the amount of data that was used to optimize the parameters. Setting the parameter too low would cause overfitting on the validation set in the hyperparameter optimization process. This issue is especially crucial in case of this research, due to large parameter space. The results presented in the table 8.2, show that some systems have overfitted on the validation set, indicated by higher error on the test set. However, it was mainly the case for the systems that performed poorly on both sets, therefore, the issue did not have a major influence on the analysis and the derived conclusions. The final parameter, namely the test set size, determined the confidence of the results, achieved in the experiments of this thesis. We have shown in section 8.2, that on some parts of the time series, for instance, in time of the bubble 'b5', the results achieved by the prediction systems are significantly better. Thus, one should also consider that the some of the results in the related work may be biased by the test set used. If the size of this set is too small or it focuses on a specific event, for instance, an uptrend, the evaluation may not be as trustworthy. Moreover, in case of this research, the test set needed to contain at least two economic bubbles to allow for answering the RQ 1.4. Therefore, we have set the size of the test set to the value larger than sizes of train and validation sets together, namely 20000.

When it comes to the hyperparameter optimization process, for each specified feature set, it found the best parameter setting, based on a 100 random search iterations. After reviewing the results from table 8.2, presenting the evaluation of the optimized prediction systems on validation and test sets, we concluded that the systems with 17 or 22 features overfit in two ways. Firstly, the models trained in the iterative training and testing procedure overfit on train batches, causing low scores on the corresponding test batches, and thus, worse results on the entire validation set than models with less features. This may be caused by too low amount of data to learn from, for such a complex algorithm, exploiting so many variables. Thus, we suppose one can remedy the problem by repeating the optimization process with increased amount of regularization applied to the systems using 17 features and above. The second type of overfitting is demonstrated by a lower error on the validation set than the test set, caused by too large parameter optimization space and too small size of the former set. In other words, the systems worked well on the former set, yet did not generalize to the latter. To prevent the issue, we would need to extend the validation set size, which may be tested in the future work. The final improvement that one can consider, is the use of more efficient hyperparameter optimization technique, for instance, a Genetic algorithm [43].

To sum up, we have addressed the research question, by proposing a suitable methodology, designing a

prediction framework that can robustly and effectively forecast Bitcoin price.

9.1.4. RQ 1.4: How well does a prediction system deal with non-stationarity and economic bubbles, and can semantic features derived from the considered data sources improve the forecast?

The last research question has been addressed by performing the experiments in chapter 8. Firstly, in section 8.1, we have investigated whether employing semantic features into the forecast boosts the results. Thus, the mean NMAE has been measured on validation and test sets for optimized systems, with different feature sets. Testing on the former set, indicates that addition of a single set of features from online sources improves the performance of a system. This is further confirmed on the latter set, for variables derived from comments and news. In order to statistically validate the significance of this performance variation, we used the Paired Sample T-Test between local NMAE scores of a given system and the 'M'. The test has proven with high confidence ($p<.05$) that 'MR' and 'MN' perform significantly better on the test set than the system based on market features only, which further supports the initial observation. However, one would not be able to prove it with higher confidence level than 95%, because the resulting p-values were close to 0.05. Thus, one can still be cautious about this finding. On the other hand, 'MT' performs worse on the test set, than 'M', which may be a result of overfitting on the validation set. Presumably, introducing modifications mentioned earlier to the hyperparameter optimization procedure, would allow it to achieve superior results than 'M'. One can also observe that the performance of the systems deteriorates, when more than one set of semantic features is employed in the forecast. Moreover, the Paired Sample T-Test has proven with high confidence ($p<.05$) that, indeed, the results significantly degraded. This is most probably caused by overfitting, due to too many features included into the model. As before, introducing improvements to the hyperparameter optimization procedure or employinga feature selection mechanism, could lead to further improvement of the forecast. Furthermore, one can also consider extracting a lower number of semantic clusters per data source. The main advantage of the performed experiment is the use of a large test set, which enabled extraction of high confidence conclusion. However, the designed methodology failed to deal with overfitting in some cases, especially for systems exploiting more than 12 variables, which is the main limiting factor of this discussion.

