



UNIVERSITY OF AMSTERDAM

Ensemble Voting for a high-frequency cryptocurrency trading strategy

July 1, 2021

Student:

Jacco Broere

Supervisor:

...

Abstract

The cryptocurrency market has rapidly grown in recent times, presenting an interesting opportunity for algorithmic trading. This study examines the profitability of algorithmic trading strategies devised through machine learning and combines them into an Ensemble Voting strategy. The models are tested and validated over two subsets of data during 2017-2018 and 2020-2021 with 5-, 15-, and 30-minute data frequencies. For each frequency, the most important features are selected through an iterative procedure called Boruta. The performance of these trading strategies is measured against a Buy-and-Hold strategy to find whether algorithmic strategies can beat the market when considering 0.1% transaction costs and when neglecting them. Our analysis shows that while the strategies can generate high returns without transaction fees, they cannot be shown to produce significantly higher returns than the Buy-and-Hold strategy when transaction costs are imposed.

Statement of Originality

This document is written by Jacco Broere who declares to take full responsibility for the contents of this document. I declare that the text and the work presented in this document are original and that no sources other than those mentioned in the text and its references have been used in creating it. The Faculty of Economics and Business is responsible solely for the supervision of the work, not for the contents.

Contents

1	Introduction	4
2	Theoretical Framework	5
2.1	Cryptocurrency characteristics	5
2.1.1	Functionality as a currency	5
2.1.2	Volatility	6
2.2	Cryptocurrency trading	7
2.2.1	Prediction strategies	7
2.2.2	Performance and returns	8
2.3	Feature engineering	9
3	Methodology	10
3.1	Data collection and preprocessing	10
3.1.1	Feature Selection and Engineering	11
3.2	Mathematical framework	13
3.2.1	Problem statement	13
3.2.2	Random Forest	15
3.2.3	Logistic Regression	16
3.2.4	Support Vector Machine	16
3.3	Backtesting	17
4	Results	18
5	Conclusion	22
	References	24
	Appendix	27

1 Introduction

Cryptocurrencies find their roots in 2008 when a pseudonymous developer Satoshi Nakamoto describes the first cryptocurrency in the original Bitcoin whitepaper (Nakamoto, 2019). Nakamoto proposed a distributed digital ledger, based on technology that is now often referred to as a blockchain. A blockchain is a decentralized network based on cryptographic protocols, enabling cryptocurrencies to be used and trusted as a payment system without the need for a central authority. Bitcoin and many modern cryptocurrencies are built around the core technology of blockchain, often using modifications and extensions (Yaga et al., 2018). For instance, a decrease in block time is a modification that is often seen in newer cryptocurrencies, resulting in faster transaction speeds compared to Bitcoin, which is frequently regarded as slow.

Cryptocurrencies were initially introduced to solve problems that were inherent to traditional fiat currencies, such as the need for a central authority. However, investors and economists often regard cryptocurrencies more as interesting investment assets, due to their speculative and volatile nature (Yermack, 2015). These considerations helped cryptocurrencies gain traction among investors and academics alike, reflected by the rapid growth of the market in cryptocurrencies, e.g., the price of Bitcoin recently surpassed the \$50.000 barrier (La Roche, 2021).

Although the advent of machine learning technology has provided promising opportunities for trading strategies such as High-Frequency Trading (HFT), research on these strategies is mostly being conducted in conventional financial markets. Cryptocurrencies present an interesting parallel to the financial market, moreover, modern technology makes HFT for cryptocurrencies accessible through freely available exchange APIs, e.g., Binance and Bitfinex. Nonetheless, similar literature on the prediction of cryptocurrency prices at high frequency is much more sparse. Fang et al. (2020) found that 85% of research on cryptocurrency price prediction emerged since 2018, indicating that this is still a novel field. They categorized this research into six distinct aspects of cryptocurrency trading, one of which is at focus in this study: the emergent trading strategies using machine learning.

The strategies in HFT revolve around numerical algorithms to predict price movements. These algorithms exploit computing power to make split-second trading decisions, far quicker than humans would ever be able to, driving the increase in their popularity. JP Morgan estimated that during peak levels of HFT in 2009, algorithms performed 61% of trading on the US equity market (Cheng, 2017). Moreover, an influx of media attention is directed at the success of HFT strategies during periods of market turmoil (Wigglesworth, 2019). This sparked debate

about whether the widespread use of HFT strategies is healthy for the financial markets on which they are working. Critics express their concern by arguing that HFT increases volatility and systemic risk (Biais, Woolley, et al., 2011). On the other hand, supporters argue that it increases market liquidity and helps price discovery (Brogaard et al., 2014).

Most research on cryptocurrency price prediction is conducted using low-frequency time horizons, such as a daily frequency. Instead, this study aims to contribute to the understudied field of HFT strategies for cryptocurrencies, by using sets of high-frequency data to create distinct algorithmic trading strategies and a combination of them in the form of an Ensemble Voting (EV) strategy (Dietterich, 2000). The raw minute-level price data is freely obtained using the API from the cryptocurrency exchange Bitfinex. This study considers data from recent and earlier time periods, to determine whether results obtained in foregoing years continue to hold using recent data, now that the cryptocurrency market has evolved in many ways. The final trading strategies are compared to a Buy-and-Hold (B&H) strategy on a set of evaluation metrics and their profitability. Unlike similar research, this study tries to resemble a legitimate trading environment by considering realistic transaction costs. The main objective of this study is to determine whether an EV classifier can make high-frequency price movement predictions on cryptocurrencies accurately enough to result in a trading strategy that can outperform a B&H strategy in the current market.

The remainder of this study is as follows. The Theoretical Framework begins by reviewing academic literature on the cryptocurrency market and its characteristics, then it focuses primarily on the current research in the field of HFT for cryptocurrencies. The Methodology describes the data set and establishes the statistical framework used throughout this study. The Results section presents the optimized models and relates their performance against the benchmark strategy. In the Conclusion, the implications and shortcomings of the findings are discussed alongside further research interests.

