Supervised Learning

Classification of Bitcoin Price Development

Lukas Hermans

Università degli Studi di Milano
lukas.hermans@studenti.unimi.it

June 25, 2021

Abstract

The market price of the cryptocurrency Bitcoin has increased impressively since Bitcoin's first occurrence in 2009. Today, its market cap is comparable to the market cap of companies like Facebook and Tesla at the stock market. The differences of Bitcoin to classical financial assets like shares lie in the underlying blockchain technology. This makes Bitcoin an interesting asset for investors and traders, and gives rise to scientific studies on the market price development of Bitcoin.

The scope of the present work is the prediction of tomorrow's Bitcoin market price based on data from today and the past days. The starting point is the development of a dataset that transforms the market price prediction into a binary classification problem. Then, the goal is to simply predict if the Bitcoin market price will increase (go "up") or decrease (go "down") but not to estimate the exact market price of tomorrow.

For this purpose, overall, four different learning algorithms are trained on the binary classification dataset. The analysis reveals that a simple majority predictor - that constantly predicts that the Bitcoin market price will increase - leads to a prediction accuracy of 55.54 % with a sensitivity of 100 % but a specificity of 0.00 %. The application of a deep neural network for various settings reproduces the majority predictor, and does not seem to be suitable for the binary classification of tomorrow's Bitcoin market price. The highest accuracy of 56.70 % is obtained with a logistic regression. The K-nearest neighbors algorithm leads to a comparably low accuracy of 51.80 %. In contrast to the majority predictor and the deep neural network, the logistic regression and the K-nearest neighbors algorithm lead to predictors with a more balanced sensitivity and specificity. For traders, the optimal choice of a prediction algorithm depends on the specific investment strategy which is beyond the scope of the present work.

It is also discussed how a successful implementation of a neural network might be possible, and how the prediction accuracies could be increased by including more information like social media and Google search analysis in the binary classification dataset.

Contents

1 Introduction

Recently, cryptocurrencies such as Bitcoin, Ether, and Dogecoin have gained increasing attention from traders and investors. In particular, the first cryptocurrency called Bitcoin - developed by an unknown individual or group under the pseudonym Satoshi Nakamoto and first presented in 2009 [1] - has achieved a market cap of about 750 billion United States dollar (USD) as of 14 June 2021. This is comparable to the market cap of companies like Facebook and Tesla at the stock market [2][3].

The original purpose of Bitcoin is the establishment of a form of decentralized digital money that can be used for uncensored transactions on the internet. The basis of Bitcoin is the so-called blockchain. As its name suggests, the blockchain is an ordered chain of blocks. Each block contains Bitcoin transactions from a certain time period (typically, roughly 10 minutes). New blocks - and thus new recorded transactions - are appended to the blockchain by solving a mathematical task. The task is solved by investing computing power. The person or entity that solves the mathematical task first is rewarded with new Bitcoins and with the transactions fees of all transactions in the new block. The addition of a new block to the blockchain and the distribution of new Bitcoins is called Bitcoin mining. A simplified illustration of the Bitcoin blockchain is depicted in Fig. 1.

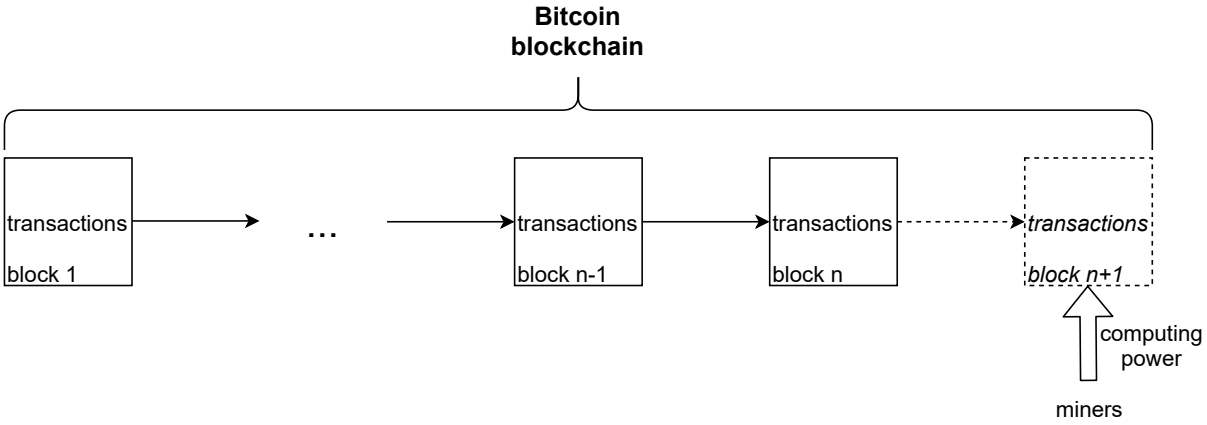


Figure 1: Simplified illustration of the Bitcoin blockchain starting from the first block nr. 1 that is also called genesis block, and was mined by Satoshi Nakamoto in 2009. Each block contains information on Bitcoin transactions for time periods of roughly 10 minutes. Suppose, the current length of the Bitcoin blockchain is given by n blocks. The addition of a new block with the nr. $n+1$ to the blockchain is called Bitcoin mining, and is achieved by investing computing power in solving a mathematical task.

Besides its original intention of a form of digital money, Bitcoins are more and more treated as a financial asset that is exchanged at specific exchanges. These exchanges are called cryptocurrency exchanges because they allow their customers to exchange Bitcoin with other cryptocurrencies and with fiat currencies such as USD. Examples for cryptocurrency exchanges are Binance, Kraken, and Coinbase. As a financial asset, the Bitcoin market price is the result of demand and supply that is limited as the distribution of new Bitcoins requires computing power. This makes the Bitcoin comparable to classical financial assets such as company shares. However, its volatility is usually much larger than share prices at the stock market [4].

For traders and investors, it is of course of interest to predict the future development of Bitcoin's market price in order to maximize returns. The present work is concerned with the prediction of tomorrow's market price based on data from today and the past days. The prediction of the Bitcoin market price is treated as a binary classification problem, such that predictions are either "up" or "down", dependent on the predicted direction of tomorrow's Bitcoin market price.

The structure of the present paper is as follows. In Section 2, time series data of the Bitcoin market price and other measures are transformed into a dataset that is suitable for a binary classification task as described above. The binary classification dataset allows the application of supervised machine learning algorithms, whose theoretical basics are summarized in Section 3. In addition, the same Section includes an explanation on techniques that are applied to optimize hyperparameters and to evaluate the performance

of these machine learning algorithms. Section 4 contains a list of the software that is applied to develop the binary classification dataset and to implement the prediction algorithms. The results of the application of the different machine learning algorithms on the Bitcoin price development classification problem are presented in Section 5. A comparison of the different learning algorithms as well as a detailed discussion of the results is given in Section 6. Finally, in Section 7, the findings of the present work are summarized.

2 Dataset

In the present work, the Bitcoin market price prediction is treated as a binary classification problem. To the best knowledge of the author of the present paper, there is no standard binary classification dataset of the Bitcoin market price direction available. Hence, this Section is concerned with the development of such a binary classification dataset starting from time series data on the Bitcoin market price and other measures, in Subsection 2.1. The transformation into a binary classification dataset is treated in Subsection 2.2.

2.1 Bitcoin Time Series Data

The Bitcoin time series data is taken from the Blockchain.com API, see [5]. In the present work, the time period from 01 September 2011 to 15 June 2021 is considered. The starting date of the time period is chosen such that there were already some established cryptocurrency exchanges that allow a reliable determination of the Bitcoin market price.

The measures that are obtained from the Blockchain.com API are displayed in Fig. 2 as a function of the date. All measures are day-by-day time series for a total of 3576 days corresponding to the considered time period mentioned above. The most important measure is the Bitcoin market price that reports the value of one Bitcoin in the fiat currency USD. In the present work, USD is chosen as the reference currency as is usually done for financial assets. The market cap describes the value of all Bitcoins that have ever been mined at the current market price. It is the product of the total number of Bitcoins and the current market price. Another measure is the number of transactions per day, i.e. how often Bitcoins are transferred on each day. The transaction volume reported in Bitcoins (BTC) is the total worth of these transactions. The average blocksize in megabyte (MB) is a measure for the number of transactions contained in each block. In combination with the relative difficulty of the mathematical task to mine new blocks, this reports how much computing power has to be applied to mine a new block. The hash rate expressed in EH/s ("Exahashes per second") is an estimate of the total computing power that all mining entities are applying on each day. Finally, the miner's revenue is the sum of the reward that miners obtain for mining a new block and the transaction fees per block. It reports how miners are rewarded for their computational efforts.

