Master's thesis
Master's Programme in Data Science

Predicting the price of Bitcoin using the sentiment of popular Bitcoin-related Tweets

Eino Keningi

April 20, 2022

Supervisor(s):	Assoc. Prof. Michael Mathioudakis
Examiner(s):	Assoc. Prof. Michael Mathioudakis
	Prof. Jiaheng Lu

UNIVERSITY OF HELSINKI
FACULTY OF SCIENCE

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty Faculty of Science	Koulutusohjelma — Utbildningsprogram — Degree programme Master's Programme in Data Science

Tekijä — Författare — Author

Eino Keningi

Työn nimi — Arbetets titel — Title

Predicting the price of Bitcoin using the sentiment of popular Bitcoin-related Tweets

Työn laji — Arbetets art — Level Master's thesis	Aika — Datum — Month and year April 20, 2022	Sivumäärä — Sidantal — Number of pages 55

Tiivistelmä — Referat — Abstract

In little over a decade, cryptocurrencies have become a highly speculative asset class in global financial markets, with Bitcoin leading the way. Throughout its relatively brief history, the price of bitcoin has gone through multiple cycles of growth and decline. As a consequence, Bitcoin has become a widely discussed – and polarizing – topic on Twitter.

This work studies whether the sentiment of popular Bitcoin-related tweets can be used to predict the future price movements of bitcoin. In total, seven different algorithms are evaluated: Vector Autoregression, Vector Autoregression Moving-Average, Random Forest, XGBoost, LightGBM, Long Short-Term Memory, and Gated Recurrent Unit. By applying lexicon-based sentiment analysis, and heuristic filtering of tweets, it was discovered that sentiment-based features of popular tweets improve the prediction accuracy over baseline features (open–high–low–close data) in five of the seven algorithms tested. The tree-based algorithms (Random Forest, XGBoost, LightGBM) generally had the lowest prediction errors, while the neural network algorithms (Light Short-Term Memory and Gated Recurrent Unit) had the poorest performance. The findings suggest that the sentiment of popular Bitcoin-related tweets can be an important feature in predicting the future price movements of bitcoin.

ACM Computing Classification System (CCS):
Information systems → Intormation retrieval → Retrieval tasks and goals → Sentiment analysis
Mathematics of computing → Probability and statistics → Statistical paradigms → Time series analysis
Computing methodologies → Machine learning → Learning paradigms → Supervised learning

Avainsanat — Nyckelord — Keywords

bitcoin, time series, sentiment analysis, machine learning, price prediction

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Contents

1. Introduction

Investor sentiment in the stock market has been researched extensively. Over time, various potential proxies for investor sentiment have been studied, such as investor surveys, retail investor trades, mutual fund flows, and trading volume [1]. With the rise of social media and mass communication, new potential proxies have emerged, as public discussions on social media have been used to estimate investor sentiment [2, 3]. With the advances in sentiment analysis methods and computing power, large-scale analysis can now be performed to extract insights about general sentiment.

Since the launch of Bitcoin in 2009, cryptocurrencies have steadily gained significance in global financial markets. In November 2021, cryptocurrency market capitalization reached a \$2.9 trillion mark, of which Bitcoin accounted for $\sim$44% [4]. From the beginning, Bitcoin has been discussed and speculated on the internet. Bitcointalk, an internet forum created in 2009 by Satoshi Nakamoto, the pseudonymous inventor of Bitcoin, has since amassed over 59 million forum posts [5]. On Twitter, Bitcoin discussion has been steadily growing; on 21.5.2021, Bitcoin was mentioned in approximately 355,000 Tweets in the span of 24 hours [6, 7]. Prominent people, such as Goldman Sach's former CEO Lloyd Blankfein, and Twitter's former CEO Jack Dorsey, among others, have voiced their public opinions on Bitcoin on Twitter [6]. Many "cryptocurrency influencers" also use the platform to promote existing and new cryptocurrencies [8]. As a recent trend, paid celebrity endorsements of cryptocurrency-related platforms and services have also appeared on Twitter as an attempt to attract new customers [9]. For example, several professional athletes have endorsed cryptocurrency trading platforms on Twitter [10].

The aim of this thesis is to investigate, whether the sentiment of popular Bitcoin-related tweets on Twitter can be a useful feature in predicting future price movements of bitcoin, and if so, which prediction methods yield the best results. By leveraging public online discussions and speculation of Bitcoin on Twitter, a compound user sentiment will be extracted, and various multivariate time series forecasting methods will be evaluated in the task of predicting the future price of bitcoin.

As a primary dataset, a collection of 17.7 million English language tweets containing the keyword "bitcoin", posted between August 2017 and January 2019, will be used as a source for the Twitter sentiment. A supplementary open–high–low–close (OHLC) price dataset of bitcoin, describing the price movements of bitcoin from the same time period will also be used. The OHLC price information has traditionally been used in quantitative price analysis [11] and will serve as a baseline for the predictions. The value of the extracted sentiment-based features will be investigated by comparing the predictive performance between two feature sets: baseline (OHLC data), and augmented (OHLC and sentiment data). Prediction lengths of 1, 3, 5, and 7 days will be evaluated.

Previous studies have shown that online sentiment can influence cryptocurrency prices, both positively and negatively [12, 13]. The role of Twitter as a source of investor sentiment has also been studied with encouraging results [14, 15]. And while the efficacy of Twitter sentiment in predicting the future price of bitcoin has been studied before [16, 17, 18, 19], there has yet to be a comparative study that evaluates the predictive performance of different statistical and machine learning-based models over the same tweet dataset and time period. Due to the differences in tweet datasets and cryptocurrency market conditions over time, the results between different studies can be hard to generalize. The contribution of this thesis is to provide a better evaluation of different time series forecasting methods in predicting the future price of bitcoin using sentiment extracted from popular tweets.

Throughout this thesis, Bitcoin (in uppercase), will refer to the protocol and the concept of Bitcoin, whereas bitcoin (in lowercase) will strictly refer to the currency (BTC).

2. Related work

This thesis builds on the existing body of research done in the areas of time series forecasting, sentiment analysis, and social media bot detection.

2.1 Sentiment Analysis

In recent years, many papers have been published on using social media posts as a proxy for cryptocurrency investor sentiment.

Hutto et al. [20] presented VADER, a rule-based sentiment analysis system geared specifically towards social media messages. Since its release, VADER has been widely used in social media sentiment analysis [21, 22, 16, 23, 19]. Oliveira et al. [24] created a stock market lexicon geared toward financial sentiment analysis. The training corpus was created from messages on StockTwits, a Twitter-like social media platform aimed at investors.

Kraaijeveld et al, [21], used bivariate Granger-causality to investigate the predictive power of Twitter sentiment in predicting the price returns for the nine largest cryptocurrencies. In order to extract sentiment from cryptocurrency-themed messages, a custom cryptocurrency-specific lexicon was used in addition to the standard lexicon. Kraajieveld et al. found that out of the nine cryptocurrencies, Twitter sentiment has predictive power for the returns of Bitcoin, Bitcoin Cash, and Litecoin.

Guegan et al. [25] studied the relationship between investor sentiment and bitcoin returns. Approximately one million messages on StockTwits were used for the analysis. A statistically significant relationship between investor sentiment and bitcoin returns for frequencies of up to 15 minutes was found. However, the overall impact of the effect was described as insufficient to gain any noticeable advantage.

Lyócsa et al. [26] studied the effect of news headlines on the volatility of the price of bitcoin. Various categories of news headlines were used, such as regulatory news, and news about hacking of Bitcoin exchanges. They discovered that the volatility of bitcoin is most strongly affected by news on Bitcoin regulation and hacking attacks on cryptocurrency exchanges.

Pano et al. [22] analyzed the optimal text preprocessing pipeline for sentiment analysis of Bitcoin tweets. Pearson autocorrelation scores were used in evaluating different approaches. A short-term correlation between sentiment and the price of Bitcoin was found. It was found that text preprocessing improves the correlation slightly.

Other sentiment analysis techniques have also been used to determine investor sentiment. Balfagih et al. [27] evaluated different sentiment classifiers, such as MLP, WiSARD, and RATS. The analyzed sentiment was used to predict the future price of bitcoin. It was found that manual Tweet labeling produced better results than machine learning-based tools. Naeem et al. [18] examined the predictive ability of Twitter Happiness Sentiment for six major cryptocurrencies using daily data between 7. August 2015 – 31. December 2019. A significant nonlinear relationship between sentiment and cryptocurrency prices was found.

2.2 Social Media Bot Detection

The widespread presence of social media bots in cryptocurrency-related tweets has been studied. Due to the variation between different datasets, varying estimates have been given.

Varol et al. [28] investigated the prevalence of bots on Twitter, as well as ways to detect them. They estimated that between 9% to 15% of active Twitter users might be bots. Similarly, Kraaijeveld et al. [21] estimated that at least 1-14% of the Tweets in their dataset were posted by bot accounts.

Mirtaheri et al. [29] identified and analyzed cryptocurrency manipulations, such as pump-and-dump schemes, on social media platforms, namely Twitter and Telegram. They estimate that the majority of the active users participating in pump-and-dump schemes were bot accounts. Similarly, Nizzoli et al. [30] charted the landscape of online cryptocurrency manipulation, attempting to detect various cryptocurrency-related Ponzi and pump-and-dump schemes on Twitter, Telegram, and Discord. According to Nizzoli, the proportion of bots that Varol et al. [28] estimated (9–15%) could be even higher in the cryptocurrency sphere, as high as 50–90%, depending on the type of posts.

2.3 Cryptocurrency Price Prediction

Several methods of predicting the future price of cryptocurrencies have been investigated, spanning a wide variety of assorted features and prediction models.

Chowdhury et al. [31] used price movement data (OHLC) to forecast the closing price of the Cryptocurrency Index 30, and nine other cryptocurrencies. Gradient boosted trees, k-Nearest Neighbor, and Long Short-Term Memory (LSTM) neural net models were used. Similarly, Patel et al. [32] studied using price movement data to predict the price of Bitcoin, Litecoin, Ripple, Monero, Tether, and IOTA. The predictions were done using LSTM and Gated Recurrent Unit (GRU) neural nets. A hybrid GRU + LSTM model was also proposed. Azari et al. [33] studied an ARIMA-based univariate approach to bitcoin price prediction. ARIMA was found to be efficient in short-period forecasts, such as predicting the next day's price.

Pant et al. [22] used recurrent neural networks to predict the price of bitcoin. The features were derived from Twitter sentiment. The sentiment analysis was performed using an NLP-based pipeline. Shen et al. [14] investigated a link between investor attention (in the form of Tweet volume) and Bitcoin returns using a vector autoregressive (VAR) model. Inamdar et al. [17] predicted cryptocurrency prices using sentiment analysis. The data for sentiment analysis consisted of Twitter messages and various news headlines. The predictive models were developed using the Random Forest algorithm. Colianni et al. [34] experimented if Twitter messages relating to cryptocurrencies can be utilized to develop advantageous cryptocurrency trading strategies using supervised machine learning models. Abraham et al. [23] used tweet volumes and VADER-based sentiment analysis to predict the prices of bitcoin and Ethereum, using multiple linear regression. Hao et al. [19] used Twitter sentiment to predict bitcoin price movements. Sentiment analysis was done with VADER, and the prediction model was using the XGBoost algorithm. A novel "interactivity" feature was also proposed, based on tweet statistics, such as the number of likes, retweets, and comments. Kim et al. [35] attempted to predict the fluctuations in the prices of Bitcoin, Ethereum and Ripple, by analyzing user comments on popular cryptocurrency forums. Sentiment analysis of the comments was done using VADER. Li et al. [36] used sentiment-based gradient boosted tree models to predict fluctuations in the price of ZClassic alt-coin. The Twitter sentiment was found to have a strong link between the price movements in ZClassic.

Features drawn from the Bitcoin network itself have also been used to predict the price of bitcoin. Dutta et al. [37] used variables drawn from bitcoin network-related statistics, such as hash rate, transaction fees, and miner revenues, to predict the price of bitcoin. GRU neural network models were used in the prediction task. Similarly, Ji et al. [38] used LSTM, and GRU neural network models to predict the price of bitcoin, deriving features from various blockchain statistics.

3. Background

3.1 Bitcoin

In the original Bitcoin whitepaper [39] released in 2008, Satoshi Nakamoto offered an alternative to existing online payment solutions by proposing a trust-free electronic cash system, backed by a cryptographically verifiable blockchain, which functions as an immutable public ledger for all transactions.

Bitcoin works in a decentralized manner. There are no regulatory entities that can control the supply of bitcoin, making artificial inflation or deflation of the currency impractical. The supply of new bitcoins is predictable and consistent; a predetermined amount of new bitcoins is created every 10 minutes when a new block is added to the blockchain. Due to the limited supply, the price of bitcoin is largely determined by its demand. This demand can often be influenced by public perceptions, opinions, or news surrounding Bitcoin. In the past, cryptocurrency markets have reacted swiftly to favorable and unfavorable news about Bitcoin. For example, previous attempts to regulate Bitcoin in China [40, 41] resulted in heavy price shifts in the cryptocurrency markets. For example, on 19.5.2021, after the Chinese government's plans to regulate cryptocurrencies were made public, the price of bitcoin fell in the span of 24 hours almost 30% from $43,546.12 to $30,681.50. Just a month earlier, on 14.4.2021, the price of bitcoin had peaked at its then all-time high of $63,109.69 [42]. Similarly, several Bitcoin-related tweets made by Tesla and SpaceX CEO Elon Musk have caused significant price swings in the price of bitcoin [43, 44].

Such events have demonstrated that the demand for bitcoin can be unpredictable. However, over time, the growth of the price of bitcoin has been exponential [45]. Due to the deflationary aspect of Bitcoin, its function as a payment system has since become less appealing [46]. Instead, the focus has shifted more towards the lucrative potential of bitcoin as an investment opportunity and a store of value, similar to gold and other precious metals [47].

3.2 Sentiment Analysis

Sentiment analysis is an area in natural language processing. Pozzi et al. [48] define sentiment analysis as the task of creating automated tools that can extract subjective characteristics, such as sentiment, opinions, and emotions, from written text [48]. In the same work, they note that sentiment analysis consists of many subtasks, such as polarity classification, opinion summarization, subjectivity classification, sarcasm detection, and detection of misleading opinions, among others. Throughout this thesis, the term "sentiment" will strictly refer to the polarity of the text, i.e., whether the text appears positive, neutral, or negative [48].

3.2.1 VADER (Valence Aware Dictionary for sEntiment Reasoning)

VADER is a sentiment analysis model, originally introduced in a paper by Hutto et al. [20]. In its documentation [49], the authors explain its scoring mechanism. For some input of text, VADER will output its *compound* sentiment score, which is a sum of all adjusted valence (intensity) scores of words in the lexicon [49]. The compound score is a decimal value between $[-1, 1]$ (-1 being the most negative, 1 being the most positive). The compound score can also be interpreted as a categorical sentiment value, if needed, by using the threshold values suggested by the authors [49]:

$$\text{Sentiment(compound)} = \begin{cases} \textsc{positive}, & \text{if compound} > 0.05 \\ \textsc{neutral}, & \text{if } -0.05 \leq \text{compound} \geq 0.05 \\ \textsc{negative}, & \text{if compound} < -0.05 \end{cases} \tag{3.1}$$

In addition to the compound score, VADER also outputs the proportions of positive, neutral, and negative words in the text [49].

Hutto et al. [20] describe VADER as a rule-based model, which features its own gold-standard lexicon, in which each word includes a valence score, a decimal value between $[-4, 4]$, indicating the *intensity* of the word. When determining the compound sentiment, VADER considers the valence of each word in the sentence [20]. For example, words "okay" and "best" have intensity scores of 0.9, and 3.2, respectively, while words "uncomfortable" and "horrific" have intensity scores of -1.6 and -3.4, respectively [49]. The gold-standard lexicon was created assessing random tweets on Twitter, making VADER particularly effective in social media contexts [20]. As a result of this, the lexicon also includes present-day Internet colloquialisms, such as initialisms (LOL, WTF), emoticons (:(, ;-)), as well as emojis (😸❤️👍) [20].