2 Theoretical Framework

2.1 Cryptocurrency characteristics

2.1.1 Functionality as a currency

Cryptocurrencies were introduced to address the shortcomings of regular fiat currencies by creating a decentralized, unregulated form of payment. Yermack (2015) explained how Bitcoin exploits cryptography to create a deterministic supply with a regulated growth rate that is

guarded through mathematical principles. In this manner, no government or any other authority can alter the predetermined supply of Bitcoin, preventing inflation induced by monetary policy from governments in financial difficulty.

Dwyer (2015) provided an in-depth overview of the technical requirements for electronic money to function as a currency, and how Bitcoin achieves these features through its peer-to-peer, blockchain network. According to Yermack (2015), a well-functioning currency should satisfy three economic criteria, namely: it should serve as a medium of exchange, a store of value, and a unit of account. He discussed how Bitcoin fares in these three categories and found that Bitcoin had too few daily transactions to accommodate for an economy. Then, it becomes difficult to consider Bitcoin as a store of value and a unit of account due to its high volatility. High volatility causes the prices of goods in Bitcoin to vary strongly, even within the same day, making it difficult to express value in terms of Bitcoin. These shortcomings make Bitcoin less suited to the role of a currency and could be better regarded as a speculative investment asset (Baek & Elbeck, 2015; Dyhrberg, 2016; Yermack, 2015).

2.1.2 Volatility

When developing a trading strategy, it can be helpful to consider the determinants and magnitude of the volatility of cryptocurrencies. Dwyer (2015) found cryptocurrencies to display higher volatility than regular stocks, and that the daily volatility is heavily right-skewed, indicating that many days occur where the price of cryptocurrencies is unstable. High-frequency trading tends to be more effective when the volatility of the commodity being traded is high, this is signalled by the influx of media attention directed at HFT strategies during periods of market turmoil (Wigglesworth, 2019). However, most research on cryptocurrency prediction uses low-frequency data, such as a daily frequency (Fang et al., 2020). Similar research using high-frequency data is more sparse, as HFT for cryptocurrencies is a more novel field. This might make it worthwhile to investigate whether HFT methods can be successful in the volatile cryptocurrency markets.

Baur and Dimpfl (2018) analysed asymmetric volatility effects for the 20 largest cryptocurrencies using the asymmetric threshold GARCH (TGARCH) model. They concluded positive price shocks to have a larger increasing effect on the volatility of cryptocurrencies than negative price shocks, an asymmetric effect that is in contrast with observations in the regular stock market. Furthermore, their results showed that cryptocurrencies can be characterized as highly volatile assets that display little correlation with traditional assets. These characteristics could

provide investors with promising opportunities. For instance, cryptocurrencies might function as a portfolio diversifier due to the lack of correlation with other assets, moreover, they might prove valuable in trading strategies because of their high volatility.

2.2 Cryptocurrency trading

Cryptocurrencies have garnered most of their popularity over the last couple of years. Fang et al. (2020) found that 85% of academic literature on cryptocurrency trading developed since 2018. They provided a comprehensive survey of 126 papers across six distinct research areas concerning cryptocurrency trading. This section focuses on the existing literature in the field of trading strategies using machine learning.

2.2.1 Prediction strategies

Chen et al. (2020) compared a set of statistical and machine learning models on their predictive power when applied to Bitcoin data. The set of statistical models included Logistic Regression (LR) and Linear Discriminant Analysis (LDA), while the machine learning methods consisted of Random Forest (RF), Gradient Decision Tree Boosting (GTB), Quadratic Discriminant Analysis (QDA), Support Vector Machine (SVM), and Long Short-Term Memory neural network (LSTM). Their models were fitted to both daily and 5-minute frequency data from 2017 to 2019. The 5-minute data encompassed only price data, while the daily data included also features on blockchain characteristics and gold spot price. Moreover, their final models were benchmarked against the similar work of McNally et al. (2018).

Madan et al. (2015) published one of the first academic papers on high-frequency cryptocurrency trading. They employed a Random Forest (RF), and a Generalized Linear Model (GLM) using 10-minute Bitcoin price data from the first five years after the instalment of Bitcoin in 2009. Moreover, they created RF, Binomial GLM, and Support Vector Machine (SVM) models, using daily data on 26 features concerning Bitcoin trading and blockchain characteristics. Similarly, Vo and Yost-Bremm (2020) designed a strategy by adopting an RF model using five financial indicators derived from Open, High, Low, and Close (OHLC) Bitcoin price data. These indicators convey information on the trend, momentum, volume, and volatility of the price. Additionally, the models were fitted using 30 different data frequencies, ranging from 1 minute to 90 days, consisting of data from 2015 to 2017. This allowed them to make inferences on the optimal trading horizon. Finally, they compared the optimal configurations of RF to a Deep Learning (DL) approach.

Borges and Neves (2020) took an alternative approach utilizing machine learning algorithms when constructing a trading strategy. They constructed an Ensemble Voting (EV) strategy by taking the unweighted average of four trading signals produced by individual models. The EV consists of two linear and two non-linear models; LR and SVM, GTB, and RF respectively. The hyperparameters of these algorithms are optimized using a grid search with negative log-loss as the scoring metric, except for the SVM, which cannot produce this metric naturally. They argued that negative log-loss, which also considers the probability estimate along with the predicted label, is suitable because investment strategies benefit from predicting the correct price movement with high confidence. The data used in their study consists of 1-minute data on 100 cryptocurrency pairs, traded on Binance from 2017 until 2018. The EV and individual models were fitted to both the original data and various resampled versions of the original data. In line with this approach, Sebastião and Godinho (2021) constructed multiple EV strategies consisting of 6 models, namely the regression and classification counterparts of a linear regression model based on Ordinary Least Squares (OLS), RF, and SVM. Three EV strategies were constructed by requiring four, five, or six of the individual models to agree on a trading decision. Instead of using high-frequency data, Sebastião and Godinho (2021) opted to use daily data from 2015-2019, combining trading information and blockchain characteristics.

