# Short-term bitcoin market prediction via machine learning

Patrick Jaquart*, David Dann, Christof Weinhardt

*Institute of Information Systems and Marketing, Karlsruhe Institute of Technology, Germany*

## Abstract

We analyze the predictability of the bitcoin market across prediction horizons ranging from 1 to 60 min. In doing so, we test various machine learning models and find that, while all models outperform a random classifier, recurrent neural networks and gradient boosting classifiers are especially well-suited for the examined prediction tasks. We use a comprehensive feature set, including technical, blockchain-based, sentiment-/interest-based, and asset-based features. Our results show that technical features remain most relevant for most methods, followed by selected blockchain-based and sentiment-/interest-based features. Additionally, we find that predictability increases for longer prediction horizons. Although a quantile-based long-short trading strategy generates monthly returns of up to 39% before transaction costs, it leads to negative returns after taking transaction costs into account due to the particularly short holding periods.

## 1. Introduction

Bitcoin is a digital currency, introduced in 2008 by Nakamoto.[1] It is enabled by the blockchain technology and allows for peer-to-peer transactions secured by cryptography.[2] In this study, we analyze the short-term predictability of the bitcoin market. Therefore, we utilize a variety of machine learning methods and consider a comprehensive set of potential market-predictive features.

Empirical asset pricing is a major branch of financial research. Machine learning methods have been applied increasingly within this domain, due to the ability to flexibly select amongst a potentially large number of features and to learn complex, high-dimensional relationships between features and target.[3] Although a considerable body of research has examined the pricing of equities and bonds, yielding in a substantial number of potentially market-predictive factors,[4] less attention has been paid to the novel stream of cryptocurrency pricing. In particular, the short-term predictability of the bitcoin market has not yet been analyzed comprehensively. Furthermore, most studies have solely considered technical features and have not analyzed the feature importance of the employed machine

---

* Corresponding author.
    *E-mail addresses:* patrick.jaquart@kit.edu (P. Jaquart), david.dann@kit.edu (D. Dann), christof.weinhardt@kit.edu (C. Weinhardt).
    Peer review under responsibility of KeAi.

learning models.[5] Against this backdrop, we tackle this research gap by comparatively analyzing different machine learning models for predicting market movements of the most relevant cryptocurrency—bitcoin. With a market capitalization of around 170 billion US dollar (September 2020), bitcoin represents about 58% of the cryptocurrency market.[6] In this context, our overarching research question is:

**RQ**: Can machine learning models predict short-term movements of the bitcoin market?

We answer this research question by comparing six well-established machine learning models trained on 9 months of minutely bitcoin-related data against each other. The results show that the trained models indeed significantly outperform random classification. Our study provides two main contributions. First, we contribute to the literature by systematically comparing the predictive capability of different prediction models (e.g., recurrent neural networks, gradient boosting classifiers), feature sets (e.g., technical, blockchain-based), and prediction horizons (1−60 min). Thereby, our study establishes a thorough benchmark for the predictive accuracy of short-term bitcoin market prediction models. The general picture emerging from the analysis is that recurrent neural networks and gradient boosting classifiers appear well-suited for this prediction problem and technical features remain most-relevant. Also, an interesting side finding is that, for longer prediction horizons, prediction accuracy tends to increase and less recent features appear to be of particular importance. Second, despite the models' ability to create viable bitcoin market predictions, our results do not violate the efficient market hypothesis, as the employed quantile-based long-short strategy yields returns that are not able to compensate for associated transaction costs.

## 2. Related work

Financial market prediction is a prominent branch of financial research and has been studied extensively.[5] There is mixed evidence regarding the predictability and efficiency of financial markets.[7,8] An established approach to analyze return-predictive signals is to conduct regression analysis on possible signals to explain asset returns.[9,10] However, linear regressions are not able to flexibly incorporate a large number of features and impose strong assumptions on the functional form of how signals indicate market movements. In contrast, machine learning methods, which often do not impose those restrictions, have been increasingly applied for financial market prediction.[11,3] Among those, neural network-based methods may be expected to be particularly well-suited, as they are already described to be the predominant method for predicting the dynamics of financial markets.[12]

### 2.1. Market efficiency and financial market prediction

#### 2.1.1. Theory on market efficiency

Within efficient financial markets, prices reflect all available information and are not predictable in order to earn abnormal returns. To determine the degree of efficiency of a market, Fama[7] defines a formal three-level framework—weak, semi-strong, and strong form market efficiency. In weak form efficient markets, prices reflect all information about past prices, whereby in semi-strong form efficient markets, prices reflect all publicly available information. In strong form efficient markets, prices additionally reflect all private information. While regulators aim to prevent investors from profiting from private information, it is generally agreed upon that major financial markets are semi-strong form efficient.[13] On the other hand, Grossman and Stiglitz[14] argue that market efficiency may not constitute a constant state of equilibrium over time. If information is costly and prices consistently reflect all available information, informed traders will stop to acquire information, which leads market prices to deviate from fundamental asset values. Besides, there is evidence in the scientific financial literature for a large number of potential market anomalies. Green et al,[15] for instance, identify more than 330 different empirically-found return-predictive signals for the US stock market that have been published between 1970 and 2010. Similarly, Lo[8] formulates the adaptive markets hypothesis, according to which markets may be temporarily inefficient. Thereby, the duration of the temporal inefficiency is influenced by the degree of competition within a market and limits to arbitrage.[16] Informed traders exploit these inefficiencies so that prices reflect all available information again. Summarizing, the question remains open, whether return-predictive signals constitute market anomalies or represent reasonably-priced risk factors. Also, some of the most prominent signals have disappeared after publication,[17] which indicates that part of the published return-predictive signals either have only existed in the sample period or have been erased due to traders adopting strategies

for exploitation. Green et al[15] infer that a unified model of market efficiency or inefficiency should account for persistent empirically identified return-predictive signals.

### 2.1.2. Bitcoin market efficiency

Several findings in the financial literature[18−21] indicate that bitcoin may constitute a new asset class. Therefore, findings regarding the weak form efficiency of other financial markets may not hold for the bitcoin market. Several researchers examine the degree of market efficiency of the bitcoin market using different time horizons. First, Urquhart[22] investigates the time series of daily bitcoin prices (August 2010 to July 2016). He finds that the bitcoin market is not even weak form efficient. However, splitting the study period reveals that the bitcoin market becomes increasingly efficient over time. Revisiting this data, Nadarajah and Chu[23] find that a power transformation of the used bitcoin returns satisfies the weak form efficient market hypothesis. Similarly, Bariviera[24] examines daily bitcoin prices (August 2011 to February 2017), using the Hurst exponent[25] and shows that the bitcoin market is not weak form efficient before 2014, but becomes weak form efficient after 2014. Vidal-Tomás and Ibañez approach the question of semi-strong form bitcoin market efficiency from an event study perspective.[26] With data on news related to monetary policy changes and bitcoin (September 2011 to December 2017), they show that the bitcoin market does not react to monetary policy changes but becomes increasingly efficient concerning bitcoin-related events. Testing for the adaptive markets hypothesis, Khuntia and Pattanayak[27] analyze daily bitcoin prices (July 2010 to December 2017), finding evidence for an evolving degree of weak form market efficiency. They conclude that this finding constitutes evidence that the adaptive market hypothesis holds for the bitcoin market.

Summarizing, there is mixed evidence among scholars regarding the efficiency of the bitcoin market. However, most researchers find that the bitcoin market has become more efficient over the years. An increasing degree of market efficiency seems intuitive, as the bitcoin market has proliferated since its inception and, therefore, has become increasingly competitive.

### 2.2. Bitcoin market prediction via machine learning

Conducting a literature review, Jaquart et al[5] analyze the literature on bitcoin market prediction via machine learning published until April 2019. They examine the body of literature with regards to applied machine learning methods, return-predictive features, prediction horizons, and prediction types. The reviewed body of literature utilizes both classification and regression models approximately equally often, while regression models are used slightly more frequently. Due to the use of different time horizons, targets and feature variables, parameter specifications, and evaluation metrics, the comparison of prediction models across different papers often remains infeasible. On the other hand, comparisons within the same paper often avoid these shortcomings, and remain especially relevant. Based on the latter, Jaquart et al[5] outline that especially recurrent neural networks yield promising results regarding bitcoin market predictions (e.g., see Karakoyun and Cibikdiken,[28] McNally et al[29]). Furthermore, they group the utilized return-predictive features into four major categories—technical, blockchain-based, sentiment-/interest-based, and asset-based features. Technical features describe features that are related to historical bitcoin market data (e.g., bitcoin returns). Blockchain-based features denote features related to the bitcoin blockchain (e.g., number of bitcoin trans-actions). Sentiment-/interest-based features describe features that are related to sentiment and internet search volume of bitcoin (e.g., bitcoin Twitter sentiment). Asset-based features are features that are related to financial markets other than the bitcoin market (e.g., gold returns, returns of the MSCI World index). More recently, Huang et al[30] utilize high-dimensional technical indicators to predict daily bitcoin returns via tree-based prediction models (January 2012 to December 2017) and find that technical analysis can be useful in markets of assets with hard-to-value fundamentals. Chen et al[31] utilize various machine learning techniques to predict the direction of bitcoin price movements. Using data between February 2017 to February 2019, they find that rather simple methods (e.g, logistic regressions) outperform more complex algorithms (e.g., recurrent neural networks). However, the use of a class split based on the directional price movement is likely to create an imbalanced training set, which may cause biased results.[32] Especially for financial time series, which are usually rather noisy,[3] an imbalanced training set may cause classifiers to generally predict the majority class—regardless of input features. If the utilized test set has a similar target class imbalance, biased classifiers may achieve a high prediction accuracy without taking input feature values into account.[33] Using blockchain-based features and data from April 2013 to December 2019, Mudassir et al[34] compare feedforward neural networks, support vector machines and long-short term memory networks to predict bitcoin price movements with

prediction horizons ranging between 1 and 90 days. They find that the support vector machine model has the highest accuracy for the shorter prediction horizons, while the long-short term memory network performs best on the long term horizons. However, they also define target classes based on bi-directional price movements—potentially creating an imbalanced training set. Dutta et al[35] compare recurrent neural networks and feedforward networks to predict daily bitcoin prices, using daily data from January 2010 to June 2019. They perform feature selection based on the variance inflation factor and find that recurrent neural networks tend to outperform feedforward networks on this task. However, in the chosen setting, the utilized feedforward networks do not receive similar temporal information as the recurrent neural networks and therefore results are expected to be biased towards a higher performance of the recurrent neural networks. In a first analysis, Jaquart et al[36] analyze the short-term predictability of the bitcoin market. Their results emphasize the potential of recurrent neural networks for predicting the short-term bitcoin market. However, questions regarding the robustness of the results, theoretical foundations and descriptions of the research approach remain unanswered.