In addition to using the gold-standard lexicon, VADER also utilizes five grammatical and syntactical heuristics, which allow for a more detailed analysis of the sentiment [20]. The five heuristics are:

1. **Punctuation**. Using punctuation, especially the exclamation point, can increase the intensity of a sentence, i.e., "Buy Bitcoin!!!" is more intense than "Buy Bitcoin" [20].

2. **Capitalization**. CAPITALIZED words can intensify a sentence, i.e., "BITCOIN BANNED IN CHINA" is more intense than "Bitcoin banned in China" [20].

3. **Degree modifiers**. The use of various degree modifiers can increase or decrease the intensity of a sentence, i.e. "This is *extremely* good for Bitcoin" is more intense than "This is *barely* good for Bitcoin" [20].

4. **Contrastive "but"**. The sentence "I love to pay with Bitcoin, *but* sadly it hasn't gained a wider acceptance." has a shift in sentiment after the word *but* [20].

5. **Trigram analysis**. According to the authors, analyzing words in groups of three can detect almost 90% of long-range negations that flip the meaning of a sentence, i.e., "The future of Bitcoin *isn't* really clear" [20].

Compared with machine learning-based sentiment analysis approaches, VADER does not require any training data, or training, since a lexicon is already included [20]. Also, being entirely rule-based, VADER can rapidly process enormous quantities of text. When using VADER through Natural Language Toolkit (NLTK) library in Python, it is possible to customize the lexicon by adding additional tokens and valence values [50].

3.3 Time series forecasting

3.3.1 Time series data

Time series data is an ordered sequence of data over some period of time; each row representing a discrete point in time. In the following table 3.1, five sample rows from a bitcoin price time series dataset are shown. Each row in the dataset describes various characteristics of bitcoin's price against the USD at a specific point in time, indicated in the "Timestamp" column as a UNIX timestamp.

Timestamp	Open	High	Low	Close	Volume_(BTC)	Volume_(Currency)	Weighted_Price
1385317920	831.00	831.69	830.00	830.00	17.443817	14484.402712	830.345922
1385317980	830.02	831.68	830.00	831.68	0.472665	392.713864	830.850613
1385318040	831.69	838.98	831.68	831.68	8.058825	6746.357068	837.139066
1385318100	830.01	830.01	829.83	830.00	9.255889	7682.377290	829.998856
1385318160	831.68	831.68	827.99	827.99	16.212175	13455.615370	829.969789

Table 3.1: Five sample rows from the BTC–USD dataset.

Time series components

Time series data can be decomposed into three distinct components [51]:

- **Trend component:** The trend component T represents long-term upward and downward tendencies in the data,
- **Seasonal component:** The seasonal component S represents any seasonal variation in the data that repeats at a regular frequency (daily, weekly, monthly, quarterly, yearly, etc.), e.g. the sale data of sunscreen has a seasonal component in that more sunscreen is sold during summer than winter months,
- **Residual component:** The residual (also known as irregular) component R represents the remaining irregular residuals that are left after removing trend and seasonal components.

Together, these three components can be used to model time series data y at period t. The decomposition can be additive, as defined in equation 3.2, or multiplicative, defined in equation 3.3 [51].

$$y_t = T_t + S_t + R_t \tag{3.2}$$

$$y_t = T_t \times S_t \times R_t \tag{3.3}$$

A multiplicative model, as shown in equation 3.3, can be transformed into an additive model using logarithmic transformation, as shown in equation 3.4 [51].

$$y_t = T_t \times S_t \times R_t \quad \text{is equivalent to} \quad \log y_t = \log T_t + \log S_t + \log R_t. \tag{3.4}$$

It depends on the variation in the data, whether to choose an additive or multiplicative decomposition [51]. In the case of the price of bitcoin, the daily variations in price are not constant and can change in orders of magnitude, suggesting that a multiplicative decomposition should be used. A multiplicative time series decomposition of the closing price of bitcoin was done using the `statsmodels` package for Python and is shown in figure 3.1, showing the trend, seasonal, and residual components of the data. The time series decomposition was done using the classical method, in which the trend component T is calculated using a moving average [51].

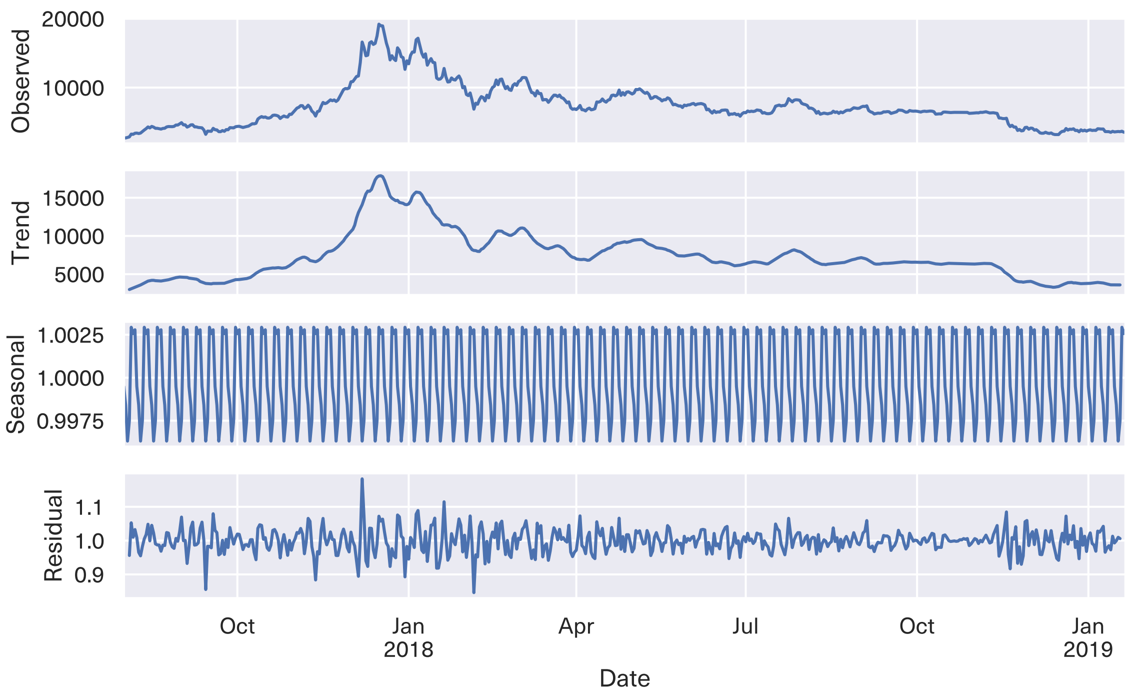


Figure 3.1: Multiplicative time series decomposition of the bitcoin closing price data. The insignificant variation in the Seasonal component indicates that there is no strong seasonal influence on the price of bitcoin.

Stationarity

One property of time series data is its stationarity. A stationary time series data does not exhibit an observable trend or seasonality, and the variation of the data remains constant [51]. Some forecasting models, such as Vector Autoregression (VAR), and Vector Autoregression Moving-Average (VARMA), require that the input data is stationary [52].

Stationarity cannot always be clearly determined by eye, which is why various tests have been developed to help objectively ascertain whether the data is stationary or not [51]. The *Augmented Dickey-Fuller* (ADF) test is one such test. It determines whether the trend in a time series is stochastic or deterministic. The null hypothesis in the ADF test assumes that the data is non-stationary. If the p-value of the ADF test is less than 0.05, the null hypothesis is rejected, and the data is assumed stationary.

It is possible to transform non-stationary time series data into stationary data by applying *differencing* to the data. The first-order difference of a time series y_t is defined in equation 3.5.

$$Dy_t = y_t - y_{t-1} \tag{3.5}$$

Differencing reduces trend and seasonality of a time series data by stabilizing its mean, as the difference is computed between consecutive timesteps [51]. With financial time series, e.g. price of a stock, the differenced data will describe returns (gain or loss), instead of the price.

A differenced time series can also be transformed back into the original, un-differenced time series by creating a vector y', where the differenced time series $\{Dy\}$ is prepended by the initial value y_0 in the original time series (equation 3.6). Original, un-differenced values can now be restored by taking the cumulative sum of each element, as shown in equation 3.7.

$$y' = \{y_0, Dy_0, Dy_1, \ldots, Dy_{t-1}\} \tag{3.6}$$

$$y_t[k] = \sum_{i=1}^{k} y'[i] \tag{3.7}$$

Logarithmic differencing can also be used, as shown in equations 3.8, 3.9, and 3.10. Logarithmic differencing works only if all data points in the time series data are positive.

$$\log Dy_t = \log y_t - \log y_{t-1} \tag{3.8}$$

$$\log y' = \{y_0, \log Dy_0, \log Dy_1, \ldots, \log Dy_{t-1}\} \tag{3.9}$$

$$y_t[k] = e^{\sum_{i=1}^{k} \log y'[i]} \tag{3.10}$$

The bitcoin closing price data is *non-stationary*, as it shows visible up and downtrends over time (see Trend subplot in figure 3.1). Bitcoin closing price data is shown before and after differencing in figure 3.2, along with p-values from the Augmented Dickey-Fuller test.

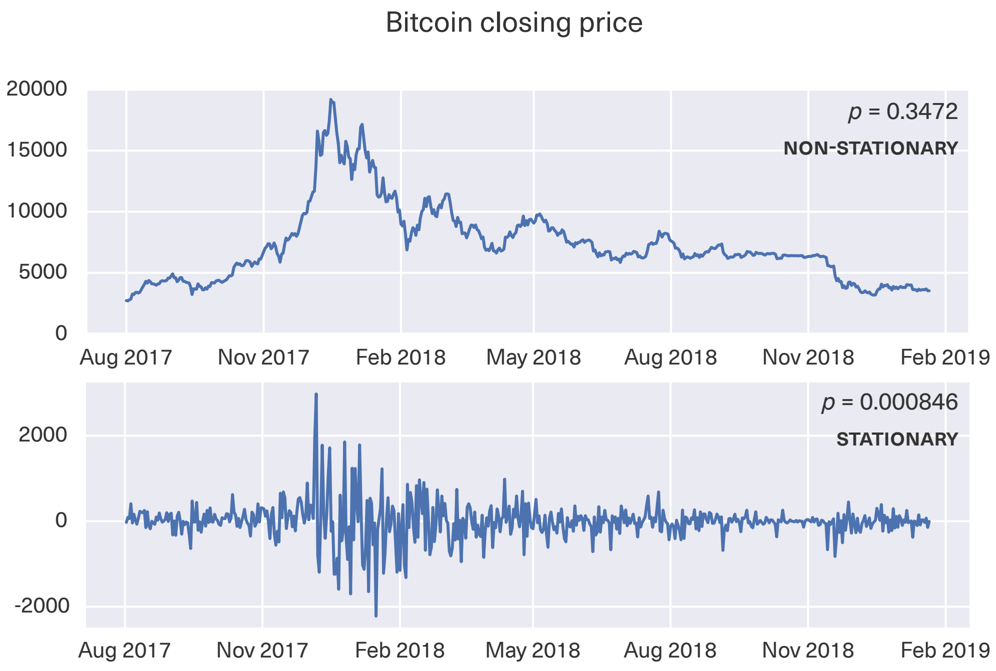


Figure 3.2: Non-stationary bitcoin closing price data, and the same data, now stationary, after applying first-order differencing. The top right corner p-values are from the Augmented Dickey-Fuller test.

3.3.2 Statistical models

Vector Autoregression (VAR)

Vector Autoregression (VAR) is a statistical time series model, originally proposed by Christopher A. Sims, in a 1980 paper *Macroeconomics and Reality* [53]. VAR merges multiple variables into a single vector y_t, where t is the number of variables. The principal feature of VAR is the *order* of the model. The order p represents the *lag* value, i.e., how many steps back the model can see when making a future prediction. For example, a VAR(5) model uses five previous time steps to predict the next time step. VAR requires all included variables to be stationary [52]. VAR(p) for t samples and a sample size T is defined in equation 3.11,

$$y_t = \nu + A_1 y_{t-1} + \cdots + A_p y_{t-p} + \varepsilon_t, \tag{3.11}$$

where $y_t = (y_t, \ldots, y_{kt})'$ is a $t \times 1$ matrix of variables, ν is a $t \times 1$ matrix of constants for each variable, p is the order of the model, i.e., the lag, A_i is a $k \times k$ matrix of coefficients for each lag, and $\varepsilon_t = (\varepsilon_t, \ldots, \varepsilon_{kt})'$ is the statistical error [54, 55].

The coefficients are estimated using a method of multivariate least squares [54]. By writing the VAR model in a compact matrix form as

$$Y = BZ + U, \tag{3.12}$$

where Y is the y value matrix $[y, \ldots, y_T]$, B is the coefficient matrix $[v, A_1, \ldots, A_p]$, Z is the matrix $[Z_0, \ldots, Z_{T-1}]$, where Z_i is a matrix of the values of y with a lag i, and U is the error matrix $[\varepsilon_0, \ldots, \varepsilon_T]$, the coefficients can be estimated using the matrix calculation described in equation 3.13 [54].

$$\hat{B} = YZ^T \left(ZZ^T\right)^{-1} \tag{3.13}$$

Vector Autoregression Moving–Average (VARMA)

Vector Autoregression Moving–Average (VARMA) is an extended version of the vector autoregression (VAR) model. In VARMA, a moving average MA(q) component is added to the VAR model. The order q in MA(q) specifies how many previous steps are included in the calculation of the moving average. VARMA is defined in equation 3.14.

$$y_t = \nu + A_1 y_{t-1} + \cdots + A_p y_{t-p} + u_t + M_1 u_{t-1} + \cdots + M_q u_{t-q}, \tag{3.14}$$

where $A_0, \ldots, A_p$ are $k \times k$ matrices of coefficients for each lag, u_t is the statistical error term, and $M_0, \ldots, M_q$ are $k \times k$ matrices of moving averages [54]. When estimating the coefficients of a VARMA model, maximum likelihood estimation is used, making it computationally more complex than a VAR model [54]. As a result, VARMA models can be time-consuming to fit with a large number of variables.

3.3.3 Tree-based machine learning models

Time series as a supervised learning problem

Time series data can be transformed into data that is compatible with supervised learning models. To predict the value after n timesteps, both the target and feature columns need to be duplicated and delayed by $n+1, n+2, \ldots, k+n$ steps, where k is the size of the lookback window. After such transformation, a single row will now include columns that include previous values, delayed by at least $n+1$ days. The method is illustrated in figure 3.3.

t	a	b
...	...	...
...	...	...
...	...	...

$$\downarrow$$

t	a	$a(n+1)$	$b(n+1)$	$a(n+2)$	...	$b(k+n)$
...	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	...

Figure 3.3: Transforming time series data into data that is compatible with supervised learning models. t is the time column, a is the target variable, b is a feature variable. $y(n)$ indicates a delay of n time steps applied to column y. k is the size of the lookback window.

Random Forest

The Random Forest algorithm is an ensemble learning method that was originally proposed in a 2001 paper by Leo Breiman [56]. It is a flexible algorithm that is suitable for both regression and classification tasks [57]. The Random Forest algorithm has shown to be performant in a large variety of classification tasks [58].

The training process is based on the tree bagging method, in which multiple smaller decision trees are trained on small, randomly sampled subsets of the data, and their predictions are aggregated, either by using a majority vote among the decision trees or a local average, depending on the classification task [57]. The features used in each decision tree split are randomized, and their importances are computed throughout the training process [56]. A schematic of the Random Forest algorithm is shown in figure 3.4.