Secondly, section 8.2 studies the forecast in time of the economic bubbles. Since, the number of these event encountered in the test set is low, namely only 2, one cannot find the actual influence of the phenomenon on the prediction. However, we investigated the results achieved by the systems locally, and found some interesting patterns. At first, the prediction results in time of 'b5' and 'b6' were compared to the outside regions. For each prediction system, we have observed a lower NMAE in time of these bubbles, which indicates that there may be certain properties of the phenomenon, enabling enhanced prediction. We have proven that the difference of error is significant ($p<.05$) for 'b5', using the Mann-Whitney U test. Since it is only a single event, one cannot conclude any general relationship, however, it indicates that bubbles may have a positive influence on the prediction and one can investigate further in future work on a more representative dataset. On the other hand, the observation regarding 'b6' is insignificant ($p>.05$), thus, may have appeared by random. The second experiment considers the results achieved by different prediction systems on each segment of the two bubbles. In both cases, we witnessed a drop of error of around 10% during, so called, boom. The observation can be supported by the fact, that at the time, the price of Bitcoin grows in a stable way, over a long period of time. Thus, the models may simply predict the positive change of the asset's value, which results in a lower error. Consequently, based on this observation, we can strengthen the argument that there may be certain structural properties of the bubbles that enable superior prediction. However, we need to highlight that further investigation is required to make that conclusion with high confidence. Another observation made in the experiment is that all the prediction systems have expected the price downfall close to the highest peak of the price. Since, a peak of a bubble mostly appears after a long-term explosive growth of the price, it should be difficult for a prediction system to foresee the change of trend. However, in that case, all of them expected it long time prior to the downfall. This may be caused by the occurrence of several shifts of trends and price peaks in the train batches of these models. Therefore, the anomaly suggests that under certain conditions, a given predictive model can forecast a shift of trend in an economic bubble, even when the price grows to the value never witnessed before. Overall, the experiments showed several interesting patterns and anomalies that highlight topics that can be studied in the future. However, one cannot make any general conclusions regarding the influence of economic bubbles on the forecast, based on the dataset, which is the main limitation of our approach. The main difficulty of researching the topic is rarity of these events, therefore, one needs to find a time series containing multiple bubbles to address the issue.

Thirdly section 8.3 performs an experiment aiming to uncover how non-stationarity influences the re-

sults. It is investigated by testing how well the selected prediction systems forecast samples that are further into the future. Based on figure 8.3, we have observed that performance of all models deteriorates, when they are used to predict further into the future. The most probable explanation of this phenomenon is the influence of the concept shift, which is often present in non-stationary environment [45]. This effect is the most evident for the prediction system based on Reddit and the least for 'MT'. The negative influence has been confirmed using Wilcoxon signed rank test for 'M', 'MR', 'MN' ($p<.05$). However, we have not managed to reject the null hypothesis for 'MT' and its high p-value, namely 0.230, indicates that the system may be more robust to non-stationarity than other systems. Yet we suppose that the test would lead to the same conclusion as for other systems, if the sample size was more significant, which can be addressed in future work. Although this approach has allowed to partially answer the final part of the research question, its main limitation is the generated sample size. If we repeated the experiment with larger maximum number of steps forward, the results would be more significant, especially in case of 'MT'. The experiment also illustrates how crucial it is to make use of methodology for dealing with concept shift in a non-stationary environment. We suppose that if one does not employ any measures for dealing with this phenomenon while predicting Bitcoin price, the trained model will not achieve a stable performance over a long period of time. In other words, if a single model was trained, and applied to predict all the test samples, its performance would deteriorate over time. This also proves that while evaluating a given prediction system, one needs to employ a relatively large test set to be certain of the system's results. As already mentioned in section 9.1.3, the results may be biased, in case this set is too small and contains only samples that appear right after the train set. Knowing that the performance of a single model may deteriorate over time, it is neccessary to confirm stability of the proposed solution on a large test sample. Otherwise, one cannot trust that the system will be as efficient in the future as it currently is.

9.1.5. RQ 1: How do the non-stationarity and speculative bubbles encountered in the Bitcoin price signal influence its prediction and how to make the forecast more robust?