So far, only few researchers (e.g., Dutta et al,[35] Poyser[37]) utilize features of *all* established feature categories. Besides, the particular feature importance across different models has received little academic attention so far. The vast majority of researchers construct their models using daily prediction horizons and only few scholars[38,39] benchmark different prediction horizons against each other.[5] Consequently, the bitcoin market dynamics concerning prediction horizons of less than 1 h are not fully understood yet.

## 3. Methodology

To tackle the previously-outlined research gap, we systematically evaluate different prediction models, features, and horizons. Thereby, we implement data gathering, preprocessing, and model building using the Python programming language and the libraries TensorFlow, scikit-learn, and XGBoost.

### 3.1. Data

We use data from Bloomberg, Twitter and Blockchain.com ranging from March 4, 2019 to December 10, 2019. Regarding Bloomberg, our data set includes minutely price data for bitcoin, gold, oil and minutely levels for the total return variants of the indices MSCI World, S&P 500, and VIX. All prices and index levels are denoted in US dollar (USD). Furthermore, the data set includes minutely exchange rates relative to the US Dollar for the currencies euro (EUR/USD), Chinese yuan (CNY/USD), and Japanese yen (JPY/USD). Fig. 1 shows the bitcoin price development for the examined time period. From Blockchain.com, the data set includes minutely data for the number of bitcoin transactions and growth of the mempool (i.e., storage of not-yet validated bitcoin transactions). Last, the data set includes sentiment data of all English Twitter Tweets in the given period that include the hashtag bitcoin ("#bitcoin").



Fig. 1. **Bitcoin price overview.** Bitcoin price development between March 2019 and December 2019.

## 3.2. Software and hardware

Python 3.7 is used for all data processing and analysis, utilizing the packages pandas[40] and numpy.[41] We rely on the NLTK library[42] for part of the Tweet preprocessing and use the Google Natural Language API[43] for sentiment analysis. We use the keras library[44] on top of the tensorflow backend[45] to build our feedforward, LSTM, and GRU networks. We build gradient boosting classifiers with xgboost[46] and random forest as well as logistic regression models using scikit-learn.[47] All neural networks and the gradient boosting classifier utilize the GPU-based Nvidia CUDA parallel computing platform (GeForce GTX 1080), while the random forest and logistic regression models are trained on CPU (Intel Core i7-7700, 3.60 GHz).

## 3.3. Features

We employ features from all four major feature categories identified by Jaquart et al,[5] as listed in Table 1. For all prediction models, we calculate minutely-updated feature values. Depending on whether the prediction model has a memory state, we further aggregate these feature values.

For the *technical* and *asset-based* features, returns are given by

$$r_{t,t-k}^a = \frac{p_t^a}{p_{t-k}^a} - 1, \tag{1}$$

where $p_t^a$ is defined as the price of asset $a$ at time $t$ and $k$ represents the number of periods over which the return is calculated. We obtain minutely-updated values for the selected *blockchain-based* features. *Sentiment-/interest-based* features are generated from the collected Tweets. We only keep Tweets that do not contain pictures or URLs, since the use of textual sentiment analysis is not able to capture all information contained in multimodal Tweets.[48] Following the suggestions of Symeonidis et al,[49] we apply various preprocessing techniques to the collected Tweet texts. First, we remove usernames, non-English characters, and additional whitespace. Second, we replace contractions (e.g., replace "isn't" with "is not"). Last, we apply lemmatization to the Tweet texts to replace inflected word forms with respective word lemmas (e.g., replace "bought" with "buy"). Next, we make use of the Google Natural Language API[43] to generate sentiment and estimates of strength of emotion for each Tweet. For every minutely instance, we calculate three different features: First, the number of bitcoin Tweets published in last minute as a proxy for the overall interest in bitcoin. Second, the sum of sentiment scores of all Tweets published in the previous minute. Third, sum of sentiment scores of all Tweets published in the previous minute, weighted by the strength of emotion per Tweet. Features 2 and 3 depend on the sentiment expressed towards bitcoin but differ in the applied weighting scheme. While

Table 1

**Feature overview.** Overview of the utilized features.

| Technical | |
|---|---|
| Bitcoin returns | |

| Asset-based | |
|---|---|
| MSCI World returns | Crude Oil WTI returns |
| SP 500 returns | EUR/USD returns |
| VIX returns | CNY/USD returns |
| Gold returns | JPY/USD returns |

| Blockchain-based | |
|---|---|
| Number of Bitcoin Transactions | Mempool growth |

| Sentiment-/interest-based | |
|---|---|
| Twitter sentiment | Number of tweets |
| Twitter sentiment weighted with strength of emotion | |

traditional prediction methods fail more often when predictors are highly correlated, machine learning models appear well-suited for these cases through the use of various variable selection methods.[3]

### 3.3.1. Feature set for models with memory function

For the machine learning models with a memory function (i.e., LSTM and GRU), we create time series for all features listed in Table 1. To facilitate model training, all feature values are standardized based on the properties of the specific feature in the training set.[50]

Following feature standardization, we create time series from the 120 most recent, minutely feature values. For the employed technical and asset-based features, the time series consist of the latest one-minute returns. However, for bitcoin, we create an additional time series by also calculating the one-week bitcoin returns for each of the most recent 120 min according to Equation (1) to give the models information about the longer-term status of the bitcoin market. Conclusively, the input for the memory models consists of 15 different time series, whereby each of the time series consists of 120 minutely time steps.

### 3.3.2. Feature set for models without memory function

The prediction models without memory function (i.e., feedforward networks, random forests, gradient boosting classifiers, and logistic regressions), require input in form of a one-dimensional vector with one observation per feature. Therefore, we create additional features by aggregating the 120-min history of the feature classes to also give the employed no-memory models temporal information about the feature values. In line with Takeuchi and Lee[51] and Krauss et al,[52] we choose a more granular resolution for the most recent feature history. Specifically, we choose the following set of intervals, $j$, to aggregate the feature history: $j \in \{(0, 1], (1, 2], (2, 3], (3, 4], (4, 5], (5, 10], (10, 20], (20, 40], (40, 60], (60, 80], (80, 100], (100, 120]\}$, whereby these intervals describe the minutes before a prediction is made. For the aggregation process, blockchain-based features, as well as sentiment-/interest-based features are summed up across the respective intervals. For the aggregated technical and asset-based features, we calculate multi-period returns over the respective intervals (Equation (1)). We build these intervals for all features used for the feature sequences of the memory models, except for the one-week bitcoin return, since this time series naturally exhibits low variation over 120 consecutive minutes. As for the memory models, we standardize all feature based on the feature properties within the training set. Consequently, our feature set for the prediction models without memory function consists of $14 \times 12 + 1 = 169$ different features.

### 3.4. Targets

We formulate a binary classification problem for four different prediction horizons. For every observation, the target class $c_m$ is formed based on the return over the next $m$ minutes, with $m \in \{1, 5, 15, 60\}$.

We place observations, for which the return over the next $m$ minutes is greater than or equal to the median $m$-minute return of all training set observations, in Class 1 and all other observations in Class 0. With regard to Equation (1), the target class at time $t$, $y_t^m$, is given by

$$y_t^m = \begin{cases} 1, & \text{if } r_{t+m,t}^{bitcoin} \geq Median(r_{u+m,u}^{bitcoin}) \, \forall \, u \in \{train\} \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

where *train* denotes all time timestamps in the training set.

Creating classes directly from the training set ensures that the prediction models are trained on equally balanced proportions and are not subject to a bias towards one specific class. During prediction, a model returns the probability for an observation to belong to a specific class. The training set median return is 7.5984E-6 for the 1-min prediction horizon, 2.8491E-5 for the 5 min prediction horizon, 7.7942E-5 for the 15 min prediction horizon and 2.7168E-4 for the 60 min prediction horizon.

### 3.5. Generation of training, validation and test sets

We convert all timestamps to Coordinated Universal Time (UTC) and create a data set for each prediction problem by aggregating features and target variable. Most bitcoin trading venues allow for continuous trading of bitcoin but for

Table 2

**Parameter tuning space.** Overview of parameter tuning grid for all models. * denotes selected parameters based on validation set accuracy.

| Model | Parameter Tuning Grid |
|---|---|
| GRU | Number of memory blocks: {64, 128, 256*, 512} |
| LSTM | Number of memory blocks: {64, 128, 256*, 512} |
| FNN | Hidden layer structure: {(512), (512−256), (512-256-128), (512-256-128-64), (512-256-128-64-32)*, (512-256-128-64-32-16)} |
| LR | — |
| GBC | Maximum tree depth: {1*, 2, 6, 10, 15, 20, None} |
| RF | Minimum fraction of instances per leaf: {1%, 5%, 10%, 20%*, 30%} |

the minutely Bloomberg bitcoin price series, there is a gap in the time series on weekends, which we exclude from our analysis. Since the utilized asset-based features are related to assets that are mainly traded on weekdays, we consider this procedure to be reasonable. Due to the 7-day bitcoin return feature, we require 7 days of history for every observation to calculate the complete feature set. Therefore, our final data sample spans a time range of 9 months, namely from March 11, 2019 to December 10, 2019. We use the first five-ninths of the data (approximately 5 months) to generate a training set. The subsequent one-ninth of the data (approximately 1 month) forms a validation set for hyperparameter tuning, including regularization techniques, such as early stopping.[53,54] The remaining one-third of data (approximately 3 months) is used to test our models and obtain a representative out-of-sample prediction accuracy.