2.2.2 Performance and returns

Chen et al. (2020) found that the final machine learning models achieved an average accuracy of 62.2%, significantly higher than the 53.0% average accuracy of the statistical methods. The best performing machine learning model was a Long Short-Term Model (LSTM), which achieved an accuracy of 67.2%, outperforming the benchmark LSTM model created by McNally et al. (2018). Neither study reports on the profitability of the final strategies, nor do they consider transaction costs. The same holds for Madan et al. (2015), they do not report on the final profitability of their RF and GLM strategies. They discovered the RF model to outperform the GLM model in terms of sensitivity, specificity, precision, and accuracy, with the RF model obtaining an accuracy of 57.4%. Both Chen et al. (2020) and Madan et al. (2015) concluded that statistical methods performed better when applied to low-frequency data, while machine learning methods performed better when using high-frequency data.

Vo and Yost-Bremm (2020) measured the performance of their models not only by evaluation metrics on the movement predictions, but also report on the returns of their trading strategies. Returns are calculated by simulating a trading environment using the out-of-sample test data.

Their findings show that the RF model performs best when using a 15-minute data frequency and outperformed its DL counterpart. However, they did not go into depth on the exact structure of the DL model, making it difficult to judge their RF model on its comparative performance. The optimal RF model attained an F1-score of 97.5% on both buy and sell decisions and yielded 657.45% annualized average returns over 3 years with a Sharpe Ratio of 8.22, transaction costs not considered. Furthermore, their findings showed that there is a significant drop in accuracy when the frequency is increased beyond 30-minute intervals.

Borges and Neves (2020) compared their EV and individuals models to a benchmark B&H strategy. Their average results for all markets and resampling methods showed that the EV classifier reported a return on investment of 615% over the period 2017-10-30 to 2018-10-30 and that it outperformed the individual models in most metrics, indicating that combining machine learning methods can be valuable for trading strategies. The individual models were also shown to outperform a B&H strategy in terms of most evaluation metrics. They noted that the profitability of their strategies is likely excessive, attributed to the fact that there is an absence of a bid-ask spread in the data.

2.3 Feature engineering

Machine learning models are trained on features, also known as explanatory variables, to predict the outcome of the target variable. Hawkins (2004) discussed a myriad of problems, such as overfitting and slow training speed, that arise when too many features are included in a predictive model. This raises questions about which features to include when developing a machine learning model. Kursa et al. (2010) designed Boruta, a novel feature selection algorithm. The method iteratively determines what features bring enough explanatory power to the model by comparing them to randomly generated features. They showed that this approach is statistically grounded and allows for a more efficient selection of relevant features. Chen et al. (2020) and McNally et al. (2018) utilized Boruta to discern the most important features for their cryptocurrency price prediction models. On the other hand, studies that did not rely on a numerical method often used existing results or common practices from financial literature to decide on what features to include (Borges & Neves, 2020; Nakano et al., 2018; Vo & Yost-Bremm, 2020).

3 Methodology

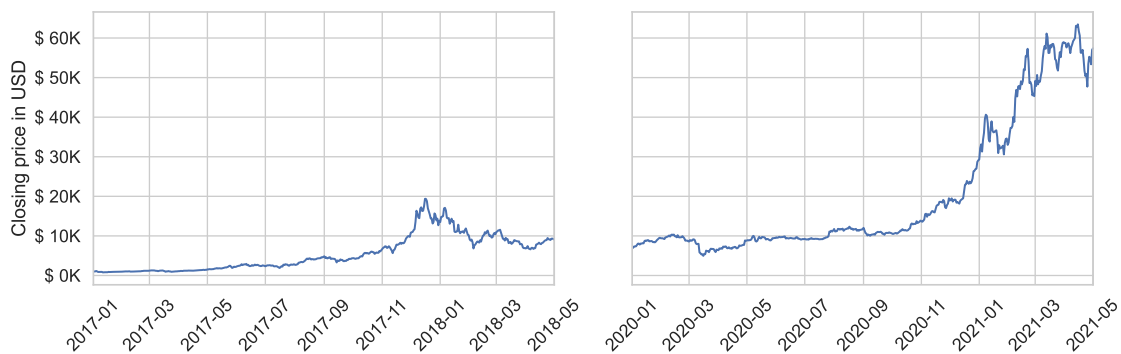
3.1 Data collection and preprocessing

The data is collected from the cryptocurrency exchange Bitfinex, where historical data on the currency pair BTC/USD from 2013-04-27 to 2021-05-03 is extracted at the 1-minute level, the highest granularity provided by the Bitfinex API. The 1-minute data set is resampled to obtain data of 5- and 15-minute frequencies. It should be noted that it is also possible to query lower-frequency data directly from the API, however, resampling is more efficient and allows for any desired data frequency longer than 1-minute. The OHLC prices are significant up to 5 digits. Furthermore, a row from the candlestick data extracted through the API has six columns and is structured as follows;

(Unix timestamp - Open - Close - High - Low - Volume).

The raw data consists of data from 2013-04-27 to 2021-05-03, which is separated into two subsets of data, the first period concerns 2017-01-01 to 2018-05-01, while the second regards 2020-01-01 to 2021-05-01. These subsets are used to compare the profitability of the final trading strategies during two different epochs, and to see whether the market has become more efficient over the years. The closing price of Bitcoin over these intervals is displayed in Figure 1 and Table 1 contains descriptive statistics on the raw data and the data subsets.

