Available online at www.sciencedirect.com

ScienceDirect

The Journal of Finance and Data Science 5 (2019) 140–155
http://www.keaipublishing.com/en/journals/jfds/

Predicting bitcoin returns using high-dimensional technical indicators[☆]

Jing-Zhi Huang [a,*], William Huang [b], Jun Ni [c]

[a] Smeal College of Business, Pennsylvania State University, University Park, PA 16802, USA
[b] Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA 15213, USA
[c] AohuiData LLC, Shanghai, 202163, China

Received 12 October 2018; accepted 29 October 2018
Available online 22 November 2018

Abstract

There has been much debate about whether returns on financial assets, such as stock returns or commodity returns, are predictable; however, few studies have investigated cryptocurrency return predictability. In this article we examine whether bitcoin returns are predictable by a large set of bitcoin price-based technical indicators. Specifically, we construct a classification tree-based model for return prediction using 124 technical indicators. We provide evidence that the proposed model has strong out-of-sample predictive power for narrow ranges of daily returns on bitcoin. This finding indicates that using big data and technical analysis can help predict bitcoin returns that are hardly driven by fundamentals.
© 2019 The Authors, Production and hosting by Elsevier on behalf of KeAi Communications Co. Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Keywords: Bitcoin return prediction; High-dimensional classification; Decision tree classification; CART; Cryptocurrency; Bitcoin; Technical indicators

1. Introduction

Cryptocurrency is a digital currency that utilizes cryptography to secure the processes involved in transactions and generation of units. As the world's first decentralized cryptocurrency, bitcoin was created in 2009 based on a white paper written by a person with the pseudonym of Satoshi Nakamoto.[1] In centralized currencies, the government or other corporate entities have control over the supply of currency by printing new fiat money. In contrast, bitcoin is a decentralized currency, meaning that no single entity is responsible for the creation of new units or bitcoins (see, e.g., Harvey[2]).

☆ The authors would like to thank Weiping Li (the editor) and an anonymous referee for helpful comments and suggestions. We also thank Terry O'Brien for his editorial assistance.
* Corresponding author.
E-mail addresses: jxh56@psu.edu (J.-Z. Huang), wh2@andrew.cmu.edu (W. Huang), junni2019@126.com (J. Ni).
Peer review under responsibility of KeAi.

https://doi.org/10.1016/j.jfds.2018.10.001
2405-9188/© 2019 The Authors, Production and hosting by Elsevier on behalf of KeAi Communications Co. Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Bitcoin provides a secure way for people to make digital transactions with anonymity. Every time transactions of bitcoins are made, there needs to be a way to organize the order of bitcoins. It is easy to know who made the transaction and who was the recipient, but not necessarily the time of the transaction. In order to get one transaction in line next, one needs to solve a certain math puzzle. Every time these math puzzles are solved, and a transaction is queued, new bitcoins are created and introduced into the system. The reward for adding one block (of transactions) to the blockchain is 12.5 newly added bitcoins as of April 2019.[1] Every single bitcoin unit in circulation has been created from these transactions. The reward for adding a transaction to the blockchain will be halved every 4 years, reaching the limit of 0 at around the year 2140. There is currently a limit of 21 million units to the amount of bitcoins in circulation. As the number of bitcoins in circulation approaches closer to the limit, bitcoins become increasingly harder to mine. This limited supply of bitcoins is presumably one reason for the high valuation of bitcoins until January 2018.

If anything, bitcoin's prices have shown extremely high volatilities. For instance, the price was a “mere” $7854 on November 15, 2017. Just a month later, however, the price hit an all-time high of $19,511 on December 15, 2017. Then on December 28, 2017, the price has sunk 27% from this high. The bitcoin price has been in the range of $3400~$11,000 since February 2018. With such high volatility, there is an opportunity to make great amounts of money as well as a chance to lose eons. Many people have been scrambling to find ways to make money off of this volatile market. Thus, a natural question is whether returns of bitcoin are predictable. This is a non-trivial question, especially given the short history of bitcoin and the fact that there are very few studies of the bitcoin return predictability.

This paper investigates whether one can predict daily returns on bitcoin based on its historical prices only. Although it is desirable to be able to predict returns, the objective of this analysis is less ambitious because of the short history of bitcoin and huge volatility of bitcoin returns. Specifically, rather than predicting the level of the next day return, we aim to forecast its range. For instance, we may predict that the next day return will be between 1% and 3% instead of that it will be 2%. To make this exercise more meaningful in practice, the predicted ranges should not be too wide. As such, we divide the domain of the next day return, $[-100\%, \infty]$, into 21 intervals in this analysis and then try to predict in which of the 21 intervals the next day return will be—extending the analysis to other numbers of intervals is straightforward.

Given the speculative nature of the bitcoin value, fundamental analysis based on economic and accounting information is unlikely very useful for the purpose of this study. As a result, we focus on technical analysis and examine the potential predictive power of high-dimensional technical predictors in this study. The advantages of such predictors are two folds. First, they require only historical data on bitcoin returns, and thus are easy to implement. Second, the high-dimensional nature of such predictors allows us to consider a large set of candidate predictor variables and construct a potentially powerful predictor using these variables. Specifically, we consider 124 technical indicators, which can be classified into five groups: overlap studies indicators, momentum indicators, cycle indicators, volatility indicators, and pattern recognition indicators.

We conduct our analysis in three steps. First, using the 124 indicators, we construct a classification tree that can predict in which of the 21 non-overlapping return intervals bitcoin's next day return lies. In other words, we construct a tree with 124 input variables and a response that has 21 outcomes. Next, we repeat the above analysis 1000 times to build 1000 independent classification trees. We then construct a model for bitcoin return prediction based on these 1000 trees. Lastly, we examine the out-of-sample predictive power of this model and, for comparison, also consider the buy-and-hold strategy and some other simple tree-based strategies.

We find evidence that the proposed model for return prediction has strong out-of-sample predictive power for narrow ranges of daily returns on bitcoin. We also find that the proposed model outperforms the buy-and-hold strategy (as well as those simple strategies) even during a strong bull market under either the information ratio or the win-to-loss ratio. Our findings provide evidence that big data and technical analysis together have potential useful application in the prediction of bitcoin daily returns that are hardly driven by fundamentals.

The rest of this paper is organized as follows. Section 2 reviews related literature. Section 3 describes the data used in this analysis. Section 4 reviews the decision tree classification method that we use to construct classification trees

[1] See, e.g., https://en.bitcoin.it/wiki/Mining.

for return prediction. Section 5 presents our proposed model for bitcoin return prediction and examines the out-of-sample performance of the model. Finally, Section 6 concludes.

2. Related literature

This study is related to the new literature on cryptocurrency. Several studies have examined the valuation of bitcoin and other cryptocurrencies. For instance, Athey, Parashkevov, Sarukkai, and Xia[3] develop a model of bitcoin pricing and provide mixed evidence about the ability of the model to explain bitcoin prices. Pagnotta and Buraschi[4] consider the valuation of bitcoin and decentralized network assets using an equilibrium model. Several other studies analyze the implications of blockchains and related technologies for other areas in finance. For example, Raskin and Yermack[5] consider the implications for central banking. Yermack[6] focuses on corporate governance. Easley, O'Hara, and Basu[7] and Huberman, Leshno, and Moallemi[8] investigate bitcoin mining costs. Lastly, Harvey[9] provides an in-depth discussion of the mechanics of cryptocurrencies.