### 3.6. Prediction models

With our set of models for bitcoin market prediction, we benchmark neural networks with and without memory components, tree-based models, regression models, and ensemble models against each other. Apart from the ensemble models, all these models have already been applied to the domain of bitcoin market predictions.[5] Besides the in the following described models, we make use of a logistic regression model (LR) as a benchmark. Table 2 gives an overview of the evaluated parameter grid for model tuning. Beyond the approach of Jaquart et al,[36] we train the stochastic prediction models (i.e., neural networks and random forests) on 10 different random seeds to reduce the impact of randomness on the results. We create the final class probabilities for these models by averaging the predicted class probabilities per seed over all utilized random seeds.
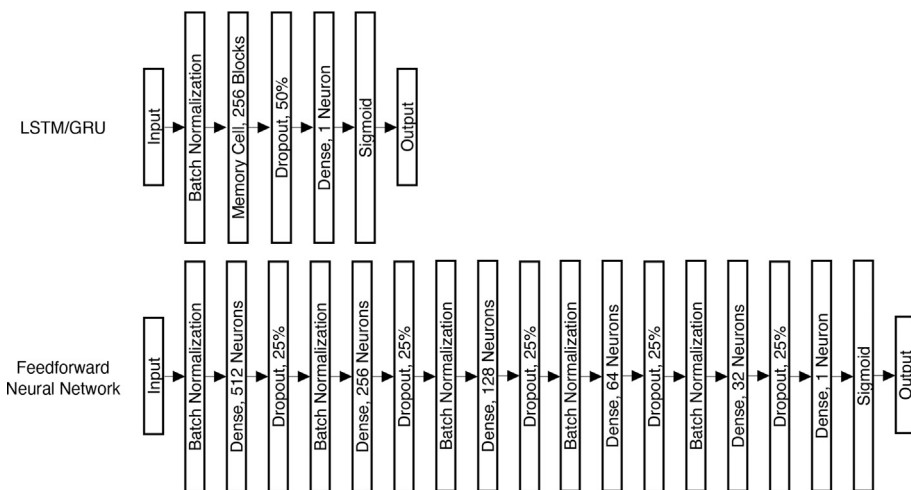


Fig. 2. **Neural network architecture.** Architecture of applied feedforward neural networks (bottom) and recurrent neural networks (top). Note: For the recurrent networks, the memory cell is either an LSTM cell or a GRU cell.
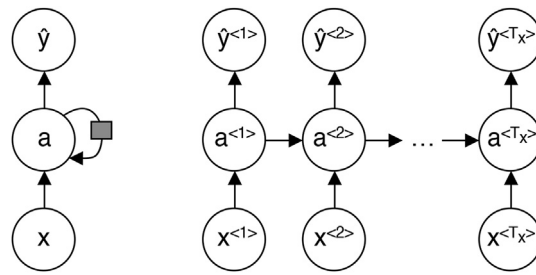
Fig. 3. **RNN structure.** RNN in a folded (left) and unfolded (right) state. Grey square indicates time step delay.

### 3.6.1. Neural networks

The structure and intended behavior of artificial neural networks is inspired by the functionality of the human brain. In analogy to the structure of a brain, which consists of billions of highly interconnected neurons, artificial neural networks consist of various, highly connected nodes. Every node receives a certain amount of input from other nodes and, dependent on the received input, each note generates output, which is then passed to subsequent nodes. Hence, information is passed through the network of nodes and transformed by every node on its path. Formally, the network learns to approximate a function $f(x)$, which maps a given input $x$ to a given output (category) $y$. All networks employed are trained with a batch size of 5000 and the established Adam optimizer[55] to minimize the binary cross-entropy loss. To further improve the level of generalization, the networks contain individual dropout layers[56] and use the early-stopping method with a patience value of $10^{53,54}$.

#### 3.6.1.1. Feedforward neural network.
Feedforward neural networks (FNN) are a basic type of neural networks. FNNs represent a directed acyclic graph in which processed information flows exclusively in one direction, information is so to say 'fed forward'. FNNs consist of three types of layers: One input layer, capturing the input information, a variable number of hidden layers, and one output layer, determining the network's final classification. The final classification is dependent on the activation of nodes in preceding layers. The activation of each node in all layers is determined by a previously-assigned activation function. This (commonly non-linear) activation function controls the output of each node. Formally, the activation state of layer $n$ is given by

$$a^{(n)} = g^{(n)}\left(W^{(n)^\top} a^{(n-1)} + b^{(n)}\right), \tag{3}$$

where $g^{(n)}$ is the activation function, $W^{(n)}$ is the weight matrix for the connections between layer $n-1$ and layer $n$, and $b^{(n)}$ is the bias for layer $n$.

Summarizing, information is passed from the input layer, processed and transformed in the different hidden layers, and finally classified in the output layer. It can be shown, that FNNs including at least one hidden layer and a non-linear differentiable activation function can approximate any non-linear function, rendering them a universal approximator.[57] However, this does not imply that a FNN generalizes in such a way that it correctly classifies previously unseen data. Choosing an appropriate network architecture and hyperparameterization constitutes an essential step towards creating a generalized network for a given task. Fig. 2 (bottom) describes the architecture of the applied FNNs.

#### 3.6.1.2. LSTM and GRU.
Long short-term memory (LSTM) and gated recurrent unit (GRU) networks belong to the category of gated recurrent neural networks (RNNs). RNNs drop FNN's condition of acyclic graphs. Relaxing this boundary allows for the existence of arbitrary feedback connections and an overall cyclic structure of the network. Hammer[58] shows that RNNs with a sufficient number of hidden nodes and non-linear activation function also satisfy the requirements of a universal approximator. Fig. 3 depicts a basic recurrent neural network in both a folded and unfolded representation. Hochreiter and Schmidhuber introduce the LSTM architecture in the late 1990's[59] with a specific focus on long-term memorization of information in sequential data. LSTMs have been used for a variety of tasks in different domains. Among these are neural language processing and speech recognition[60−63], handwriting recognition and handwriting generation[64−67], music generation,[68] analysis of financial data,[69] as well as more generic

scenarios such as tasks that require to remember specific numbers along long sequences.[70,59] Their architecture replaces the nodes in the hidden layers with memory blocks. Each block usually consists of one memory cell and varying number of gates, which can manipulate the internal values of the cell. All blocks are connected via the cell states $c^{<t>}$, and the output of the memory blocks $a^{<t>}$. The gates enable the network to maintain the influence of inputs along longer time sequences. The original LSTM has two different gates: an input and an output gate. Each gate utilizes a sigmoid activation function. Gers[71] extend the original LSTM with an additional forget gate, which allows the cell to reset itself. Formally, the output of the LSTM is given by:

$$\widehat{y}^{<t>} = g^{<t>}(a^{<t>}), \tag{4}$$

whereby $g^{<t>}$ is the network's activation function at $t$ and

$$a^{<t>} = o^{<t>} \circ \tanh(c^{<t>}) \tag{5}$$

$$c^{<t>} = i^{<t>} \circ \widetilde{c}^{<t>} + f^{<t>} \circ c^{<t-1>} \tag{6}$$

$$\widetilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c) \tag{7}$$

$$i^{<t>} = \sigma(W_i[a^{<t-1>}, x^{<t>}] + b_i) \tag{8}$$

$$f^{<t>} = \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f) \tag{9}$$

$$o^{<t>} = \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o), \tag{10}$$

where $i^{<t>}, f^{<t>}, o^{<t>}$ denote the state of the input, forget, and output gate with their respective weight matrix $W$ and bias $b$, $\widetilde{c}^{<t>}$ is the candidate for updating the current cell state $c^{<t>}$. Fig. 4 depicts the composition of an LSTM memory block.

GRUs differ from the LSTMs insofar, as they use one unified gate unit to control the forget and the update gate simultaneously. Although the number of learnable parameters of GRUs is thereby smaller than that of LSTMs, their performance in various domains is comparable.[72] Fig. 2 (top) outlines the architecture of the applied RNNs.
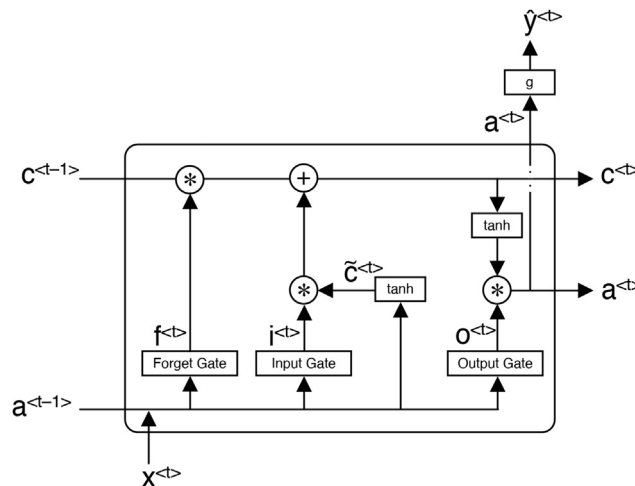


Fig. 4. **LSTM memory block.** LSTM memory block with input, forget, and output gate (based on Olah[73]).

### 3.6.2. Tree-based models

Tree-based models use a decision tree to learn attribute-class relationships, which are fast to train and well interpretable. However, unpruned, single decision trees are prone to overfit to training data.

*3.6.2.1. Random forest.* Introduced by Ho,[74] random forests (RF) aim to overcome tree-based models' tendency to overfit by means of an ensemble method. Here, multiple decision trees are generated, and each of them is trained on different parts of the training data. The output of the final model for overall classification is the average output of all individual trees. The random forest applied is subject to the parameterization of 100 trees and a minimum number of instances per leaf of 20%. For all remaining parameters, we use the default values of the Python scikit-learn library.[47]

*3.6.2.2. Gradient boosting classifier.* Similar to random forests, gradient boosting classifiers (GBC) leverage the input of multiple decision trees.[75] In addition, boosting classifiers also train individual weights for all included models. In this way, the output classification of the better-adapted models is weighted more strongly in the model's final classification decision. In our analysis, we use the extreme gradient boosted (XGBoost) trees, parameterized with a binary logistic objective function to build a gradient boosting classifier, whereby individual trees have a max-depth of 1.

### 3.6.3. Ensemble models

Similar to random forest and gradient boosting, ensemble models rely on the classification output of multiple models. However, in an ensemble, different model-types (e.g., neural networks and tree-based models) can be combined into a meta-model. The output of the meta-model constitutes the averaged predictive probability vector of all models included. In this way, method-specific misclassifications should be "overruled" by the other models in the ensemble. We apply an meta-model consisting of all individual models.

## 3.7. Evaluation

The prediction models are evaluated and analyzed regarding various aspects. First, we compare the models on a prediction level. Second, we analyze and compare feature importance for each model and prediction target. Third, we examine economic implications of our bitcoin market predictions by employing a long-short portfolio strategy.

### 3.7.1. Forecast evaluation

We compare the forecasts of our prediction models based on the predictive accuracy on the test set. Also, for our stochastic prediction models, we examine the impact of using multiple random seeds on model accuracy and stability. Furthermore, we compare the significance of the differences in model-prediction based on Diebold−Mariano tests.[76] Additionally, we estimate the probability that the models make predictions by chance. If the true accuracy of a binary classification model is 50%, the number of correctly classified targets follows the distribution

$$X \sim B(n = \text{test}, p = 0.5, q = 0.5), \tag{11}$$

where #test is the number of observations in the test sample (e.g., 94834 for the 1-min horizon). Based on this binomial distribution, we calculate the probabilities that a prediction model has a true probability of 50%.

### 3.7.2. Feature importance

The feature importance for all models is determined by the measure of permutation feature importance.[77] This ensures comparability between the resulting importance scores across all models. We randomly permute every feature vector with a random standard normally distributed vector and calculate the decrease in prediction accuracy, which we interpret as feature importance. A high decrease in prediction accuracy implies that the model strongly relies on the feature for its predictions. To decrease the impact of randomness on the results, we average the permutation feature importance across a set of 10 different random seeds, $s \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. In the rare case that the random permutation increases the prediction accuracy, we set this feature's importance to zero. For our stochastic prediction models, we calculate the feature importance for every model seed (i.e., based on 10 feature importance seeds per model seed) and average the results over all model seeds. To enhance interpretability, we follow the recommendations

of Gu et al[3] and normalize the feature importances in a way that all feature importance scores for a prediction model sum to one.