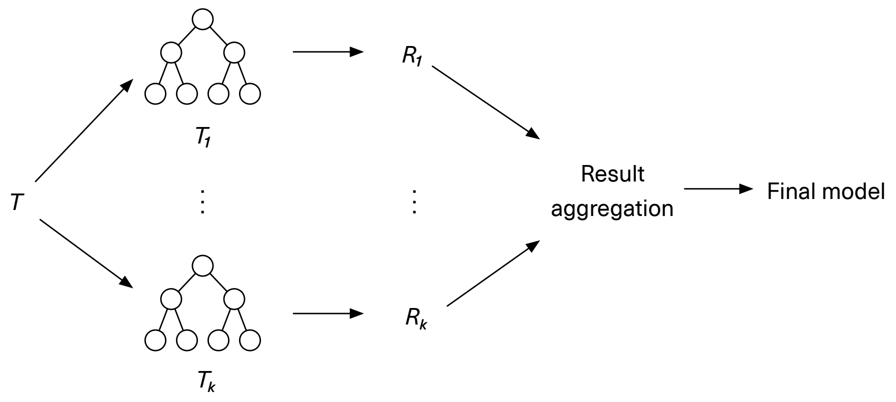


Figure 3.4: A schematic view of the Random Forest algorithm. Training data T is divided into smaller subsets, from which decision trees $T_1 \dots T_k$ are created. The results $R_1 \dots R_k$ get aggregated, and the final model is created.

The Random Forest algorithm has adjustable hyperparameters, which can be modified to refine the performance of the model. Many parameters in the learning process can be adjusted for the type of data that is being used. The number of decision trees, and their depth, can be adjusted, as well as the number of randomly selected features in the decision tree splits.

XGBoost and LightGBM

XGBoost is another decision tree-based model, originally presented in a 2016 paper by Chen et al. [59]. Compared to Random Forests, which used tree bagging during the training process, XGBoost leverages an iterative method called tree boosting. In tree boosting, the training process consists of creating a sequence of decision trees, in which each new decision tree focuses on improving the residuals of the previous ones [60]. Each new decision tree is created so that it maximizes the fit of the residuals, or gradients, of the previous trees, hence, *boosting* the gradient. In contrast to Random Forests, XGBoost ranks each split point using a histogram-based method, which ranks and chooses the best split point for each variable [61]. A schematic of the tree boosting method is shown in figure 3.5.

LightGBM is another gradient-boosting decision tree model, first published in a 2017 paper by Ke et al. [62]. LightGBM introduces an efficient tree-splitting method called Gradient-Based One-Side Sampling, which improves on the histogram-based method by automatically discarding trees with low residuals, and instead focusing on trees with higher gradients [62]. LightGBM also utilizes Exclusive Feature Bundling, which increases the speed of training by combining mutually correlated features together [62].

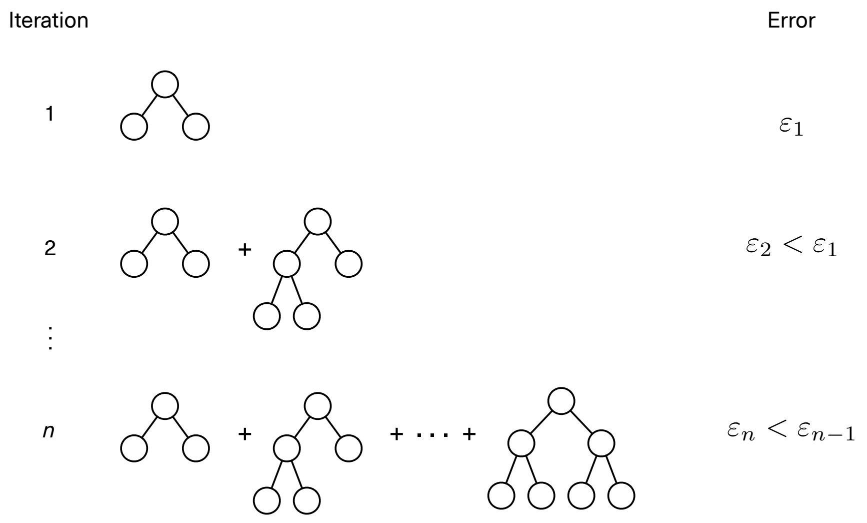

Iteration
Error
1
2
n
ε_1
$\varepsilon_2 < \varepsilon_1$
$\varepsilon_n < \varepsilon_{n-1}$
+
+ · · · +

Figure 3.5: A schematic view of the iterative tree boosting method. Each new iteration targets the residuals of the previous trees, attempting to lower the preceding error.

3.3.4 Recurrent Neural Networks (RNN)

Recurrent neural networks are a family of neural networks that have the capability to learn sequences of ordered data, such as time series, and natural language. Conventional neural networks process each input individually and do not maintain any record of past inputs. Recurrent neural networks maintain an internal state of previously processed inputs, which allows the extraction of long-range patterns from sequentially ordered data [63].

In a recurrent neural network, the input at time step t consists of x_t, and h_{t-1}, the output of the previous time step. Both inputs have associated weight matrices, W_x and W_h, for inputs x_t and h_{t-1}, respectively. The output h_t is calculated using the following equation

$$h_t = \sigma\left(W_x x_t + W_h h_{t-1} + b_t\right), \tag{3.15}$$

where σ is an activation function (sigmoid, or other), and b is a constant bias vector [64]. A simplified, unrolled schematic of a sequence–to–sequence recurrent neural network is shown in figure 3.6.

A simple implementation of a recurrent neural network is susceptible to the *vanishing gradient problem.* As the input passes through an increasing number of layers, the gradients begin to disintegrate, resulting in smaller and smaller changes applied to the weight matrices [64]. This can make the model untrainable, as it will never converge to a satisfactory solution. Due to this limitation, a simple recurrent neural network is capable of remembering only a few preceding time steps, making it unable to detect any long-term patterns [64]. Several improved models have been proposed to tackle the issue of vanishing gradients, such as Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU).

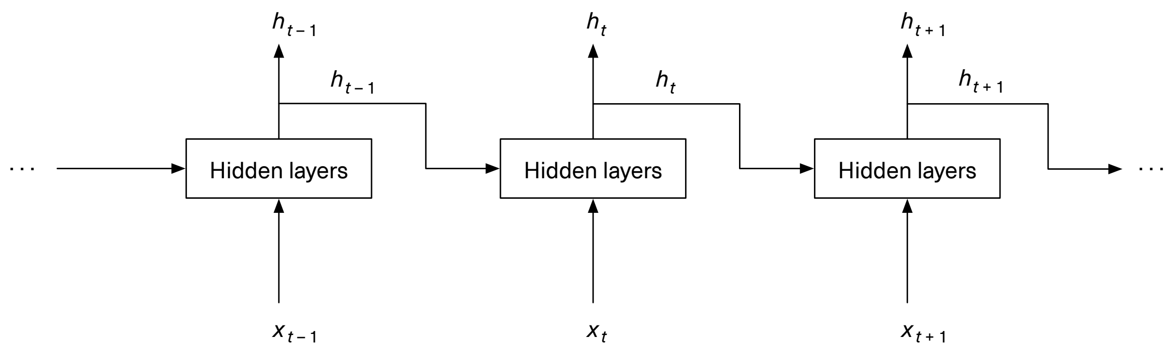


Figure 3.6: A schematic of an unrolled, recurrent neural network.

Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a recurrent neural network architecture, originally proposed by Hochreiter and Schmidhuber in 1997 [65]. According to the authors, LSTM is capable of retaining information across thousands of time steps. A schematic of an LSTM cell is shown in figure 3.7.

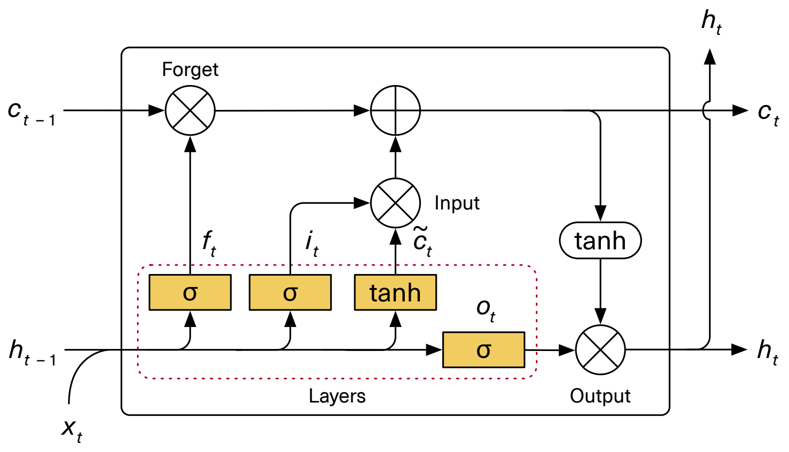


Figure 3.7: A diagram of an LSTM cell. $\otimes$ is elementwise matrix product, and $\oplus$ is direct matrix addition. [66]

LSTM was designed to solve the issue of vanishing gradients by adding a parallel, long-term memory state c to the model. The data in the long-term memory is controlled by three gates: *forget*, *input*, and *output*. The forget gate uses a sigmoid layer to determine what data should be discarded from the long-term state (f_t). The input gate uses a sigmoid (i_t) and a hyperbolic tangent ($\tilde{c}_t$) layer to determine what new data will be added to the long-term state. Lastly, the output gate uses a sigmoid layer (o_t) to determine what part of the long-term state c should be added to the current output h_t. [64]

The internal state of each component in an LSTM cell is calculated using the following equations [64]:

$$
\begin{aligned}
f_t &= \sigma\left(W_{f_h}\left[h_{t-1}\right], W_{f_x}\left[x_t\right], b_f\right) \\
i_t &= \sigma\left(W_{i_h}\left[h_{t-1}\right], W_{i_x}\left[x_t\right], b_i\right) \\
\tilde{c}_t &= \tanh\left(W_{c_h}\left[h_{t-1}\right], W_{c_x}\left[x_t\right], b_c\right) \\
c_t &= f_t \otimes c_{t-1} + i_t \otimes c_t \\
o_t &= \sigma\left(W_{o_h}\left[h_{t-1}\right], W_{o_x}\left[x_t\right], b_o\right) \\
h_t &= o_t \otimes \tanh\left(c_t\right)
\end{aligned}
$$

With the added long-term state c, LSTM is better suited to recalling previously encountered data than a simple recurrent network. During training, LSTM can learn which features of the data should be stored in the long-term memory state. Important features can be stored temporarily and extracted when needed, making it possible to capture long-range dependencies that were previously impossible with simple recurrent networks.

Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) is a recurrent neural network model, originally proposed by Cho et al. in 2014 [67]. GRU differs from the existing LSTM architecture by using only one state vector h_t. In addition, GRU has no output gate. A schematic of a GRU cell is shown in figure 3.8.

There are two gates in a GRU cell that are used to control the long-term data of the model: *update*, and *reset*. The update gate uses a sigmoid layer (z_t) to determine how much of the past data should be added to the state vector. Based on the output of z_t, data is either stored or erased from the state vector using gates similar to those in LSTM. The reset gate uses a sigmoid layer (r_t) to determine how much of the previous state should be shown to the main layer $(\tilde{h}_t)$. [67]

The internal state of each component in a GRU cell is calculated using the following equations [69]:

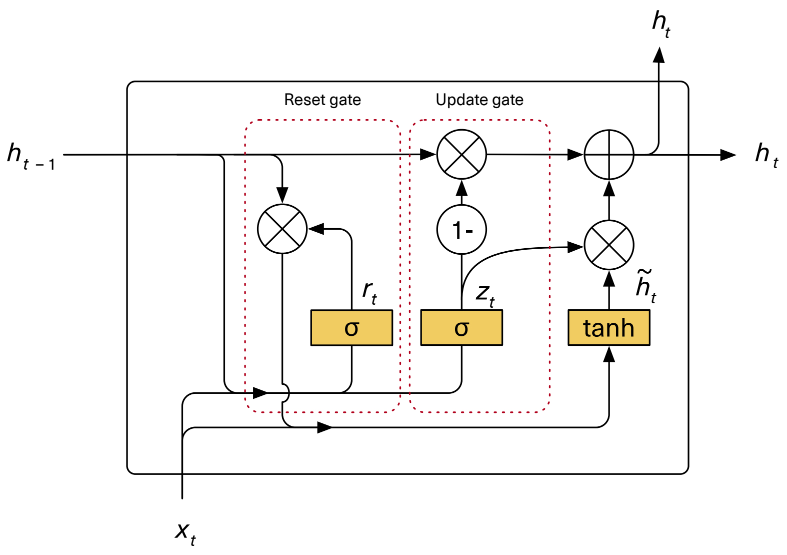


Figure 3.8: A diagram of a GRU cell. $\otimes$ is elementwise matrix product, $\oplus$ is direct matrix addition. [68]

$$
\begin{aligned}
h_t &= (1 - z_t) \otimes h_{t-1} + z_t \otimes \tilde{h}_t \\
z_t &= \sigma \left(W_{z_h} \left[h_{t-1} \right], W_{z_x} \left[x_t \right], b_z \right) \\
\tilde{h}_t &= \tanh \left(W_{h_{\tilde{h}}} \left[r_t \otimes h_{t-1} \right], W_{h_x} \left[x_t \right], b_h \right) \\
r_k &= \sigma \left(W_{r_h} \left[h_{t-1} \right], W_{r_x} \left[x_t \right], b_r \right)
\end{aligned}
$$

In a paper by Chung et al. [70], it was shown that in modeling of polyphonic music and speech, GRU performs similarly to LSTM. In some cases, GRU was found to converge faster, with better final results, especially with smaller sample sizes, although according to the authors this may heavily depend on the type of data and the modeling task, as no conclusive results could be obtained.

4. Experiments

4.1 Data

4.1.1 Twitter data

The standard Twitter search API allows searching for tweets published within the last seven days; retrieval of older tweets requires an enterprise API account [71]. Due to the limited time scope of this thesis, it was not feasible to carry out continuous scraping for a prolonged period of time in order to gather a sufficient quantity of tweets. Thus, it was decided best to use an existing tweet dataset. One such was found on Kaggle, an online community pertaining to data science and machine learning practitioners, offering over 50,000 public datasets for analysis [72]. The tweet dataset [73] comprises of 17.7 million English language tweets containing the keyword "bitcoin". The tweets were collected between 1. August 2017 and 22. January 2019.

Each tweet was supplied with additional metadata. The description of the dataset is shown in table 4.1 below.

COLUMN	DESCRIPTION
`username`	Author's Twitter username
`date`	Date of publication
`retweets`	Amount of retweets (shares) the tweet has received
`favorites`	Amount of favorites (likes) the tweet has received
`text`	Tweet text
`geo`	Geographical information
`mentions`	Mentions of other Twitter users in the Tweet
`hashtags`	Hashtags used in the tweet
`id`	Twitter's ID of the Tweet
`permalink`	URL to the tweet on `twitter.com`

Table 4.1: Tweet dataset description.

4.1.2 Historical bitcoin price data

The historical BTC–USD price dataset was downloaded from Kaggle [74]. The price data was acquired from Bitstamp, a cryptocurrency exchange. The data was collected between January 2012 and March 2021 and was recorded at one-minute intervals. The daily closing price of bitcoin (against USD) is shown in figure 4.1. The dataset is described in table 4.2 below.

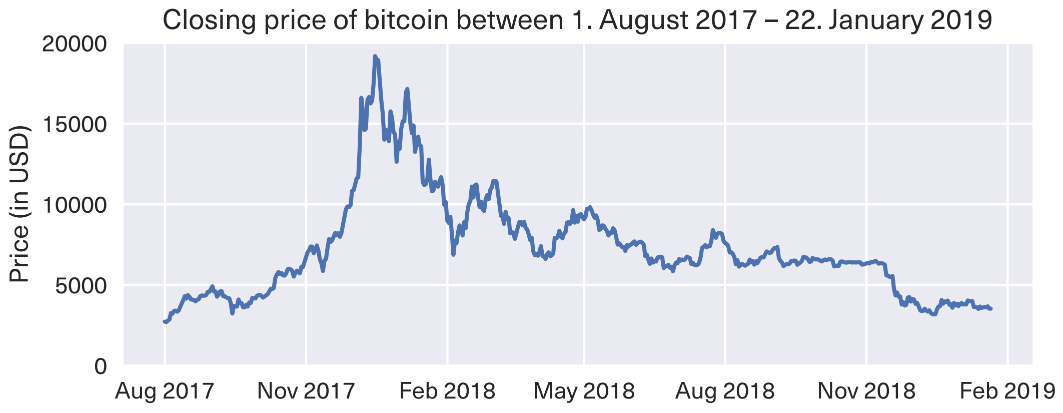


Figure 4.1: The daily closing price of bitcoin between 1. August 2017 – 22. January 2019.