This study also fits in the large literature on technical analysis. For instance, since the article by Brock, Lakonishok, and LeBaron,[10] many studies have examined different aspects of moving average (MV) strategies in financial markets. Bessembinder and Chan,[11] LeBaron[12] and Sullivan, Timmermann, and White[13] focus on the profitability of these strategies in equity markets. Han, Yang, and Zhou[14] and Shynkevich[15] compare some specific MA strategies with the buy-and-hold strategy based on certain equity portfolios. Neely, Rapach, Tu, and Zhou[16] forecast the equity risk premium using technical indicators. Huang and Huang[17] test MV strategies using equity exchange traded funds (ETFs).

Besides MV strategies, other technical trading rules are also studied in the literature. Examples of such studies include Allen and Karjalainen,[18] Brown, Goetzmann, and Kumar,[19] Lo, Mamaysky, and Wang,[20] and Hsu, Hsu, and Kuan,[21] etc. In addition, Hsu, Taylor, and Wang[22] test some technical analysis rules using data on foreign exchanges.

This paper differs from the aforementioned studies in that it is among the first to examine the predictability of bitcoin returns using high-dimensional technical indicators. Among other things, this study provides evidence on the potential application of technical analysis on bitcoin. As such, the paper makes contribution to both strands of literature, namely, bitcoin and technical analysis.

3. Data description

The data used in this paper is a BTC-USD data set from investing.com, which includes the daily open, high, low and close prices of bitcoin from January 1st, 2012 to December 29th, 2017. After cleaning the data set, we have 2168 observations in total. We divide the whole sample into three subsamples. The first one covers the period January 2, 2012−April 29, 2012, consisting of 120 observations. This subsample is used to calculate the initial values of technical indicators that serve as inputs (predictors) in our decision-tree analysis. The second subsample is from April 30, 2012 to July 19, 2016 and used as the so-called training set in the decision-tree analysis. The third subsample is the period July 20, 2016−December 29, 2017 and used as the test set. The split between the training and test samples here is done such that the size of the training sample is about 3 times of the test sample. As a result, these two subsamples include 1539 and 509 observations, respectively. For convenience, the training and test samples together are referred to as the full sample in the balance of the paper.

Fig. 1 plots the daily price of bitcoin over the full sample period April 30, 2012−December 29, 2017. As can be seen from the figure, the bitcoin price is relatively low before 2017 and the return is not that volatile relatively. However, since the beginning of 2017, the price goes up tremendously and the return volatility becomes crazily huge. In other words, although our sample ends in 2017, it does include some periods with extremely high volatility.

Indeed, the price of bitcoin has gone through various cycles of appreciation and depreciation referred to by some as bubbles and busts over past several years. For instance, in 2011, the value of one bitcoin rapidly rose from about USD $0.30 to USD $32 before returning to USD $2. In the second half of 2012 and during the 2012−13 Cypriot financial crisis, the bitcoin price began to rise, reaching a high of USD $266 on April 10, 2013, before crashing to around $50. On November 29, 2013, the price of one bitcoin rose to a peak of $1242. In 2014, the price fell sharply, and as of April 2014 remained to be at little more than half of the 2013 prices and was under US $600 as of August 2014. Then the bitcoin price went to the roof during 2017. For a more detailed review of the bitcoin price history, see, e.g., Wiki.[23]

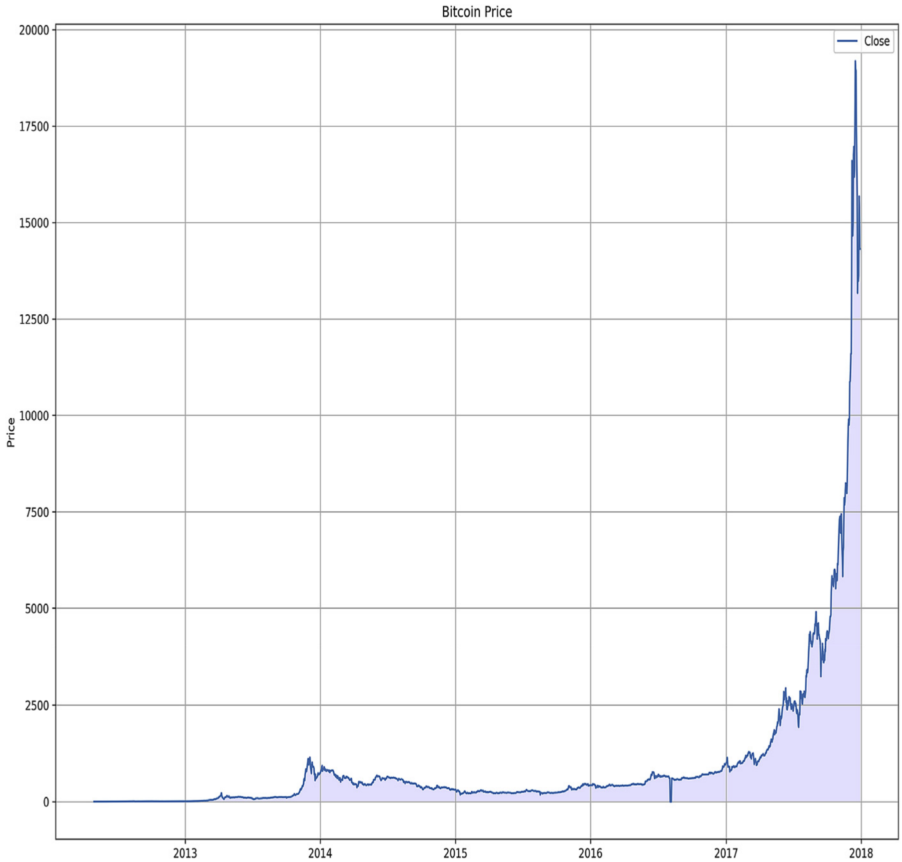

Fig. 1. **Daily Prices of bitcoin over the period April 30, 2012–December 29, 2017**. This figure plots the daily closing prices of bitcoin over the full sample period April 30, 2012–December 29, 2017. This sample period is split into the training period April 30, 2012–July 19, 2016 and the test period July 20, 2016–December 29, 2017. The sample over January 1, 2012–April 29, 2012, not included in the figure, is used to construct the initial values of input variables (124 technical indicators).

3.1. Response

As noted earlier, the objective of this study is not to predict the level of the next day return of bitcoin; instead, we aim to forecast the range of the next day return using the set of 124 technical indicators considered. For example, rather than predicting that the next day return will be, say, 0%, we try to predict that the return level will be between −0.2% and 0.2%. Note that if target return ranges are too wide, then the predictability of such ranges is less useful in practice. On the other hand, if a return range is too narrow, then it is more like the predictability of levels of the bitcoin price and a simple model may be less effective. To balance between these two considerations, we divide the return domain [−100%, ∞) into 21 non-overlapping return ranges (intervals that are closed on left, open on right) in the empirical analysis that follows. The main objective of this analysis is to try to predict in which of the 21 ranges the next day return will be. It is straightforward to extend our analysis to different partitions of the return domain.

The 21 return ranges consist of ten intervals that include only positive returns, ten that include only negative returns, and one "neutral" range that includes the return of zero. Specifically, the first ten return intervals are (−100%, −11.00%), (−11.00%, −9.00%), (−9.00%, −7.00%), (−7.00%, −5.00%), (−5.00%, −3.00%), (−3.00%, −1.00%), (−1.00%, −0.80%), (−0.80%, −0.60%), (−0.60%, −0.40%), and (−0.40%, −0.20%). We label these 10 ranges as "−10," "−9," …, and "−1," respectively. The second ten return ranges used are (0.20%, 0.40%), (0.40%, 0.60%), (0.60%, 0.80%), (0.80%, 1.00%), (1.00%, 3.00%), (3.00%, 5.00%), (5.00%, 7.00%), (7.00%, 9.00%), (9.00%, 11.00%), and (11.00%, ∞). These ranges are labeled as "1," "2," …, and "10," respectively. The "neutral" return range is (−0.20%, 0.20%), labeled as range "0."