Figure 1: Closing price of Bitcoin in both subsets



In the case of a missing value, the complete row for the concerning timestamp is missing in this data. A missing value occurs when the API failed to record data at the time. There are many missing values in the early years of interval, during this time a missing value often means there were simply no trades on the exchange at that point in time. Missing values are a lot less prevalent in recent years since cryptocurrencies are now continuously being traded. To

Table 1: Descriptive statistics on raw data and data subsets at 1-minute frequency

<i>Sample period</i>	27 April 2013 – 3 May 2021	1 January 2017 – 1 May 2018	1 January 2020 – 1 May 2021
# 1-min observations	4,216,320	698,401	699,841
# missing observations	909,520	21,178	10,904
<i>Closing price (USD)</i>			
Mean	5,947.80	5,398.52	20,209.23
Standard deviation	10,153.35	4,398.28	17,094.83
Maximum	64,787.76	19,891.00	64,787.76
Minimum	63.04	744.48	4,030.10
<i>Volume (BTC)</i>			
Total volume	63,251,207.14	19,881,380.68	4,237,735.74
Mean volume per minute	14.98	28.14	6.05

impute the missing data, forward fill imputation is used, which means the observation from the preceding timestamp is copied to the current one, similar to Vo and Yost-Bremm (2020).

3.1.1 Feature Selection and Engineering

Boruta (Kursa et al., 2010), was used to select the features to be included in the models. The procedure starts by generating copies of all features, and shuffling them to remove all correlation with the dependent variable. That is, the $n \times k$ matrix X , containing n observations and k features, is extended to the $n \times 2k$ matrix $\tilde{X}$. Let x_i , $\tilde{x}_i$ denote the column vectors of their respective matrices, then the column vectors $\tilde{x}_i$ are constructed as follows using the random uniform shuffling function $v(\cdot)$,

$$\tilde{x}_i = \begin{cases} x_i & \text{for } i \in \{1, \dots, k\}, \\ v(x_{i-k}) & \text{for } i \in \{k+1, \dots, 2k\}. \end{cases} \quad (1)$$

The resulting matrix $\tilde{X}$ is used to train an RF classifier for a predetermined number, m , iterations and at each iteration, j , the feature importance scores; Z_{ij} for $i \in \{1, \dots, k\}$, are compared to the maximum feature importance scores of the random features; $Z_j^{\max} = \max\{Z_{k+1j}, \dots, Z_{2kj}\}$. Let $I(\cdot)$ and $E(\cdot)$ denote the indicator function and the expected value operator, respectively, and $l \leq m$ is the current number of iterations. If we assume the following

under H_0 ;

$$W_i = \sum_{j=1}^l I(Z_{ij} > Z_j^{\max}) \stackrel{H_0}{\sim} \text{BIN}(l, p = 0.5), \quad (2)$$

then the following hypotheses can be tested using two one-sided tests of equality at each iteration,

$$H_0 : E(W_i) = \frac{l}{2}, \quad H_a : E(W_i) > \frac{l}{2}, \quad (3)$$

$$H_0 : E(W_i) = \frac{l}{2}, \quad H_a : E(W_i) < \frac{l}{2}. \quad (4)$$

If the hypothesis in equation (3) is rejected, the feature is deemed important, if the hypothesis in equation (4) is rejected, the feature is rejected and $\tilde{x}_i$ dropped from $\tilde{X}$. When hypotheses (3) and (4) are both not rejected, the feature is flagged tentative, however, retained. After every iteration, the matrix $\tilde{X}$ is reconstructed using the remaining features and the hypothesis tests are performed again. This is repeated for the predetermined number of iterations and leaves a subset of features to be included in the model. The p -values of these tests should be corrected for multiple testing as the procedure is executed on many features simultaneously, this study employed the package BorutaPy, which utilizes the Benjamini-Hochberg correction procedure (Benjamini & Hochberg, 1995). The procedure can be made more or less conservative by changing the confidence level, α , of the test, which is set to $\alpha = 0.05$ in this study.

Boruta is employed on a feature set of 192 financial indicators available through the Pandas TA package in Python¹. Some available indicators are prevalent in existing literature for cryptocurrency trading, such as the Relative Strength Index (RSI), Stochastic Oscillator (SO), and Moving Average Convergence-Divergence (MACD), among others (Borges & Neves, 2020; Nakano et al., 2018; Vo & Yost-Bremm, 2020). Features that were still deemed tentative after a hundred iterations are decided not to be included, as a simple model is generally preferred (Hawkins, 2004). Furthermore, all continuous features are scaled using min-max normalization, that is, the normalized value of x is given by:

$$x_{\text{norm}} = \frac{x - \min x}{\max x - \min x}. \quad (5)$$

¹The list of all 192 indicators can be found here: <https://github.com/twopirllc/pandas-ta/blob/master/README.md>

3.2 Mathematical framework

3.2.1 Problem statement

Let p_t^c denote the closing price of a cryptocurrency at time t and $h \in \{1, \dots, 2 \times 7 \times 24 \times 4 = 1344\}$ the time in two weeks at 15-minute frequency. Then the price prediction problem concerns the prediction of $y_{t+h+1} \in \{0, 1\}$, where

$$y_{t+h+1} \equiv I(p_{t+h+1}^c > (1 + \tau)p_{t+h}^c). \quad (6)$$

Let τ denote the relative transaction costs, this study considers $\tau \in \{0, 0.001\}$ equivalently 0% and 0.1%. The model at time t is trained on the $n \times k$ window matrix X_t , with k features and a window size of eight weeks, i.e., 5376 observations at 15-minute frequency. The model is retrained at the end of a block of two weeks using the next window, that is, the first two weeks of data in X_t are dropped and the new two weeks of data are added to construct the next window matrix X_{t+1344} , in the case of a 15-minute frequency. The same structure is applied to 5-minute and 30-minute data, which results in window sizes of $n = 16128$ and $n = 2688$ respectively.