From Fig. 2, it seems that the market price and the market cap reveal the exact same development. The reason for this is that the market price is the USD value of one Bitcoin, and the market cap the USD value of all available Bitcoins. In other words, the market cap is the market price multiplied by the total number of available Bitcoins. Thus, the market cap is not useful to predict the Bitcoin market price of tomorrow if the market price of today and the past days is already considered. Also, the block difficulty and the miner's revenue show a time series similar to the market price. In contrast, the transaction volume appears to be fluctuating with a constant magnitude without any trend over the time. From this perspective, also the transaction volume cannot be used to predict the market price because it contains only time independent information.

These qualitative impressions can be quantified by plotting the correlation matrix of the different time series, as in Fig. 3. Indeed, the market cap has a correlation of 1 with the market price. In addition, the miner's revenue and the difficulty have a correlation coefficient of 0.8 or higher with the market price. Thus, these measures are not further considered since they are not expected to be useful for the prediction of tomorrow's Bitcoin market price because they simply follow the development of the Bitcoin market price. Furthermore, the transaction volume - as already observed above - is only weakly correlated to the market price and the other measures. Hence, also the transaction volume is not helpful for the prediction of the Bitcoin market price and is dropped.

An overview of the remaining measures can be found in Fig. 4. The Figure reports a scatterplot matrix of the four remaining measures that appear to be useful for the prediction of tomorrow's Bitcoin market price direction. The diagonal contains the densities of the measures. In addition, Pearson's correlation coefficients - the same as in Fig. 3 - are stated. As the correlation coefficients are computed from time series data, the measures are not only moderatly correlated with the Bitcoin market price but they are also correlated among themselves.

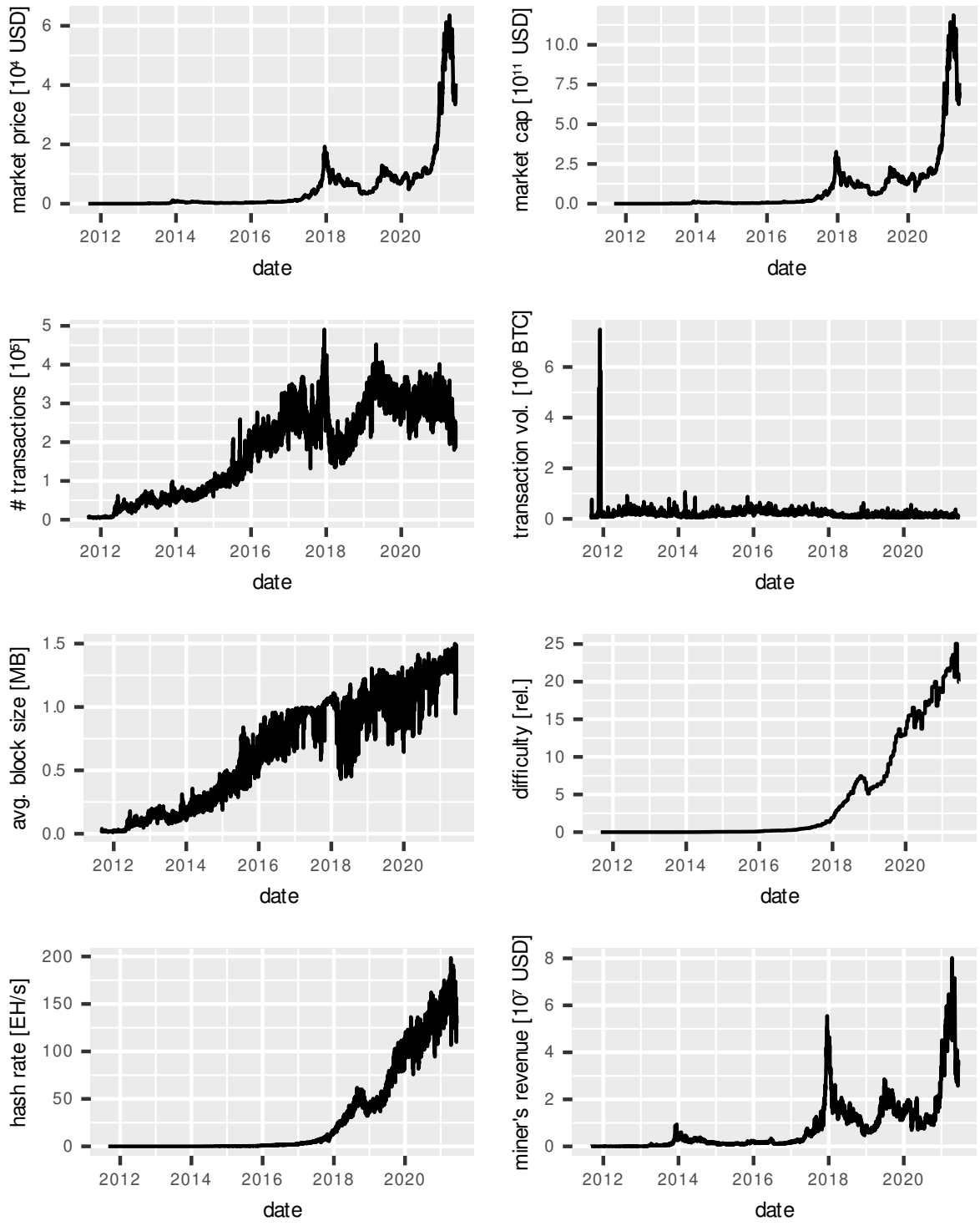


Figure 2: Bitcoin day-by-day time series data obtained from the Blockchain.com API, see [5]. The time series of the Bitcoin market price and the other displayed measures are reported for the time period from 01 September 2011 to 15 June 2021. This is a total of 3576 days. The different measures are explained in the main text.

2.2 Transformation to Binary Classification Dataset

In order to obtain a binary classification dataset for the prediction of tomorrow's direction of the Bitcoin market price, the four selected time series including the market price from the previous Subsection are used.

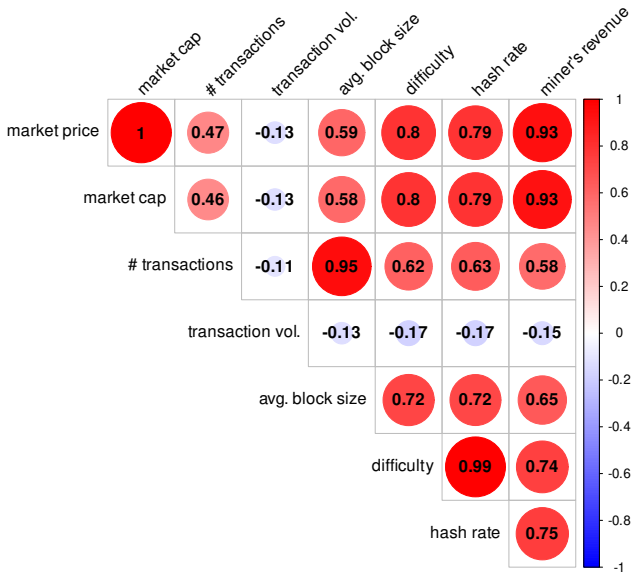


Figure 3: Pearson's correlation matrix of the time series from Fig. 2. As the market cap, the difficulty, and the miner's revenue have a correlation coefficient of 0.8 or higher, these measures are not expected to be useful for the prediction of tomorrow's Bitcoin market price. In contrast, the transaction volume is weakly correlated with the Bitcoin market price and the other measures, and does not contain any time-dependent information. Thus, it is also dropped. An overview of the remaining measures can be found in Fig. 4.

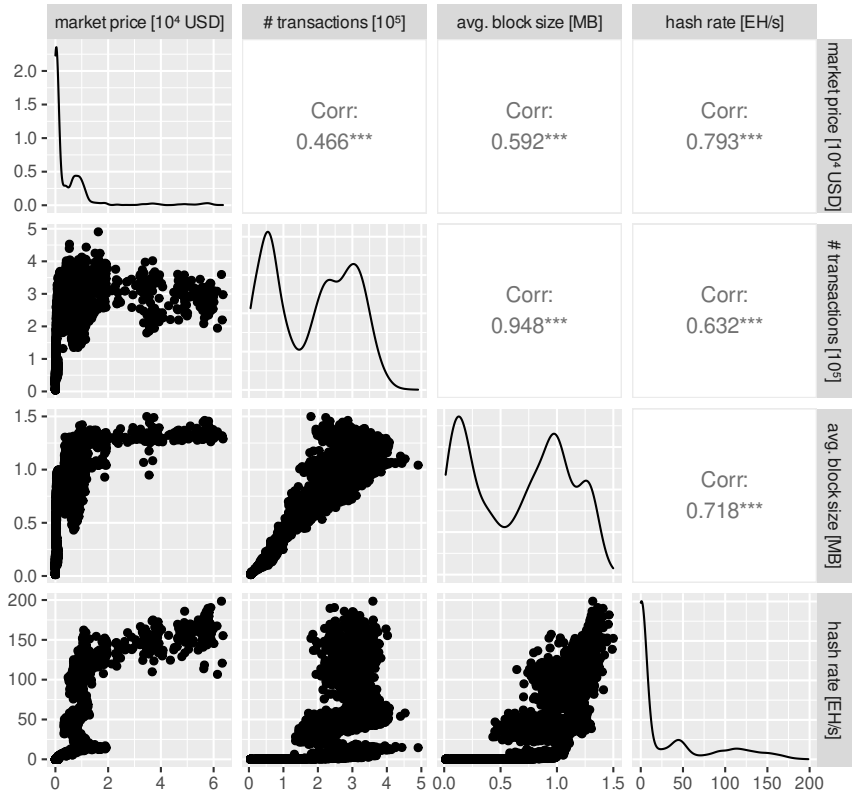


Figure 4: Scatterplot matrix of the remaining four measures selected from the original time series dataset in Fig. 2 that appear to be useful for the prediction of tomorrow's Bitcoin market price. The line plots on the diagonal are the densities of the respective measures. The upper righthand part of the Figure reports Pearson's correlation coefficients - the same as in Fig. 3 - between the measures.