Let y denote the response variable. It follows that by construction, the 21 return ranges are indexed by y, where $y \in \{-10, -9, \ldots, -1, 0, 1, \ldots, 9, 10\}$.

3.2. Predictors

Predictors to be used (input variables) are 124 technical indicators that are included in the TA-Lib library[24]; see Appendix A for a complete list of these indicators. All the indicators depend only on past prices of bitcoin. The 124 indicators are grouped into five categories by TA-Lib: overlap study indicators, momentum indicators, cycle indicators, volatility indicators, and pattern recognition indicators. Below we briefly describe each of these five categories of indicators (see Benediktsson *et al*.[24] for a detailed description of the 124 indicators).

3.2.1. Overlap study indicators

In technical analysis, the primary purpose of an overlap study indicator is to objectively identify the direction of a trend by smoothing out the volatile nature of the price action. There are many different trend indicators that can be used for technical analysis. Note that, however, these indicators tend to be lagging indicators as they are usually based on past prices. That is, these technical indicators follow price action rather than lead it. Some trend indicators, such as the moving average convergence/divergence (MACD) indicator of Appel,[25] also measure the strength of the trend. The most widely used trend indicator is the moving average (MA). Other popular trend indicators include Parabolic SAR (stop and reverse) of Wilder,[26] Kaufman Adaptive Moving Average KMAM,[27] and trend lines. These indicators help develop trend-following trading strategies, among other things.

3.2.2. Cycle indicators

Cycle indicators are oscillating indicators that are used to analyze market cycles in technical analysis. According to cycle theory, there exist periodic rhythms in stock markets (moving from bull market periods to bearish market periods and back to bull market periods). Such cycles are repeated periodically so they can help investor predict the market. One example of cycle indicators is the Schaff Trend Cycle (STC) indicator (see, e.g., Twomey[28]). The STC indicator combines a Slow Stochastics indicator with MACD and is often used to predict foreign exchange rates.

3.2.3. Momentum indicators

Momentum indicators are useful for determining how fast the price of the underlying security changes. These indicators plot the rate of price change rather than the price change itself and are usually leading, oscillating indicators. Some of these indicators, such as the Stochastic Oscillator and the Relative Strength Index (RSI) proposed by Wilder,[26] have an upper and lower boundary (usually 100 and −100), while others, such as the Commodity Channel Index (CCI) introduced by Lambert,[29] are unbounded. The other popular momentum indicators include the Chande Momentum Oscillator,[30] TRIX (triple exponential; see Wiki[31]) and Williams' %R (see, e.g., Wiki[32]).

3.2.4. Volatility indicators

Volatility indicators, such as the Average True Range developed by Wilder,[26] attempt to measure the volatility of a security's price action. Day traders prefer increased volatility as the more volatile the price is, the more money can be gained (or lost) in a short time. Examples of popular volatility indicators include Bollinger Bands (see, e.g., Bollinger[33]), Chaikin's Volatility (see, e.g., Kaufman and Chaikin[34]) Relative Volatility Index (RVI) of Dorsey[35], and Standard Deviation.

3.2.5. Pattern recognition indicators

Pattern recognition indicators are subroutines used to detect the specific price chart patterns, including two crows, three stars in the south, breakaway, dark cloud cover, etc.

4. Methodology

As mentioned before, the main idea of this paper is to predict narrow ranges of bitcoin's next day return using a tree-based classifier. In this section we first review the so-called decision tree classification method (see, e.g., Hastie, Tibshirani[36], and Friedman and Kuhn and Johnson[37] and references therein). We then briefly discuss how to implement this method in our analysis.

4.1. Decision trees

Decision tree learning is one of the most widely used and practical methods for inductive inference (see, e.g., Hastie et al.[36]) and, for a less technical introduction, Wiki.[38] More specifically, it is a method for predicting the value of a discrete-valued target via a decision tree. Moreover, as a non-parametric supervised learning method, the decision tree (the learned function) is robust to noisy data and capable of learning disjunctive expressions. Through learning, the resulting trees can also be re-represented as sets of if-then rules to improve human readability. This method has been applied to a broad range of tasks, such as medical diagnosis and assessing credit risk of loan applications. In this study, we explore the application of decision tree learning in cryptocurrency return prediction.

The core algorithm for building decision trees goes back to the so-called ID3 (Iterative Dichotomiser 3) developed by Quinlan[39] (see also, e.g., Wiki[40] for a non-technical review of ID3). This algorithm employs a top-down, greedy search through the space of possible branches with no backtracking. Also, decision trees are constructed using recursive partitioning that is based on entropy and information gain. Algorithms C4.5,[41] C5.0 and CART (Classification and Regression Trees) are improvements on ID3. C4.5 extends ID3 by removing the restriction that features must be categorical, because it dynamically defines a discrete attribute (based on numerical variables) that partitions the continuous attribute value into a discrete set of intervals. C4.5 converts the trained trees (i.e., the output of the ID3 algorithm) into sets of if-then rules. Algorithm C5.0 is the latest version released by Quinlan.[42] It is similar to CART, the method introduced by Breiman, Friedman, Olshen, and Stone[43] and developed independently of ID3. However, while C5.0 includes a scheme for deriving rule sets, CART supports numerical target variables (regression tress) and does not compute rule sets (see, e.g., Hastie et al.[36] and Kuhn and Johnson[37]). Also, CART builds binary trees using the feature and threshold that yield the largest information gain at each node.

A decision tree is built top-down from a root node and involves recursively partitioning the data space into subsets that contain instances with similar values. Suppose the number of observations in training data is N_{train}. Let $x_i \in R^P$ and $y_i \in R$ denote the i-th observations of P input variables and the response variable y, respectively, for $i = 1, 2, \ldots, N_{train}$. Note that in our case, $N_{train} = 1{,}538$, $P = 124$ (technical indicators or features), and $y \in \{-10, -9, \ldots, 9, 10\}$.

A decision tree will partition the data space such that the subsamples of data with the same labels are grouped together. If input variables are continuous (as in our case), the standard practice is to use a binary space partitioning based on thresholds of inputs. Our target is then to select a pair $\theta = (j, t_j)$—a feature j and its threshold t_j—to split the data on this feature, recursively. For illustration, below we briefly describe how to implement this idea, following largely Hastie et al.[36]