Once the answers of the research subquestions are presented, one can address the main objective of this study. Firstly, 9.1.4 describes how the non-stationarity and speculative bubbles influence the forecast, thus, we will briefly summarize the findings. We have discovered that non-stationarity has a negative effect on the forecast, mainly caused by the concept shift. If a model is trained on a portion of the current data, and applied to predict the future samples, it will gradually lose its predictive power, due to the detected patterns getting outdated. In case of economic bubbles, it was not possible to find the relationship between them and the forecast's results, due to the limitations of the dataset. However, some interesting observations have been made. The results of all prediction systems were significantly better in time of one of 'b5' and slightly better for 'b6'. Furthermore, there was a specific behaviour observed for every prediction systems in time of bubble's boom. Thus, the results suggest that there may be some structural properties of economic bubbles, that allow boosting the performance of the models, however, this hypothesis needs to be addressed in the future work.

In this study we approached the topic of increasing robustness of the prediction system to non-stationarity in two ways, namely by designing appropriate methodology and adding features from online sources. At first we ensured stationarity of the variables using relative differencing. Presumably, the step had a critical effect on the stability of the results, taking into consideration the explosive growth of the signal. Violation of the sample's identical distribution would most probably cause major deterioration of the model's performance. Then, the iterative approach for training and testing the models has been applied, which further increased robustness of the built prediction system, thanks to forecasting only the most recent samples, and then, retraining of the models. The experiment in section 8.3 indicated that using a single model to predict samples further into the future would cause results deterioration, which proves the necessity of the employed approach. Therefore, the main recommendation for building a robust system for predicting Bitcoin price is to frequently update the trained model with the most recent data. Otherwise, there is a high chance that its performance will deteriorate in the future.

Furthermore, the semantic features have been included into the forecast, to make the models more robust to non-stationarity. The same experiment shows that there is no major increase of model's robustness, and even, the Reddit-based variables boost the deterioration of the results. Only the 'MT' presented some promising results, however, it was not possible to make any conclusions, due to small data sample size. Even though the approach on its own does not boost model's robustness, applying it together, with the appropriate forecasting methodology improves efficiency of the system on the entire test set. Since in this thesis we have built the prediction framework that often retrains the models, these models are not required to form a stable

performance over long time, but rather high efficiency on the most recent data. In the experiment in section 8.1, we have shown that adding a single set of semantic features, based on Reddit or online news portals, has significantly enhanced the results in the non-stationary environment, on the test set of 20000 samples. Therefore, once the robustness of the system is assured by the frequent retraining of the models, one can employ some additional features to boost the results overall. However, with adding more variables, there is a higher risk of overfitting and performance deterioration.

9.2. Conclusion

In this research we aimed at researching the topic of Bitcoin price prediction, with special attention paid to forecasting a non-stationary variable with occurrence of economic bubbles. In order to do that, the data from multiple online sources, such as exchange markets, Twitter, Reddit and news portals was collected and analysed. After that, the processing steps extracted semantic features from text data and transformed the samples into the multivariate time series. Further, the appropriate methodology was designed to forecast Bitcoin price, which resulted in stable and efficient results. Finally, performed experiments allowed us to make conclusions, regarding the influence of non-stationarity on the forecast, and proved the efficiency our approach for predicting the variable. Moreover, semantic features from Reddit and news portals have brought a boost of performance to the prediction system. Apart from that, some interesting patterns and anomalies were detected, while forecasting in time of the economic bubbles, which can lead to discoveries made in the future work.

Bibliography

[1] Yaser S. Abu-Mostafa and Amir F. Atiya. Introduction to financial forecasting. *Applied Intelligence*, 6: 205–213, 1996.

[2] Ratnadip Adhikari and Ramesh K. Agrawal. An introductory study on time series modeling and forecasting. *LAP Lambert Academic Publishing*, 2013.

[3] Basant Agarwal and Namita Mittal. Text classification using machine learning methods-a survey. In B. V. Babu, Atulya Nagar, Kusum Deep, Millie Pant, Jagdish Chand Bansal, Kanad Ray, and Umesh Gupta, editors, *Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012*, pages 701–709, New Delhi, 2014. Springer India.

[4] Franklin Allen, Stephen Morris, and Andrew Postlewaite. Finite bubbles with short sale constraints and asymmetric information. *Journal of Economic Theory*, 61(2):206 – 229, 1993.

[5] Oren Anava, Elad Hazan, Shie Mannor, and Ohad Shamir. Online learning for time series prediction. *CoRR*, 2013.

[6] Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 6:1345–1382, December 2005.