First of all, Fig. 5 displays the autocorrelation of the Bitcoin market price as a function of the lag. The autocorrelation plot gives an idea on the number of days in the past including today that might contain information on tomorrow's Bitcoin price. As can be seen in the Figure, the autocorrelation drops below a value of 0.5 after a lag of about 100 days. It is expected that this period of time might be useful for the prediction of tomorrow's Bitcoin market price.

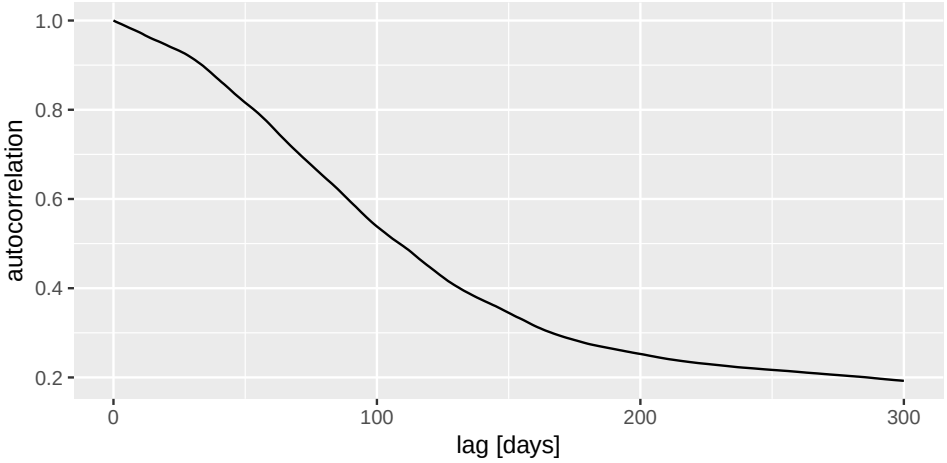


Figure 5: Autocorrelation of the Bitcoin market price. The autocorrelation drops below a value of 0.5 after a lag of about 100 days. This information is used in the construction of the binary classification dataset by applying the moving window method, see the example structure in Fig. 6.

For the construction of the binary classification dataset, the *moving window approach* (also called sliding window approach) is applied. The starting point of the moving window approach is to consider the first m days of the four selected time series presented in the last Subsection. This leads to four new time series of length m from date t_{m-2} until t_{+1}, where t_{+1} is the date of "tomorrow", t_0 is the date of "today", and so on. The dates from t_{m-2} until t_0 are considered as the features of the supervised learning approach, while the date t_{+1} is transformed into a label. In the present work, the label is "1" if the Bitcoin market price on date t_{+1} is larger than on date t_0, and as "0" if it is lower. In the following, the notations "1" and "up", and "0" and "down" are used interchangeably. This combination of features and label is the first example of the binary classification dataset. After that, the starting date of the window is shifted by one day, and another example is constructed. This process is repeated until the last date t_{+1} of the moving window coincides with the last day of the overall time series. The result is a set of labeled examples as depicted in Fig. 6 that together build the binary classification dataset. In the present work, based on the autocorrelation plot from Fig. 5, a window length of $m = 100$ is chosen.
In applying the moving window approach, for each window, the four time series - before considering them as features - are standardized using the min-max-transformation

$$\hat{x}_t = \frac{x_t - \min_{t' \in \{t_{m-2},\dots,t_0\}}(x_{t'})}{\max_{t' \in \{t_{m-2},\dots,t_0\}}(x_{t'}) - \min_{t' \in \{t_{m-2},\dots,t_0\}}(x_{t'})},$$

where $\hat{x}_t \in [0, 1]$ is the standardized window time series, and x_t the intial window time series (e.g. the Bitcoin market price). Of course, the date is $t \in \{t_{m-2}, \dots, t_0\}$ because the date t_{+1} is used for the label, and is not included in the features.

In total, 3477 examples are generated using the moving window technique.

Finally, the binary classification dataset constructed using the moving window approach is split into a training and a test part, after the 3477 examples have been shuffled randomly. The training set contains 80 % of the examples in the original dataset, while the remaining 20 % are the test set. This leads to a training set with 2782 examples, and a test set with 695 examples. Both sets are slightly unbalanced towards an increasing Bitcoin market price, see Fig. 7.

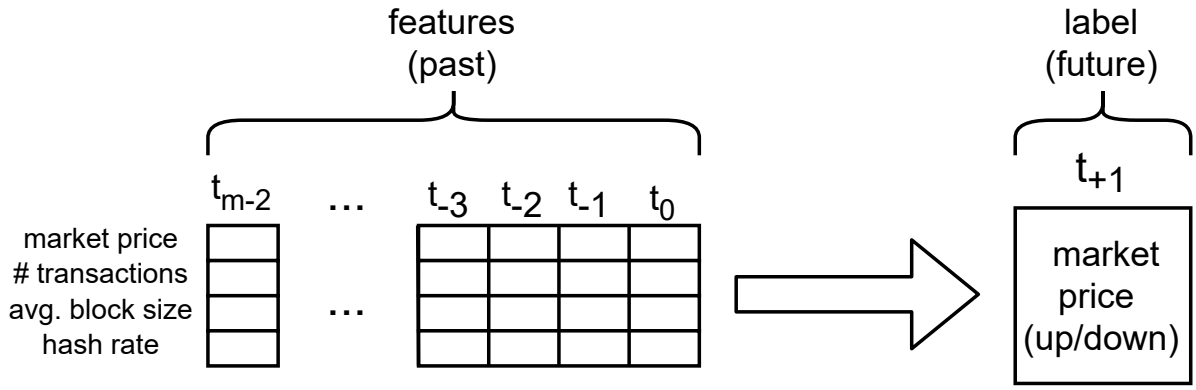
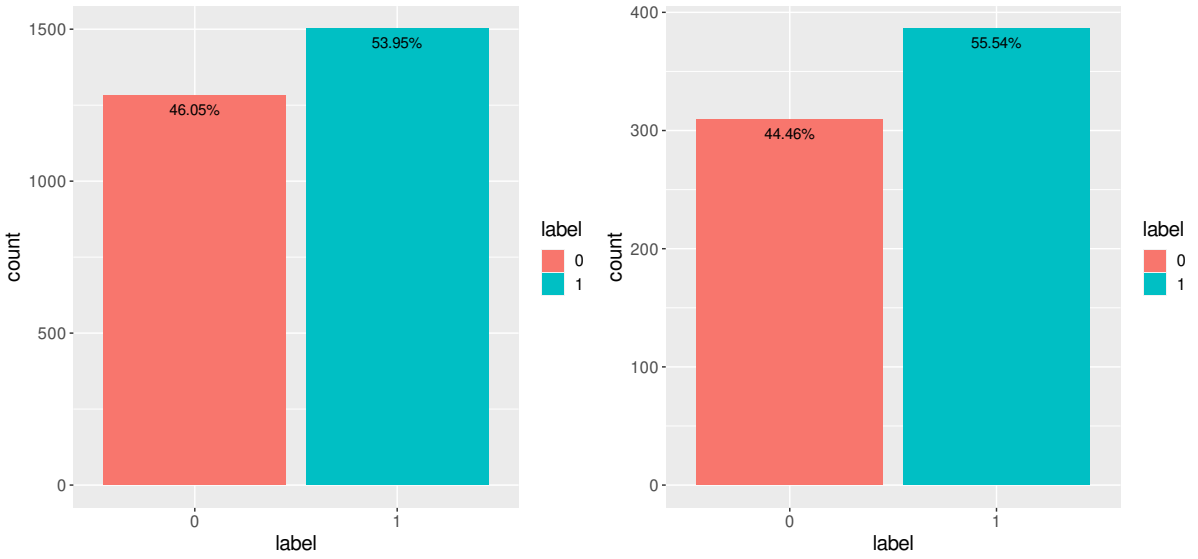


Figure 6: Structure of the examples in the binary classification dataset obtained by applying the moving window approach on the four selected time series. There are $m-1$ dates for each selected time series that are treated as features. The label of each example is the direction of tomorrow's Bitcoin market price, and is used as the label. In the present work, based on the autocorrelation plot from Fig. 5, $m = 100$ is used as the length of the moving window.