Let node Q denote all the data in one given step. We denote the data by Q_{left} if the j-th feature value of the samples in Q is less than or equal to t_j; otherwise, they are grouped as Q_{right}. More specifically, Q_{left} and Q_{right} are defined as follows:

$$Q_{left}(\theta) = \{(\mathrm{x}, \mathrm{y}), \mathrm{x_j} \le \mathrm{t_j}\}; \tag{1}$$

$$Q_{right}(\theta) = Q - Q_{left} \tag{2}$$

To choose the pair θ, we need to solve an optimization problem in each splitting to make the nodes of the split become purer (i.e. to partition the data into more homogenous groups). To this end, consider the following objective function:

$$G(Q, \theta) = \frac{n_{left}}{N_m} H\left(Q_{left}(\theta)\right) + \frac{n_{right}}{N_m} H\left(Q_{right}(\theta)\right) \tag{3}$$

where n_{left} denotes the number of data points in Q_{left}, n_{right} the number of data points in Q_{right}, N_m the total number of data points in Q, and $H(\cdot)$ is an impurity function to be defined below. The pair that leads to the best split is

$$\theta^* = argmin_\theta\, G(Q, \theta)$$

There are three commonly used types of impurity function: Cross-Entropy, Gini Index, and Misclassification Rate. In order to introduce their definitions, consider an auxiliary variable p_{mk} for class k in node m, where $k \in \{1, 2, \ldots, K\}$ and K is the number of classification outcomes (in our analysis $K = 21$). If the data in node m region is denoted by R_m and the number of observations is N_m, then the proportion of class k observations in node m is defined as:

$$p_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k) \tag{4}$$

Cross-Entropy is defined as:

$$H_{CE}(R_m) = -\sum_k p_{mk} * \log(p_{mk}) \tag{5}$$

The Gini Index is defined as follows:

$$H_{gini}(R_m) = \sum_k p_{mk} * (1 - p_{mk}) \tag{6}$$

Misclassification Rate is defined by:

$$H_{MC}(R_m) = 1 - \max_k p_{mk} \tag{7}$$

Among these three types of impurity function, cross-entropy and the Gini index are more suitable than the misclassification rate when growing a tree.[36] As such, we focus on the former two measures of impurity—cross-entropy and the Gini index—in our empirical analysis.

4.2. Implementation

In our analysis, the implementation of the 124 technical indicators is done using the package from TA-Lib. Decision trees for return prediction are constructed using the CART method. We implement this algorithm using the Scikit-learn function "DecisionTreeClassifer".[44] Among its inputs we specify, cross-entropy as given in Eq. (5) is used as the measure of impurity, partitioning is done until all terminal nodes (leaves) are pure, and the number of features (technical indicators) to consider when looking for the best split is set to 11 (the integer part of sqrt(124)). Note that at each split, candidate features to be considered are selected randomly and, importantly, by design more than 11 features will be considered if necessary.

As it is known that decision trees are sensitive to input data, we need to use multiple trees in order to construct a more reliable model for return prediction. In our implementation, we use 1000 different trees to train our return prediction model. We generate such trees simply by running the code 1000 times, given the fact that during the construction of a tree, candidate features (technical indicators) to be considered at each node are selected randomly. The construction of the model for return prediction is based on the in-sample performance of the "average tree" of those 1000 trees built in the training period. The out-of-sample performance of the proposed model is then based on out-of-sample return predictions (for the test period) of the same 1000 trees built in the training period.

5. Empirical analysis and results

In this section we conduct an empirical analysis to examine the predictive power of our proposed decision tree model and the ability of our trading strategy to generate trading profits. Section 5.1 reports and discusses summary statistics of bitcoin daily returns. Section 5.2 constructs the proposed tree model and examines its in-sample performance over the training sample. Section 5.3 presents the results on the out-of-sample performance of a trading strategy based on the decision tree model.

5.1. Summary statistics

Fig. 1 plots the daily price of the bitcoin over the full sample period April 30, 2012–December 29, 2017. Recall that this sample period includes both the training and test samples (April 30, 2012–July 19, 2016 and July 20, 2016−December 29, 2017) but not the initial sample period (January 1, 2012−April 29) used to estimate the initial values of those 124 technical indicators. As can be seen from the figure, the bitcoin price is relatively low before 2017 and the return is not that volatile, but since the beginning of 2017, the price goes up tremendously and the return volatility becomes crazily huge.

Table 1 reports summary statistics of bitcoin daily returns over both the full sample and its two subsamples. The reported variables include the average daily return, daily volatility, annualized volatility, the annualized geometric average of daily returns ($\overline{r}_G$), the annualized arithmetic average of daily returns ($\overline{r}_A$), and the information ratios based on these two annualized average returns. Let $\{r_i,\ i = 1, 2, \ldots, n\}$ denote the daily return series of bitcoin. The annualized geometric and arithmetic averages of daily returns are calculated as follows:

$$\overline{r}_G = \left[\prod_{i=1}^{n}(1+r_i)\right]^{\frac{365}{n}} - 1; \tag{8}$$

$$\overline{r}_A = \left[1 + \frac{\sum_{i=1}^{n} r_i}{n}\right]^{365} - 1 \tag{9}$$

As can be seen from the table, the annualized volatility is about 88.96%, 90.77%, and 83.17% for the full, training, and test samples, respectively; indeed, bitcoin is highly volatile through the full sample period. The geometric average return is also high: $\overline{r}_G$ is about 305.46%, 221.12%, and 719.13% for the full, training, and test samples, respectively. As a result, the information ratio based on this average return varies significantly over different sample periods. For example, the ratio is about 2.44 for the training sample, 8.65 for the test period, and 3.43 for the full sample. The annualized arithmetic average return $\overline{r}_A$ is around 501.42%, 385.09% and 1049.15% for the full, training and test samples, respectively. In other words, $\overline{r}_A$ is higher than $\overline{r}_G$, regardless of the samples considered here. Therefore, the information ratio based on the former is also higher than that based on the latter, being 5.64, 4.24 and 12.62 for the full, training and test samples, respectively. These results indicate that the bitcoin market performs extremely well over the test period 07/20/2016−12/29/2017.

However, we find that the proportion of days when the market is going upward is around 57.45 and the proportion of market going down is around 41.76; that is, the win/loss ratio is around 1.376 in the test period (untabulated). This ratio is not overly impressive, however, as it means that if you just use a buy-and-hold strategy, you will have one day losing money and 1.376 days making money on average; namely, the profit is quite volatile.

Next, we consider summary statistics of different ranges of bitcoin daily returns. Recall from Section 3.1 that we use 21 different return ranges in our analysis, which are indexed by the response variable $y \in \{-10, -9, \ldots, 9, 10\}$.

Table 2 provides a count of the number of days (the number of occurrences) when bitcoin daily returns lie in each of the 21 ranges specified in the analysis. The frequency count is reported for the full sample (April 30, 2012−December 29, 2017) as well as the training sample (April 30, 2012−July 19, 2016) and the test sample (July 20, 2016−December 29, 2017). Note that as the ranges become increasingly far away from range "0," the number of occurrences initially decreases but peaks at ranges "−5" and "5," regardless of whether it is the full sample, training sample or test sample. Also, among all the 21 return ranges considered, the number of occurrences for positive ranges ($y \geq 1$) is larger than

Table 1
Summary Statistics of Bitcoin Returns. This table reports summary statistics of bitcoin daily returns over the full sample (April 30, 2012−December 29, 2017) as well as the training sample (April 30, 2012−July 19, 2016) and the test sample (July 19, 2016−December 29, 2017). Annualized returns $\overline{r}_G$ and $\overline{r}_A$ are based on the geometric and arithmetic averages of daily returns, respectively. Information ratio based on $\overline{r}_G$ ($\overline{r}_A$) is the ratio of $\overline{r}_G$ ($\overline{r}_A$) to annualized volatility.