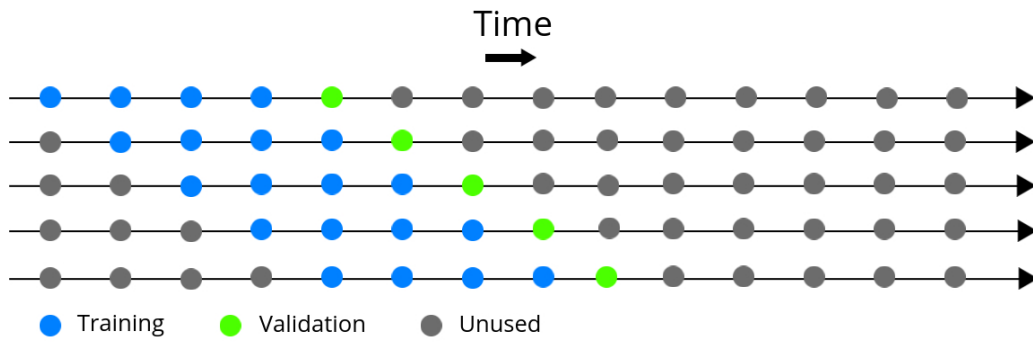
Once the model is trained on X_t , it can be deployed at time $t + h$ to predict the next movement of the closing price, $\hat{y}_{t+h+1}$. Then a trading strategy is constructed as follows; take or hold a long position if $\hat{y}_{t+h+1} = 1$, and relinquish the long position or stay out of the market if $\hat{y}_{t+h+1} = 0$. It should be noted that shorting opportunities are not available on the market from which the data has been gathered, and is therefore not considered in the trading strategy.

For all three models discussed below in sections 3.2.2–4, a grid search with cross-validation is employed to find the optimal combination of hyperparameters. Hyperparameters are settings that dictate the learning process of the algorithms and can be adjusted in an attempt to find the best configuration for the prediction problem at hand. The cross-validation procedure for the grid search is structured in conformity with the prediction strategy described at the beginning of this section. That is, in each iteration of cross-validation eight consecutive weeks are used as training data, and the ninth and tenth weeks serve as the validation set. Five iterations of this concept are illustrated in Figure 2, every dot represents a block of two weeks. The procedure is applied over the first 18 weeks of data, which means that five iterations of cross-validation are performed for each configuration of hyperparameters.

Finally, the three models with optimized configurations are combined into an EV classifier, through a process called hard-majority voting. Hard-majority voting is a voting process where

the class with the majority of votes is yielded as the final prediction. In our case, the EV classifier yields the price movement prediction that is produced by two out of the three individual models. Additionally, a soft-majority voting approach is employed for the 15-minute strategy. Soft-majority voting takes the average of the probability estimates produced by the individual models and predicts the future price movement with the highest average probability. Platt scaling is used to produce probability estimates for the SVM model discussed in section 3.2.4, as SVM models cannot produce probability estimates naturally (Platt et al., 1999).

Figure 2: Cross-validation structure for a sliding window model



The hyperparameter space used in the grid search for each of the models can be found in Table 2, the parameter space for all numeric parameters are determined through trial and error, such that the optimal outcomes are not on the borders of the parameter space. Moreover, this grid search performs its evaluation of the models based on a scoring metric. This study opts to use the precision score on buy decisions as the scoring metric for all models, as it is important to predict the positive price movements with high confidence and limit the amount of losing trades. The precision score is defined as the ratio of true positives and the sum of true positives and false positives, which can be formalized as follows:

$$pr = \frac{tp}{tp + fp}.$$

The main objective of this study is to make inferences on the following hypotheses. Let $E(R_s)$ denote the expected weekly return on investment of a certain algorithmic trading strategy, s , and let $E(R_h)$ denote the expected weekly return on investment of the B&H strategy. The hypotheses can then be formalized as follows:

$$H_0 : E(R_s) = E(R_h), \quad H_a : E(R_s) > E(R_h), \quad (7)$$

which can be tested using the Wilcoxon Signed-Rank test on the series of weekly returns of the

trading strategies (Wilcoxon, 1992). This test is performed for 48 hypotheses, each hypothesis considers a trading strategy based on a different combination of time period, transaction costs, data frequency, and type of algorithm. As multiple hypotheses are tested simultaneously, the resulting p -values should be corrected for multiple testing. This study employs the Benjamini-Hochberg correction procedure (Benjamini & Hochberg, 1995), controlling the false discovery rate (FDR) at 5%.

Table 2: Subspace of values for grid search with cross-validation

<i>Algorithm</i>	<i>Hyperparameter</i>	<i>Space of values</i>
<i>Random Forest</i>	Number of trees	250
	Number of features per tree	$\sqrt{\text{num. features}}$
	Min. number of obs. to split a node	{11, 101, 501, 1001}
	Max. depth of each tree	{5, 10, 20, 50}
	Splitting measure	Gini impurity
	Scoring metric	precision
<i>Logistic Regression</i>	Regularization	l_2
	λ	{0.1, 1, 10, 100, 1000, 10000}
	Intercept	Included
	Scoring metric	precision
<i>Support Vector Machine</i>	λ	{0.1, 1, 10, 100, 1000}
	Kernel	Linear
	Scoring metric	precision

3.2.2 Random Forest

A Random Forest is an ensemble learning method consisting of a collection of individual decision trees (Breiman, 2001). The decisions made by each decision tree are aggregated to yield a final, more robust prediction. The individual decision trees are not trained on the entire data set, instead, different samples are created using Bootstrap Aggregation (bagging) (Breiman, 1996). A bootstrap sample is generated by taking random samples with replacement from the original data. Then each decision tree is trained on a different bootstrap sample, furthermore, the individual trees do not have access to the complete set of features either. Namely, in each decision tree, a random subset of features is used, forcing the individual decision trees to create different splitting criteria from each other.