COLUMN	DESCRIPTION
`Timestamp`	Start time of time window (1 minute), in UNIX time
`Open`	Open price at the start of a time window
`High`	High price within a time window
`Low`	Low price within a time window
`Close`	Close price at end of a time window
`Volume_(BTC)`	The volume of BTC transacted in this window
`Volume_(Currency)`	The volume of USD transacted in this window
`Weighted_Price`	VWAP (volume-weighted average price)

Table 4.2: Price dataset description.

4.2 Tweet preprocessing

4.2.1 Sentiment analysis

The sentiment analysis of the tweets was carried out using VADER, a rule-based sentiment analysis tool. As well as relying on its own lexicon, VADER also uses additional lexical and syntactical heuristics to provide more accurate sentiment analysis.

As a result, it was decided to not preprocess the tweets before carrying out the sentiment analysis.

Expanding the lexicon

The default gold-standard lexicon in VADER contains 7,520 frequently used words or terms used in everyday online discussions [49]. However, due to the inherent financial nature of cryptocurrencies, much of the online discussions surrounding cryptocurrencies can contain less common finance-related terms, such as *bear*, *bull*, *bubble*, and *crash*, which have different meanings in everyday language. Furthermore, various online communities discussing cryptocurrencies have developed their own slang with expressions such as *hodl*, *to the moon*, and *buy the dip*, which can appear obscure to the uninitiated [75].

By design, if VADER encounters a word that is not in its lexicon, it considers the word to be neutral [49]. This can be a problem, when analyzing domain-specific text, as many terms which carry significant sentiment, such as *cyberattack*, or *shorting*, can appear neutral to the sentiment analyzer. This can cause the compound sentiment to appear more neutral than it really is.

The lexicon of VADER can be expanded with additional tokens and their valence values. The following additional lexicons were considered.

1. **Master Dictionary by Loughran and McDonald.** The Master Dictionary by Loughran and McDonald is a lexicon geared toward financial applications [76]. It contains 86,531 words, which have been categorized into eight sentiment categories: negative, positive, uncertainty, litigious, strong modal, weak modal, constraining, and complexity. The dictionary does not provide intensity values for the tokens, but rather the frequencies, at which they appeared in the training corpora.

   Words belonging to the *negative*, *positive*, and *uncertainty* sentiment classes were added to the sentiment analyzer. The valence scores were set to $+1$ for positive words, -3 for negative words, and -1 for uncertain words. Words in the *uncertainty* class included tokens such as *possibly*, *rumors*, and *unidentified*. Such words could elicit doubt and suspicion in investors, which is why their valence scores were set to slightly negative (-1).

2. **Stock Lexicon by Oliveira et al.** The Stock Lexicon by Oliveira et al. is another finance-related lexicon, created by analyzing stock market discussion on StockTwits and Twitter [24]. The Stock Lexicon contains 20,551 finance-related tokens. Each token includes a sentiment score for positive and negative contexts.

The authors use the word "explosive" as an example; usually, the word carries a negative connotation, but it can also be interpreted positively in financial contexts (e.g. "explosive rise") [24]. The authors calculated the sentiment scores for each token by combining several frequency-based metrics of the word in the training corpus [24]. As a result, the sentiment scores have arbitrary upper and lower ranges.

These sentiment scores were used as valence scores in the expanded lexicon. For each token, the valence score was calculated by taking the arithmetic mean of its positive and negative sentiment scores, after which the sentiment scores were scaled to a range of $[-4, 4]$, the valence range used by VADER, using the scaling function 4.1 shown below.

$$f(s_i) = \begin{cases} \frac{s_i}{\max(s)} \times 4 & \text{if } s_i > 0, \\ \frac{s_i}{\min(s)} \times -4 & \text{if } s_i < 0. \end{cases} \tag{4.1}$$

If the same token appeared in both the Master Dictionary by Loughran and MacDonald, and Stock Lexicon by Oliveira et al., the valence score from the Stock Lexicon was preferred, as it featured calculated sentiment scores, while the Master Dictionary did not.

3. **Cryptocurrency slang lexicon.** As stated by Kraaijeveld et al.[21], compiling an additional slang lexicon will help the sentiment analyzer assign sentiment to sentences involving slang terms, that would otherwise be classified as neutral. While both the Master Dictionary by Loughran and MacDonald, and the Stock Lexicon by Oliveira et al., feature finance-related terms, they do not contain some of the more obscure terms which are used prolifically in online cryptocurrency discussions, such as *hodl* (from "hold", i.e., urging people to refrain from selling), or *copium* (a word meaning to cope with a grim situation). Therefore, a custom cryptocurrency slang lexicon was manually compiled, based on the author's own knowledge, containing 59 slang terms that are often used in online cryptocurrency discussions. The words were classified as positive or negative. Fixed valence scores of $+3$ and -3 were used for positive and negative words, respectively.

The Pearson correlation values were calculated for the price and sentiment data. Different combinations of additional lexicons were experimented with. It was established that the combination of all three additional lexicons resulted in the best correlation scores between the price and sentiment data. Some example sentences before and after applying lexicon updates are shown in table 4.3 below.

EXAMPLE SENTENCE	COMPOUND SCORE	
	DEFAULT	EXPANDED LEXICON
All signs point to a bitcoin bull run next year.	0	0.1113
The price of bitcoin took a tumble today.	0	-0.2019
Bitcoin is mooning; keep HODLing!	0	0.2642

Table 4.3: Three example sentences, along with their compound sentiment scores before and after expanding the lexicon of the sentiment analyzer. The default lexicon in VADER is not geared towards finance-related applications; as a result, all three sentences were classified as neutral. After expanding the lexicon, VADER classified all three sentences as having either a positive or negative sentiment.

After performing sentiment analysis on the tweets, the tweet dataset was augmented with new sentiment-based features, shown in table 4.4.

FEATURE	DESCRIPTION
`compound`	Mean compound sentiment of the tweet
`pos` `neu` `neg`	The proportion of positive/neutral/negative words out of all words in the tweet

Table 4.4: Additional features that were obtained using sentiment analysis.

4.2.2 Tweet filtering

The tweets in the original dataset represent a raw sample of Bitcoin discussions on Twitter at that time, as they have not been filtered or processed in any way. Such volume of data can be noisy, which can in turn make capturing overall sentiment difficult. To limit the amount of background noise in the data, it is necessary to attempt to filter tweets that add little or no value to the overall sentiment.

Filtering bots and spam

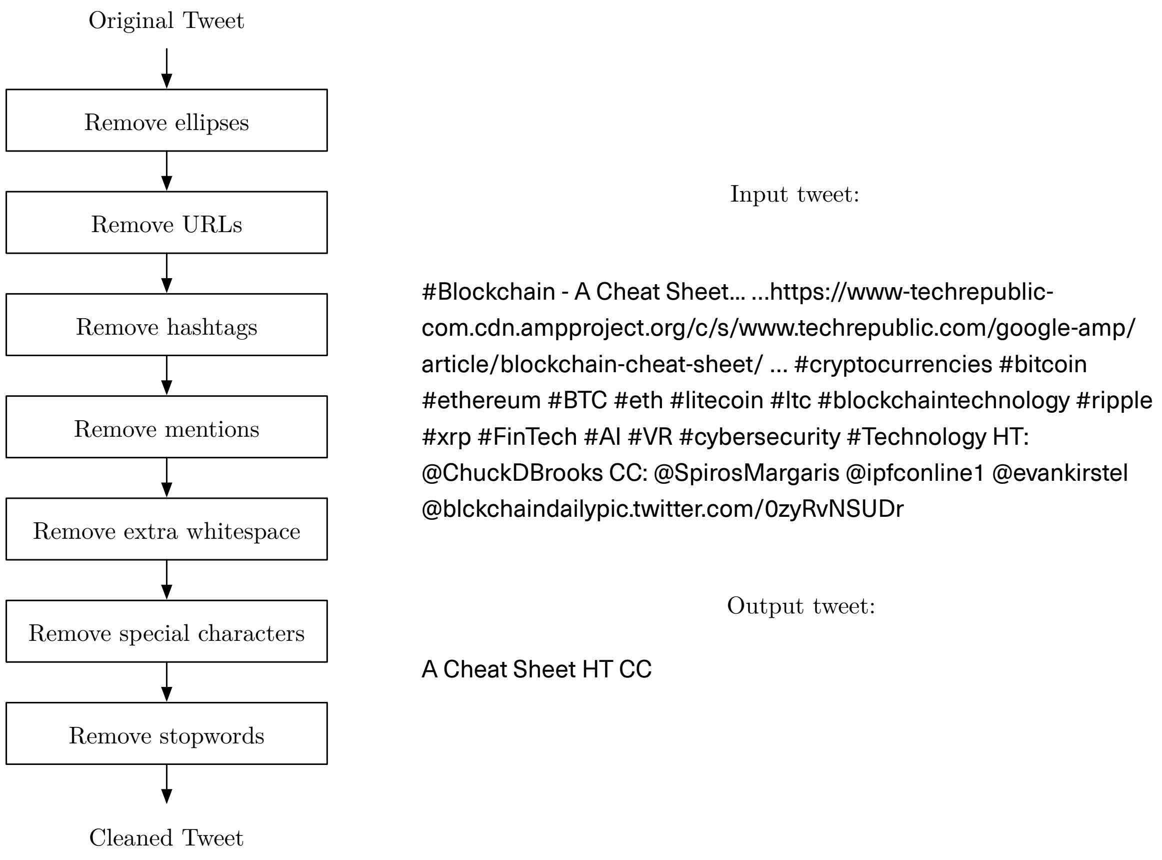
In a paper by Varol et al. [28], it is estimated that 9–15% of all Twitter users could be bots, i.e. automated programs that interact on Twitter without direct human involvement. In the cryptocurrency sphere, Nizzioli et al. [30] estimated that 56% of tweets sharing invite links to cryptocurrency-related Telegram or Discord channels were tweeted by bots or accounts that have since been suspended. Many of these channels were either promoting cryptocurrency-related Ponzi schemes, or "pump–and–dump" schemes, in which cooperating participants attempt to raise the price of a cryptocurrency by buying it ("pump"), thus inflating its price, and attracting broader interest among other investors, only to sell it to them for a profit, causing the price to collapse ("dump") [30]. To increase the visibility and reach of these invite links, the tweets can contain popular cryptocurrency-related keywords, such as "bitcoin" or "ethereum". In the tweet dataset, 184,741 tweets (1.0%) contained the keyword "telegram", or a Telegram channel invite URL (`t.me`). Similarly, 10,498 tweets (0.06%) in the dataset contained the keyword "discord".

The dataset also contained a significant amount of low-quality "spam" tweets that included several popular hashtags, including #bitcoin, to increase their visibility. Considering human sentiment, these tweets served no purpose.

The following attempts were made to filter out bots and spam tweets.

- **Filter users with a large number of tweets.** The tweet dataset contained tweets collected between 1. August 2017 and 22. January 2019, a period of 540 days. Any user with more than 540 tweets in the dataset, i.e. more than 1 tweet per day on average, was filtered, as there was a high probability that it was a bot account. The unfiltered tweet dataset contained 17,739,629 tweets made by 1,517,940 unique accounts. Of these, 3,506 (0.002%) accounts had made more than 540 tweets each. In total, these 3,506 accounts had produced 9.1 million (51.4%) of the total 17.7 million tweets. The account with the highest tweet count in the dataset had posted 134,434 Bitcoin-related tweets in the span of 540 days.
- **Filter users that had made near-identical tweets.** The tweets were processed using a processing pipeline as described in figure 4.2. If there were multiple tweets belonging to the same account that produced identical processed outputs, they were likely written using a template, indicating either a "spammy" tweet, or a bot. Accounts that had such tweets were filtered from the dataset.
- **Filter tweets that contain several hashtags, mentions, or URLs.** A spam tweet would often include multiple hashtags and mentions to increase its visibility and reach. In order to filter such tweets from the dataset, the length of the original tweet was compared to the length of the processed tweet (see figure 4.2). A threshold of 30% was set as the lower limit. Any processed tweets that were less than 30% of the length of the original tweet were removed from the dataset, as they were likely to contain many hashtags, mentions, or long external URLs, indicating a tweet unsuited for sentiment analysis.
- **Filter tweets that contain suspicious words.** Any users that had tweeted links containing Telegram or Discord invites were filtered. In addition, a tri-, and

quad-gram search was performed on the tweets. The results were then aggregated, revealing the most common three- and four-word substrings found in the tweets. The 100 most common tri-, and quadgrams were manually evaluated, filtering any suspicious entries. For example, one of the most common trigrams was "$btc $eth $xrp", most likely originating from template-based "stock ticker" tweets. One of the most common quadgrams was "join the gym rewards", referring to promotion tweets for a now-defunct crypto token [77].



Input tweet:

#Blockchain - A Cheat Sheet… …https://www-techrepublic-com.cdn.ampproject.org/c/s/www.techrepublic.com/google-amp/article/blockchain-cheat-sheet/ … #cryptocurrencies #bitcoin #ethereum #BTC #eth #litecoin #ltc #blockchaintechnology #ripple #xrp #FinTech #AI #VR #cybersecurity #Technology HT: @ChuckDBrooks CC: @SpirosMargaris @ipfconline1 @evankirstel @blckchaindailypic.twitter.com/0zyRvNSUDr

Output tweet:

A Cheat Sheet HT CC

Figure 4.2: Tweet processing pipeline. On the right, a spam tweet is shown before (396[†] characters) and after processing (19 characters).

[†]Although Twitter has a 280-character limit for tweets, URLs always count as 23 characters, despite their actual length.[78]

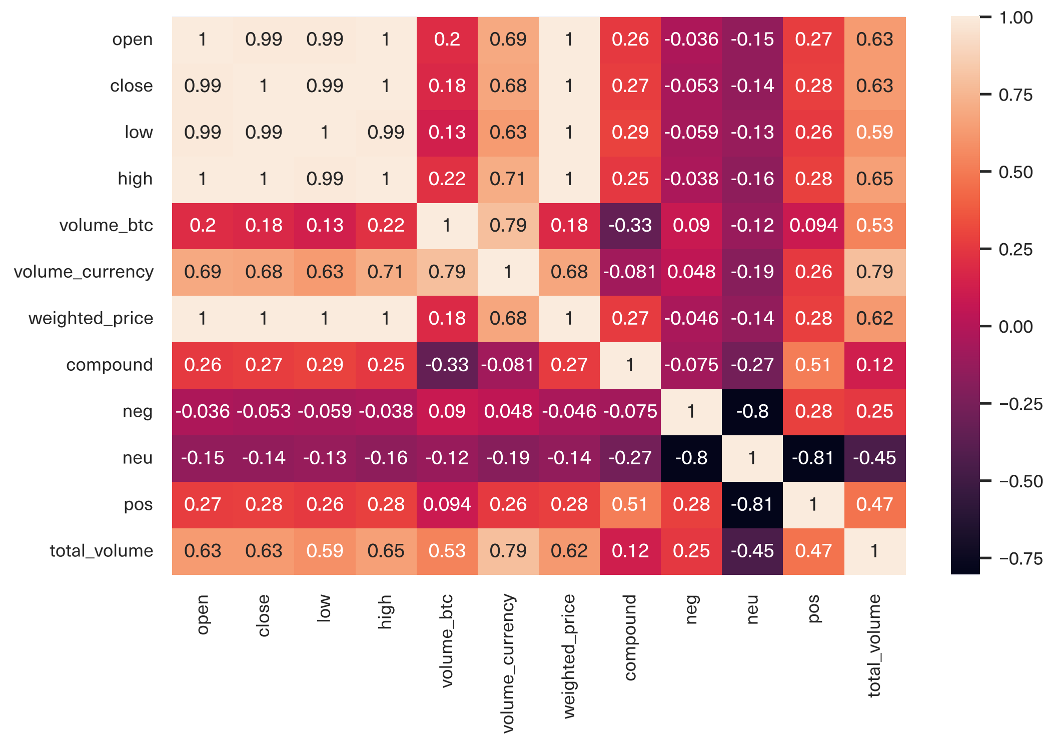
Filtering unimportant tweets

Amount of likes	0	1	2	3	4	5 or more
Percentage of tweets	73.3%	13.7%	4.4%	2.1%	1.2%	5.3%

Table 4.5: Distribution of likes in the tweet dataset.