	Full sample	Training sample	Test sample
	04/30/2012−12/29/2017	04/30/2012−07/19/2016	07/20/2016−12/29/2017
Daily return mean	0.493%	0.434%	0.671%
Daily return volatility	4.656%	4.751%	4.353%
Annualized volatility	88.957%	90.767%	83.167%
Annualized return $\overline{r}_G$	305.456%	221.121%	719.133%
Info ratio based on $\overline{r}_G$	3.434	2.436	8.647
Annualized return $\overline{r}_A$	501.416%	385.086%	1049.152%
Info ratio based on $\overline{r}_A$	5.637	4.243	12.615

Table 2

Trend Statistics over the Full Sample Period. This table provides a count of the number of days when bitcoin daily returns lie in each of the 21 right-open intervals specified in the analysis. These 21 intervals/ranges are indexed by the response variable, $y = -10, -9 \ldots 0, 9 \ldots 9, 10$. The values of $y = -10$ and $y = 10$ represent the return ranges of (−100%, −11.00%) and (11.00%, ∞), respectively. The range in between, (−11.00%, 11.00%), is divided into 19 non-overlapping intervals, which are represented by $y = -9 \ldots 9$, respectively. The frequency count is reported for the full sample (April 30, 2012–December 29, 2017) as well as the training sample (April 30, 2012–July 19, 2016) and the test sample (July 20, 2016–December 29, 2017).

Number of occurrences for each value of the response variable y

y	−10	−9	−8	−7	−6	−5	−4
Return Range	(−100%, −11.00%)	(−11.00%, −9.00%)	(−9.00%, −7.00%)	(−7.00%, −5.00%)	(−5.00%, −3.00%)	(−3.00%, −1.00%)	(−1.00%, −0.80%)
Full sample	28	14	40	55	101	296	57
Training sample	23	9	28	37	78	238	47
Test sample	5	5	12	18	23	58	10
y	−3	−2	−1	0	1	2	3
Return Range	(−0.80%, −0.60%)	(−0.60%, −0.40%)	(−0.40%, −0.20%)	(−0.20%, 0.20%)	(0.20%, 0.40%)	(0.40%, 0.60%)	(0.60%, 0.80%)
Full sample	58	84	87	178	87	73	66
Training sample	43	60	65	136	73	58	50
Test sample	15	24	22	42	14	15	16
y	4	5	6	7	8	9	10
Return Range	(0.80%, 1.00%)	(1.00%, 3.00%)	(3.00%, 5.00%)	(5.00%, 7.00%)	(7.00%, 9.00%)	(9.00%, 11.00%)	(11.00%, ∞)
Full sample	67	383	180	79	43	22	50
Training sample	45	289	118	53	31	18	39
Test sample	22	94	62	26	12	4	11

the number of occurrences for negative ranges ($y \le -1$), due to the fact that the price of bitcoin has increased dramatically during our sample, especially in 2017.

5.2. *Construction of the decision tree model and its in-sample performance*

In this subsection we first construct a decision tree model for return prediction using data in the training sample. The model takes the 124 technical indicators discussed in Section 3.2 as inputs and predicts the range of the bitcoin return on next day. We then consider a trading strategy based on this trained model and examine the in-sample performance of this strategy. Without loss of generality, we assume that an investor has one dollar in her or his account initially in all trading strategies considered in Section 5. Also, we ignore transaction costs.

Our training data consist of daily observations of 124 technical indicators and the response variable over the training period. Denote the data by $(X, Y)_{train} = \{(x_i, y_i), i = 1, \ldots, N_{train}\}$, where $x_i = (x_{i1}, \ldots, x_{iP})$ with $P = 124$ represents the time-i vector of the 124 technical indicators (predictors or inputs). Given $(X, Y)_{train}$, we can generate a decision tree from our implementation of the CART algorithm.

Let $\hat{y}_i$ denote the i-th predicted value of y from a resulting tree. We now design a trading strategy based on this tree. For illustration, consider a signal generated from the tree, say, the signal "$\hat{y} = 3$," i.e., the return range = 3 (a positive range), where the time index of this signal is omitted to simply the notation. On a given day in the training sample, if the tree forecasts that the next day return is in range "3", then the investor takes a long position in her or his bitcoin account; that is, the investor buys one dollar of bitcoin. The next day realized return may be positive or negative. If it is positive, then the investor makes money on this trade; otherwise, if the next day return turns out to be negative, the investor loses money on this trade. Among the number of times that the investor makes a trade based on the signal $\hat{y} = 3$, the proportion of making money on the following day is defined as the win ratio of $\hat{y} = 3$, and the proportion of losing money on the following day is called the loss ratio of the same signal.

Next, consider a negative signal, say, "−3" (i.e., $\hat{y} = -3$). On a given day in the training sample, if the tree forecasts that the next day return is in range "−3", then the investor takes a short position in bitcoin. If the next day realized return of bitcoin is positive, then the investor loses money on the trade; otherwise, if the next day realized return of bitcoin is negative, then the investor makes money on this trade. It follows that the win and loss ratios of this signal "$\hat{y} = -3$" can also be calculated for the training sample.

Lastly, if the forecast range from the tree is range "0" (i.e., $\hat{y} = 0$) on a given day, then the investor takes no positions on the next day (keeping cash). Both win and loss ratios of this "0" signal are zero.

Taken together, it follows from the above discussion that given the training data, we can obtain a decision tree along with the win and loss ratios of 21 different trading strategies that are based on 21 predicted signals of the response.

In our implementation, we do this exercise 1000 times using the training data. Then we compute the mean and standard deviation of both win and loss ratios for each signal (i.e., each value of $\hat{y}$) in order to investigate the accuracy of return-range prediction of the resulting trees. We report the results in Table 3. As can be seen from the table, for the return ranges of −8, −7, 4, 5, 6, 7, 9 and 10, the average win ratios range from 53.43% to 71.78%, while the average loss ratios range from 28.22% to 39.51%. In other words, the win ratios are significantly higher than the loss ratios for signals $\hat{y} = -8, -7, 4, 5, 6, 7, 9$ and 10. This finding implies that on average, the 1000 constructed decision trees perform better in predicting these eight return ranges. For the other 13 return ranges, the average win ratio does not dominate the average loss ratio. This result implies that if a tree forecasts a return range in those 13 groups, the investor should not trade on the signal.

To summarize, we have constructed a decision tree model for return predictions using the training data, based on the "average tree". Under this model, the set of trading signals consists of $\hat{y} = -8, -7, 4, 5, 6, 7, 9$ and 10.

5.3. Out-of-sample performance of the proposed model for return prediction

Having constructed a model for return prediction based on 1000 trees in Section 5.2, we now examine its out-of-sample performance over the test period July 20, 2016−December 29, 2017.

Table 3
In-Sample Win and Loss Ratios of Classification Trees. This table reports the results on the in-sample performance of classification trees constructed using the training sample from April 30, 2012 to July 19, 2016. Inputs used are price-based 124 technical indicators. The response $y \in \{-10, -9, \dots -1, 0, 1, \dots 9, 10\}$ takes 21 values, representing 21 non-overlapping, right-open intervals of bitcoin daily returns. The values of $y = -10$ and $y = 10$ represent the return ranges of (−100%, −11.00%) and (11.00%, ∞), respectively; the interval (−11.00%, 11.00%) is divided into 19 small ones indexed by $y = -9 \dots 9$, respectively. Variable $\hat{y}$ denotes the predicted y from a tree. For any given tree, we calculate win and loss ratios of a trading strategy—where an investor is long in bitcoin if $\hat{y} > 0$, short if $\hat{y} < 0$, or takes no position (keeping cash) otherwise—based on each signal. We repeat this exercise 1000 times using 1000 independent trees. Summary statistics reported in the table are based on win and loss ratios of the trading strategy across these 1000 trees.