(a) training examples

(b) test examples

Figure 7: Split of the binary classification dataset into a training (80 %) and a test (20 %) part. Both sets are slightly unbalanced towards an increasing Bitcoin market price.

3 Theory

The development of a binary classification dataset in the last Section has reduced the Bitcoin prediction task into a supervised learning problem. There are several machine learning algorithms that can be used for binary classification. In the present work, the logistic regression, the K-nearest neighbor algorithm, and a deep neural network are considered as algorithms to predict tomorrow's Bitcoin price direction. In this Section, a few theoretical basics of these machine learning algorithms are summarized. The Section is concluded with a Subsection on techniques that allow the optimization of hyperparameters as well as the perfomance evaluation of these algorithms on the binary classification dataset. The discussion in this theory Section is by no means complete. For more information on the covered topics, the reader is refered to the literature, e.g. [6].

3.1 Logistic Regression

The logistic regression is the classification counterpart of the linear regression. It models the probability p for an increasing Bitcoin market price tomorrow with

$$p(\vec{x}) = \frac{e^{\beta_0 + \vec{\beta}\cdot\vec{x}}}{1 + e^{\beta_0 + \vec{\beta}\cdot\vec{x}}}, \tag{1}$$

where $\vec{x}$ contains all features (i.e. the Bitcoin market price, the number of transactions, and so on), and β_0 and $\vec{\beta}$ are trainable coefficients. The analogy to the linear regression becomes clear after rewriting Eq. 1 as follows:

$$\log\left(\frac{p(\vec{x})}{1 - p(\vec{x})}\right) = \beta_0 + \vec{\beta}\cdot\vec{x}. \tag{2}$$

This quantity is also called the *log-odds*. The righthand side of Eq. 2 is a linear function of $\vec{x}$. The parameters β_0 and $\vec{\beta}$ are estimated given a training dataset by computing the so-called maximum-likelihood estimation.

3.2 K-Nearest Neighbors Algorithm

The K-nearest neighbor algorithm interprets the feature vectors $\vec{x}$ as points in a (high-dimensional) space. It makes predictions by computing the Euclidian distance of all training points to a new point. The predicted label is the majority label of the K nearest neighbors to the new point. Typically, K is an odd number. It is also possible to compute prediction probabilities by considering the label distribution of the K nearest neighbors training points. From the machine learning algorithms presented in the present work, it is the only non-parametric one.

3.3 Deep Neural Network

Arguably, deep neural networks are the most complex machine learning algorithm applied in the present work. They consist of several layers of neurons, inspired by the structure of the human brain. The first layer is called input layer. The number of neurons of the input layer is determined by the number of features, i.e. the dimension of $\vec{x}$. The neurons of the input layer are connected with the neurons of the first hidden layer. These neurons are again connected with the neurons of the second hidden layer, and so on. The final layer is called output layer. The number of neurons of the output layer is specified by the number of classes of the label. In the case of binary classification it consists of two neurons.

Hyperparameters of a deep neural network are the number of hidden layers, and the number of neurons in each hidden layer. In addition, the so-called activation function of the neurons has to be choosen. In the present work, all the neurons in the hidden layer use the ReLU activation function. The final layer applies the softmax function which allows the interpretation of the output of the deep neural network as probabilities. Finally, the number of epochs N_{epoch} can be specified. It describes the number of times for which the deep neural network should be trained over the whole training set. For the training, the training set is divided into N_{batch} batches. Here, $N_{batch} = 32$ batches are used.

The deep neural network is trained using an optimizer. In the present work, the Adam optimizer is used. As the loss function, the binary cross-entropy is used.

3.4 Hyperparameter Optimization & Performance Evaluation

As already explained in Sec. 2, the binary classification dataset is split into a training and a test set. The training set is used to train the machine learning algorithms. For the optimization of hyperparameters during the training phase, the training set is further split into a new (slightly smaller) training set and into a validation set. For each hyperparameter, the machine learning algorithm is trained on the new training set and its performance is evaluated on the validation set. In the present work, the performance is simply measured by the prediction accuracy on the respective set. The best hyperparameter maximizes the performance of the machine learning algorithm on the validation set. Then, the machine learning algorithm for the optimal hyperparameters is retrained on the whole training set. Finally, the performance of the machine learning algorithm is evaluated on the test set. The idea is that the test set should only be used after the training and the search for the best hyperparameters in order to be able to estimate the performance of the machine learning algorithm on unseen data. This approach is called *training-, validation-, and test-split.*

A more sophisticated approach is to optimize the hyperparameters by repeating the split into training and validation set M times. The machine learning algorithm is then trained on all of the training sets and its performance is evaluated on the respective validation set. This approach is called *M-fold cross-validation.* In this case, the average performance of the machine learning algorithm on the validation sets is a more accurate estimate for the actual test accuracy on unseen data. In the present work, $M = 10$ is choosen.

4 Implementation & Software

The following list gives an overview of the software that is used to develop the binary classification dataset and to implement the machine learning algorithms:

- R (version 3.6.3): development of dataset & implementation of logistic regression and K-nearest neighbors algorithm
- RStudio (version 1.4.1717): IDE for R coding
- Python (version 3.7.10): implementation of deep neural network
- Anaconda (version 4.10.1): managing of Python modules
- Tensorflow (version 2.4.1): Python module for machine learning, especially deep learning

All R and Python code, the dataset, and the LaTeX-code of this paper can be found in the following GitHub repository: `https://github.com/lukher98/stat-learn`. The relevant folder for the present work is called "supervised". In particular, the folder "data" contains the raw data obtained from the Blockchain.com API, the time series dataset ("data.csv"), and the binary classification dataset ("m=100.csv").

All of the code was executed on a Linux machine, but it should also work on a Mac machine. For Windows, the user might have to change some path names inside the code.

5 Results

In this Section, the results of the implementation of the machine learning algorithms on the binary classification dataset are presented. For each machine learning algorithm, the first step is to find the most suitable size of the moving window m between 2 (features only for the current day) and 100 (size of the developed binary classification dataset). Note that m can be regarded as a hyperparameter. If applicable, for each value of m the intrinsic hyperparameters (such as K in the K-nearest neighbors algorithm) are optimized. The results are reported with suitable graphs and similar representations. Finally, for each machine learning algorithm, the performance in terms of prediction accuracy, sensitivity, and specificity on the test set is evaluated.

Besides the three machine learning algorithms described in the theory Section, also the performance of a simple majority predictor is evaluated. The majority predictor - that is a naive approach to solve the classification problem - is used as a reference for the performance of the other machine learning algorithms.

5.1 Majority Predictor

The majority predictor is a naive approach for solving the binary classfication problem. It constantly predicts the label that occurs most often in the training set, which in the present work is an increasing Bitcoin market price. Fig. 8 reports the training accuracy (on the whole training set) and the 10-fold CV accuracy for the majority predictor as a function of the moving window length m. As expected, both accuracies are the same and coincide with the 53.95 % labels "1" in the training set, see Fig. 7. There is also no dependence of the accuracy on the length m of the moving window because the majority predictor depends only on the labels but not on the features of the examples in the training set. From this point of view, the majority predictor is a random classifier that classifies everything with a probability of 100 % as "up".

Tab. 1 reports the confusion matrix of the majority predictor on the test set. Clearly, the majority predictor predicts only the majority label "1". This explains also the sensitivity and specificity of the majority predictor in Tab. 2. The Table additionally reports the test accuracy of the majority predictor which is given by 55.54 %. Again, this is simply the frequency of the label "1" in the test set, see Fig. 7.

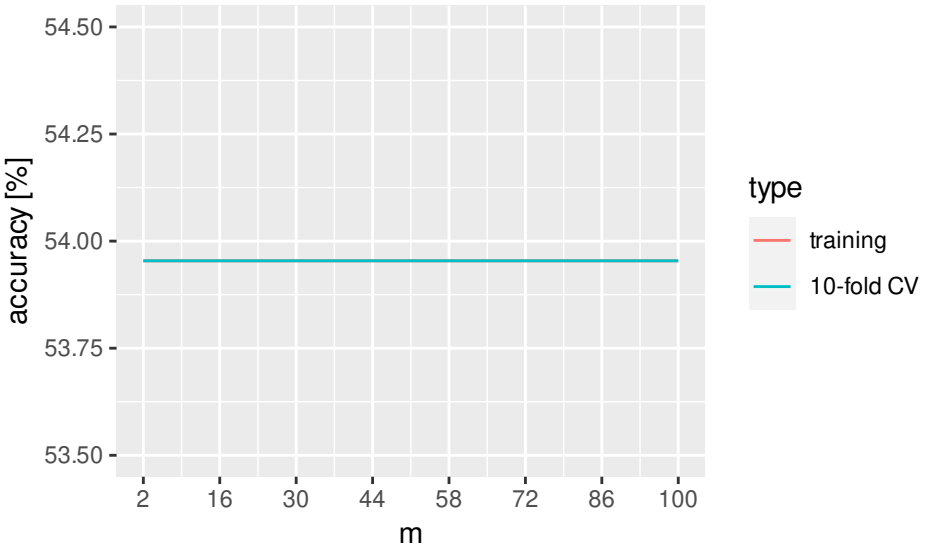


Figure 8: Training and 10-fold CV accuracy of the majority predictor as a function of the moving window length m.