### 3.7.3. Trading strategy

We analyze the economic implications of the bitcoin market predictions by testing a straightforward trading strategy. To approximate an ex ante trading strategy, we calculate the 99%-quantiles of all the classes' probabilities from the training set predictions. If, for instance, the predicted probability in the test set for Class 1 is higher than the respective threshold probability, we take a long position. Vice versa, we take a short position, if a predicted class probability for Class 0 is above the respective threshold probability. Table A.1 lists these threshold probabilities per model and prediction horizon. At the end of the prediction horizon, the position is closed. We calculate the return of this strategy before and after transaction costs. Similar to Fischer et al,[78] we assume round-trip transaction costs of 30 basis points (bps).

## 4. Results

### 4.1. Predictive accuracy

We compare the model predictions based on the accuracy scores, which are presented in Table 3. We find that all tested models' predictive accuracy is above the 50% threshold. Furthermore, all models have a probability of less than 3.08E-09 for a true accuracy of 50% (see Table 4). Thereby, the use of multiple random seeds increases stability in the predictions of the stochastic prediction models. With regards to that, Table A.2 lists the predictive accuracies for the individual random seeds. Second, we find that the average prediction accuracy monotonically increases for longer prediction horizons. Third, we find that RNNs or GBCs constitute the best-performing methods across all prediction horizons. Specifically, the LSTM performs best on the 1-min prediction horizon. As shown in Table A.3, Diebold−Mariano tests[76] reveal that LSTM predictions are more accurate than the predictions of all other models, apart from the GRU ($\alpha$: 5%). On the 5-min horizon, the GBC model shows the highest predictive accuracy, yielding significantly more accurate forecasts than all remaining models. The ensemble model is the most accurate method on the 15-min prediction horizon, but does not produce significantly more accurate predictions than the GRU and GBC models. The LSTM model yields the most accurate predictions on the 60-min horizon, which are significantly more accurate than the predictions of all other models. Summarizing, we find that the RNNs and GBC models, on average, provide more accurate predictions for the bitcoin market compared to the other models. Both RNNs show comparable predictive accuracy. On the 1-min and the 5-min prediction horizon, the predictive accuracy of the GRU and the LSTM does not differ significantly, while the GRU yields more accurate predictions on the 15-min horizon and the LSTM yields more accurate forecasts on the 60-min horizon. Equivalently, the predictive accuracy of the tree-based models (i.e., GBC and RF) does only significantly differ on the 5-min prediction horizon, where the GBC yields more accurate forecasts.

Table 3

**Accuracy overview.** Predictive accuracy of the machine learning models for the different prediction horizons.

| Model | Accuracy | | | |
| --- | --- | --- | --- | --- |
| | 1-Min Predictions | 5-Min Predictions | 15-Min Predictions | 60-Min Predictions |
| GRU | 0.518411 | 0.524562 | 0.536490 | 0.556653 |
| LSTM | 0.519286 | 0.524931 | 0.531967 | 0.560067 |
| FNN | 0.509438 | 0.521988 | 0.520820 | 0.529587 |
| LR | 0.511272 | 0.517926 | 0.519595 | 0.538552 |
| GBC | 0.511093 | 0.529268 | 0.537282 | 0.557026 |
| RF | 0.511947 | 0.526662 | 0.534641 | 0.556356 |
| E (All) | 0.514626 | 0.526092 | 0.537863 | 0.557579 |

Table 4
**Probabilities for true model accuracy of 50%.** Probability for the prediction models to have a true accuracy of 50%, derived from the binomial distribution described in Equation (11).

|  | Probability | | | |
|---|---|---|---|---|
| Model | 1-Min Predictions | 5-Min Predictions | 15-Min Predictions | 60-Min Predictions |
| GRU | 4.18E-30 | 5.66E-52 | 6.19E-112 | 8.44E-265 |
| LSTM | 7.65E-33 | 1.75E-53 | 1.99E-86 | 2.28E-297 |
| FNN | 3.08E-09 | 4.64E-42 | 7.17E-38 | 7.02E-74 |
| LR | 1.92E-12 | 1.26E-28 | 8.99E-34 | 6.90E-124 |
| GBC | 4.18E-12 | 6.67E-73 | 9.47E-117 | 2.93E-268 |
| RF | 9.31E-14 | 7.30E-61 | 4.23E-101 | 4.75E-262 |
| E (All) | 1.05E-19 | 2.22E-58 | 2.40E-120 | 1.94E-273 |

## 4.2. Feature importance

The predominant feature for both RNNs is the minutely bitcoin return time series. The relative importance of this feature decreases for longer prediction horizons from about 80% (1-min horizon) to less than 50% (60-min horizon). It becomes apparent that, for longer time horizons, additional time series besides the minutely bitcoin returns are increasingly relevant for both RNNs. Among those are the number of transactions per second, the number of Tweets, the weekly bitcoin returns, and the weighted sentiment scores.

Subsequent analysis of the models without memory function provides further insights into the temporal distribution of feature importance. While the most important feature for the GBC and RF is consistently related to bitcoin returns, for more extended prediction horizons, less recent bitcoin returns become more important. On the 1-min horizon, the most recent minutely return is most relevant, while on the 5-min horizon, the bitcoin returns from the period 10 to 5 min before the prediction point constitute the most important feature. Accordingly, for these models, the bitcoin returns from 20 to 10 min before prediction are most important on the 15-min horizon and the bitcoin returns from 40 to 20 min before prediction constitute the most important feature on the 60-min horizons.

Table 5
**Trading returns per model.** Trading returns (*TR*) and number of trades (*#trades*) for the long-short 1%-quantile strategy before transaction costs over the 3 months of testing data.

| | 1-Min Predictions | | 5-Min Predictions | |
|---|---|---|---|---|
| Model | TR | #trades | TR | #trades |
| GRU | −0.0933 | 557 | −0.0816 | 586 |
| LSTM | 0.0281 | 584 | −0.0869 | 701 |
| FNN | 0.0354 | 507 | 0.1089 | 583 |
| LR | 0.0599 | 664 | 0.3405 | 822 |
| GBC | 0.0637 | 1305 | 0.0097 | 1601 |
| RF | 0.0307 | 798 | 0.1881 | 574 |
| E (All) | 0.0203 | 582 | 0.1224 | 710 |
| | 15-Min Predictions | | 60-Min Predictions | |
| Model | TR | #trades | TR | #trades |
| GRU | 0.0632 | 2164 | 0.2087 | 2806 |
| LSTM | −0.2563 | 1268 | 1.1569 | 2582 |
| FNN | 0.0509 | 924 | 0.4699 | 661 |
| LR | 0.2059 | 881 | −0.8732 | 929 |
| GBC | 0.4749 | 1971 | 0.3155 | 1602 |
| RF | 0.2909 | 385 | 0.5483 | 497 |
| E (All) | 0.2829 | 1421 | 0.4156 | 1244 |

Similar to the finding for the RNNs, for longer prediction horizons, the relative importance of the predominant feature drops for the GBC (60-min horizon: 70%, 1-min horizon: 30%) and the RF (60-min horizon: 45%, 1-min horizon: 30%). Besides technical features, mainly blockchain-based features (e.g., transactions per second, mempool size growth), as well as sentiment-/interest-based features (e.g., number of Tweets) remain important for the tree-based models. Compared to the GBC and RF, for the FNN and LR, feature importance is distributed along several features, which may be explained by the rather shallow parameterization of the tree-based models. We provide graphical representations of all feature importances in the Appendix B.

### 4.3. Trading strategy

Table 5 lists the results of our quantile-based trading strategy before transaction costs. In comparison, a buy and hold strategy yields a return of −0.2958 during the test set period. Since the threshold class probabilities are calculated on predictions on the training set, the number of trades varies between methods and prediction horizons. Table presents the exact threshold class probabilities for the different prediction models and horizons. The results of the trading strategy yield three key insights. First, there is a rather large variance in trading results between the different prediction models. Higher predictive model accuracy does not necessarily translate into better trading results. We explain this by the fact that we do not set up our prediction problem to *optimize* trading performance, but rather to *predict* directional market movements. Additionally, based on our trading strategy, only a rather small proportion of observations is traded, which presumably increases variance. Trading returns based on the ensemble model are generally positive and near the average of the trading returns of the individual models, which indicates that combining predictions of individual prediction models may reduce the variance in trading results. Second, the average return per trade tends to increase with longer prediction horizons. Third, considering transaction costs of 30 bps per round-trip, trading performance becomes negative for all methods. These negative returns may be explained by the models' short-term prediction horizons. Based on the transaction costs, making 1000 trades would cause transaction costs of 300%.

## 5. Discussion

This study demonstrates that machine learning models are able to predict short-term movements of the bitcoin market. Clearly, the forecasting accuracy of slightly over 50% indicates that the bitcoin market predictability is somewhat limited. A limited bitcoin market predictability is comparable to findings related to the market predictability of other financial assets, such as stocks.[11,3] This may be due to multiple reasons, for instance, an immediate market reaction to the utilized features or a potentially large amount of information beyond these features that influence the bitcoin market. Furthermore, our results are consistent with the findings that the bitcoin market has become more efficient over the last years.[22,26,27]

Since trading results based on the market predictions are negative after transaction costs, our work does not represent a challenge to bitcoin market efficiency. However, in this study, we analyze the predictability of the bitcoin market movement and do not train our models to maximize trading performance. Nevertheless, our results indicate that empirical trading strategies should be implemented on the basis of models with more extended prediction horizons. This would correspond to longer holding periods, for which the relative impact of transaction costs is presumably lower. Complementary, our finding that predictive accuracy increases for longer prediction horizons paves the path for further research opportunities.

Next, we find that RNN and GBC models are particularly well-suited for predicting the short-term bitcoin market. Yet, as the field of machine learning is evolving constantly, we may speculate about future specialized machine learning models performing even better on this task. The implemented RNN and GBC models clearly distinguish in the weighting of features, mainly relying on a set of few features. For these prediction models, technical features appear to be most influential, followed by blockchain-based and sentiment-/interest-based features. However, the exact source of predictive power for these features remains ambiguous. Possible sources of explanations may be theoretical equilibrium models. For instance, Biais et al[79] determine a quasi-fundamental value for bitcoin based on, for instance, transactional costs and benefits. Some of the used features (e.g., transactions per second) may partially approximate these factors. Furthermore, Detzel et al[80] present an equilibrium model showing how technical indicators

are able to affect the prices of assets with hard-to-value fundamental values. Subsequent studies may explore whether market anomalies, such as the momentum effect,[81] exist within the bitcoin market and test whether additional technical features, such as bitcoin trading volume, exhibit predictive power. Besides, future research may examine whether behavioral financial market biases (e.g., the disposition effect[82]) are more pronounced for bitcoin, as it does not exhibit a fundamental value in the traditional sense. Since financial markets are dynamic, future research could also evaluate whether the identified Bitcoin market mechanisms remain in place or to what extent the Bitcoin market structure has changed.