The majority (73.3%) of the tweets in the dataset have received no likes from other users, as can be seen in table 4.5. Such tweets may appear insignificant on their own, but overall, they can add valueless background noise to the data, which may reduce the captured sentiment. It can be said that a tweet that has received 100 likes is more *popular* than a tweet that has received no likes. Furthermore, it can be assumed that the more popular a tweet is, the more likely it is to be retweeted, reach other people, and align with the popular sentiment. Therefore, a choice was made to focus only on such popular tweets.

After filtering out potential bot and spam tweets using methods outlined in the previous section, the dataset was aggregated to the frequency of one day. The sentiment data was aggregated by calculating the arithmetic mean of sentiment scores across all tweets over each 24-hour time window. Next, columnwise Pearson correlations were calculated for the price and sentiment data. After experimenting with different thresholds, it was discovered that excluding tweets with less than 40 likes produced the best correlation coefficient values between the compound sentiment and price data. The difference between the correlation coefficients of the full tweet dataset and the filtered dataset in regard to the closing price is shown in table 4.6. The full columnwise Pearson correlation matrix of the filtered dataset is shown in figure 4.3.



	open	close	low	high	volume_btc	volume_currency	weighted_price	compound	neg	neu	pos	total_volume
open	1	0.99	0.99	1	0.2	0.69	1	0.26	-0.036	-0.15	0.27	0.63
close	0.99	1	0.99	1	0.18	0.68	1	0.27	-0.053	-0.14	0.28	0.63
low	0.99	0.99	1	0.99	0.13	0.63	1	0.29	-0.059	-0.13	0.26	0.59
high	1	1	0.99	1	0.22	0.71	1	0.25	-0.038	-0.16	0.28	0.65
volume_btc	0.2	0.18	0.13	0.22	1	0.79	0.18	-0.33	0.09	-0.12	0.094	0.53
volume_currency	0.69	0.68	0.63	0.71	0.79	1	0.68	-0.081	0.048	-0.19	0.26	0.79
weighted_price	1	1	1	1	0.18	0.68	1	0.27	-0.046	-0.14	0.28	0.62
compound	0.26	0.27	0.29	0.25	-0.33	-0.081	0.27	1	-0.075	-0.27	0.51	0.12
neg	-0.036	-0.053	-0.059	-0.038	0.09	0.048	-0.046	-0.075	1	-0.8	0.28	0.25
neu	-0.15	-0.14	-0.13	-0.16	-0.12	-0.19	-0.14	-0.27	-0.8	1	-0.81	-0.45
pos	0.27	0.28	0.26	0.28	0.094	0.26	0.28	0.51	0.28	-0.81	1	0.47
total_volume	0.63	0.63	0.59	0.65	0.53	0.79	0.62	0.12	0.25	-0.45	0.47	1

Figure 4.3: Columnwise Pearson correlation coefficients of the price and sentiment data. A coefficient of 1 indicates a perfect correlation, a coefficient of 0 indicates no correlation, and a coefficient of −1 indicates a perfect inverse correlation.

From the correlation matrix, we can observe that there exists a weak correlation between the closing price and the compounded (0.27) and positive sentiment (0.28). There is a moderate correlation (0.63) between the price and the total volume of tweets, indicating that as the price of bitcoin increases, so does the Bitcoin-related discussion on Twitter. The same phenomenon has also been noted by Twitter [6].

VARIABLE	CORRELATION COEFFICIENT	
	FULL	FILTERED
`open`	0.99	0.99
`low`	0.99	0.99
`high`	1.00	1.00
`volume_btc`	0.18	0.18
`volume_currency`	0.68	0.68
`weighted_price`	1.00	1.00
`compound`	-0.02	0.27
`neg`	0.06	-0.05
`neu`	-0.12	-0.14
`pos`	0.10	0.28
`total_volume`	0.66	0.63

Table 4.6: Correlation coefficients of the variables in the full and filtered datasets, in regard to the closing price of bitcoin. From the table can be seen that the correlation coefficients of the mean compound sentiment (`compound`), and the mean proportion of positive words in the tweets (`pos`) increased significantly ($-0.02 \rightarrow 0.27$, and $0.10 \rightarrow 0.28$, respectively), after focusing only on popular tweets.

The final, filtered tweet dataset contained 107,591 tweets, 0.6% of the original dataset.

4.3 Validation

To get a good sense of the forecasting capabilities of each model, several prediction lengths were considered. All models were trained to predict the price of bitcoin for the next 1, 3, 5, and 7 days.

4.3.1 Time series validation

To assess whether the models can predict the price in varying market conditions, all models were evaluated using a validation method shown in figure 4.4. Ten random dates between 5. September 2018 and 15. January 2019 were selected from a discrete uniform distribution, giving each split at least 400 historical data points for training. Each randomly selected date functioned as a split between the training and testing data. In each split, seven different time series forecasting models (VAR, VARMA, Random Forest, XGBoost, LightGBM, LSTM, and GRU) were used to predict the closing price of bitcoin for the next 1, 3, 5, and 7 days, based on previous historical data up to the split point. The training data was used to train the model, and the testing data was used to measure the accuracy of the predictions, reflecting a real-life situation where the model must predict a future, unseen price. All models produced out-of-sample forecasts of the future price, and the respective test errors of each model were averaged for each split and prediction length, resulting in an average validation error for each prediction length for all prediction methods.

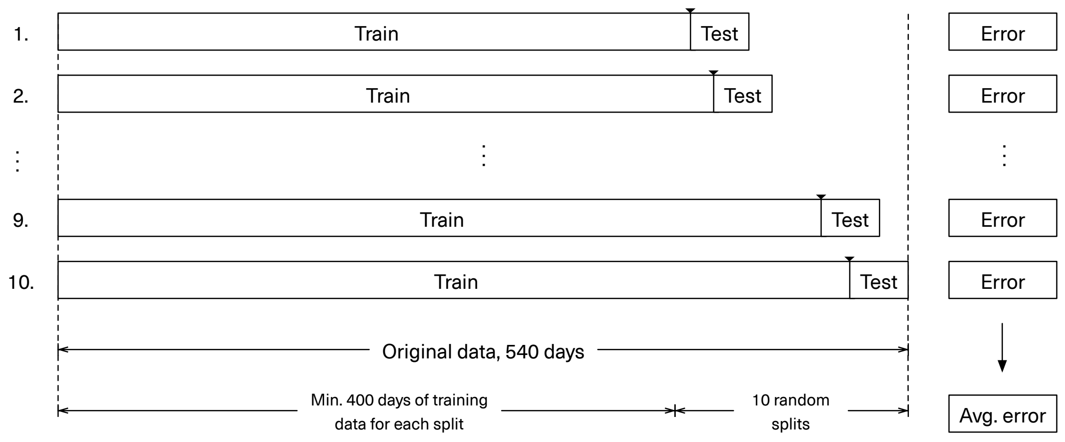


Figure 4.4: Time series validation with ten randomly selected splits.

4.3.2 Evaluation metrics

The models were evaluated using three evaluation metrics: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). The error E_i is defined as the difference between the real value of the closing price y_i and the predicted value of the closing price x_i, where i is the index of the prediction among n samples.

$$E_i = y_i - x_i. \tag{4.2}$$

Mean Squared Error measures the arithmetic average of the squares of the errors. Due to the squaring, errors are penalized more heavily. The formula for MSE is shown in equation 4.3.

$$\text{MSE} = \frac{\sum_{i=1}^{n} E_i^2}{n}. \tag{4.3}$$

Root Mean Squared Error is the square root of MSE. RMSE is easier to interpret, as the error is on the same scale as the predicted variable. The formula for RMSE is shown in equation 4.4.

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{\sum_{i=1}^{n} E_i^2}{n}}. \tag{4.4}$$

Mean Absolute Percentage Error measures the average absolute percentage error of the predictions. The formula for MAPE is shown in equation 4.5.

$$\text{MAPE} = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{E_i}{y_i} \right|. \tag{4.5}$$

To calculate accuracy, precision, and recall of the predictions, each predicted timestep was also categorized as follows:

$$y_i, x_i = \begin{cases} \text{True Positive (TP)} & \text{if } (y_i > y_{i-1}) \;\;\&\;\; (x_i > x_{i-1}), \\ \text{True Negative (TN)} & \text{if } (y_i < y_{i-1}) \;\;\&\;\; (x_i < x_{i-1}), \\ \text{False Positive (FP)} & \text{if } (y_i < y_{i-1}) \;\;\&\;\; (x_i > x_{i-1}), \\ \text{False Negative (FN)} & \text{if } (y_i > y_{i-1}) \;\;\&\;\; (x_i < x_{i-1}). \end{cases}$$

The formulas for accuracy, precision, and recall are shown below.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4.6}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{4.7}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{4.8}$$

Lastly, a *trend* score was also calculated. If in the last predicted timestep both the predicted price and actual price are higher than the last known value in the training data, the model has successfully predicted an upward trend in price. Similarly, if both prices are lower than the last known price, the model has predicted a downward trend in price.

4.4 Forecasting

4.4.1 Feature Selection

After applying sentiment analysis to the tweets, the dataset is now completed. The full dataset is described in table 4.7 below.

FEATURE	DESCRIPTION
`open`	Opening price at the start of the day
`close`	Closing price at the end of the day
`low`	The lowest price during the day
`high`	The highest price during the day
`volume_btc`	Volume of BTC transacted during the day
`volume_currency`	Volume of USD transacted during the day
`weighted_price`	Volume-weighted average price
`compound`	Mean compound sentiment of the day
`pos` `neu` `neg`	Mean proportion of positive/neutral/negative words in all tweets categorized that day
`total_volume`	Total number of tweets in the dataset for the day

Table 4.7: Features of the completed dataset.

As shown in figure 4.3, many features in the dataset have a near-perfect correlation with each other. For example, the opening, high, low, closing, and weighted prices correlate perfectly with each other (0.99–1.00). Having almost identical features in the feature set adds unnecessary overhead and lengthens the training time [79]. Similarly, including features with low correlation offers trivial benefit for the model. Therefore, a feature selection was performed, favoring features with a higher correlation with the target variable, that is, the closing price. Four features were selected, and they are described in table 4.8.

SET	FEATURE	DESCRIPTION
Baseline	`volume_currency`	Volume of USD transacted during the day
Sentiment	`compound`	Mean compound sentiment of the day
	`pos`	Mean proportion of positive words in the tweets that day
	`total_volume`	Total number of tweets in the dataset for the day

Table 4.8: Final features after feature selection.

To investigate, whether the Twitter sentiment can be a useful feature for predicting future bitcoin prices, two sets of features were used for each training configuration. In the baseline predictions, only the USD volume was used as an explanatory variable for the closing price. In sentiment-based predictions, a larger feature set was used, which included the USD volume and three additional sentiment-based explanatory values: compound sentiment, proportion of positive words in the tweets, and the total number of tweets.

For each model, the future closing price of bitcoin was predicted on ten randomly selected days. The random days were generated once and remained the same for all models. For each selected day, the closing price of bitcoin was predicted for the next 1, 3, 5, and 7 days with all models. A sample 3-day prediction is shown in figure 4.5, showing the predictions of all models.

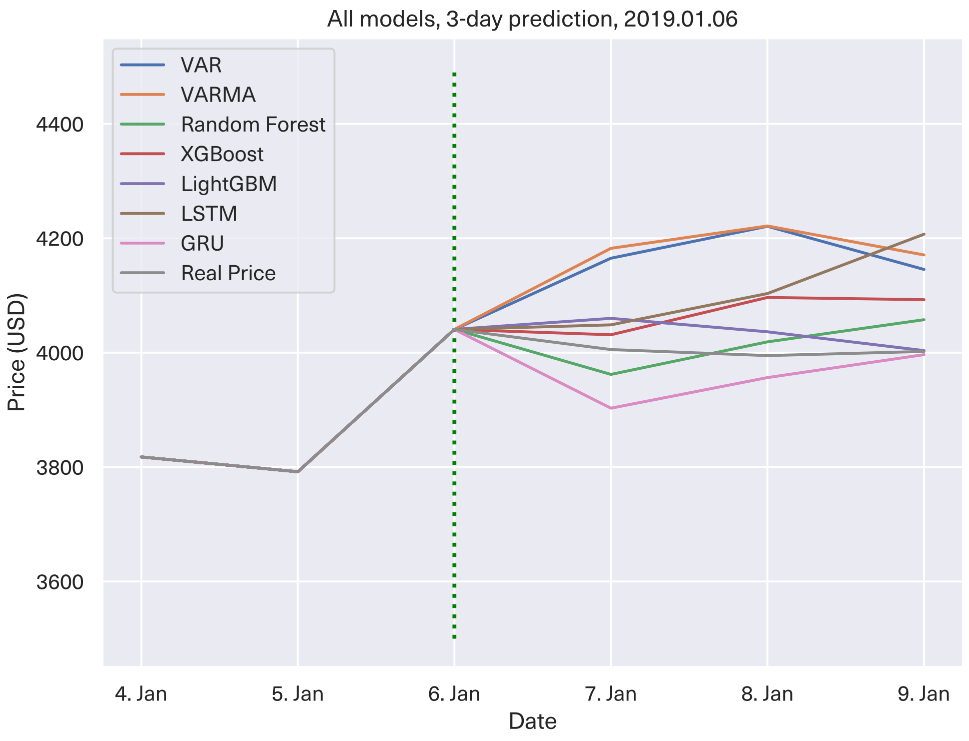


Figure 4.5: A sample prediction, showing all seven models predicting the price of bitcoin for the next 3 days. These predictions were done using the augmented feature set.

4.4.2 VAR

The VAR models were implemented using the `statsmodels` library for Python. As VAR requires stationarity for all variables, the variables in the dataset were tested for stationarity using the Augmented Dickey-Fuller test, results of which are shown in table 4.9. Subsequently, all non-stationary variables were transformed into stationary variables using logarithmic first-order differencing.

FEATURE	p-VALUE	RESULT
`close`	0.347	Non-stationary
`volume_currency`	0.536	Non-stationary
`compound`	0.064	Non-stationary
`pos`	0.084	Non-stationary
`total_volume`	0.110	Non-stationary

Table 4.9: The results of the Augmented Dickey-Fuller test applied to the variables.

With VAR, there is a tradeoff to be made between the number of variables, and the maximum order of the model, as setting either value too high can make the model too large to estimate, due to the excessive number of equations. The best results were achieved by setting the maximum lag order p to 14. The optimal order for each model was chosen during fitting, using the Akaike information criterion (AIC), which estimates the optimal order of the model.

The mean prediction scores for VAR models are shown in table 4.10 below.