5.2 Logistic Regression

A more sophisticated algorithm compared to the majority predictor is the logistic regression. Fig. 9 shows the training and 10-fold CV accuracy of the logistic regression fit as a function of the moving window

predicted \ true	0	1
0	0	0
1	309	386

Table 1: Confusion matrix of the majority predictor on the test set.

accuracy [%]	sensitivity [%]	specificity [%]
55.54	100.00	0.00

Table 2: Accuracy, sensitivity, and specificity of the majority predictor on the test set.

length m. Besides m, there are no other hyperparameters to optimize. The 10-fold CV accuracy of the logistic regression fit is maximized for a moving window length of $m = 4$. The training accuracy increases significantly with m. The main reason for this is that an increase of m is equivalent to the inclusion of more features. The model uses these features to "store" the training examples, which leads to comparably high training accuracies. However, the general performance of the logistic regression does not improve with increasing m as can be verified by the slightly decreasing 10-fold CV accuracy.

Fig. 10 shows the so-called receiver operating characteristic (ROC) curve of the logistic regression fit for $m = 4$.
Recall that the logistic regression fit assigns a probability for the label "1" to all possible feature combinations. Predictions are made by applying a probability threshold. Typically, the threshold is set to 50 %.
The ROC curve displays the sensitivity and specificity for all possible values of the threshold from 0 % in the upper righthand corner to 100 % in the lower lefthand corner. E.g., for a threshold of 0 % everything is classified as "up", and the majority predictor is obtained. As the majority predictor has a sensitivity of 100 % and a specificity of 0 %, it is located in the upper righthand corner of the ROC plot. The dashed diagonal line in the Figure is the ROC curve of a random classifier that does not make any use of information from the features.
The ROC curve of the logistic regression fit lies above the random classifier which indicates a better sensitivitiy-specificity relation. This qualitative impression is quantified by the so-called area under the curve (AUC). For the random classifier the AUC is simply 0.5, while the logistic regression fit obtains an AUC of 0.5486.

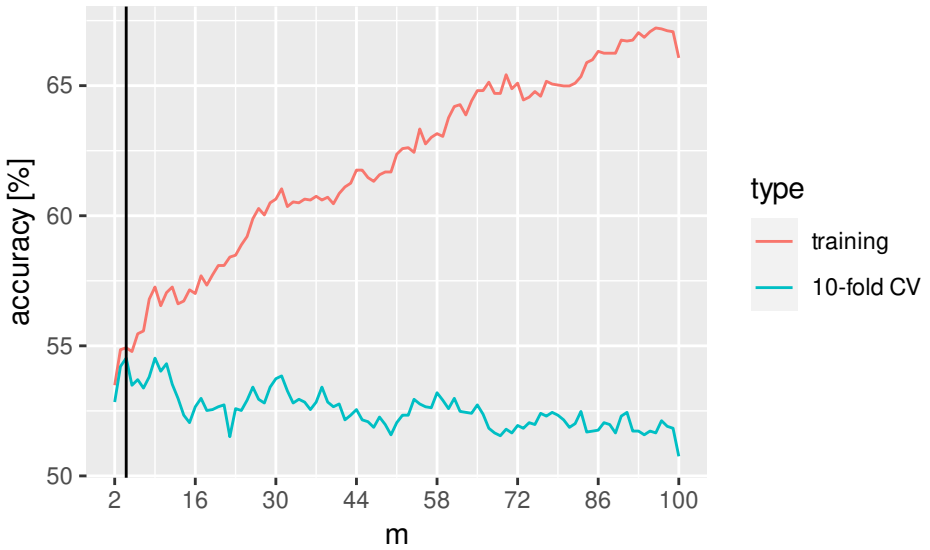


Figure 9: Training and 10-fold CV accuracy of the logistic regression fit as a function of the moving window length m for a prediction threshold of 50 %. The vertical black line indicates the value of $m = 4$ for which the 10-fold CV accuracy of the logistic regression fit is maximized.

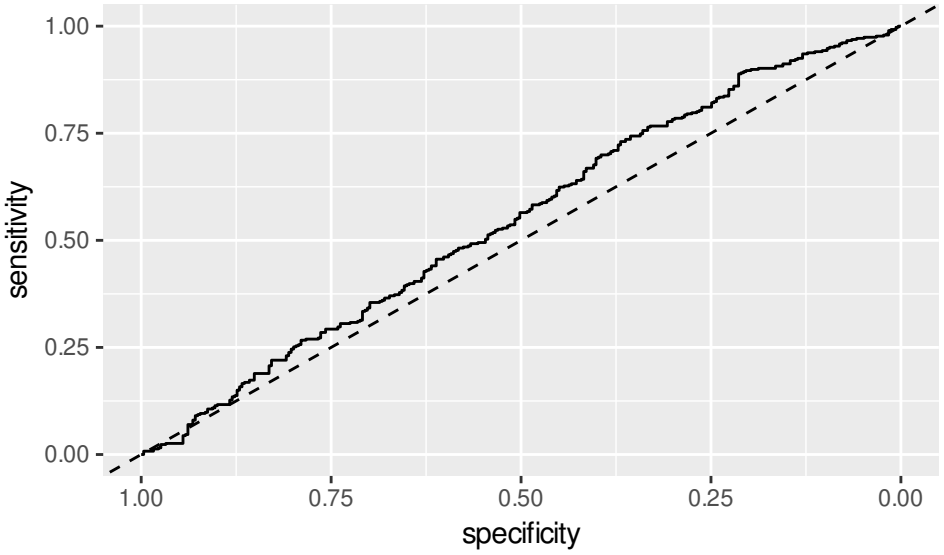


Figure 10: ROC curve of the logistic regression fit with a moving window length of $m = 4$. The dashed diagonal line is the ROC curve of a random classifier that has an AUC of 0.5. For the logistic regression fit, the AUC is 0.5486.

The confusion matrix of the logistic regression fit for $m = 4$ with a threshold of 50 % with respect to the test set, reported in Tab. 3, verifies the observation from the ROC plot and the AUC. Indeed, the logistic regression fit is able to classify both increasing and decreasing tomorrow's Bitcoin market prices correctly. In particular, the specificity is larger than 0 %, as can been seen in Tab. 4. However, it is still significantly smaller than the sensitivity. So, the logistic regression fit predicts increasing Bitcoin market prices more often correctly than decreasing market prices.

predicted \ true	0	1
0	77	73
1	232	313

Table 3: Confusion matrix of the logistic regression fit on the test set for a moving window length of $m = 4$, and a prediction threshold of 50 %.

accuracy [%]	sensitivity [%]	specificity [%]
56.70	81.09	24.92

Table 4: Accuracy, sensitivity, and specificity of the logistic regression fit on the test set for a moving window length of $m = 4$, and a prediction threshold of 50 %.