[7] Jonathan Todd Barker. Why is bitcoin's value so volatile? `https://www.investopedia.com/articles/investing/052014/why-bitcoins-value-so-volatile.asp`. Accessed: 09-12-2017.

[8] Michèle Basseville and Igor V. Nikiforov. *Detection of Abrupt Changes - Theory and Application*. Prentice Hall, Inc., 1993.

[9] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, February 2012.

[10] Young Bin Kim, Jun Gi Kim, Wook Kim, Jae Ho Im, Tae Hyeong Kim, Shin Jin Kang, and Chang Hun Kim. Predicting fluctuations in cryptocurrency transactions based on user comments and replies. In *PloS one*, 2016.

[11] Olivier Blanchard and Mark Watson. Bubbles, rational expectations and financial markets. 07 1983.

[12] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3: 993–1022, March 2003.

[13] Michal Campr and Karel Ježek. Comparing semantic models for evaluating automatic document summarization. In Pavel Král and Václav Matoušek, editors, *Text, Speech, and Dialogue*, pages 252–260, Cham, 2015. Springer International Publishing.

[14] Kyunghyun Cho, B van Merrienboer, Caglar Gulcehre, F Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014.

[15] Stuart Colianni, Stephanie C. Rosales, and Michael Signorotti. Algorithmic trading of cryptocurrency based on twitter sentiment analysis. 2015.

[16] Hércules A. do Prado, Edilson Ferneda, Luis C.R. Morais, Alfredo J.B. Luiz, and Eduardo Matsura. On the effectiveness of candlestick chart analysis for the brazilian stock market. *Procedia Computer Science*, 22: 1136 – 1145, 2013. 17th International Conference in Knowledge Based and Intelligent Information and Engineering Systems - KES2013.

[17] Pedro Domingos. A few useful things to know about machine learning. 55:78–87, 10 2012.

[18] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[19] Ryan Elwell and Robi Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, Oct 2011.

[20] Zalan Forró. *Detecting Bubbles in Financial Markets: Fundamental and Dynamical Approaches*. ETH-Zürich, 2015.

[21] Kenneth Froot and Maurice Obstfeld. Intrinsic bubbles: The case of stock prices. 81:1189–214, 01 1991.

[22] João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. 8:286–295, 09 2004.

[23] Ifigeneia Georgoula, Demitrios Pournarakis, Christos Bilanakos, Dionisios N. Sotiropoulos, and George M. Giaglis. Using time-series and sentiment analysis to detect the determinants of bitcoin prices. In *MCIS*, 2015.

[24] C. Lee Giles, Steve Lawrence, and Ah Chung Tsoi. Noisy time series prediction using recurrent neural networks and grammatical inference. *Machine Learning*, 44(1):161–183, Jul 2001.

[25] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[26] Alex Greaves and Benjamin Au. Using the bitcoin transaction graph to predict the price of bitcoin. 2015.

[27] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. 28:2222–2232, 03 2015.

[28] James D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.

[29] Jan Hauke and Tomasz Kossowski. Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data. 30:87–93, 06 2011.

[30] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[31] Guido W. Imbens and Tony Lancaster. Efficient estimation and stratified sampling. *Journal of Econometrics*, 74(2):289 – 318, 1996.

[32] Rafal Józefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In *ICML*, 2015.

[33] Andrej Karpathy, Justin Johnson, and Fei-Fei Li. Visualizing and understanding recurrent networks. *CoRR*, 2015.

[34] Charles .P. Kindleberger, Robert Z. Aliber, and R. Solow. *Manias, Panics and Crashes: A History of Financial Crises, Sixth Edition*. Palgrave Macmillan, 2011.

[35] David Kriesel. *A Brief Introduction to Neural Networks*. 2007. URL `availableathttp://www.dkriesel.com`.

[36] Vivek Kumar Rangarajan Sridhar. Unsupervised topic modeling for short texts using distributed representations of words. pages 192–200, 01 2015.

[37] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages 1188–1196, 2014.

[38] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. 521, 05 2015.

[39] Georg Lindgren. *Stationary Stochastic Processes: Theory and Applications*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, 2012.

[40] Isaac Madan and Shaurya Saluja. Automated bitcoin trading via machine learning algorithms. 2014.