$\hat{y}$	−10	−9	−8	−7	−6	−5	−4
Return Range	(−100%, −11.00%)	(−11.00%, −9.00%)	(−9.00%, −7.00%)	(−7.00%, −5.00%)	(−5.00%, −3.00%)	(−3.00%, −1.00%)	(−1.00%, −0.80%)
win ratio (mean)	0.4717	0.3675	0.6450	0.6238	0.3632	0.5183	0.4458
win ratio (std)	0.1543	0.1988	0.0772	0.1150	0.0241	0.0288	0.0856
loss ratio (mean)	0.5283	0.6325	0.3550	0.3762	0.6232	0.4814	0.5542
loss ratio (std)	0.1543	0.1988	0.0772	0.1150	0.0264	0.0285	0.0856
$\hat{y}$	−3	−2	−1	0	1	2	3
Return Range	(−0.80%, −0.60%)	(−0.60%, −0.40%)	(−0.40%, −0.20%)	(−0.20%, 0.20%)	(0.20%, 0.40%)	(0.40%, 0.60%)	(0.60%, 0.80%)
win ratio (mean)	0.4388	0.3195	0.3963	0	0.5149	0.2499	0.5188
win ratio (std)	0.2172	0.0662	0.1184	0	0.0741	0.1771	0.2206
loss ratio (mean)	0.5612	0.6805	0.6003	0	0.4851	0.7501	0.4812
loss ratio (std)	0.2172	0.0662	0.1181	0	0.0741	0.1771	0.2206
$\hat{y}$	4	5	6	7	8	9	10
Return Range	(0.80%, 1.00%)	(1.00%, 3.00%)	(3.00%, 5.00%)	(5.00%, 7.00%)	(7.00%, 9.00%)	(9.00%, 11.00%)	(11.00%, ∞)
win ratio (mean)	0.6825	0.6396	0.5343	0.6104	0.2167	0.7139	0.7178
win ratio (std)	0.1694	0.0160	0.0512	0.0883	0.1660	0.1088	0.1752
loss ratio (mean)	0.3015	0.3543	0.3951	0.3896	0.7792	0.2861	0.2822
loss ratio (std)	0.1636	0.0159	0.0616	0.0883	0.1726	0.1088	0.1752

To proceed, we take those 1000 trees (constructed with the training data) as given and do not generate any new trees using any data from the test sample. For any given (constructed) tree, on a given day in the test sample, say, day i, we generate a signal $\hat{y}_{\mathrm{i}}$ (the next day return range) using the tree with the input x_{i} (the values of the 124 technical indicators on day i). Repeating this calculation for the rest of the test sample as well as the remaining trees, we obtain one set of pairs, $(X, \hat{Y})_{\mathrm{test}} = \{(x_{\mathrm{i}}, \hat{y}_{\mathrm{i}}), i = 1, \ldots, N_{test}\}$, for each of those 1000 trees.

Given these 1000 sets of $(X, \hat{Y})_{\mathrm{test}}$, we next examine the out-of-sample performance of the proposed return-range predictor/model (based on the "average tree"). Recall that under this model, an investor makes a trade only if the signal from a tree is one of those eight signals whose win ratios are significantly higher than their loss ratios over the training period. That is, the investor makes a trade only if $\hat{y} \in \{-8, -7, 4, 5, 6, 7, 9, 10\}$. More specifically, on a given day in the test period, the investor goes long on bitcoin if the signal $\hat{y} \in \{4, 5, 6, 7, 9, 10\}$, takes a short position in bitcoin if $\hat{y} \in \{-8, -7\}$, or maintains a cash position otherwise.

For comparison, we also consider ten simple, naive trading strategies based on different cutoff levels of signal $\hat{y}$. Let $y_{\mathrm{cutoff}} \in \{1, \ldots, 10\}$ denote the cutoff level of the bitcoin daily return range. The simple strategy based on y_{cutoff} is defined as follows: an investor takes a long position in bitcoin if $\hat{y} \geq y_{\mathrm{cutoff}}$, a short position in bitcoin if $\hat{y} \leq -y_{\mathrm{cutoff}}$, or no position (keeping cash) otherwise, where $\hat{y} \in \{-10, -9, \ldots, 9, 10\}$. In addition, we consider a buy-and-hold strategy.

Table 4 reports the results on the out-of-sample performance of the decision tree model-based trading strategy and the ten simple strategies. Specifically, reported variables include the average annualized volatility, average $\overline{r}_G$, average $\overline{r}_A$, average information ratio based on $\overline{r}_G$ or $\overline{r}_A$, average win and loss ratios, and average win-to-loss ratio, for each of the eleven trading strategies. Note that the averages reported in Table 4 are all based on those 1000 trees constructed in Section 5.2. For example, consider $\overline{r}_G$, the annualized geometric average return. Given a simple trading strategy based on, say, $y_{\mathrm{cutoff}} = 4$, we obtain this strategy's $\overline{r}_G$ in the test period for each of the 1000 trees. The average of these 1000 $\overline{r}_G$s is the "avg. $\overline{r}_G$" for the strategy reported in Table 4.

As can be seen from the table, all ten simple strategies have positive average returns but have different average information ratios, win and loss ratios, and win-to-loss ratios. Under the information ratio, the strategy based on $y_{\mathrm{cutoff}} = 4$ is the best among the ten simple strategies, whose average information ratios based on the geometric and arithmetic average returns are 2.77 and 4.20, respectively. However, this strategy's average win to loss ratio (about 1.30) is lower than that of the buy-and-hold strategy (about 1.38 as noted in Section 5.1).

Table 4

Out-of-Sample Performance of the Proposed Return Prediction Model. This table reports the results on the out-of-sample performance of the proposed model for narrow ranges of bitcoin daily returns over the test period July 20, 2016–December 29, 2017. The model is constructed, based on 1000 independent classification trees generated using training data over the period April 30, 2012–July 19, 2016. For comparison, also reported are the results on the performance of ten simple strategies based on different cutoff levels (y_{cutoff}) of the bitcoin daily return range, where $y_{\mathrm{cutoff}} \in \{1, \ldots, 10\}$. Under a given simple strategy based on y_{cutoff}, an investor is long in bitcoin if $\hat{y} \geq y_{\mathrm{cutoff}}$, short if $\hat{y} \leq -y_{\mathrm{cutoff}}$, or takes no position (keeping cash) otherwise, where the predicted response $\hat{y} \in \{-10, -9, \ldots, 0, \ldots, 9, 10\}$ and is an index of 21 non-overlapping, right-open intervals (ranges) of bitcoin daily returns specified in Section 3.1. For instance, $\hat{y} = -10$ and 10 represent the return ranges of (−100%, −11.00%) and (11.00%, ∞), respectively. Returns $\overline{r}_G$ and $\overline{r}_A$ are annualized geometric and arithmetic averages of daily returns, respectively. The mean values reported in the table are all based on the 1000 trees constructed using the training data. For example, we obtain $\overline{r}_G$ under the proposed model for each tree and report the average of values of $\overline{r}_G$ from those 1000 trees.