Tab. 5 reports the coefficients of the logistic regression fit for a moving window length of $m = 4$, and a prediction threshold of 50 %. Almost all of the coefficients are not significant based on their relatively high standard deviation, and on the low z scores close to zero. This has mainly two reasons. On the one hand, the actual relationship between the features and the label is not modeled correctly by the logistic regression. On the other hand, some of the features are correlated, as already observed in the development of the binary classification dataset. The influence of the correlation of the features on the standard deviatin of the estimated coefficients can be quantified by computing the so-called variance inflation coefficients (VIF). In Tab. 6, the square-root of the VIF coefficients for all of the estimated logistic regression coefficients are reported. The square-root of the VIF indicates how much larger the standard deviation of each estimated coefficient is compared to the case where there were no correlation between the features. Typically, a value of about 2 is taken as a threshold for a negligible influence of feature correlation. It is observed that especially the estimated coefficients related to the Bitcoin market price have a high VIF. This is caused by the high autocorrelation of the Bitcoin market price for lags smaller than 100 days, and is actually

desired for a prediction of tomorrow's Bitcoin market price.
All in all, the fact that the binary classification is not modelled correctly by the logistic regression, and the poor statistics make a statistical inference from the coefficients difficult. It cannot be deduced which feature is responsible for an increasing Bitcoin market price, and which for a decreasing market price because the feature correlation makes the logistic regression fit results unstable. However, the logistic regression model can still be used to make predictions, and its performance can be evaluated on the independent test set.

coefficient	estimate	std.	z score	$P(>\|z\|)$ [%]
(intercept)	−0.15	0.22	−0.688	49.13
market price (t_{-2})	0.45	−0.60	0.764	44.51
market price (t_{-1})	0.71	0.82	0.874	38.20
market price (t_0)	−0.65	0.57	−1.121	26.22
# transactions (t_{-2})	−0.28	0.27	−1.040	29.81
# transactions (t_{-1})	0.20	0.32	0.624	53.28
# transactions (t_0)	0.11	0.27	0.394	69.32
avg. block size (t_{-2})	−0.10	0.28	−0.252	80.10
avg. block size (t_{-1})	−0.54	0.32	−1.682	9.25
avg. block size (t_0)	0.69	0.28	2.501	1.24
hash rate (t_{-2})	−0.10	0.26	−0.171	86.44
hash rate (t_{-1})	−0.10	0.28	−0.043	96.56
hash rate (t_0)	−0.10	0.26	−0.029	97.67

Table 5: Estimated coefficients of the logistic regression fit for a moving window length of $m = 4$, and a prediction threshold of 50 %. For each coefficient, the standard deviation, the z score, and the probability to obtain a z score with an absolute value larger than the obtained one based on the z statistic are given.

coefficient	$\sqrt{VIF}$
market price (t_{-2})	4.69
market price (t_{-1})	6.64
market price (t_0)	4.82
# transactions (t_{-2})	1.68
# transactions (t_{-1})	2.01
# transactions (t_0)	1.70
avg. block size (t_{-2})	1.78
avg. block size (t_{-1})	2.08
avg. block size (t_0)	1.79
hash rate (t_{-2})	1.45
hash rate (t_{-1})	1.55
hash rate (t_0)	1.46

Table 6: Square-root of the VIF for all of the estimated coefficients in the logistic regression fit from Tab. 5. Typically, a value of about 2 is taken as a threshold for a negligible influence of the correlation between the features.

5.3 K-Nearest Neighbors Algorithm

The next algorithm that is trained on the binary classification dataset is the K-nearest neighbors algorithm. The K-nearest neighbors algorithm has the hyperparameter K - the number of nearest neighbors that is considered for each prediction. Thus, for each moving window length m, additionally the hyperparameter K is optimized via 10-fold CV. The optimal hyperparameter K_{max} that maximizes the 10-fold CV for each moving window length m is reported in Fig. 11. Fig. 12 shows the accuracies of the best predictor using K_{max} for each moving window length m. The highest 10-fold CV accuracy is obtained for $m = 91$ using $K_{max} = 29$. Whenever, the highest 10-fold CV accuracy is obtained with $K_{max} = 1$, the training accuracy is 100 %.

For $m = 91$, Fig. 13 shows the search for the optimal hyperparameter explicitly. As expected, the training accuracy is always larger than the 10-fold CV accuracy, and drops significantly for an increasing number K of nearest neighbors. The 10-fold CV accuracy increases slightly with K which indicates a higher prediction performance. The optimal value of K is obtained when the 10-fold CV accuracy is maximized. Here, this is the case for $K_{max} = 29$.

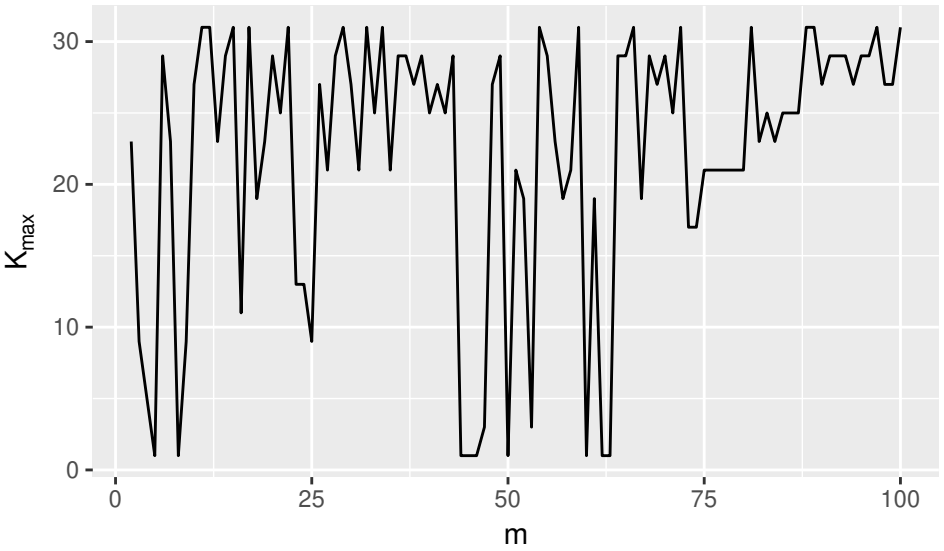


Figure 11: Number of nearest neighbors K_{max} for which the 10-fold CV accuracy using a prediction threshold of 50 % for each moving window length m is maximized.

Similar to the analysis of the logistic regression fit in the last Subsection, Fig. 14 shows the ROC curve of the K-nearest neighbors algorithm for $m = 91$ and $K_{max} = 29$ for different prediction thresholds in comparison to a random classifier. The ROC curve of the K-nearest neighbors algorithm lies only slightly above the ROC curve of a random classifier. In addition, the K-nearest neighbor algorithms leads to an AUC of only 0.5038. Thus, the K-nearest neigbor algorithm performs only slightly better than a random classifier.
The confusion matrix of the K-nearest neighbors algorithm is reported in Tab. 7. The prediction accuracy, sensitivity, and specficitiy of the K-nearest neighbors algorithm is summarized in Tab. 8. Also the test accuracy of 51.80 % is only slightly higher than 50 %.

predicted \ true	0	1
0	115	133
1	194	253

Table 7: Confusion matrix of the K-nearest neighbors algorithm on the test set for a moving window length of $m = 91$, $K_{max} = 29$, and a prediction threshold of 50 %.

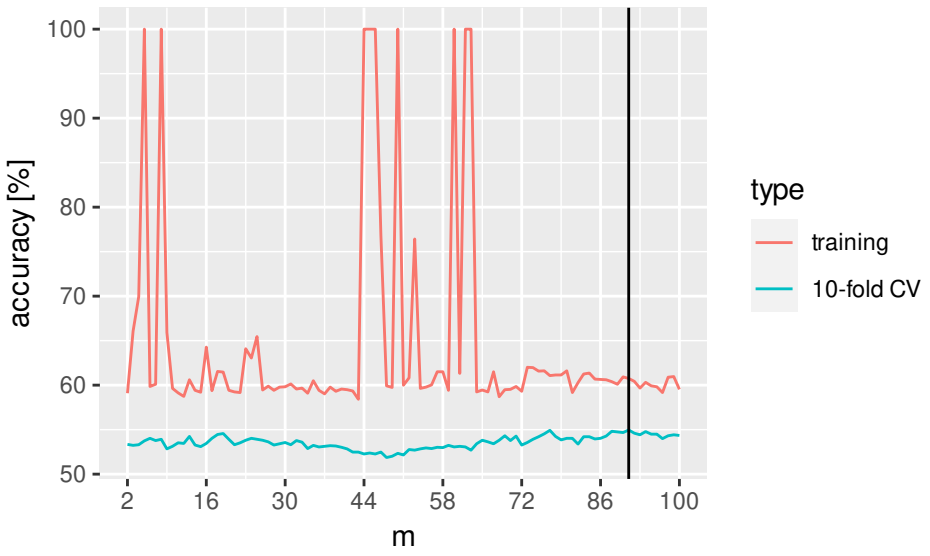


Figure 12: Training and 10-fold CV accuracy for the K-nearest neighbors algorithm as a function of the moving window length m for a prediction threshold of 50 %. For each m, the accuracies for the optimal number of nearest neighbors K_{max} from Fig. 11 is reported. The vertical black line indicates the value of $m = 91$ for which the 10-fold CV accuracy of the K-nearest neighbors regression using $K_{max} = 29$ is maximized.