DAYS	METHOD	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
1	Baseline	112.48	18493.53	0.0237	0.3000	0.5000	0.4286	**0.3**
	Sentiment	**98.36**	**13199.57**	**0.0227**	0.1000	0.0000	0.0000	0.1
3	Baseline	169.40	32281.60	0.0341	0.5001	0.6000	0.5556	0.1
	Sentiment	**144.21**	**23531.97**	**0.0303**	0.3665	0.4500	0.4166	**0.5**
5	Baseline	172.07	**33205.06**	0.0339	0.5000	0.4333	0.5557	**0.6**
	Sentiment	**166.93**	34213.29	**0.0322**	0.4000	0.4000	0.3958	0.5
7	Baseline	198.49	**45953.52**	0.0395	0.5285	0.4917	0.5018	0.4
	Sentiment	**186.05**	46186.04	**0.0362**	0.4003	0.3916	0.3333	**0.5**

Table 4.10: Mean prediction scores for VAR.

On average, the sentiment-based VAR model performs better than the baseline VAR in all prediction lengths. It can also be seen that the prediction error increases as the prediction length grows, which is to be expected. As for predicting upward and downward trends, VAR performed poorly, especially with 1-day predictions.

4.4.3 VARMA

The VARMA model was implemented using the `statsmodels` library for Python. As VARMA is computationally very costly to fit on higher orders, p and q were set to 7 and

3, respectively. As with VAR, VARMA also requires stationarity for all variables; all non-stationary variables were made stationary using logarithmic first-order differencing.

The mean prediction scores for VARMA models are shown in table 4.11 below.

DAYS	METHOD	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
1	Baseline	103.33	15439.64	0.0241	0.3000	0.1667	0.3333	**0.3**
	Sentiment	**88.54**	**9819.59**	**0.0195**	0.1000	0.1667	0.2000	0.1
3	Baseline	151.95	25989.17	0.0314	0.4998	0.5500	0.6041	**0.5**
	Sentiment	**139.27**	**23066.89**	**0.0279**	0.4667	0.6500	0.5666	0.3
5	Baseline	173.18	36422.31	0.0337	0.4800	0.5001	0.4722	**0.5**
	Sentiment	**170.72**	**32640.43**	**0.0328**	0.4200	0.5667	0.4250	0.3
7	Baseline	**191.65**	48115.13	**0.0372**	0.4716	0.4916	0.4316	**0.6**
	Sentiment	195.66	**45254.47**	0.0374	0.4145	0.5500	0.4183	0.5

Table 4.11: Mean prediction scores for VARMA.

The sentiment-based VARMA model performs noticeably better than the baseline model in 1-day and 3-day predictions. VARMA performs similarly to VAR, being slightly better for 1-day and 3-day predictions, suggesting that adding a moving average component to the model improves the forecasting results in short-term predictions. In 7-day predictions, the sentiment-based VARMA models performed slightly poorer on average than the baseline models. The reason for this might be in the selected p and q values, which were set to low values to keep the fitting time at a manageable duration. Perhaps with 7-day predictions, higher p and q values could improve the error scores of the sentiment-based models, as more historical sentiment data would be available to the model. Unfortunately, this could not be investigated further at this time, as out of all the models tested, VARMA models were already taking the longest time to fit.

Like VAR, VARMA also performed poorly for predicting downward and upward trends on 1-day predictions.

4.4.4 Random Forest

The Random Forest model was implemented using the `scikit-learn` library for Python. In contrast to VAR and VARMA, there are no strict requirements regarding the stationarity of the input data that has been transformed using the method described in section 3.3.3. During testing of the models, however, it was established that using non-stationary data during the training process resulted in consistently poorer prediction results when compared to tests done with stationary data. Therefore, all non-stationary variables were transformed into stationary variables using logarithmic first-order differencing.

The size of the lookback window was set to 30 days. Higher lookback window values, such as 60 or 90 days, did not result in better prediction results.

The number of estimators was set to 1000, and the maximum number of features was set to $\sqrt{n}$, where n is the total number of features. The time series data was transformed using the method described in section 3.3.3.

The mean prediction scores for Random Forest models are shown in table 4.12 below.

DAYS	METHOD	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
1	Baseline	74.47	8286.71	0.0176	0.4000	0.5000	0.5000	0.4
	Sentiment	**46.23**	**2963.45**	**0.0112**	0.8000	0.6667	1.0000	**0.8**
3	Baseline	127.10	20400.68	0.0253	0.3667	0.4000	0.3667	0.6
	Sentiment	**110.58**	**17430.71**	**0.0216**	0.4332	0.4000	0.4074	**0.8**
5	Baseline	176.20	35922.89	0.0328	0.5600	0.5167	0.6111	**0.3**
	Sentiment	**156.60**	**26966.42**	**0.0280**	0.5400	0.5500	0.5167	**0.3**
7	Baseline	220.15	55417.49	0.0424	0.3715	0.3583	0.3519	0.2
	Sentiment	**218.56**	**52448.64**	**0.0418**	0.5713	0.5416	0.6001	**0.5**

Table 4.12: Mean prediction scores for Random Forest.

The sentiment-based Random Forest models outperform the baseline models in all prediction lengths. When compared to VAR and VARMA, Random Forest outperforms them in 1-day, 3-day, and 5-day predictions. Sentiment-based Random Forest models had the lowest average RMSE among all 1-day and 3-day predictions (46.23, 110.58). Sentiment-based Random Forest models also performed well in predicting downward and upward price trends in 1-day, and 3-day predictions, with an 80% success rate in both groups.

4.4.5 XGBoost

The XGBoost model was implemented using the `xgboost` library for Python. Similar to the Random Forest models, XGBoost models performed significantly better with stationary data. All non-stationary variables were transformed into stationary variables using logarithmic first-order differencing. During testing of the models, a lookback window of 30 days produced satisfactory results; higher values did not result in better predictions. The number of estimators was set to 500, and the learning rate to 0.01. The maximum tree depth was set to 5.

The mean prediction scores for XGBoost models are shown in table 4.13 below.

DAYS	METHOD	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
1	Baseline	78.87	9094.80	0.0178	0.4000	0.6667	0.5000	0.4
	Sentiment	**72.94**	**6344.49**	**0.0153**	0.6000	0.5000	0.7500	**0.6**
3	Baseline	205.74	54305.60	0.0363	0.3666	0.5500	0.4000	**0.7**
	Sentiment	**123.21**	**21141.95**	**0.0227**	0.4333	0.4000	0.3834	0.6
5	Baseline	244.52	76990.66	0.0423	0.4200	0.5167	0.4683	**0.3**
	Sentiment	**165.09**	**36378.45**	**0.0310**	0.5000	0.5668	0.5001	**0.3**
7	Baseline	288.25	106001.57	0.0488	0.4285	0.5750	0.4384	0.3
	Sentiment	**234.74**	**66233.60**	**0.0443**	0.4714	0.6251	0.4950	**0.4**

Table 4.13: Mean prediction scores for XGBoost.

On average, sentiment-based XGBoost models perform noticeably better than the baseline models in all prediction lengths. The difference is more prominent than in Random Forest models. The mean RMSE of the baseline models was 205.74, and 244.52, for 3-day, and 5-predictions. For sentiment-based models the RMSEs were 123.21 (40.1% decrease), and 165.09 (32.5% decrease), respectively. Compared to Random Forest, XGBoost performs worse on average. However, for 1-day, 3-day, and 5-day predictions, XGBoost performs better than VAR and VARMA.

XGBoost performed slightly below Random Forest in predicting the upward and downward trends in 1-day and 3-day predictions.

4.4.6 LightGBM

The LightGBM model was implemented using the `lightgbm` library for Python. As with Random Forest and XGBoost, logarithmically differenced stationary data yielded consistently better results with LightGBM as well. The size of the lookback window was set to 30 days. After hyperparameter testing, the number of estimators was set to 500, the learning rate to 0.001, and the maximum depth of a tree to 5.

The mean prediction scores for LightGBM models are shown in table 4.14 below.

DAYS	METHOD	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
1	Baseline	71.37	6899.81	0.0166	0.3000	0.1667	0.3333	**0.3**
	Sentiment	**62.62**	**6264.31**	**0.0146**	0.3000	0.1667	0.3333	**0.3**
3	Baseline	120.85	20196.31	0.0238	0.3667	0.3000	0.3333	0.4
	Sentiment	**113.92**	**17654.84**	**0.0220**	0.3333	0.3000	0.2593	**0.6**
5	Baseline	165.65	32002.02	0.0306	0.4400	0.4667	0.4233	0.2
	Sentiment	**149.16**	**26977.35**	**0.0276**	0.4000	0.3500	0.3666	**0.4**
7	Baseline	190.71	43642.27	0.0366	0.4857	0.4334	0.4033	**0.3**
	Sentiment	**181.16**	**39698.35**	**0.0348**	0.5285	0.5499	0.5083	0.2

Table 4.14: Mean prediction scores for LightGBM.

The sentiment-based LightGBM models perform better than the baseline models in all prediction lengths. LightGBM performs better than XGBoost, which can be expected, as LightGBM was proposed as an improvement upon XGBoost. LightGBM performed the best out of all models in 5-day and 7-day predictions. However, LightGBM did not perform as well as Random Forest for predicting price trends.

4.4.7 LSTM & GRU

Both the LSTM and GRU models were implemented using the Keras library for Python. A schematic of the models is shown in figure 4.6.

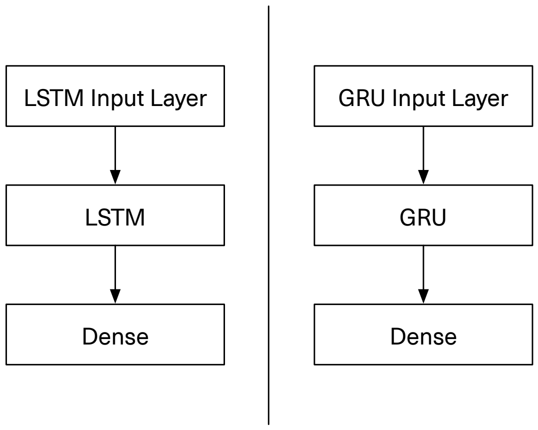


Figure 4.6: Schematics of the LSTM and GRU models that were used in predictions.

A single layer with 8 neurons produced the best results for both models. Adding additional layers, or increasing the neuron count resulted in consistently poorer predictions. The best batch size was found to be 32. Instead of differencing, all data columns were normalized between a range of $[0 \ldots 1]$, using a MinMax scaler, to expedite convergence.

The lookback window k was set to 50 days. Neither increasing nor decreasing k resulted in significant changes to the prediction performance. The network was set up as a sequence-to-sequence model, i.e. for each batch, the model predicted the next n steps, n being the prediction length. For each prediction, the original training data was partitioned into smaller training batches of size $k + n$. A simplified schematic showing partitioning for $n = 3$ predictions is shown in figure 4.7.

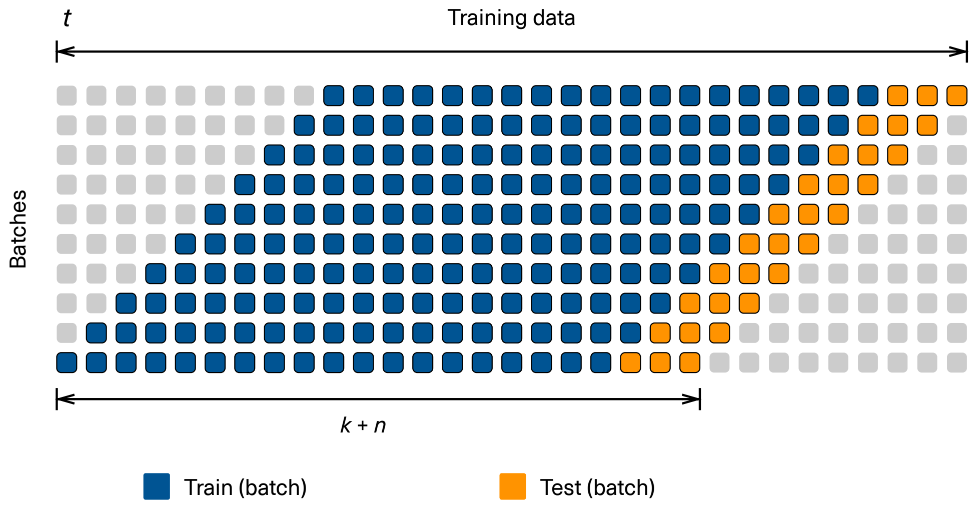


Figure 4.7: A simplified schematic showing training data partitioning for 3-day predictions.

Both models were compiled with an Adam gradient optimizer and Mean Absolute Error (MAE) as a loss function. The learning rate of the Adam optimizer was set to 0.01. 5% of the training batches were used for validation during training. The maximum training amount was set to 300 epochs, with the early stop feature enabled.

The mean prediction scores are shown in tables 4.15 and 4.16, for LSTM and GRU, respectively.

DAYS	METHOD	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
1	Baseline	**125.01**	**21711.92**	0.029	**0.500**	**0.500**	**0.600**	**0.5**
	Sentiment	131.33	25675.00	**0.030**	0.400	**0.500**	0.500	0.4
3	Baseline	**191.79**	**45100.02**	0.037	0.367	0.350	0.317	**0.7**
	Sentiment	192.11	47655.96	**0.041**	**0.433**	**0.550**	**0.483**	0.3
5	Baseline	**208.34**	**51467.38**	0.041	**0.560**	**0.617**	**0.625**	**0.7**
	Sentiment	277.53	88910.57	**0.057**	0.460	0.517	0.442	0.6
7	Baseline	**285.66**	**105100.59**	0.055	**0.557**	**0.592**	**0.565**	**0.6**
	Sentiment	532.22	462666.05	**0.118**	0.500	0.492	0.530	0.3

Table 4.15: Mean prediction scores for LSTM.

DAYS	METHOD	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
1	Baseline	**87.10**	**14322.28**	**0.0220**	0.600	0.333	**1.000**	0.6
	Sentiment	144.61	30764.41	0.0344	**0.700**	**0.667**	0.800	**0.7**
3	Baseline	**145.67**	**24874.36**	**0.0281**	**0.600**	**0.750**	**0.550**	0.5
	Sentiment	221.02	57031.16	0.0420	0.400	0.450	0.400	**0.6**
5	Baseline	**241.01**	**79991.34**	**0.0482**	0.520	0.517	0.518	0.5
	Sentiment	294.54	101811.09	0.0579	**0.560**	**0.667**	**0.607**	**0.7**
7	Baseline	**272.98**	**110469.70**	**0.0589**	**0.486**	0.525	**0.512**	**0.8**
	Sentiment	397.96	246997.36	0.0848	0.386	**0.583**	0.385	0.6

Table 4.16: Mean prediction scores for GRU.

From the results can be seen that both models perform better with the baseline features only. Adding sentiment-based features resulted in considerably poorer scores. This is counter to previous results with other models. In addition, in both LSTM and GRU, all error metrics (RMSE, MSE, and MAPE), were significantly higher than in all earlier non-RNN models. Considering that neural networks represent the "state of the art" in current machine learning methods, these results appear perplexing. Despite extensive testing with different hyperparameters, lookback window sizes, and features, the performance could not be improved from these results.

The amount of available training data could be one reason behind the inferior performance of RNNs in these prediction tasks. RNNs are capable of making long-range connections between thousands of timesteps, but the available amount of training data, in this case, was limited to approximately 400–500 days, depending on the split point. Perhaps with more training data, the prediction results could be improved.

To test the plausibility of this idea, four identical 10-split time series validations were executed. In each round, the training data was progressively truncated from the beginning. In the first round, 20% of the training data was cut from the beginning, and in the last, 80%. The results of the mean RMSE scores are plotted in figure 4.8.