## 6. Conclusion

In our empirical analysis, we analyze the short-term predictability of the bitcoin market, leveraging different machine learning models on four different prediction horizons. We find that all tested models make statistically viable predictions. The models are able to predict the binary market movement with accuracies ranging from 50.9% to 56.0% whereby predictive accuracy tends to increase for longer forecast horizons. We identify that especially recurrent neural networks, as well as gradient boosting classifiers, are well-suited for this prediction task. Comparing feature groups of technical, blockchain-based, sentiment-/interest-based, and asset-based features shows that, for most methods, technical features remain prevalently important. For longer prediction horizons, the relative importance appears to spread across multiple features (e.g., transactions per second, weighted sentiment), whereby less recent technical features become increasingly relevant. A quantile-based trading strategy based on the market predictions yields up to 116% return over three months before transaction costs. However, due to the particularly short holding periods and correspondingly frequent trading activities, these returns cannot compensate for incurring transaction costs.

## Declaration of competing interest

All authors have none to declare.

## Acknowledgements

## Appendix A. Supplemental tables

Table A.1

**Trading class probability thresholds.** Overview of the class probability thresholds for trading. A buy (sell) trade is initiated if the model's predicted probability that a observation belongs to Class 1 (Class 0) exceeds the listed probability threshold.

| Model | 1-Minute Predictions | | 5-Minute Predictions | |
|---|---|---|---|---|
| | Buy | Sell | Buy | Sell |
| GRU | 0.6021 | 0.6029 | 0.6103 | 0.5874 |
| LSTM | 0.6040 | 0.6108 | 0.5909 | 0.5755 |
| FNN | 0.5379 | 0.5498 | 0.5788 | 0.5730 |
| LR | 0.5804 | 0.5971 | 0.6354 | 0.6112 |
| GBC | 0.5727 | 0.5888 | 0.5973 | 0.5814 |
| RF | 0.5095 | 0.5113 | 0.5110 | 0.5111 |

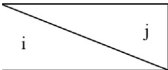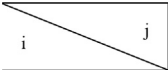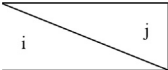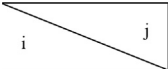| Model | 15-Minute Predictions | | 60-Minute Predictions | |
|---|---|---|---|---|
| | Buy | Sell | Buy | Sell |
| GRU | 0.6136 | 0.5932 | 0.5999 | 0.5972 |
| LSTM | 0.6065 | 0.5899 | 0.6071 | 0.6043 |
| FNN | 0.5601 | 0.5608 | 0.5872 | 0.5858 |
| LR | 0.6633 | 0.6292 | 0.6962 | 0.6604 |
| GBC | 0.6216 | 0.5996 | 0.6645 | 0.6210 |
| RF | 0.5141 | 0.5147 | 0.5173 | 0.5182 |

Table A.2

**Model accuracies per random seed.** Overview of the predictive accuracies of all stochastic prediction models for the individual random seeds.

| Model | Accuracy 1-Minute Predictions | | | | |
|---|---|---|---|---|---|
| | Seed 0 | Seed 1 | Seed 2 | Seed 3 | Seed 4 |
| GRU | 0.5164 | 0.5159 | 0.5166 | 0.5193 | 0.5153 |
| LSTM | 0.5136 | 0.5169 | 0.5183 | 0.5205 | 0.5171 |
| FNN | 0.5166 | 0.5088 | 0.5151 | 0.5124 | 0.5049 |
| RF | 0.5170 | 0.5142 | 0.5182 | 0.5118 | 0.5110 |

| Model | Accuracy 1-Minute Predictions | | | | |
|---|---|---|---|---|---|
| | Seed 5 | Seed 6 | Seed 7 | Seed 8 | Seed 9 |
| GRU | 0.5194 | 0.5204 | 0.5179 | 0.5174 | 0.5190 |
| LSTM | 0.5156 | 0.5172 | 0.5192 | 0.5194 | 0.5184 |
| FNN | 0.5033 | 0.5142 | 0.5103 | 0.4943 | 0.5094 |
| RN | 0.5134 | 0.5109 | 0.5113 | 0.5123 | 0.5102 |

| Model | Accuracy 5-Minute Predictions | | | | |
|---|---|---|---|---|---|
| | Seed 0 | Seed 1 | Seed 2 | Seed 3 | Seed 4 |
| GRU | 0.5228 | 0.5219 | 0.5260 | 0.5242 | 0.5267 |
| LSTM | 0.5241 | 0.5202 | 0.5223 | 0.5218 | 0.5228 |
| FNN | 0.5192 | 0.5190 | 0.5210 | 0.5207 | 0.5090 |
| RF | 0.5292 | 0.5231 | 0.5269 | 0.5258 | 0.5261 |

| Model | Accuracy 5-Minute Predictions | | | | |
|---|---|---|---|---|---|
| | Seed 5 | Seed 6 | Seed 7 | Seed 8 | Seed 9 |
| GRU | 0.5221 | 0.5186 | 0.5212 | 0.5243 | 0.5244 |
| LSTM | 0.5218 | 0.5209 | 0.5210 | 0.5225 | 0.5235 |
| FNN | 0.5167 | 0.5203 | 0.5183 | 0.5140 | 0.5172 |
| RN | 0.5265 | 0.5265 | 0.5242 | 0.5270 | 0.5259 |

| Model | Accuracy 15-Minute Predictions | | | | |
|---|---|---|---|---|---|
| | Seed 0 | Seed 1 | Seed 2 | Seed 3 | Seed 4 |
| GRU | 0.5321 | 0.5349 | 0.5325 | 0.5385 | 0.5367 |
| LSTM | 0.5295 | 0.5313 | 0.5279 | 0.5284 | 0.5324 |
| FNN | 0.5189 | 0.5292 | 0.5146 | 0.5224 | 0.5040 |
| RF | 0.5361 | 0.5322 | 0.5341 | 0.5332 | 0.5350 |

| Model | Accuracy 15-Minute Predictions | | | | |
|---|---|---|---|---|---|
| | Seed 5 | Seed 6 | Seed 7 | Seed 8 | Seed 9 |
| GRU | 0.5326 | 0.5344 | 0.5332 | 0.5313 | 0.5366 |
| LSTM | 0.5305 | 0.5309 | 0.5240 | 0.5338 | 0.5260 |
| FNN | 0.5111 | 0.5272 | 0.5194 | 0.5173 | 0.5172 |
| RN | 0.5357 | 0.5323 | 0.5342 | 0.5332 | 0.5325 |

| Model | Accuracy 60-Minute Predictions | | | | |
|---|---|---|---|---|---|
| | Seed 0 | Seed 1 | Seed 2 | Seed 3 | Seed 4 |
| GRU | 0.5555 | 0.5403 | 0.5471 | 0.5569 | 0.5545 |
| LSTM | 0.5548 | 0.5502 | 0.5549 | 0.5505 | 0.5485 |
| FNN | 0.5466 | 0.5164 | 0.5474 | 0.5448 | 0.5085 |
| RF | 0.5528 | 0.5563 | 0.5535 | 0.5531 | 0.5598 |

| Model | Accuracy 60-Minute Predictions | | | | |
|---|---|---|---|---|---|
| | Seed 5 | Seed 6 | Seed 7 | Seed 8 | Seed 9 |
| GRU | 0.5560 | 0.5579 | 0.5517 | 0.5374 | 0.5515 |
| LSTM | 0.5523 | 0.5526 | 0.5602 | 0.5453 | 0.5563 |
| FNN | 0.5318 | 0.5475 | 0.5115 | 0.5092 | 0.5061 |
| RN | 0.5565 | 0.5536 | 0.5549 | 0.5567 | 0.5540 |

Table A.3
**Diebold-Mariano tests.** Diebold-Mariano test p-values to reject the null hypothesis towards the alternative hypothesis that the forecast of model *i* on the test sample is more accurate than the forecast of model *j*.

15-Minute Predictions

| i \ j | GRU | LSTM | FNN | LR | GBC | RF | E (All) |
|---|---|---|---|---|---|---|---|
| GRU | - | 0.8542 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0001 |
| LSTM | 0.1458 | - | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| FNN | 1.0000 | 1.0000 | - | 0.8890 | 0.8409 | 0.9456 | 0.9999 |
| LR | 1.0000 | 1.0000 | 0.1110 | - | 0.4610 | 0.6478 | 0.9809 |
| GBC | 1.0000 | 1.0000 | 0.1591 | 0.5390 | - | 0.8054 | 0.9994 |
| RF | 1.0000 | 1.0000 | 0.0544 | 0.3522 | 0.1946 | - | 0.9935 |
| E (All) | 0.9999 | 1.0000 | 0.0001 | 0.0191 | 0.0006 | 0.0065 | - |

5-Minute Predictions

| i \ j | GRU | LSTM | FNN | LR | GBC | RF | E (All) |
|---|---|---|---|---|---|---|---|
| GRU | - | 0.6373 | 0.0464 | 0.0000 | 0.9978 | 0.9110 | 0.9128 |
| LSTM | 0.3627 | - | 0.0246 | 0.0000 | 0.9945 | 0.8682 | 0.8425 |
| FNN | 0.9536 | 0.9754 | - | 0.0021 | 1.0000 | 0.9974 | 0.9995 |
| LR | 1.0000 | 1.0000 | 0.9979 | - | 1.0000 | 1.0000 | 1.0000 |
| GBC | 0.0022 | 0.0055 | 0.0000 | 0.0000 | - | 0.0412 | 0.0141 |
| RF | 0.0890 | 0.1318 | 0.0026 | 0.0000 | 0.9588 | - | 0.3476 |
| E (All) | 0.0872 | 0.1575 | 0.0005 | 0.0000 | 0.9859 | 0.6524 | - |

15-Minute Predictions

| i \ j | GRU | LSTM | FNN | LR | GBC | RF | E (All) |
|---|---|---|---|---|---|---|---|
| GRU | - | 0.0000 | 0.0000 | 0.0000 | 0.6756 | 0.1154 | 0.8878 |
| LSTM | 1.0000 | - | 0.0000 | 0.0000 | 0.9986 | 0.9599 | 1.0000 |
| FNN | 1.0000 | 1.0000 | - | 0.2029 | 1.0000 | 1.0000 | 1.0000 |
| LR | 1.0000 | 1.0000 | 0.7971 | - | 1.0000 | 1.0000 | 1.0000 |
| GBC | 0.3244 | 0.0014 | 0.0000 | 0.0000 | - | 0.0518 | 0.6491 |
| RF | 0.8846 | 0.0401 | 0.0000 | 0.0000 | 0.9482 | - | 0.9866 |
| E (All) | 0.1122 | 0.0000 | 0.0000 | 0.0000 | 0.3509 | 0.0134 | - |