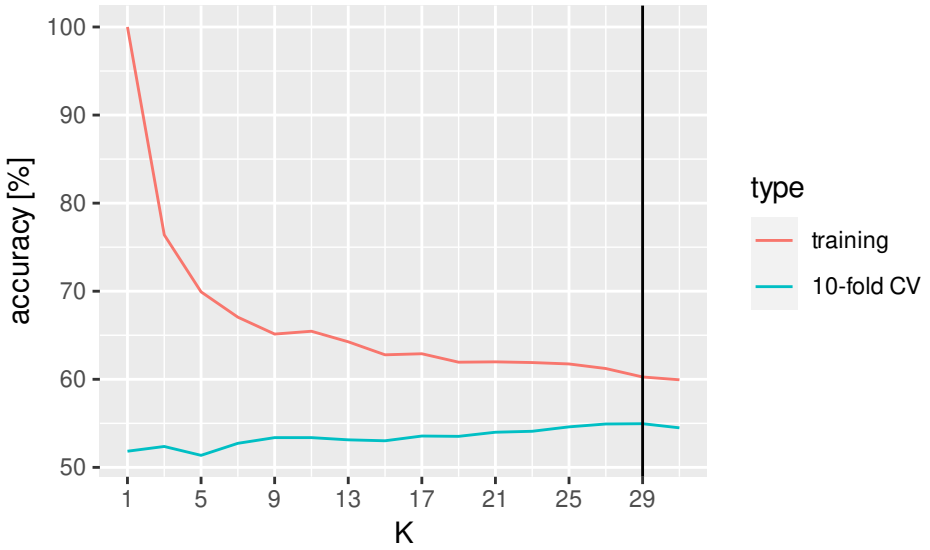


Figure 13: Training and 10-fold CV accuracy for the K-neareast neighbors algorithm as a function of K for the optimal moving window size $m = 91$ using a prediction threshold of 50 %. The vertical black line indicates the value of $K_{max} = 29$ for which the 10-fold CV accuracy of the K-nearest neighbors algorithm is maximized.

accuracy [%]	sensitivity [%]	specificity [%]
51.80	65.54	37.21

Table 8: Accuracy, sensitivity, and specificity of the K-nearest neighbors algorithm on the test set for a moving window length of $m = 4$, $K_{max} = 29$, and a prediction threshold of 50 %.

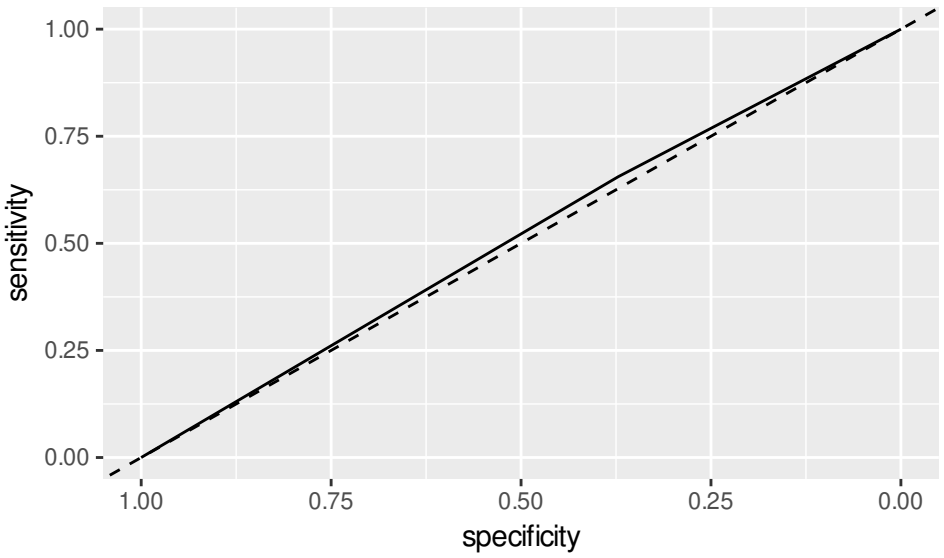


Figure 14: ROC curve of the K-nearest neighbors algorithm with a moving window length of $m = 91$, and $K_{max} = 29$. The dashed diagonal line is the ROC curve of a random classifier that has an AUC of 0.5. For the K-nearest neighbors algorithm, the AUC is 0.5038.

5.4 Deep Neural Network

The last machine learning algorithm applied on the binary classification dataset is a deep neural network. The results in this Subsection refer to a deep neural network with four hidden layers with the following number of neurons: 20-30-20-10. This amounts to a total of 5962 trainable parameters. However, several other architectures - varying in the number of hidden layers and the number of neurons - lead to the same results.

The training of the deep neural network depends on the number of epochs N_{epoch} over the whole training set. Fig. 15 reports the optimal number of epochs $N_{epoch,max}$ for each moving window length m. For the evaluation of the deep neural network, the simple *training-, validation-, and test-set* approach is used for computational reasons. Almost for all values of m the maximum validation accuracy is reached for a small number of epochs $N_{epoch,max}$ of the order of 1 and 2. This indicates that the training of the deep neural network does not improve its performance.

This impression is verified when looking examplarily at the training and validation accuracies of the deep neural network for a moving window length of $m = 57$ as a function of the number of epochs N_{epoch}, see Fig. 16. It is observed that both the training and validation accuracies saturate after $N_{epoch} = 3$ epochs. The validation accuracy is actually the fraction of the examples in the validation set that are labeled with "1". The training accuracy is smaller than the validation accuracy because dropout with a rate of 0.5 between all layers of the deep neural network is applied.

Fig. 17 reports the training and validation accuracy for the deep neural network as a function of the moving window length m for the optimal epoch numbers in Fig. 15. The Figure shows that the observed phenomenon takes place for all values of m.

Tab. 9 and Tab. 10 report the confusion matrix as well as the prediction accuracy, the sensitivity, and the specificity of the deep neural network on the test set. The obtained results show that the deep neural network reproduces the majority predictor, and does not lead to an improvement over the logistic regression fit, and the K-nearest neighbors algorithm.

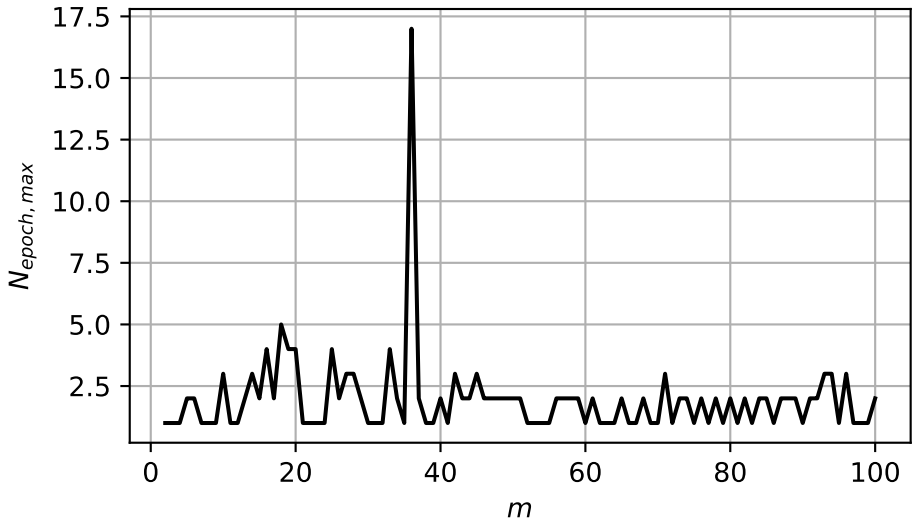


Figure 15: Number of epochs $N_{epoch,max}$ for which the validation accuracy using a prediction threshold of 50 % for each moving window length m is maximized.

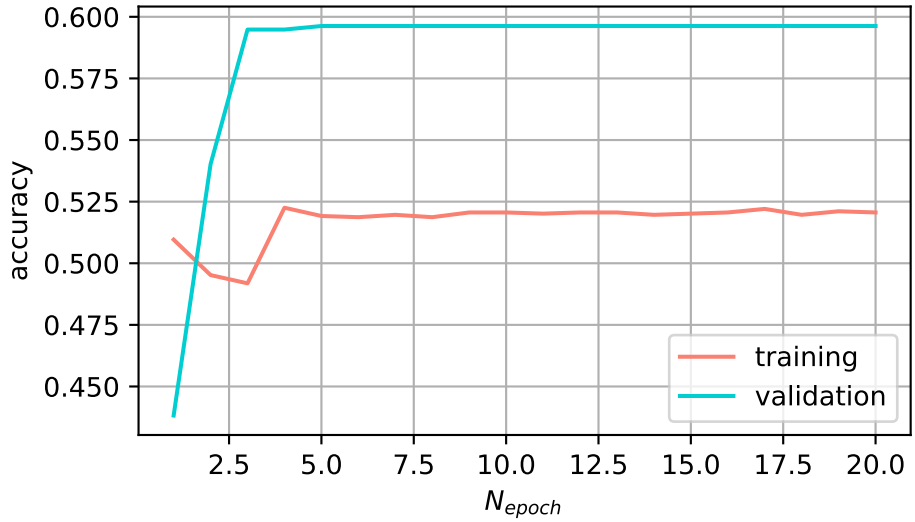


Figure 16: Training and validation accuracy of the deep neural network as a function of N_{epoch} for a moving window length of $m = 57$ using a prediction threshold of 50 %.