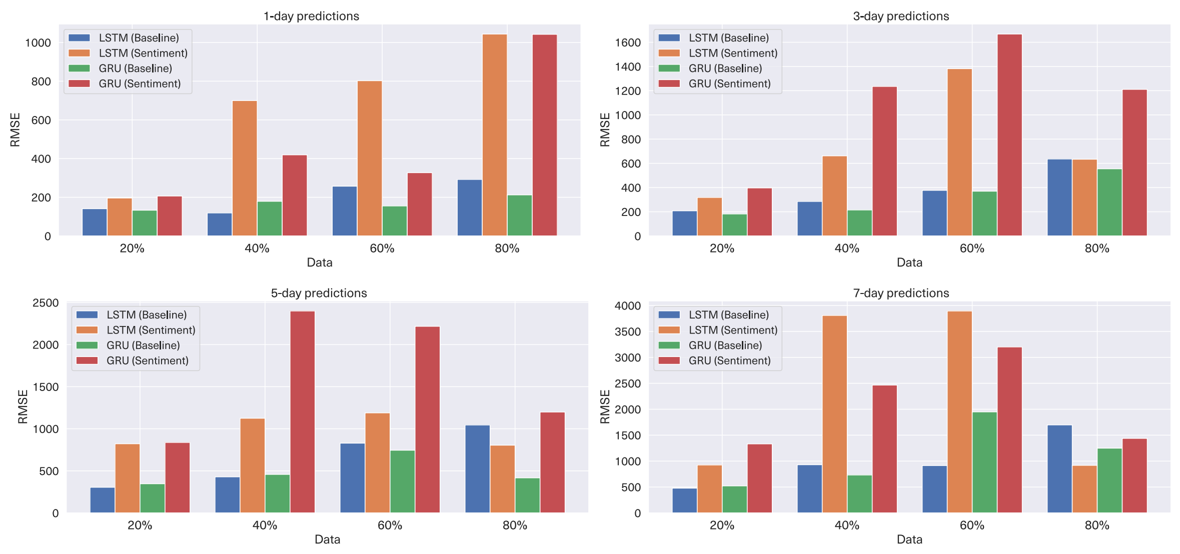


Figure 4.8: Mean RMSE scores of progressively truncated training data. Labels on the x-axis represent the amount of data that was cut from the beginning.

As can be seen from figure 4.8, there is a propensity for the error to grow when the amount of training data is decreased. The effect becomes especially noticeable with the full sentiment-based feature set. With the baseline dataset, the effect is less pronounced. While these results are insufficient to draw wider conclusions, it can be said that the amount of training data can have a significant effect on the prediction results. Perhaps with a larger training dataset, RNNs could approach the results of statistical and tree-based methods.

4.5 Summary of sentiment-based results

Below are the summarized results of all sentiment-based models in 1-day, 3-day, 5-day, and 7-day predictions.

4.5.1 1-day predictions

The average errors (RMSE, MSE, and MAPE) of all sentiment-based models have been plotted in figure 4.9. Detailed results are in table 4.17.

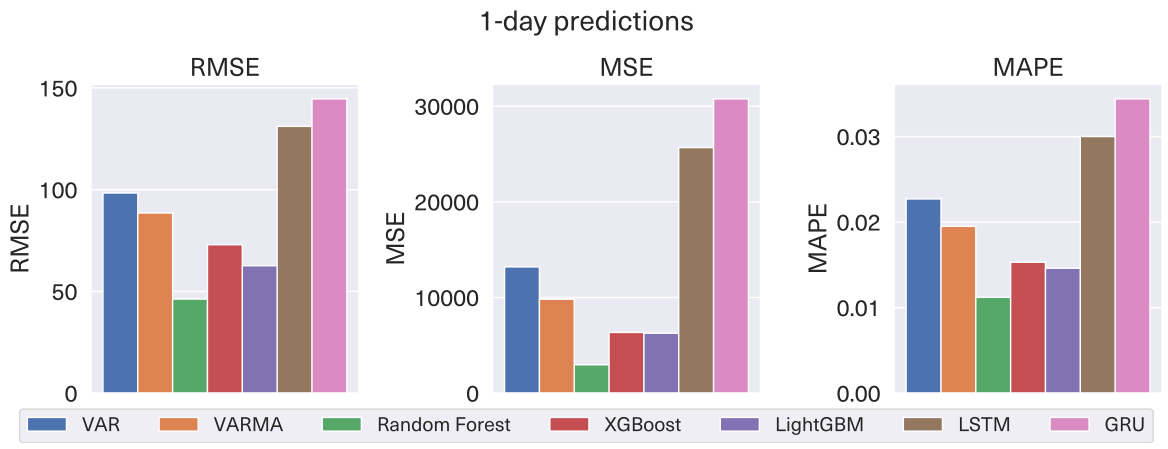


Figure 4.9: Average 1-day prediction errors of all sentiment-based models.

MODEL	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
VAR	98.36	13199.57	0.0227	0.1000	0.0000	0.0000	0.1
VARMA	88.54	9819.59	0.0195	0.1000	0.1667	0.2000	0.1
Random Forest	**46.23**	**2963.45**	**0.0112**	0.8000	0.6667	1.0000	**0.8**
XGBoost	72.94	6344.49	0.0153	0.6000	0.5000	0.7500	0.6
LightGBM	62.62	6264.31	0.0146	0.3000	0.1667	0.3333	0.3
LSTM	131.33	25675.00	0.030	0.400	0.500	0.500	0.4
GRU	144.61	30764.41	0.0344	0.700	0.667	0.800	0.7

Table 4.17: Mean 1-day prediction scores for all sentiment-based models.

For 1-day predictions, Random Forest models had the lowest average RMSE, MSE, and MAPE scores (46.23, 2963.45, and 0.0112, respectively). It was also the best model for predicting, whether the price would go up or down (80% accuracy). All tree-based models performed significantly better than statistical models (VAR and VARMA). LSTM and GRU models performed noticeably poorer than other models.

4.5.2 3-day predictions

Summarized results for 3-day sentiment-based predictions are shown in figure 4.10, and in table 4.18 below.

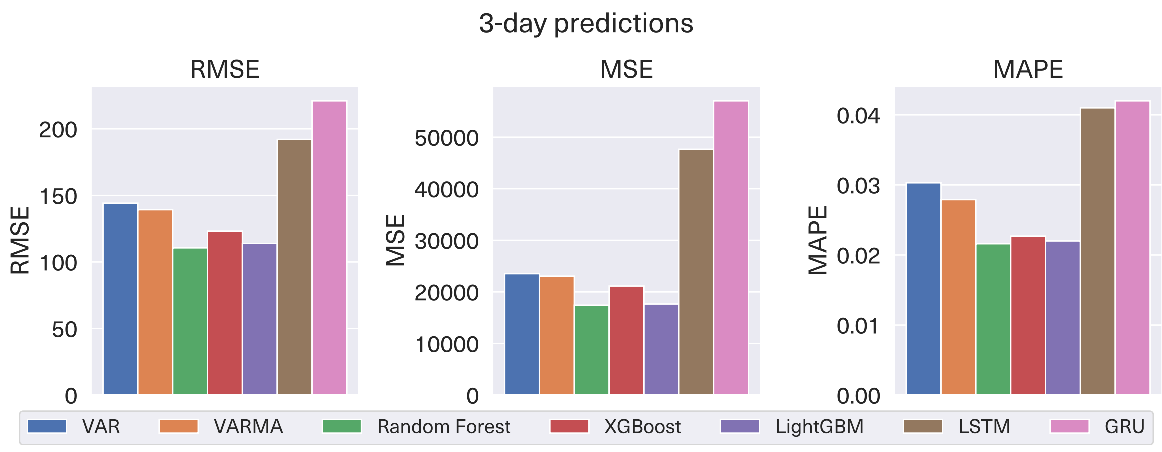


Figure 4.10: Average 3-day prediction errors of all sentiment-based models.

MODEL	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
VAR	144.21	23531.97	0.0303	0.3665	0.4500	0.4166	0.5
VARMA	139.27	23066.89	0.0279	0.4667	0.6500	0.5666	0.3
Random Forest	**110.58**	**17430.71**	**0.0216**	0.4332	0.4000	0.4074	**0.8**
XGBoost	123.21	21141.95	0.0227	0.4333	0.4000	0.3834	0.6
LightGBM	113.92	17654.84	0.0220	0.3333	0.3000	0.2593	0.6
LSTM	192.11	47655.96	0.041	0.433	0.550	0.483	0.3
GRU	221.02	57031.16	0.0420	0.400	0.450	0.400	0.6

Table 4.18: Mean 3-day prediction scores for all sentiment-based models.

For 3-day predictions, Random Forest had the best overall performance in terms of RMSE, MSE, and MAPE (110.58, 17430.71, and 0.0216, respectively). Random Forest was also the best model for predicting the trend of the price (80% accuracy). Tree-based models performed noticeably better than statistical models in 3-day predictions as well.

4.5.3 5-day predictions

The results for 5-day sentiment-based predictions are shown in figure 4.11 and table 4.19.

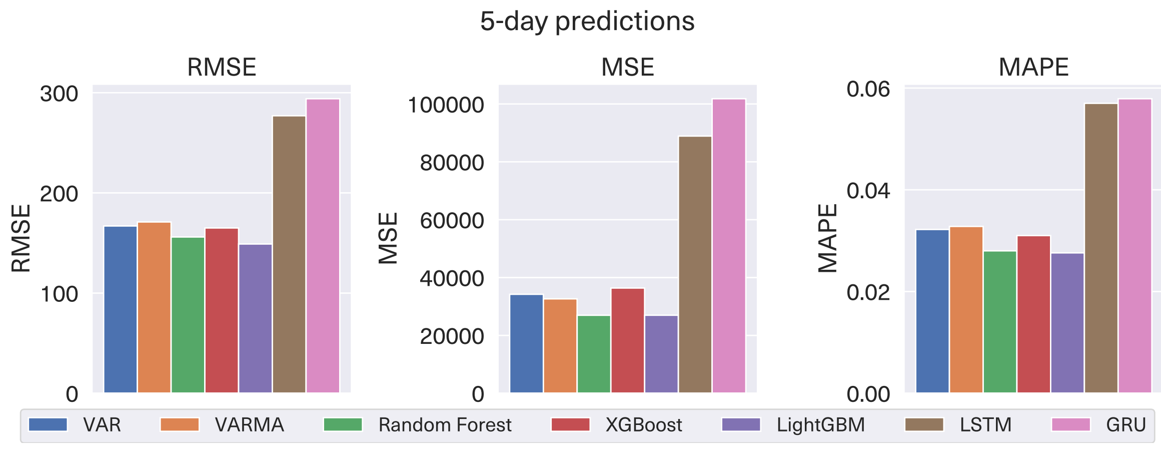


Figure 4.11: Average 5-day prediction errors of all sentiment-based models.

MODEL	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
VAR	166.93	34213.29	0.0322	0.4000	0.4000	0.3958	0.5
VARMA	170.72	32640.43	0.0328	0.4200	0.5667	0.4250	0.3
Random Forest	156.60	**26966.42**	0.0280	0.5400	0.5500	0.5167	0.3
XGBoost	165.09	36378.45	0.0310	0.5000	0.5668	0.5001	0.3
LightGBM	**149.16**	26977.35	**0.0276**	0.4000	0.3500	0.3666	0.4
LSTM	277.53	88910.57	0.057	0.460	0.517	0.442	0.6
GRU	294.54	101811.09	0.0579	0.560	0.667	0.607	**0.7**

Table 4.19: Mean 5-day prediction scores for all sentiment-based models.

For 5-day predictions, LightGBM had the lowest RMSE, and MAPE scores (149.16, and 0.0276, respectively). Random Forest had a slightly lower MSE at 26966.42, suggesting that some 5-day predictions of the LightGBM model had comparatively larger errors than those of the Random Forest model. As a result, the mean MSE of the Random Forest model was marginally lower. The trend was predicted poorly by all models when compared to 1-day and 3-day predictions.

4.5.4 7-day predictions

The results for 7-day sentiment-based predictions are shown in figure 4.12 and table 4.20.

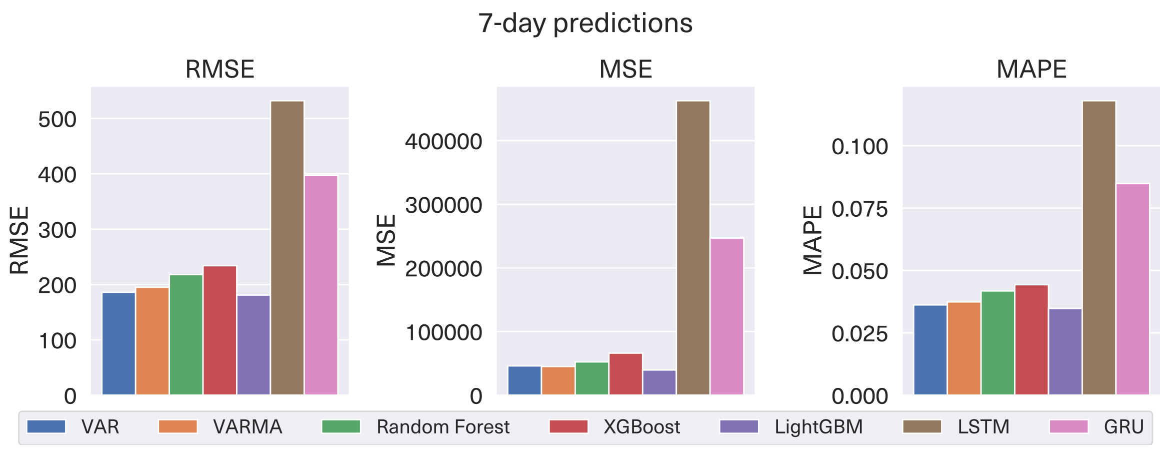


Figure 4.12: Average 7-day prediction errors of all sentiment-based models.

MODEL	RMSE	MSE	MAPE	ACCURACY	RECALL	PRECISION	TREND
VAR	186.05	46186.04	0.0362	0.4003	0.3916	0.3333	0.5
VARMA	195.66	45254.47	0.0374	0.4145	0.5500	0.4183	0.5
Random Forest	218.56	52448.64	0.0418	0.5713	0.5416	0.6001	0.5
XGBoost	234.74	66233.60	0.0443	0.4714	0.6251	0.4950	0.4
LightGBM	**181.16**	**39698.35**	**0.0348**	0.5285	0.5499	0.5083	0.2
LSTM	532.22	462666.05	0.118	0.500	0.492	0.530	0.3
GRU	397.96	246997.36	0.0848	0.386	0.583	0.385	**0.6**

Table 4.20: Mean 7-day prediction scores for all sentiment-based models.

Overall, LightGBM had the best performance in 7-day predictions, with VAR being a close second. In 7-day predictions, statistical models performed significantly better than Random Forest and XGBoost. The MSE of the RNNs, especially LSTM, is an order of magnitude larger than those of other models in 7-day predictions. The longer the prediction, the more erratically RNNs seem to behave.

5. Conclusion

5.1 Conclusions

This work investigated whether the Twitter sentiment of popular Bitcoin-related tweets can be a useful feature in predicting the future price of bitcoin. Furthermore, various multivariate time series forecasting methods were compared, including statistical, tree-based, and neural network models. It was demonstrated that compared to the baseline OHLC price data, sentiment-based features decreased the error of predictions in virtually all models and prediction lengths tested, excluding the RNN-based models, where the effect was the opposite.

Tree-based models (Random Forest, XGBoost, LightGBM) performed consistently better than statistical models (VAR, VARMA) in 1-day, 3-day, and 5-day predictions. All statistical and tree-based models performed significantly better than the neural net models. The Random Forest performed the best in 1-day and 3-day predictions, while LightGBM had the best performance in 5-day and 7-day predictions. In 5-day and 7-day predictions, the models started to become increasingly inaccurate. Predicting upward or downward price movements worked the best with Random Forest and XGBoost models, and in shorter intervals (1-day and 3-day predictions). However, as a whole, the trend prediction was unreliable.

Overall, RNNs performed poorly when compared to statistical and tree-based models. A probable explanation for this is that the amount of available training data was insufficient, as deep learning models are more efficient with larger datasets. It can therefore be suggested that with smaller training sets, tree-based models may be a better choice than neural nets for this kind of application.