60-Minute Predictions

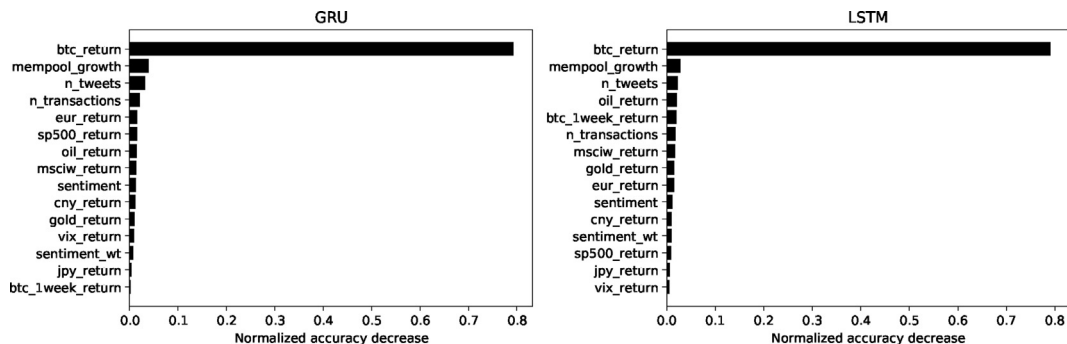| i \ j | GRU | LSTM | FNN | LR | GBC | RF | E (All) |
|---|---|---|---|---|---|---|---|
| GRU | - | 0.9999 | 0.0000 | 0.0000 | 0.5792 | 0.4194 | 0.7790 |
| LSTM | 0.0000 | - | 0.0000 | 0.0000 | 0.0500 | 0.0052 | 0.0213 |
| FNN | 1.0000 | 1.0000 | - | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| LR | 1.0000 | 1.0000 | 0.0000 | - | 1.0000 | 1.0000 | 1.0000 |
| GBC | 0.4208 | 0.9500 | 0.0000 | 0.0000 | - | 0.3433 | 0.6387 |
| RF | 0.5806 | 0.9948 | 0.0000 | 0.0000 | 0.6567 | - | 0.8145 |
| E (All) | 0.2210 | 0.9787 | 0.0000 | 0.0000 | 0.3613 | 0.1855 | - |

## Appendix B. Supplemental graphical material



Figure B.1. **Feature importance 1-minute memory models.** Feature importance of the models with memory function on the 1-minute prediction horizon.
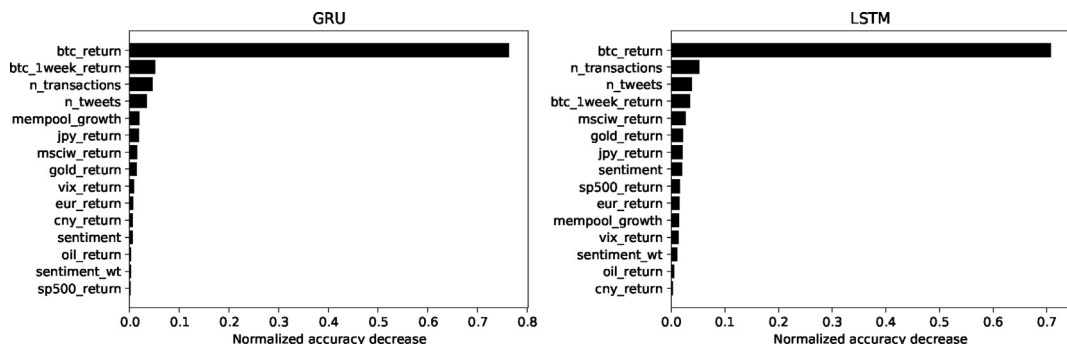


Figure B.2. **Feature importance 5-minute memory models.** Feature importance of the models with memory function on the 5-minute prediction horizon.
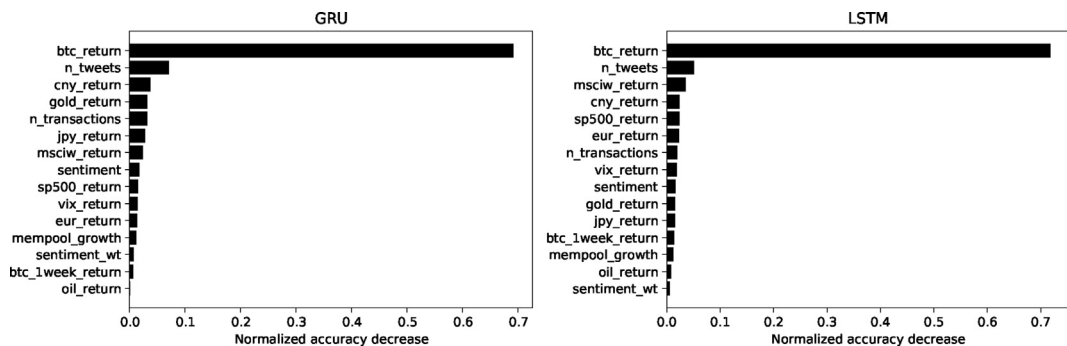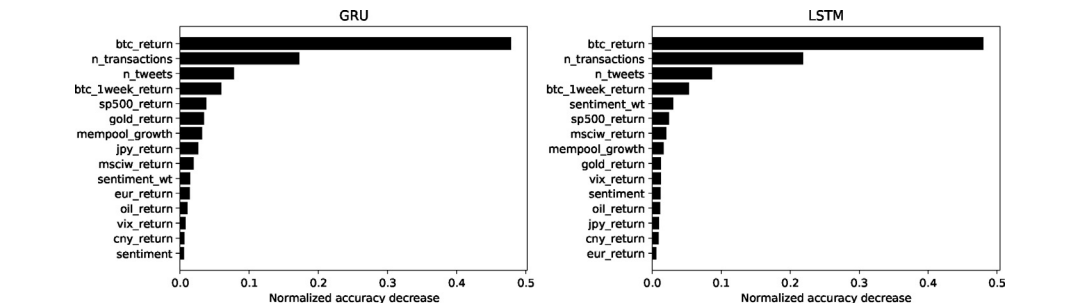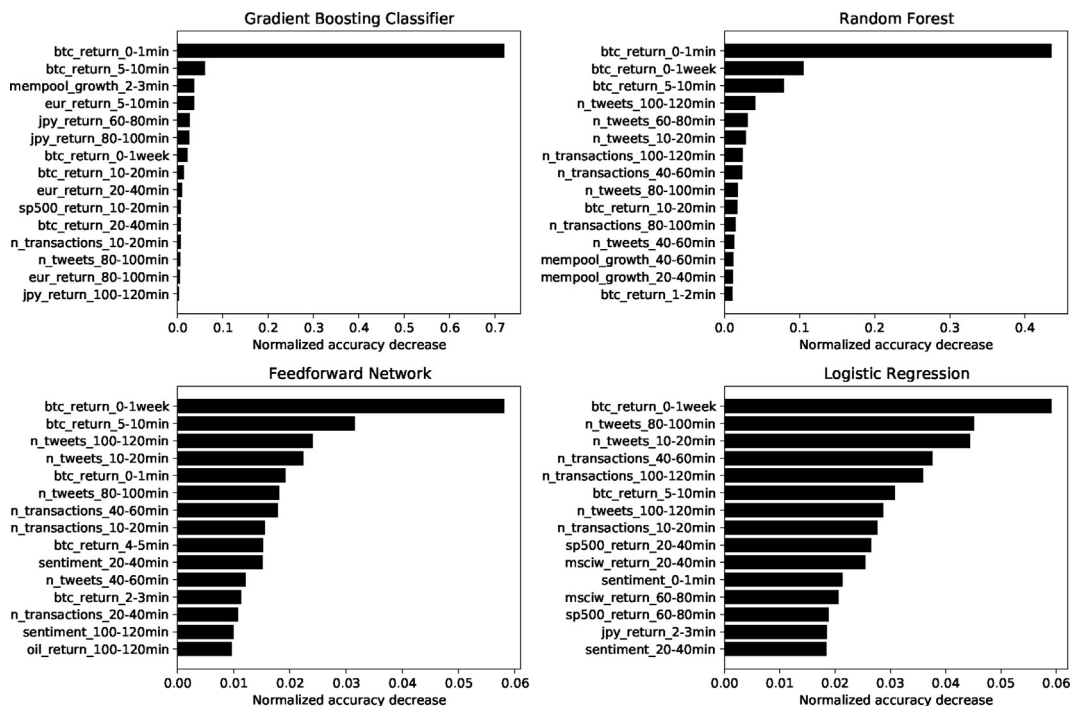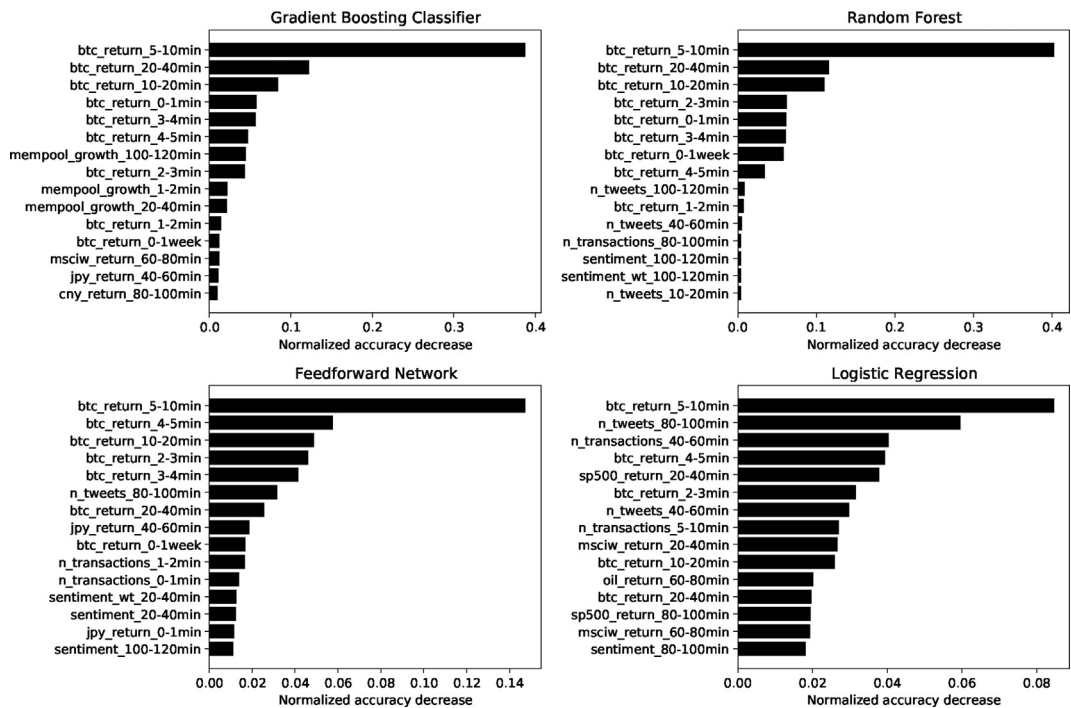


Figure B.3. **Feature importance 15-minute memory models.** Feature importance of the models with memory function on the 15-minute prediction horizon.