3.2.3 Logistic Regression

Binomial Logistic Regression models a binary target variable, by assuming a linear relationship between the features, x_{ti} for $i \in \{1, \dots, k\}$, and the log-odds of the target variable, $y_t \in \{0, 1\}$, written in terms of the conditional probability (Hosmer Jr et al., 2013),

$$\log \left(\frac{P(y_t = 1|x_t)}{1 - P(y_t = 1|x_t)} \right) = \beta_0 + \sum_{i=1}^k \beta_i x_{ti} \equiv x_t^T \beta. \quad (8)$$

The above can be used to arrive at an expression for the conditional probability, in terms of the logistic sigmoid function, $\sigma(\cdot)$.

$$P(y_t = 1|x_t) = \frac{\exp(x_t^T \beta)}{\exp(x_t^T \beta) + 1} = \frac{1}{1 + \exp(-x_t^T \beta)} \equiv \sigma(x_t^T \beta). \quad (9)$$

The expression in equation (9) can be used to construct the negative log-likelihood of the sample $\mathcal{F}_t = \{y_1, \dots, y_t\} \cup \{x_{t1}, \dots, x_{tk}\}$,

$$-\log \mathcal{L}(\beta_0, \dots, \beta_k; \mathcal{F}_t) = - \sum_{t=1}^n \left\{ y_t \log \left(\sigma(x_t^T \beta) \right) + (1 - y_t) \log \left(1 - \sigma(x_t^T \beta) \right) \right\}. \quad (10)$$

This expression can be minimized using computerized algorithms. An l_1 or l_2 regularization function, $R(\cdot)$, can be included to reduce overfitting (Hoerl & Kennard, 1970; Tibshirani, 1996).

3.2.4 Support Vector Machine

A Support Vector Machine (SVM) (Cortes & Vapnik, 1995), is a machine learning algorithm that can be used for classification and regression tasks. SVM attempts to separate a set of data points using a hyperplane, which is defined to maximize the distance to the nearest point of each group. Let $w \in \mathbb{R}^k$ denote the normal vector of the hyperplane, and let $b \in \mathbb{R}$ be the offset from the origin. The hyperplane then corresponds to the vectors, $z \in \mathbb{R}^k$, satisfying the equation;

$$w^T z - b = 0. \quad (11)$$

Contrary to previous algorithms, SVM defines the labels of the target variable as -1 for the negative class and 1 for the positive class, such that $y_t \in \{-1, 1\}$. In optimizing the algorithm, the normal vector, w , and offset, b , are chosen to maximize the distance from the hyperplane to the data points, $x_t \in \mathbb{R}^k$ corresponding to the target variable y_t . The algorithm achieves this by maximizing the margin, which is defined as the distance between the maximum and minimum

margin hyperplane, satisfying respectively:

$$w^T z - b = 1, \quad w^T z - b = -1. \quad (12)$$

Geometrically the margin can then be shown to be equal to $\frac{2}{\|w\|}$, now maximizing the margin is clearly equivalent to minimizing $\|w\|$. However, the data is often not perfectly linearly separable, in this case, the hinge loss function, $L(y_t) = \max\{0, y_t(w^T x_t - b)\}$, can be introduced. This leads to the minimization problem with regularization parameter $\lambda \in \mathbb{R}_+$,

$$\min_{w,b} \quad \lambda \|w\|^2 + \frac{1}{n} \sum_{t=1}^n L(y_t). \quad (13)$$

This variation uses a so-called linear kernel, however, SVM can be extended easily to non-linear classification by using a different kernel, also known as a kernel trick (Bishop, 2006). This study considers only a linear kernel due to the increase in computational complexity when employing a kernel trick and the size of the data set.

3.3 Backtesting

To evaluate the algorithms on their profitability, a backtesting framework is employed using Backtrader² for Python. This framework allows us to simulate a trading environment by letting the algorithms trade on historical data. The trading environment can be customized to include transaction costs and what price the orders should be matched against, i.e., the opening price of the next time interval or the closing price of the current interval, this study uses the former as it is not realistic to trade on the closing price of an interval after it has already passed. This study compares the performance of the final strategies using different values for the relative transaction costs, namely $\tau \in \{0, 0.001\}$, with trading strategies based on g -minute frequencies, with $g \in \{5, 15, 30\}$. The performance of these models is measured using the unseen test set from both 2017/18 and 2020/21, containing the remaining 51 weeks of data that follow after the 18 weeks used in the grid search over the hyperparameters.

The framework provides a log of the trading history containing the prices at which the orders were filled, along with the net and gross profit of each trade. Where the net profit is calculated by subtracting the transaction costs of both buying and selling from the gross profit. Furthermore, it provides performance metrics on a trading strategy, s , such as annualized weekly returns, R_s^a , defined as:

$$R_s^a = \left((1 + R_s)^{52} - 1 \right), \quad (14)$$

²<https://www.backtrader.com/>

where R_s denotes the mean weekly return on investment of trading strategy s . Now, the annualized Sharpe ratio, S_s^a , can be constructed as:

$$S_s^a = \frac{R_s^a - R_f^a}{\hat{\sigma}_s^a}, \quad (15)$$

where R_f^a denotes the annual risk-free return and $\hat{\sigma}_s^a$ denotes the annualized sample standard deviation of the weekly returns. In this study, an annual risk-free return of $R_f^a = 1\%$ is used. Other metrics include the maximum drawdown (MDD), which is the maximum observed decline in portfolio value from a peak to a trough. The maximum drawdown for a period of n observations can be formalized as follows:

$$\text{MDD}_n = \max_{t \in \{1, \dots, n\}} \left(\max_{h \in \{1, \dots, t-1\}} \left(\frac{V_h - V_t}{V_t} \right) \right), \quad (16)$$

with V_t denoting the value of the portfolio of a trading strategy at time t . MDD is used as a measure of the risk involved in a trading strategy. Furthermore, the F_1 -scores on both buy and sell decisions are reported, which is defined as the harmonic mean of precision, $pr = tp/(tp + fp)$, and recall, $re = tp/(tp + fn)$, where tp, fp, tn, fn denote the number of true/false positives and negatives respectively, then the F_1 -score can be formalized as follows:

$$F_1 = \frac{2}{pr^{-1} + re^{-1}} = 2 \cdot \frac{pr \cdot re}{pr + re}. \quad (17)$$