5.2 Limitations

As was mentioned earlier, the inferior performance of RNNs was perhaps partly due to the small size of the training dataset. It is reasonable to assume that with a larger, multi-year dataset, RNNs could eventually leverage their capability of making long-term connections in the data, and produce similar or even better results than the statistical and tree-based models. Unfortunately, the restrictions placed on the Twitter API make gathering historical tweets exceedingly difficult.

One inherent limitation of studies like this is that there is rarely any information on how the models perform in the real markets. What may work in a controlled setting may not work at all in the real world. On the other hand, evaluating performance in the real world would require developing an algorithmic trading robot, which is a whole topic unto itself.

5.3 Future work

This work could be extended by adding more historical data and exploring additional explanatory variables and their combinations. Instead of Twitter, sentiment could be extracted from other social media platforms, such as StockTwits, or Reddit. Sentiment data could also be augmented with other topical information, such as Google search trends, Wikipedia page views, or news headlines about bitcoin. Exploring the price prediction of other cryptocurrencies would also be a fascinating topic. While bitcoin may be the most popular cryptocurrency right now, other popular contenders have emerged in recent years, with expanded technological capabilities, such as Ethereum.

Predicting the future price movements of a highly volatile asset remains a challenging task. Markets can act quickly and irrationally, and due to the potential rewards, there is no shortage of competition. François Chollet, the creator of the Keras library wrote that successful trading amounts to “information arbitraging: exploiting a piece of information others don’t know” [80]. In situations like this, even a small edge over the majority can prove valuable. That is, at least until other people find it as well.

Bibliography

[1] M. Baker and J. Wurgler, “Investor sentiment in the stock market,” *Journal of economic perspectives*, vol. 21, no. 2, pp. 129–152, 2007.

[2] D. Guégan and T. Renault, “Does investor sentiment on social media provide robust information for bitcoin returns predictability,” *Finance Research Letters*, vol. 38, p. 101494, 2021.

[3] *On the predictability of stock market behavior using stocktwits sentiment and posting volume*, vol. Portuguese conference on artificial intelligence, Springer, 2013.

[4] “CoinMarketCap — Global Cryptocurrency Market Charts,” Mar 2022. https://coinmarketcap.com/charts [Online; accessed 11. Feb. 2022].

[5] “bitcointalk.org — Bitcoin Forum,” Sep 2021. https://bitcointalk.org [Online; accessed 23. Aug. 2021].

[6] “Cryptocurrency: The Discussion on Twitter Keeps Growing,” Feb 2022. https://blog.twitter.com/en_us/topics/insights/2017/Cryptocurrency-The-Discussion-on-Twitter-Keeps-Growing [Online; accessed 11. Feb. 2022].

[7] “BitInfoCharts — Bitcoin Tweets Chart,” Sep 2021. https://bitinfocharts.com/comparison/bitcoin-tweets.html.

[8] “Crypto influencers you should follow in 2022,” Feb 2022. https://economictimes.indiatimes.com/markets/cryptocurrency/crypto-influencers-you-should-follow-in-2022/articleshow/89661967.cms [Online; accessed 11. Mar. 2022].

[9] “Kim Kardashian, Floyd Mayweather and a crypto token’s wild ride,” Jan 2022. https://www.ft.com/content/a6dd4d6f-6a86-48cc-992c-f8a32c64fdd7 [Online; accessed 22. Feb. 2022].

[10] “Spanish watchdog clashes with footballer Andrés Iniesta over Binance tweet,” Nov 2021. https://www.ft.com/content/6ebf5176-a0b6-4370-866b-103de0389004 [Online; accessed 22. Feb. 2022].

[11] P. Khuwaja, S. A. Khowaja, and K. Dev, "Adversarial learning networks for fintech applications using heterogeneous data sources," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[12] *Sentiment-Driven Price Prediction of the Bitcoin based on Statistical and Deep Learning Approaches*, vol. 2020 International Joint Conference on Neural Networks (IJCNN), IEEE, 2020.

[13] C. Chen, L. Liu, and N. Zhao, "Fear sentiment, uncertainty, and bitcoin price dynamics: The case of COVID-19," *Emerging Markets Finance and Trade*, vol. 56, no. 10, pp. 2298–2309, 2020.

[14] D. Shen, A. Urquhart, and P. Wang, "Does twitter predict bitcoin," *Economics Letters*, vol. 174, pp. 118–122, 2019.

[15] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *Journal of computational science*, vol. 2, no. 1, pp. 1–8, 2011.

[16] A. Salač, *Forecasting of the cryptocurrency market through social media sentiment analysis.* PhD thesis, University of Twente, 2019.

[17] *Predicting Cryptocurrency Value using Sentiment Analysis*, vol. 2019 International Conference on Intelligent Computing and Control Systems (ICCS), IEEE, 2019.

[18] M. A. Naeem, I. Mbarki, M. T. Suleman, X. V. Vo, and S. J. H. Shahzad, "Does twitter happiness sentiment predict cryptocurrency," *International Review of Finance*, vol. 21, no. 4, pp. 1529–1538, 2021.

[19] *Predicting Cryptocurrency Price Movements Based on Social Media*, IEEE, 2019.

[20] *Vader: A parsimonious rule-based model for sentiment analysis of social media text*, vol. Proceedings of the International AAAI Conference on Web and Social Media 8(1), 2014.

[21] O. Kraaijeveld and J. De Smedt, "The predictive power of public twitter sentiment for forecasting cryptocurrency prices," *Journal of International Financial Markets, Institutions and Money*, vol. 65, p. 101188, 2020.

[22] T. Pano and R. Kashef, "A complete VADER-based sentiment analysis of bitcoin (BTC) tweets during the era of COVID-19," *Big Data and Cognitive Computing*, vol. 4, p. 33, 2020.

[23] J. Abraham, D. Higdon, J. Nelson, and J. Ibarra, "Cryptocurrency price prediction using tweet volumes and sentiment analysis," *SMU Data Science Review*, vol. 1, no. 3, p. 1, 2018.

[24] N. Oliveira, P. Cortez, and N. Areal, "Stock market sentiment lexicon acquisition using microblogging data and statistical measures," *Decision Support Systems*, vol. 85, pp. 62–73, 2016.

[25] D. Guégan and T. Renault, "Does investor sentiment on social media provide robust information for bitcoin returns predictability," *Finance Research Letters*, vol. 38, p. 101494, 2021.

[26] S. Lyocsa, P. Molnar, T. Plihal, and M. Siranova, "Impact of macroeconomic news, regulation and hacking exchange markets on the volatility of bitcoin," *Journal of Economic Dynamics and Control*, vol. 119, p. 103980, 2020.

[27] *Evaluating sentiment c1assifiers for bitcoin tweets in price prediction task*, vol. 2019 IEEE International Conference on Big Data (Big Data), IEEE, 2019.

[28] *Online human-bot interactions: Detection, estimation, and characterization*, vol. Proceedings of the international AAAI conference on web and social media 11(1), 2017.

[29] M. Mirtaheri, S. Abu-El-Haija, F. Morstatter, G. V. Steeg, and A. Galstyan, "Identifying and analyzing cryptocurrency manipulations in social media," *arXiv*, p. 1902.03110v2, 2019.

[30] L. Nizzoli, S. Tardelli, M. Avvenuti, S. Cresci, M. Tesconi, and E. Ferrara, "Charting the landscape of online cryptocurrency manipulation," *IEEE Access*, vol. 8, pp. 113230–113245, 2020.

[31] R. Chowdhury, M. A. Rahman, M. S. Rahman, and M. Mahdy, "An approach to predict and forecast the price of constituents and index of cryptocurrency using machine learning," *Physica A: Statistical Mechanics and its Applications*, vol. 551, p. 124569, 2020.

[32] M. M. Patel, S. Tanwar, R. Gupta, and N. Kumar, "A deep learning-based cryptocurrency price prediction scheme for financial institutions," *Journal of Information Security and Applications*, vol. 55, p. 102583, 2020.

[33] A. Azari, "Bitcoin price prediction: An ARIMA approach," *arXiv*, p. 1904.05315v1, 2019.

[34] S. Colianni, S. Rosales, and M. Signorotti, "Algorithmic trading of cryptocurrency based on twitter sentiment analysis," *CS229 Project*, pp. 1–5, 2015.

[35] Y. Kim, J. Kim, W. Kim, J. Im, T. Kim, S. Kang, and C. Kim, "Predicting fluctuations in cryptocurrency transactions based on user comments and replies.," *PLoS One*, vol. 11, no. 8, p. e0161197, 2016.

[36] T. R. Li, A. S. Chamrajnagar, X. R. Fong, N. R. Rizik, and F. Fu, "Sentiment-based prediction of alternative cryptocurrency price fluctuations using gradient boosting tree model," *Frontiers in Physics*, vol. 7, 2019.

[37] A. Dutta, S. Kumar, and M. Basu, "A gated recurrent unit approach to bitcoin price prediction," *Journal of Risk and Financial Management*, vol. 13, no. 2, p. 23, 2020.

[38] S. Ji, J. Kim, and H. Im, "A comparative study of bitcoin price prediction using deep learning," *Mathematics*, vol. 7, no. 10, p. 898, 2019.

[39] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[40] "Bitcoin slides to six-month low as China warns of risks," Nov 2019. https://www.ft.com/content/1fad6628-0f72-11ea-a7e6-62bf4f9e548a [Online; accessed 22. Feb. 2022].

[41] "Cryptocurrencies take new dive to end turbulent week," May 2021. https://www.ft.com/content/72ae88cd-bcbf-4779-ae2a-c743b09c8b73 [Online; accessed 22. Feb. 2022].

[42] C. Bovaird, "Bitcoin has fallen more than 50high—what's next?," *Forbes*, 2021. https://www.forbes.com/sites/cbovaird/2021/05/19/bitcoin-has-fallen-more-than-50-from-its-all-time-high-whats-next/ [Online; accessed 7. Jun. 2021].

[43] "Tesla sends bitcoin to record high with $1.5bn investment," Feb 2021. https://www.ft.com/content/5e83f15e-ea2c-4d2f-8ae8-bf72fc5effd0 [Online; accessed 22. Feb. 2022].

[44] "Musk says Tesla no longer plans to accept payment in bitcoin," May 2021. https://www.ft.com/content/052853fa-9816-4624-8dd3-6321c01ac875 [Online; accessed 22. Feb. 2022].

[45] R. Marshall, "Bitcoin: Where two worlds collide," *Bond L. Rev.*, vol. 27, p. 89, 2015.

[46] D. G. Baur, K. Hong, and A. D. Lee, “Bitcoin: Medium of exchange or speculative assets,” *Journal of International Financial Markets, Institutions and Money*, vol. 54, pp. 177–189, 2018.

[47] The Economist, “The price of bitcoin has soared to record heights,” Jan 2021. https://www.economist.com/graphic-detail/2021/01/04/the-price-of-bitcoin-has-soared-to-record-heights [Online; accessed 7. Jun. 2021].

[48] F. Pozzi, E. Fersini, E. Messina, and B. Liu, *Sentiment analysis in social networks*. Morgan Kaufmann, 2016.

[49] “Vader Sentiment Analysis – GitHub repository,” Sep 2021. https://github.com/cjhutto/vaderSentiment.

[50] “NLTK documentation of the VADER module,” Nov 2021. https://www.nltk.org/api/nltk.sentiment.vader.html [Online; accessed 11. Nov. 2021].

[51] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.

[52] K. Holden, “Vector auto regression modeling and forecasting,” *Journal of Forecasting*, 1995.

[53] C. A. Sims, “Macroeconomics and reality,” *Econometrica: journal of the Econometric Society*, pp. 1–48, 1980.

[54] H. Lütkepohl, *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[55] A. Hatemi-J, “Multivariate tests for autocorrelation in the stable and unstable VAR models,” *Economic Modelling*, vol. 21, no. 4, pp. 661–683, 2004.

[56] L. Breiman, “Random forests,” vol. 45(1), pp. 5–32, 2001.

[57] G. Biau and E. Scornet, “A random forest guided tour,” *arXiv*, p. 1511.05741v1, 2015.

[58] M. Fernández-Delgado, E. Cernadas, and S. Barro. . . , “Do we need hundreds of classifiers to solve real world classification problems,” *The journal of machine . . .* , 2014.

[59] *XGBoost: A Scalable Tree Boosting System*, (New York, NY, USA), ACM, 2016.

[60] S. Liu, J. Xiao, J. Liu, X. Wang, J. Wu, and J. Zhu, “Visual diagnosis of tree boosting methods,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 163–173, 2018.

[61] S. Yang and H. Zhang, “Comparison of several data mining methods in credit card default prediction,” *Intelligent Information Management*, vol. 10, no. 5, pp. 115–122, 2018.

[62] G. Ke, Q. Meng, T. Finley, T. Wang, and W. Chen. . . , “Lightgbm: A highly efficient gradient boosting decision tree,” *Advances in neural . . .* , 2017.

[63] M. Boden, “A guide to recurrent neural networks and backpropagation,” *the Dallas project*, 2002.

[64] *The performance of LSTM and BiLSTM in forecasting time series*, vol. 2019 IEEE International Conference on Big Data (Big Data), IEEE, 2019.

[65] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[66] “Schematic of the Long-Short Term Memory cell, a component of recurrent neural networks,” 2018. https://commons.wikimedia.org/wiki/File:LSTM_Cell.svg [Online; accessed 23. Feb. 2022] Original schematic by Guillaume Chevalier, adapted under CC BY-SA 4.0 license.

[67] K. Cho, B. v. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” *arXiv*, p. 1406.1078v3, 2014.

[68] “Schematic of the Gated Recurrent Unit memory cell, a component of recurrent neural networks,” 2018. https://commons.wikimedia.org/wiki/File:Gated_Recurrent_Unit,_type_2.svg [Online; accessed 23. Feb. 2022] Original schematic by Jeblad, adapted under CC BY-SA 4.0 license.

[69] L. Ungurean, M. V. Micea, and G. Cârstoiu, “Online state of health prediction method for lithium-ion batteries, based on gated recurrent unit neural networks,” *International Journal of Energy Research*, vol. 44, no. 8, pp. 6767–6777, 2020.

[70] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv*, p. 1412.3555v1, 2014.

[71] “Standard search API, Twitter.com,” https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/api-reference/get-search-tweets [Online; accessed 12. Jul. 2021].

[72] “Kaggle: Your Machine Learning and Data Science Community,” https://www.kaggle.com/ [Online; accessed 12. Jul. 2021].

[73] J. Badiola, “17.7 million Bitcoin Tweets,” November 2020. https://www.kaggle.com/jaimebadiola/177-million-bitcoin-tweets [Online; accessed 12. Jul. 2021, version 1].

[74] M. Zielinski, “Bitcoin Historical Data,” April 2021. https://www.kaggle.com/mczielinski/bitcoin-historical-data, [Online; accessed 1. Jun. 2021, version 7].

[75] “’Hodl’ ’Stonks’ ’To The Moon’: Your Guide to Internet Crypto-Slang,” Nov 2021. https://www.news18.com/news/buzz/hodl-stonks-to-the-moon-your-guide-into-internet-crypto-slang-3946820.html [Online; accessed 10. Nov. 2021].

[76] “Software Repository for Accounting and Finance,” Nov 2021. https://sraf.nd.edu/textual-analysis/resources/ [Online; accessed 10. Nov. 2021].

[77] “GYM Rewards (GYM) token,” Nov 2021. https://www.coingecko.com/en/coins/gym-rewards [Online; accessed 10. Nov. 2021].

[78] “Counting characters when composing Tweets,” Nov 2021. https://developer.twitter.com/en/docs/counting-characters [Online; accessed 9. Nov. 2021].

[79] *Parkinson disease prediction using feature selection technique in machine learning*, IEEE, 2021.

[80] “François Chollet on Twitter,” Mar 2022. https://twitter.com/fchollet/status/1177633368130781184 [Online; accessed 20. Mar. 2022].