Figure B.4. **Feature importance 60-minute memory models.** Feature importance of the models with memory function on the 60-minute prediction horizon.
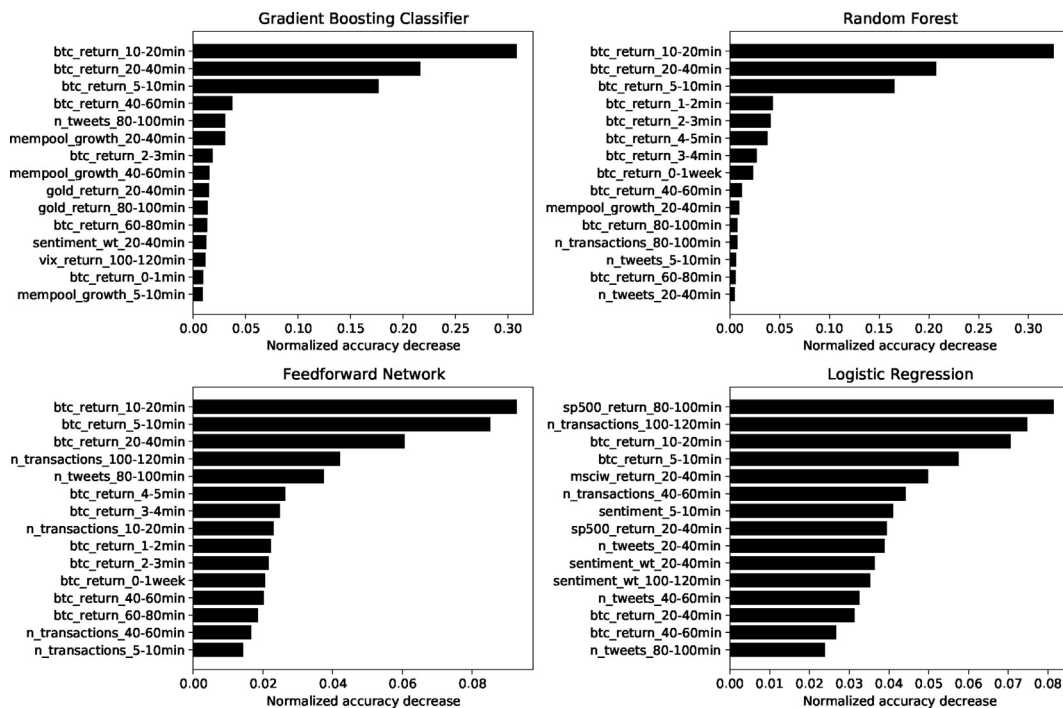
Figure B.5. **Feature importance 1-minute no-memory models.** Feature importance of the models without memory function on the 1-minute prediction horizon.



Figure B.6. **Feature importance 5-minute no-memory models.** Feature importance of the models without memory function on the 5-minute prediction horizon.

Figure B.7. **Feature importance 15-minute no-memory models.** Feature importance of the models without memory function on the 15-minute prediction horizon.
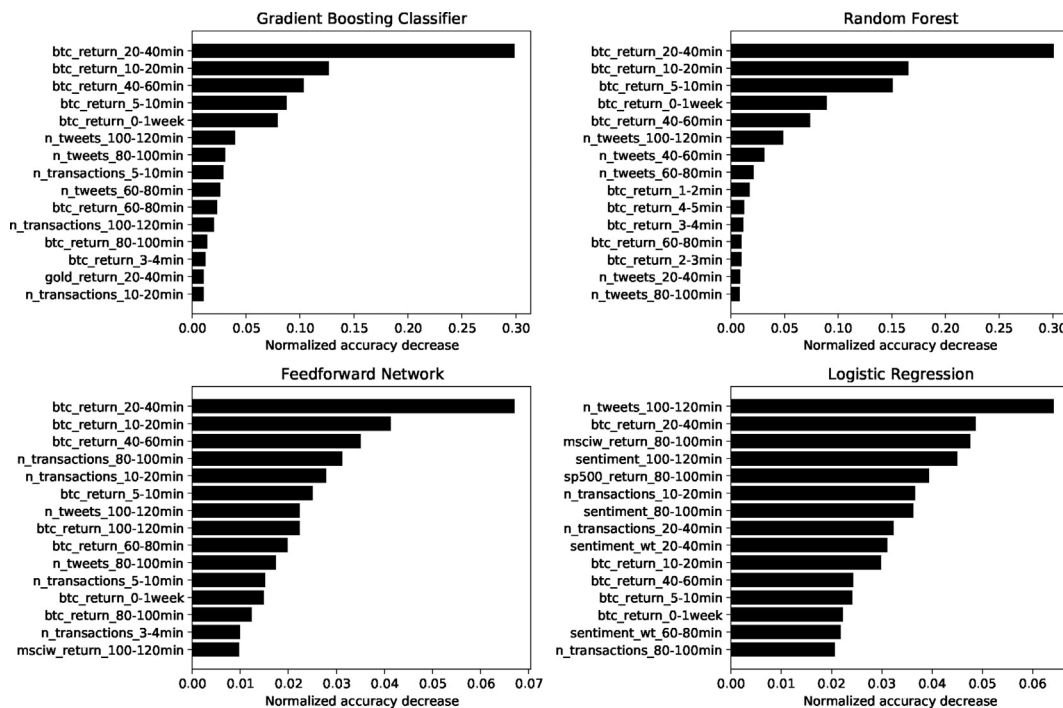


Figure B.8. **Feature importance 60-minute no-memory models.** Feature importance of the models without memory function on the 60-minute prediction horizon.

# References

1. Nakamoto S. Bitcoin: a peer-to-peer electronic cash system. *Work Pap*; 2008. https://bitcoin.org/bitcoin.pdf.
2. Beck R, Avital M, Rossi M, Thatcher JB. Blockchain technology in business and information systems research. *Bus Inf Syst Eng*. 2017;59:381–384. https://doi.org/10.1007/s12599-017-0505-1.
3. Gu S, Kelly B, Xiu D. Empirical asset pricing via machine learning. *Rev Financ Stud*. 2020;33:2223–2273. https://doi.org/10.1093/rfs/hhaa009.
4. Feng G, Giglio S, Xiu D. Taming the factor zoo: a test of new factors. *J Finance*. 2020;75:1327–1370. https://doi.org/10.1111/jofi.12883.
5. Jaquart P, Dann D, Martin C. Machine learning for bitcoin pricing - a structured literature review. In: *WI 2020 Proceedings, GITO Verlag*. 2020:174–188. https://doi.org/10.30844/wi_2020_b4-jaquart.
6. *Coinmarketcap*; 2020. URL: https://coinmarketcap.com/. Accessed September 15, 2020.
7. Fama EF. Efficient capital markets: a review of theory and empirical work. *J Finance*. 1970;25:383–417. https://doi.org/10.1111/j.1540-6261.1970.tb00518.x.
8. Lo AW. The adaptive markets hypothesis. In: *Adaptive Markets*. vol. 30. Princeton University Press; 2019:176–221. https://doi.org/10.2307/j.ctvc7778k.9.
9. Fama EF, MacBeth JD. Risk, return, and equilibrium: empirical tests. *J Polit Econ*. 1973;81:607–636. https://doi.org/10.1086/260061.
10. Fama EF, French KR. Dissecting anomalies. *J Finance*. 2007;63:1653–1678. https://doi.org/10.2139/ssrn.911960.
11. Fischer T, Krauss C. Deep learning with long short-term memory networks for financial market predictions. *Eur J Oper Res*. 2018;270:654–669. https://doi.org/10.1016/j.ejor.2017.11.054.
12. Krollner B, Vanstone B, Finnie G. Financial time series forecasting with machine learning techniques: a survey. In: *Proceedings of European Symposium on Artificial Neural Networks: Computational and Machine Learning*. Springer; 2010:1–7.
13. Fama EF. Market efficiency, long-term returns, and behavioral finance. *J Financ Econ*. 1997;49:283–306. https://doi.org/10.2139/ssrn.15108.
14. Grossman SJ, Stiglitz JE. On the impossibility of informationally efficient markets. *Am Econ Rev*. 1980;70:393–408.
15. Green J, Hand JRM, Zhang F. The supraview of return predictive signals. *Rev Account Stud*. 2012;18:692–730. https://doi.org/10.2139/ssrn.2062464.
16. Shleifer A, Vishny R. The limits of arbitrage. *J Finance*. 1995;52:35–55. https://doi.org/10.1111/j.1540-6261.1997.tb03807.x.
17. Schwert GW. Anomalies and market efficiency. In: *Handbook of the Economics of Finance, Handbook of the Economics of Finance*. Elsevier; 2003:939–974. https://doi.org/10.1016/S1574-0102(03)01024-0.
18. Glaser F, Zimmermann K, Haferkorn M, Weber MC, Siering M. Bitcoin-asset or currency? revealing users' hidden intentions. In: *Proceedings of 22nd European Conference on Information Systems*. 2014:1–15.
19. Dyhrberg AH. Bitcoin, gold and the dollar - a GARCH volatility analysis. *Finance Res Lett*. 2016;16:85–92. https://doi.org/10.1016/j.frl.2015.10.008.
20. Burniske C, White A. *Bitcoin: Ringing the Bell for a New Asset Class, Working Paper*. 2017.
21. Hu AS, Parlour CA, Rajan U. Cryptocurrencies: stylized facts on a new investible instrument. *Financ Manag*. 2019;48:1049–1068. https://doi.org/10.1111/fima.12300.
22. Urquhart A. The inefficiency of bitcoin. *Econ Lett*. 2016;148:80–82. https://doi.org/10.1016/j.econlet.2016.09.019.
23. Nadarajah S, Chu J. On the inefficiency of Bitcoin. *Econ Lett*. 2017;150:6–9. https://doi.org/10.1016/j.econlet.2016.10.033.
24. Bariviera AF. The inefficiency of Bitcoin revisited: a dynamic approach. *Econ Lett*. 2017;161:1–4. https://doi.org/10.1016/j.econlet.2017.09.013.
25. Hurst HE. Long-term storage capacity of reservoirs. *Trans Am Soc Civ Eng*. 1951;116:770–799.
26. Vidal-Tomás D, Ibañez A. Semi-strong efficiency of bitcoin. *Finance Res Lett*. 2018;27:259–265. https://doi.org/10.1016/j.frl.2018.03.013.
27. Khuntia S, Pattanayak J. Adaptive market hypothesis and evolving predictability of bitcoin. *Econ Lett*. 2018;167:26–28. https://doi.org/10.1016/j.econlet.2018.03.005.
28. Karakoyun ES, Cibikdiken AO. Comparison of ARIMA time series model and LSTM deep learning algorithm for bitcoin price forecasting. In: *Proceedings of 13th Multidisciplinary Academic Conference*. 2018:171–180.
29. McNally S, Roche J, Caton S. Predicting the price of bitcoin using machine learning. In: *Proceedings of 2018 Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. 2018:339–343. https://doi.org/10.1109/PDP2018.2018.00060.
30. Huang J-Z, Huang W, Ni J. Predicting bitcoin returns using high-dimensional technical indicators. *J Finance Data Sci*. 2019;5:140–155. https://doi.org/10.1016/j.jfds.2018.10.001.
31. Chen Z, Li C, Sun W. Bitcoin price prediction using machine learning: an approach to sample dimension engineering. *J Comput Appl Math*. 2020;365, 112395. https://doi.org/10.1016/j.cam.2019.112395.
32. Kubat M, Matwin S. Others, Addressing the curse of imbalanced training sets: one-sided selection. In: *Proceedings of the 14th International Conference on MachineLearning*. Citeseer, Morgan Kaufmann; 1997:179–186.
33. Barandela R, Valdovinos RM, Sánchez JS, Ferri FJ. The imbalanced training sample problem: under or over sampling?. In: *Lecture Notes in Computer Science*. Springer, Springer; 2004:806–814. https://doi.org/10.1007/978-3-540-27868-9_88.
34. Mudassir M, Bennbaia S, Unal D, Hammoudeh M. Time-series forecasting of Bitcoin prices using high-dimensional features: a machine learning approach. *Neural Comput Appl*. 2020:1–15. https://doi.org/10.1007/s00521-020-05129-6.
35. Dutta A, Kumar S, Basu M. A gated recurrent unit approach to bitcoin price prediction. *J Risk Financ Manag*. 2020;13. https://doi.org/10.3390/jrfm13020023. Article 23.