[41] Andrew Malinowski, Tomasz J. Cholewo, and Jacek M. Zurada. Capabilities and limitations of feed-forward neural networks with multilevel neurons. In *Circuits and Systems, 1995. ISCAS '95., 1995 IEEE International Symposium on*, volume 1, pages 131–134, 1995.

[42] Martina Matta, Maria Ilaria Lunesu, and Michele Marchesi. Bitcoin spread prediction using social and web search media. In *UMAP Workshops*, 2015.

[43] John McCall. Genetic algorithms for modelling and optimisation. *Journal of Computational and Applied Mathematics*, 184(1):205 – 222, 2005.

[44] Sean McNally. Predicting the price of bitcoin using machine learning. 2016.

[45] Marcelo Mendoza, Barbara Poblete, Felipe Bravo-Marquez, and Daniel Gayo-Avello. Long-memory time series ensembles for concept shift detection. In *Proceedings of the 2Nd International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, BigMine '13, pages 23–30, New York, NY, USA, 2013. ACM.

[46] Tomas Mikolov, Kai Chen, G.s Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. 2013, 01 2013.

[47] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119. Curran Associates Inc., 2013.

[48] Ryan Mitchell. *Web scraping with Python: collecting data from the modern web*. " O'Reilly Media, Inc.", 2015.

[49] Norazian Mohamed Noor, Mohd Mustafa Al Bakri Abdullah, Ahmad Shukri Yahaya, and Nor Azam Ramli. Comparison of linear interpolation method and mean method to replace the missing values in environmental data set. 803:278–281, 08 2014.

[50] Arman Khadjeh Nassirtoussi, Saeed Reza Aghabozorgi, Ying Wah Teh, and David Chek Ling Ngo. Text mining for market prediction: A systematic review. *Expert Systems with Applications*, 41:7653–7670, 2014.

[51] Christopher Olah. Understanding lstm networks. `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`. Accessed: 2018-03-03.

[52] Oleg Ostashchuk. *Time Series Data Prediction and Analysis*. Master's thesis, Czech Technical University in Prague, Prague, 2017.

[53] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *ICML*, 2013.

[54] Peter C. B. Phillips and Jun Yu. Dating the timeline of financial bubbles during the subprime crisis. *Quantitative Economics*, 2(3):455–491.

[55] Peter C. B. Phillips, Shuping Shi, and Jun Yu. Specification sensitivity in right-tailed unit root testing for explosive behaviour. *Oxford Bulletin of Economics and Statistics*, 76(3):315–333, .

[56] Peter C. B. Phillips, Shuping Shi, and Jun Yu. Testing for multiple bubbles: Historical episodes of exuberance and collapse in the s&p 500. *International Economic Review*, 56(4):1043–1078, .

[57] Dixa Saxena. Survey paper on feature extraction methods in text categorization. 166:11–17, 05 2017.

[58] Deerwester Scott, Dumais Susan T., Furnas George W., Landauer Thomas K., and Harshman Richard. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6): 391–407.

[59] Thomas Spooner, John Fearnley, Rahul Savani, and Andreas Koukorinis. Market Making via Reinforcement Learning. *ArXiv e-prints*, April 2018.

[60] Arjun Srinivas Nayak, Ananthu P Kanive, Naveen Chandavekar, and Balasubramani . Survey on preprocessing techniques for text mining. 5:2319–7242, 06 2016.

[61] Troy Steinbaur. Information and social analysis of reddit. *Retrieved from TROYSTEINBAUER@ CS. UCSB. EDU*, 2012.

[62] Evita Stenqvist and Jacob Lonno. Predicting Bitcoin price fluctuation with Twitter sentiment analysis. Master's thesis, KTH Royal Institute OF Technology, Stockholm, Sweden, 2017.

[63] Francesco Strati. Bitcoin, bubbles and proxies. page 15, 12 2017.

[64] Peter D. Turney. The management of context-sensitive features: A review of strategies. *CoRR*, 2002.

[65] Dennis Wackerly, William Mendenhall, and Richard L. Scheaffer. *Mathematical Statistics with Applications*. International student edition / [Brooks-Cole]. Cengage Learning, 2014.

[66] Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, Apr 1996.

[67] Wesley Willett, Jeffrey Heer, and Maneesh Agrawala. Strategies for crowdsourcing social data analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 227–236, New York, NY, USA, 2012. ACM.