		Simple strategies based on different cutoff levels									
		y_{cutoff}									
	The proposed model	1	2	3	4	5	6	7	8	9	10
Avg. volatility (%)	64.230	81.891	80.922	79.605	78.800	76.157	51.717	40.007	36.285	26.667	20.853
Avg. $\overline{r}_G$ (%)	577.898	116.980	131.354	196.863	217.847	199.396	3.713	63.169	36.961	17.102	4.308
Avg. info ratio based on $\overline{r}_G$	8.985	1.431	1.626	2.477	2.770	2.632	0.083	1.575	1.009	0.667	0.239
Avg. $\overline{r}_A$ (%)	842.024	201.225	218.615	304.502	330.182	297.383	18.476	76.601	46.168	21.335	6.728
Avg. info ratio based on $\overline{r}_A$	13.132	2.461	2.705	3.830	4.196	3.920	0.368	1.906	1.259	0.822	0.347
Win ratio (mean)	0.306	0.498	0.474	0.453	0.443	0.419	0.150	0.088	0.048	0.026	0.014
Loss ratio (mean)	0.179	0.426	0.398	0.354	0.341	0.319	0.139	0.064	0.037	0.020	0.010
Win/Loss ratio (mean)	1.709	1.169	1.191	1.280	1.299	1.313	1.079	0.125	1.297	1.300	1.400

On the other hand, the proposed model for return prediction has an average of 577.90% for $\overline{r}_G$ and an average of 8.99 for the $\overline{r}_G$-based information ratio. The model/strategy has an average return of 842.02% for $\overline{r}_A$ and an average of 13.13 for the $\overline{r}_A$-based information ratio. Note that regardless of whether the geometric or arithmetic average returns are used, the average information ratio of the proposed model is notably higher than either those of ten simple strategies or that of the buy-and-hold strategy (which is about 8.65 based on $\overline{r}_G$ and 12.62 based on $\overline{r}_A$, as shown in Table 1). It is also worth noting that the proposed model/strategy has an average annualized volatility of 64.23%, much lower than that of the buy-and-hold strategy (83.17%, as shown in Table 1).

Moreover, although the average win ratio of the proposed model (0.306) is lower than those of the five simple strategies based on $y_{\text{cutoff}} = 1, \ldots, 5$, the average loss ratio of the proposed model/strategy (0.179) is substantially lower than its counterpart for each of the five simple strategies. As a result, the average win-to-loss ratio of the tree-based strategy is 1.71, which is substantially higher than those of the ten simple strategies or that of the buy-and-hold strategy (about 1.38 as noted in Section 5.1).

To summarize, the results in this section provide evidence suggesting that the proposed decision tree model has strong predictive power for bitcoin daily return ranges. The results also show that the proposed model dominates the buy-and-hold and simple strategies under either the information ratio or the win-to-loss ratio.

6. Conclusion

There has been a long-standing debate over the effectiveness of technical trading rules in financial markets in the academic literature. The evidence in this literature so far has been based on the markets where asset prices are driven at least partially by fundamental factors. In this study, we consider bitcoin whose value is believed to be driven factors other than fundamental factors. Specifically, we construct a tree-based model for bitcoin return prediction using the decision tree classification method with 124 price-based technical indicators.

We find that the proposed model has strong (out-of-sample) predictive power for narrow ranges of bitcoin daily returns. For example, the model outperforms the buy-and-hold strategy even in a strong bull market. As such, we provide evidence suggesting that technical analysis is useful in a market like bitcoin whose value is mainly driven by by non-fundamental factors. As a result, this study contributes to both the new literature on cryptocurrency and the literature on technical analysis.

As one of the first applications of technical analysis to bitcoin, we focus on the predictability of narrow ranges of bitcoin returns (a discrete response) in this study. One straightforward extension of this analysis is to consider a continuous response and examine the predictive power of technical trading rules for bitcoin return levels. We leave this to future inquiry.

Conflicts of interest

All authors have none to declare.

Appendix A. Technical indicators used

This appendix provides a complete list of 124 technical indicators used in the empirical analysis conducted in this study. The list is divided into five categories (as in Benediktsson *et al*.[24]) in order for the reader to see the main idea of these indicators.

A.1 Overlap Studies

BBAND WIDTH, BBAND UPPER SIGNAL, BBAND LOWER SIGNAL: Bollinger Bands
DEMA: Double Exponential Moving Average
EMA: Exponential Moving Average
H TRENDLINE: Hilbert Transform - Instantaneous Trendline
KMAM: Kaufman Adaptive Moving Average
MIDPOINT: MidPoint over period
MIDPRICE: Midpoint Price over period

SAR: Parabolic SAR
SAREXT: Parabolic SAR - Extended
SMA3, SMA5, SMA10, SMA20: Simple Moving Average T: Triple Exponential Moving Average (T3)
TEMA: Triple Exponential Moving Average
TRIMA: Triangular Moving Average
WMA: Weighted Moving Average

A.2 Momentum Indicators

ADX14, ADX20: Average Directional Movement Index
ADXR: Average Directional Movement Index Rating
APO: Absolute Price Oscillator
AROONOSC: Aroon Oscillator
BOP: Balance of Power
CCI3, CCI5, CCI10, CCI14: Commodity Channel Index
CMO: Chande Momentum Oscillator
DX: Directional Movement Index
MACD, MACDSIGNAL, MACDHIST: Moving Average Convergence/Divergence
MINUS_DI: Minus Directional Indicator
MINUS_DM: Minus Directional Movement
MOM1, MOM3, MOM5, MOM10: Momentum
PLUS DI: Plus Directional Indicator
PLUS DM: Plus Directional Movement
PPO: Percentage Price Oscillator
ROC: Rate of change: ((price/prevPrice)-1) * 100
ROCP: Rate of change Percentage: (price-prevPrice)/prevPrice
ROCR: Rate of change ratio: (price/prevPrice)
ROCR100: Rate of change ratio 100 scale: (price/prevPrice) * 100
RSI5, RSI10, RSI14: Relative Strength Index.
SLOWK, SLOWD: Stochastic
FASTK, FASTD: Stochastic Fast
TRIX: 1-day Rate-Of-Change (ROC) of a Triple Smooth EMA
ULTOSC: Ultimate Oscillator
WILLR: Williams' %R

A.3 Volatility Indicators

ATR: Average True Range
NATR: Normalized Average True Range
TRANGE: True Range

A.4 Pattern Recognition

CDL2CROWS: Two Crows
CDL3BLACKCROWS: Three Black Crows
CDL3INSIDE: Three Inside Up/Down
CDL3LINESTRIKE: Three-Line Strike
CDL3OUTSIDE: Three Outside Up/Down
CDL3STARSINSOUTH: Three Stars in The South
CDL3WHITESOLDIERS: Three Advancing White
DLABANDONEDBABY: Abandoned Baby
CDLADVANCEBLOCK: Advance Block

CDLBELTHOLD: Belt-hold
CDLBREAKAWAY: Breakaway
CDLCLOSINGMARUBOZU: Closing Marubozu
DLCONCEALBABYSWALL: Concealing Baby Swallow
CDLCOUNTERATTACK: Counterattack
CDLDARKCLOUDCOVER: Dark Cloud Cover
DLDOJI: Doji
CDLDOJISTAR: Doji
DLDRAGONFLYDOJI: Dragonfly Doji
DLENGULFING: Engulfing Pattern
CDLEVENINGDOJISTAR: Evening Doji Star
CDLEVENINGSTAR: Evening Star
CDLGAPSIDESIDEWHITE: Up/Down-gap side-by-side white lines
CDLGRAVESTONEDOJI: Gravestone Doji
CDLHAMMER: Hammer
CDLHANGINGMAN: Hanging Man
CDLHARAMI: Harami Pattern
DLHARAMICROSS: Harami Cross Pattern
CDLHIGHWAVE: High-Wave Candle
CDLHIKKAKE: Hikkake Pattern
CDLHIKKAKEMOD: Modified Hikkake Pattern
CDLHOMINGPIGEON: Homing Pigeon
CDLIDENTICAL3CROWS: Identical Three Crows
CDLINNECK: In-Neck Pattern
DLINVERTEDHAMMER: Inverted Hammer
DLKICKING: Kicking
CDLKICKINGBYLENGTH: Kicking - bull/bear determined by the longer marubozu
CDLLADDERBOTTOM: Ladder Bottom
CDLLONGLEGGEDDOJI: Long Legged Doji
CDLLONGLINE: Long Line Candle
CDLMARUBOZU: Marubozu
CDLMATCHINGLOW: Matching Low
CDLMATHOLD: Mat Hold
CDLMORNINGDOJISTAR: Morning Doji Star
CDLMORNINGSTAR: Morning Star
CDLONNECK: On-Neck Pattern
CDLPIERCING: Piercing Pattern
DLRICKSHAWMAN: Rickshaw Man
CDLRISEFALL3METHODS: Rising/Falling Three Methods
CDLSEPARATINGLINES: Separating Lines
CDLSHOOTINGSTAR: Shooting Star
CDLSHORTLINE: Short Line Candle
CDLSPINNINGTOP: Spinning Top
CDLSTALLEDPATTERN: Stalled Pattern
CDLSTICKSANDWICH: Stick Sandwich
CDLTAKURI: Takuri (Dragonfly Doji with very long lower shadow)
CDLTASUKIGAP: Tasuki Gap
CDLTHRUSTING: Thrusting Pattern
DLTRISTAR: Tristar Pattern
CDLUNIQUE3RIVER: Unique 3 River
CDLUPSIDEGAP2CROWS: Upside Gap Two Crows
CDLXSIDEGAP3METHODS: Upside/Downside Gap Three Methods