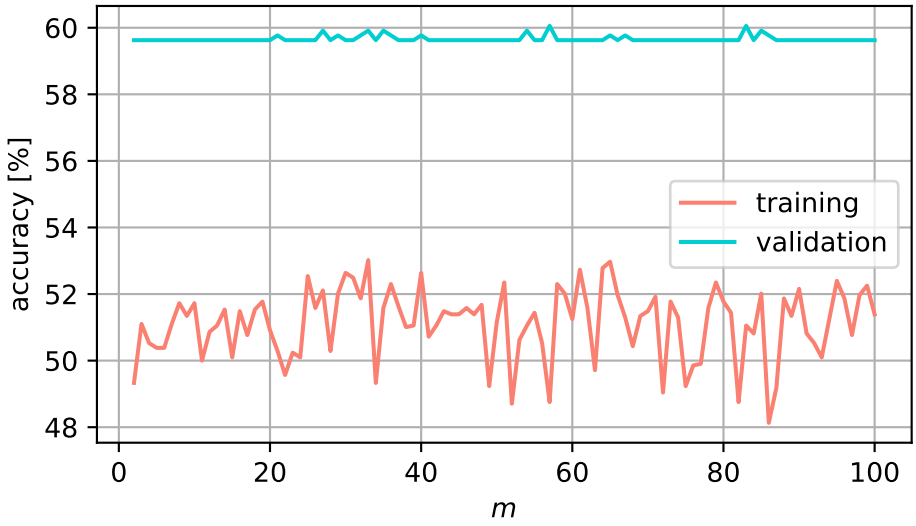


Figure 17: Training and validation accuracy of the deep neural network as a function of the moving window length m for a prediction threshold of 50 %. The accuracies for the optimal epoch numbers in Fig. 15 are reported.

predicted \ true	0	1
0	0	0
1	309	386

Table 9: Confusion matrix of the deep neural network on the test set for a moving window length of $m = 57$, $N_{epoch,max} = 3$, and a prediction threshold of 50 %. The deep neural network reproduces the confusion matrix of the majority predictor.

accuracy [%]	sensitivity [%]	specificity [%]
55.54	100.00	0.00

Table 10: Accuracy, sensitivity, and specificity of the deep neural network on the test set for a moving window length of $m = 53$, $N_{epoch,max} = 3$, and a prediction threshold of 50 %. The deep neural network reproduces the values of the majority predictor.

6 Discussion

Tab. 11 summarizes the results from the previous Section. The Table states the prediction accuracy, the sensitivity, and the specificity obtained after optimizing the hyperparameters and the length m of the moving window for every learning algorithm.

Overall, the logistic regression fit leads to the highest test accuracy of 56.70 %. The majority predictor (and the deep neural network) has the second highest test accuracy of 55.54 %. The lowest test accuracy of 51.80 % is achieved by the K-nearest neighbors algorithm.

Apart from their different test accuracies, the algorithms differ also in their sensitivity and specificity. The majority predictor does not predict decreasing Bitcoin market prices at all. The logistic regression fit and the K-nearest neighbors algorithm have larger specificities at the cost of lower sensitivities.

Given these differences between the machine learning algorithms, the choice of the most suitable algorithm for traders depends on the specific investment strategy. E.g., for some traders it might be more important to predict decreasing Bitcoin market prices. Then, algorithms with a larger specificity like the K-nearest neighbors algorithm might be desirable. The development and evaluation of an investment strategy goes beyond the scope of the present work, and is discussed in other publications [7][8].

	accuracy [%]	sensitivity [%]	specificity [%]
major. pred.	55.54	100.00	0.00
log. reg.	56.70	81.09	24.92
KNN	51.80	65.54	37.21
DNN	55.54	100.00	0.00

Table 11: Comparison of the prediction accuracy, the sensitivity, and the specificity of the majority predictor and the different machine learning algorithms on the test set. The reported values refer to the optimal hyperparameters that are obtained in Section 5.

The comparably poor performance of the deep neural network might be resolved by applying more complex neural networks. In particular, long short-term memory neural networks could be suitable as they implement a hierarchy in the input data, and are reported to perform well for time series forecasting applications.

In general, there are other techniques that are specified for time series forecasting applications that do not require a supervised learning dataset but work with the whole time series, and implement cyclic and seasonal effects. Using these techniques, higher performances could be possible.

Moreover, the prediciton performance of the machine learning algorithms can be improved by considering other features, e.g. Bitcoin-related Google searches or trends deduced from social networks like Twitter [9][10]. The inclusion of these features might also allow the prediction of crashes of the Bitcoin market price which are often the consequence of previous events. E.g., in May 2021 the Bitcoin market price collapsed when Tesla-CEO Elon Musk criticized the high CO_2 emissions of Bitcoin miners, China acted against Bitcoin mining, and an investigation against the world's largest cryptocurrency exchange Binance was opened in the USA [11][12][13].

7 Conclusion

The starting point of the present work is the development of a binary classification dataset that can be applied to train supervised machine learning algorithms for the prediction of tomorrow's Bitcoin market price development. The labels in the binary classification dataset take two possible classes "up" and "down" that specify if the Bitcoin market price of tomorrow is increasing or decreasing. The features are the past development of the Bitcoin market price and related measures like the miner's revenue.

Four different algorithms are trained on the binary classification dataset. A simple majority predictor - that always predicts an increasing Bitcoin market price - leads to a prediction accuracy of 55.54 %, combined with a sensitivity of 100 % and a specificity of 0 %. A variety of deep neural network architectures appear to be naive majority predictors with the same results. The application of neural networks like long short-term memory networks that are more adequate for time series forecasting applications could improve these results. The highest prediction accuracy of 56.70 % is obtained with a logistic regression fit, though a logistic regression does not coincide with the true underlying model of the binary classification dataset. The K-nearest neighbor algorithm leads to the lowest prediction accuracy of 51.80 %. Both the logistic regression fit and the K-nearest neighbors algorithm have non-zero specificities at the cost of lower sensitivities compared to the majority predictor.

An improvement of the prediction performance is possible by including more features like social media data and Google searches, and by the application of algorithms that are specialized for the application on time series forecasting problems.

All in all, the present work shows that the prediction of the future Bitcoin market price development can be treated as a supervised learning task.

References

[1] Satoshi Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System." In: *Cryptography Mailing list at https://metzdowd.com* (Mar. 2009).

[2] Statista. *The 100 largest companies in the world by market capitalization in 2021*. 2021. URL: https://www.statista.com/statistics/263264/top-companies-in-the-world-by-market-capitalization/ (visited on 06/14/2021).

[3] CoinMarketCap. *Today's Cryptocurrency Prices by Market Cap*. 2021. URL: https://coinmarketcap.com/ (visited on 06/14/2021).

[4] Antony Lewis. *The Basics of Bitcoins and Blockchains: An Introduction to Cryptocurrencies and the Technology That Powers Them*. 2018. ISBN: 1633538001.

[5] Blockchain.com. *Blockchain Charts & Statistics API*. 2021. URL: https://www.blockchain.com/api/charts_api (visited on 06/16/2021).

[6] Gareth James et al. *An Introduction to Statistical Learning: With Applications in R*. 2014. ISBN: 1461471370.

[7] Suhwan Ji, Jongmin Kim, and Hyeonseung Im. "A Comparative Study of Bitcoin Price Prediction Using Deep Learning." In: *Mathematics* 7.10 (2019). ISSN: 2227-7390. URL: https://www.mdpi.com/2227-7390/7/10/898.

[8] Takuya Shintate and Lukáš Pichl. "Trend Prediction Classification for High Frequency Bitcoin Time Series with Deep Learning." In: *Journal of Risk and Financial Management* 12.1 (2019). ISSN: 1911-8074. URL: https://www.mdpi.com/1911-8074/12/1/17.

[9] Martina Matta, Maria Ilaria Lunesu, and Michele Marchesi. "Bitcoin Spread Prediction Using Social And Web Search Media." In: June 2015.

[10] Germán Cheuque and Juan Reutter. "Bitcoin Price Prediction Through Opinion Mining." In: May 2019, pp. 755–762. ISBN: 978-1-4503-6675-5. DOI: 10.1145/3308560.3316454.

[11] Elon Musk. *Tweet*. 2021. URL: https://twitter.com/elonmusk/status/1392602041025843203 (visited on 06/15/2021).

[12] CNBC. *Bitcoin price falls after China calls for crackdown on bitcoin mining and trading behavior*. 2021. URL: https://www.cnbc.com/2021/05/21/bitcoin-falls-after-china-calls-for-crackdown-on-bitcoin-mining-and-trading-behavior.html (visited on 06/15/2021).

[13] Bloomberg. *Binance Faces Probe by U.S. Money-Laundering and Tax Sleuths*. 2021. URL: https://www.bloomberg.com/news/articles/2021-05-13/binance-probed-by-u-s-as-money-laundering-tax-sleuths-bore-in (visited on 06/15/2021).