Lastly, the win rate and the time in the market are reported by the backtesting framework, these are defined respectively as the percentage of profitable trades and the amount of time a long position is held. The win rate considers transaction costs when determining whether a trade is winning or not. That is, a trade is considered winning when it is profitable after the subtraction of the transaction costs of both buying and selling. The time in the market can be used as a measure for the aggressiveness of a trading strategy, such that a lower percentage of time spent in the market indicates that a strategy is more conservative.

4 Results

Table 3 summarizes the performance at a 15-minute data frequency of all three individual models, the ensemble classifier, and a B&H strategy when no transaction costs are considered. As shown, all algorithmic strategies outperform the B&H strategy over the test set in 2017/18. The RF model performed best in 2017/18, achieving annualized returns of 3643% and an annualized Sharpe ratio of 4.58. While RF is outperformed in terms of returns by the B&H strategy dur-

Table 3: Results of all strategies using a 15-minute frequency and no transaction fee

Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses					
<i>Algorithm</i>	B&H	RF	LR	SVM	EV
<i>Metric</i>					
Accuracy	50.54% (50.65%)	52.27% (52.77%)	51.87% (52.42%)	51.80% (52.26%)	51.83% (52.40%)
F_1 -score (Buy)	67.15% (67.24%)	50.45% (54.27%)	50.10% (54.18%)	50.23% (55.46%)	50.04% (54.97%)
F_1 -score (Sell)	0.00% (0.00%)	53.97% (51.17%)	53.51% (50.52%)	53.27% (48.56%)	53.49% (49.52%)
Precision (Buy)	50.54% (50.65%)	53.07% (53.25%)	52.62% (52.88%)	52.53% (52.57%)	52.58% (52.77%)
Annualized returns (%)	795.86% (408.34%)	571.52% (3642.87%)	295.11% (1004.26%)	213.89% (814.17%)	280.62% (803.27%)
Annualized Sharpe ratio (%)	3.51 (2.00)	4.43 (4.58)	3.33 (3.30)	2.79 (3.00)	3.22 (3.03)
Win rate (%)	N/A N/A	69.52% (64.51%)	70.38% (70.49%)	68.57% (69.73%)	69.73% (70.19%)
Maximum Drawdown (%)	29.97% (69.54%)	20.26% (38.09%)	25.35% (38.48%)	25.89% (54.74%)	25.36% (46.05%)
Time in market (%)	100% (100%)	45.67% (52.61%)	45.92% (53.16%)	46.30% (56.52%)	45.88% (55.03%)

Table 4: Results of all strategies using a 15-minute frequency and a 0.1% transaction fee

Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses					
<i>Algorithm</i>	B&H	RF	LR	SVM	EV
<i>Metric</i>					
Accuracy	34.14% (41.20%)	66.15% (59.01%)	62.99% (58.05%)	63.41% (58.03%)	63.43% (58.14%)
F_1 -score (Buy)	50.90% (58.35%)	11.93% (24.23%)	28.97% (28.83%)	26.94% (28.80%)	27.11% (28.64%)
F_1 -score (Sell)	0.00% (0.00%)	79.05% (71.91%)	74.97% (70.26%)	75.59% (70.25%)	75.59% (70.39%)
Precision (Buy)	34.14% (41.20%)	53.34% (50.80%)	42.00% (47.86%)	42.31% (47.82%)	42.43% (48.10%)
Annualized returns (%)	795.67% (407.03%)	20.90% (-33.68%)	-67.47% (-64.45%)	-66.68% (-61.38%)	-65.01% (-57.05%)
Annualized Sharpe ratio (%)	3.51 (2.00)	0.90 (-0.48)	-3.44 (-1.62)	-3.43 (-1.25)	-3.32 (-1.09)
Win rate (%)	N/A N/A	53.90% (57.05%)	44.36% (51.76%)	45.06% (52.85%)	45.48% (52.85%)
Maximum Drawdown (%)	29.97% (69.53%)	16.42% (53.42%)	62.07% (70.94%)	62.03% (73.15%)	60.80% (71.42%)
Time in market (%)	100% (100%)	4.30% (12.90%)	17.97% (17.75%)	15.94% (17.75%)	16.03% (17.46%)

ing 2020/21, it performs better than the B&H strategy in terms of Sharpe ratio and Maximum Drawdown. RF and B&H attain an annualized Sharpe ratio of 4.43 and 3.51 respectively during 2020/21. These findings suggest that the RF strategy could be considered less volatile than a B&H strategy when transaction costs are ignored.