A.5 Cycle Indicators

HT DCPERIOD: Hilbert Transform - Dominant Cycle Period HT
DCPHASE: Hilbert Transform - Dominant Cycle Phase HT
TRENDMODE: Hilbert Transform - Trend vs Cycle Mode

References

1. Nakamoto S. *Bitcoin: A Peer-to-Peer Electronic Cash System*; 2008. https://bitcoin.org/bitcoin.pdf.
2. Harvey C. *Bitcoin Myths and Facts*. Working Paper. Duke University; 2014. Available at: http://ssrn.com/abstract=2479670.
3. Athey S, Parashkevov I, Sarukkai V, Xia J. *Bitcoin Pricing, Adoption, and Usage: Theory and Evidence*. Working Paper. Stanford University; 2016.
4. Pagnotta E, Buraschi A. *An Equilibrium Valuation of Bitcoin and Decentralized Network Assets*; 2018. Available at: SSRN: https://ssrn.com/abstract=3142022.
5. Raskin M, Yermack D. *Digital Currencies, Decentralized Ledgers, and The Future of Central Banking*. Working Paper. National Bureau of Economic Research; 2016.
6. Yermack D. Corporate governance and blockchains. *Rev Finance*. 2017;21(1):7−31.
7. Easley D, O'Hara M, Basu S. *From Mining to Markets: The Evolution of Bitcoin Transaction fees*. Working Paper. Cornell University; 2017.
8. Huberman G, Leshno JD, Moallemi CC. *Monopoly without a Monopolist: An Economic Analysis of the Bitcoin Payment System*. Columbia Business School Research; 2017. Paper No. 17−92.
9. Harvey C. *Cryptofinance*. Working Paper. Duke University; 2016. http://ssrn.com/abstract=2438299.
10. Brock W, Lakonishok J, LeBaron B. Simple technical trading rules and the stochastic properties of stock returns. *J Financ*. 1992;47(5):1731−1764.
11. Bessembinder H, Chan K. Market efficiency and the returns to technical analysis. *Financ Manag*. 1998;27(2):5−17.
12. LeBaron B. Technical trading rule profitability and foreign exchange intervention. *J Int Econ*. 1999;49(1):125−143.
13. Sullivan R, Timmermann A, White H. Data-snooping, technical trading rule performance, and the bootstrap. *J Financ*. 1999;54(5):1647−1691.
14. Han Y, Yang K, Zhou G. A new anomaly: the cross-sectional profitability of technical analysis. *J Financ Quant Anal*. 2013;48(5):1433−1461.
15. Shynkevich A. Performance of technical analysis in growth and small cap segments of the US equity market. *J Bank Financ*. 2012;36(1):193−208.
16. Neely CJ, Rapach DE, Tu J, Zhou G. Forecasting the equity risk premium: the role of technical indicators. *Manag Sci*. 2014;60(7):1772−1791.
17. Huang J-Z, Huang ZJ. *Testing Moving-Average Trading Strategies on ETFs*; 2018. Available at: https://doi.org/10.2139/ssrn.3138690.
18. Allen F, Karjalainen R. Using genetic algorithms to find technical trading rules. *J Financ Econ*. 1999;51(2):245−271.
19. Brown S, Goetzmann W, Kumar A. The Dow theory: William Peter Hamilton's track record reconsidered. *J Financ*. 1998;53(4):1311−1333.
20. Lo A, Mamaysky H, Wang J. Foundations of technical analysis: computational algorithms, statistical inference, and empirical implementation. *J Financ*. 2000;55(4):1705−1765.
21. Hsu P-H, Hsu Y-C, Kuan C-M. Testing the predictive ability of technical analysis using a new stepwise test without data snooping bias. *J Empir Financ*. 2010;17(3):471−484.
22. Hsu P-H, Taylor MP, Wang Z. Technical trading: is it still beating the foreign exchange market? *J Int Econ*. 2016;102:188−208.
23. Wiki. *Bitcoin*; 2017. https://en.wikipedia.org/wiki/Bitcoin/#cite_note-150.
24. Benediktsson J, Other Contributors. *TA-lib Indicators*; 2017. https://mrjbq7.github.io/ta-lib/doc_index.html.
25. Appel G. *Technical Analysis: Power Tools for Active Investors*. FT Press; 2005.
26. Wilder JW. *New concepts in technical trading systems*. Trend Research; 1978.
27. Kaufman PJ. *Smart Trading: Improving Performance in Changing Markets*. McGrow-Hill; 1996.
28. Twomey B. *Schaff Trend: A Faster and More Accurate Indicator*; 2018. https://www.investopedia.com/articles/forex/10/schaff-trend-cycle-indicator.asp.
29. Lambert D. Commodity channel index: tool for trading cyclic trends. *Stocks Commod*. 1980;1(5):120−122.
30. Chande TS, Kroll S. *The New Technical Trader: Boost Your Profit by Plugging into the Latest Indicators*. John Wiley & Sons Inc; 1994.
31. Wiki. *Trix (Technical Analysis)*; 2018. https://en.wikipedia.org/wiki/Trix/_(technical_analysis).
32. Wiki. *Williams %R*; 2018. https://en.wikipedia.org/wiki/Williams_%25R.
33. Bollinger J. *Bollinger on Bollinger Bands*. New York: McGraw-Hill; 2002.
34. Kaufman SK, Chaikin M. The use of price-volume crossover patterns in technical analysis. *MTA J*. 1991;37:35−41.
35. Dorsey D. Refining the relative volatility index. *Stocks Commod*. 1995;13:388−391.
36. Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York, NY: Springer; 2008.
37. Kuhn M, Johnson K. *Applied Predictive Modeling*. New York, NY: Springer; 2013.
38. Wiki. Decision tree. https://en.wikipedia.org/wiki/Decision_tree; 2017.
39. Quinlan JR. Induction of decision trees. *Mach Learn*. 1986;1(1):81−106.
40. Wiki. *ID3 Algorithm*; 2017. https://en.wikipedia.org/wiki/ID3_algorithm.

41. Quinlan JR. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc; 1993.
42. Quinlan JR. *C5.0:*; 2004. https://www.rulequest.com/see5-info.html.
43. Breiman L, Friedman JH, Olshen RA, Stone CJ. *Classification and Regression Trees*. Wadsworth International Group; 1984.
44. Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-learn: machine learning in Python. *J Mach Learn Res*. 2011;12:2825−2830.