36. Jaquart P, Dann D, Weinhardt C. Using machine learning to predict short-term movements of the bitcoin market. In: *Proceedings of 10th International Workshop on Enterprise Applications, Markets and Services in the Finance Industry*. Springer; 2020:21−40. https://doi.org/10.1007/978-3-030-64466-6_2.

37. Poyser O. Exploring the dynamics of Bitcoin's price: a Bayesian structural time series approach. *Eurasia Econ Rev*. 2019;9:29−60. https://doi.org/10.1007/s40822-018-0108-2.

38. Madan I, Saluja S, Zhao A. *Automated Bitcoin Trading via Machine Learning Algorithms*. Working Paper; 2015.

39. Smuts N. What drives cryptocurrency prices? An investigation of Google trends and telegram sentiment. *ACM SIGMETRICS Perform Eval Rev*. 2019;46:131−134. https://doi.org/10.1145/3308897.3308955.

40. McKinney W. Data structures for statistical computing in Python. In: *Proceedings of 9th Python in Science Conference*. vol. 445. TX: Austin; 2010:56−61. https://doi.org/10.25080/Majora-92bf1922-00a.

41. van der Walt S, Colbert SC, Varoquaux G. The NumPy array: a structure for efficient numerical computation. *Comput Sci Eng*. 2011;13:22−30. https://doi.org/10.1109/MCSE.2011.37.

42. Bird S, Klein E, Loper E. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Inc.; 2009.

43. Google, Google. *Cloud Natural Language API*; 2020. URL: https://cloud.google.com/natural-language/docs.

44. Chollet F. *Keras*; 2015. URL: https://keras.io/.

45. Abadi M, Barham P, Chen J, et al. Tensorflow: a system for large-scale machine learning. In: *Proceedings of 12th USENIX Symposium on Operating Systems Design and Implementation*. 2016:265−283.

46. Chen T, Guestrin C, Boost XG. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM; 2016:785−794. https://doi.org/10.1145/2939672.2939785.

47. Pedregosa F, Varoquaux G, Gramfort A, et al. Others, scikit-learn: machine learning in Python. *J Mach Learn Res*. 2011;12:2825−2830.

48. Kumar A, Garg G. Sentiment analysis of multimodal twitter data. *Multimed Tool Appl*. 2019;78:24103−24119. https://doi.org/10.1007/s11042-019-7390-1.

49. Symeonidis S, Effrosynidis D, Arampatzis A. A comparative evaluation of pre-processing techniques and their interactions for twitter sentiment analysis. *Expert Syst Appl*. 2018;110:298−310. https://doi.org/10.1016/j.eswa.2018.06.022.

50. Goodfellow I, Bengio Y, Courville A. *Deep Learning*. MIT press; 2016.

51. Takeuchi L, Lee Y-YA. *Applying Deep Learning to Enhance Momentum Trading Strategies in Stocks*. Working Paper; 2013.

52. Krauss C, Do XA, Huck N. Deep neural networks, gradient-boosted trees, random forests: statistical arbitrage on the S&P 500. *Eur J Oper Res*. 2017;259:689−702. https://doi.org/10.1016/j.ejor.2016.10.031.

53. Finnoff W, Hergert F, Zimmermann HG. Improving model selection by nonconvergent methods. *Neural Networks*. 1993;6:771−783. https://doi.org/10.1016/S0893-6080(05)80122-4.

54. Prechelt L. Early stopping - but when?. In: *Neural Networks: Tricks of the Trade*. Springer; 2012:53−67. https://doi.org/10.1007/978-3-642-35289-8_5.

55. Kingma DP, Ba J, Adam. A method for stochastic optimization. In: *Proceedings of 3rd International Conference on Learning Representations*. 2015.

56. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res*. 2014;15:1929−1958.

57. Hornik K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*. 1991;4:251−257. https://doi.org/10.1016/0893-6080(91)90009-T.

58. Hammer B. On the approximation capability of recurrent neural networks. *Neurocomputing*. 2000;31:107−123. https://doi.org/10.1016/S0925-2312(99)00174-5.

59. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9:1735−1780. https://doi.org/10.1162/neco.1997.9.8.1735.

60. Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*. 2005;18:602−610. https://doi.org/10.1016/j.neunet.2005.06.042.

61. Graves A, Fernández S, Gomez F, Schmidhuber J. Connectionist temporal classification. In: *Proceedings of the 23rd International Conference on Machine Learning*. ACM Press; 2006:369−376. https://doi.org/10.1145/1143844.1143891.

62. Graves A, Mohamed A-r, Hinton G. Speech recognition with deep recurrent neural networks. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE; 2013:6645−6649. https://doi.org/10.1109/ICASSP.2013.6638947.

63. Graves A, Jaitly N. Towards end-to-end speech recognition with recurrent neural networks. In: *Proceedings of International Conference on Machine Learning*. 2014:1764−1772.

64. Graves A. *Generating Sequences with Recurrent Neural Networks*. Working Paper; 2013.

65. Liwicki M, Graves A, Fernàndez S, Bunke H, Schmidhuber J. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In: *Proceedings of 9th International Conference on Document Analysis and Recognition*. ICDAR; 2007.

66. Graves A, Liwicki M, Bunke H, Schmidhuber J, Fernández S. Unconstrained on-line handwriting recognition with recurrent neural networks. In: *Proceedings of Advances in Neural Information Processing Systems Conference*. 2008:577−584.

67. Graves A, Liwicki M, Fernandez S, Bertolami R, Bunke H, Schmidhuber J. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans Pattern Anal Mach Intell*. 2009;31:855−868. https://doi.org/10.1109/TPAMI.2008.137.

68. Eck D, Schmidhuber J. Finding temporal structure in music: blues improvisation with LSTM recurrent networks. In: *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*. IEEE; 2002:747−756. https://doi.org/10.1109/NNSP.2002.1030094.

69. Mäkinen M, Iosifidis A, Gabbouj M, Kanniainen J. *Predicting Jump Arrivals in Stock Prices Using Neural Networks with Limit Order Book Data*. 2018. https://doi.org/10.2139/ssrn.3165408. Working Paper.

70. Gers FA, Schraudolph NN, Schmidhuber J. Learning precise timing with LSTM recurrent networks. *J Mach Learn Res*. 2002;3:115−143.

71. Gers F. Learning to forget: continual prediction with LSTM. In: *Proceedings of 9th International Conference on Artificial Neural*. vol. 1999. IEEE; 1999:850–855. https://doi.org/10.1049/cp:19991218.

72. Chung J, Gülçehre Ç, Cho K, Bengio Y. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. Working Paper; 2014. arXiv:1412.3555.

73. Olah C. Understanding lstm networks. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/; 2015. Accessed September 1, 2020.

74. Ho T. Random decision forests. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. vol. 1. IEEE Comput. Soc. Press; 1995:278–282. https://doi.org/10.1109/ICDAR.1995.598994.

75. Friedman JH. Greedy function approximation: a gradient boosting machine. *Ann Stat*. 2001;5:1189–1232.

76. Diebold F, Mariano R. Comparing predictive accuracy. *J Bus Econ Stat*. 1994;20:134–144. https://doi.org/10.1198/073500102753410444.

77. Breiman L. Random forests. *Mach Learn*. 2001;45:5–32. https://doi.org/10.1023/A:1010933404324.

78. Fischer T, Krauss C, Deinert A. Statistical arbitrage in cryptocurrency markets. *J Risk Financ Manag*. 2019;12:31. https://doi.org/10.3390/jrfm12010031.

79. Biais B, Bisiere C, Bouvard M, Casamatta C, Menkveld AJ. Equilibrium Bitcoin Pricing, Working Paper. https://doi.org/10.2139/ssrn.3261063; 2018.

80. Detzel A, Liu H, Strauss J, Zhou G, Zhu Y. *Learning and Predictability via Technical Analysis: Evidence from Bitcoin and Stocks with Hard-To-Value Fundamentals*. Financial Management In Press; 2020. https://doi.org/10.1111/fima.12310.

81. Jegadeesh N. Evidence of predictable behavior of security returns. *J Finance*. 1990;45:881–898. https://doi.org/10.1111/j.1540-6261.1990.tb05110.x.

82. Shefrin H, Statman M. The disposition to sell winners too early and ride losers too long: theory and evidence. *J Finance*. 1985;40:777–790. https://doi.org/10.1111/j.1540-6261.1985.tb05002.x.