Table 4 contains the performance metrics of the 15-minute frequency strategies when transaction costs of 0.1% are included. The table shows that the F_1 -score on the buy decision dropped once the models need to predict price movements larger than the 0.1% threshold imposed by the transaction costs. Namely, the average F_1 -score on buy decisions dropped from 52% to an average of 26%. In both time periods, all algorithmic trading strategies now attain negative returns apart from the RF strategy in 2020/21, which attained annualized returns of 20.9% and an annualized Sharpe ratio of 0.90. Moreover, in Table 6–9 in the appendix, results can be found for 5-minute and 30-minute frequencies. As shown, none of the algorithmic at 5- and 30-minute frequencies are profitable when transaction costs of 0.1% are factored in, except for the 5-minute RF strategy in 2017/18. In contrast, the 5-minute strategies are highly profitable when no transaction fees are imposed. Then, the most profitable strategy, RF, attains returns of 322932% and 1149% in 2017/18 and 2020/21 respectively. The disparity between the inclusion and exclusion of transaction costs could be attributed to the small profit margins that are

Table 5: Results of Wilcoxon Signed-Rank test for hypothesis in equation (7)

Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses				
	No transaction costs		0.1% transaction costs	
	Test statistic	p -value	Test statistic	p -value
<i>5-minute frequency</i>				
RF	499 (799)	0.377 (<0.001*)	199 (355)	1.000 (0.923)
LR	298 (684)	0.983 (0.005*)	186 (343)	1.000 (0.942)
SVM	405 (657)	0.794 (0.013*)	201 (332)	0.999 (0.956)
EV	432 (687)	0.690 (0.005*)	212 (353)	0.999 (0.926)
<i>15-minute frequency</i>				
RF	430 (612)	0.698 (0.047)	229 (300)	0.998 (0.982)
LR	333 (464)	0.955 (0.543)	142 (259)	1.000 (0.995)
SVM	316 (490)	0.971 (0.419)	154 (261)	1.000 (0.995)
EV	320 (473)	0.968 (0.500)	158 (265)	1.000 (0.994)
<i>30-minute frequency</i>				
RF	414 (487)	0.762 (0.433)	184 (279)	1.000 (0.990)
LR	406 (401)	0.791 (0.649)	191 (317)	1.000 (0.970)
SVM	464 (391)	0.543 (0.839)	199 (283)	1.000 (0.989)
EV	463 (399)	0.548 (0.814)	191 (308)	1.000 (0.977)
<i>Note:</i>		<i>*significant after Benjamini-Hochberg correction</i>		

exploited by high-frequency strategies. These margins are abolished by the 0.1% transaction fee, often resulting in an unprofitable strategy.

In Table 4, it is noteworthy that compared to the other three algorithmic trading strategies, RF is more than three times as conservative, spending only 4.3% of time in the market in 2020/21 as opposed to an average of 16.6% among the other three strategies. The results also show that after the introduction of transaction costs, the models have become more conservative, namely, the average time spent in the market has dropped from 48% to 15% across both time periods and each of the frequencies. Table 10 and 11 in the appendix contain the results of using soft-majority voting at 15-minute frequency. Table 11 shows that the EV strategy now attains returns of 118% in 2017/18 with 0.1% transaction costs, in contrast with the negative returns of the hard-majority voting EV strategy in the same period. Furthermore, the table shows that the soft voting EV strategy has become more conservative, spending 13% and 5% of time in the market during 2017/18 and 2020/21 respectively.

Boruta selected 48, 40, and 30 features for the 5-, 15-, and 30-minute data frequencies respectively. Figure 3 and 4 in the appendix show the importance scores for the features that were selected by Boruta at 15-minute frequency, for the inclusion and exclusion of transaction costs. Both figures show, among others, two momentum indicators from the Elder-Ray Index to be important, namely the Bull Power (BULLP) and Bear Power (BEARP). Further explanation on all indicators can be found in the Pandas TA documentation³.

Table 5 contains the test statistics and corresponding p -values of testing the main hypothesis in equation (7) using a Wilcoxon Signed-Rank test (Wilcoxon, 1992). The test is performed on all four strategies at each of the frequencies, $g \in \{5, 15, 30\}$, for both time periods. Findings revealed that after correcting the p -values using the Benjamini-Hochberg procedure, there is not enough statistical evidence to reject the null hypothesis of equal expected returns between the B&H and any of the alternative strategies when transaction costs of 0.1% are imposed at a 5% confidence level. However, when transaction costs are excluded, there is enough statistical evidence to reject the null hypothesis of equal expected returns in favour of the alternative hypothesis of higher expected returns for all four strategies based on a 5-minute frequency.

³Explanation on indicators can be found here: <https://github.com/twopirllc/pandas-ta/blob/master/README.md>

5 Conclusion

This study examined the profitability of multiple trading strategies based on machine learning algorithms, namely Random Forests, Logistic Regressions, and Support Vector Machines. Along with a combination of these algorithms in the form of an Ensemble Voting classifier. The performance of the trading strategies is measured against test samples from a recent and earlier timeframe, namely in 2017/18 and 2020/21, to examine if the cryptocurrency market has evolved and whether the results from research in the foregoing years continue to hold.

This work contributed to the understudied field of high-frequency algorithmic cryptocurrency trading, by exploring strategies based on data frequencies up to 30-minutes, in contrast to similar studies that are often conducted using lower frequency data. The subset features that are most relevant at each data frequency are determined through the iterative feature selection tool Boruta. The selected features are used in a sliding window model to predict a binary target variable, the movement of the closing price in the next interval.

By means of quantitative analysis through a backtesting framework for the algorithmic strategies, it can be concluded that a high-frequency trading strategy based on machine learning could be employed as a profitable investment strategy. However, based on a statistical test, the expected returns of these strategies are not significantly higher than the returns from an elementary Buy-and-Hold strategy when transaction costs are imposed on the model in either testing period. In a theoretical scenario without transaction costs, strategies based on 5-minute data can be shown to have significantly higher returns than the Buy-and-Hold strategy during 2017/18. Furthermore, from the analysis, it can be concluded that the exploitability of the cryptocurrency market through algorithms has declined over time, as the models have been found to perform worse in 2020/21 compared to 2017/18.

There is no clearly distinguishable winner among the machine learning models in terms of performance, neither can we conclude that the Ensemble Voting strategy performs better than the individual models. The comparison between the behaviour of the trading strategies when considering transaction costs and neglecting them provides insight into how the behaviour of the algorithms changes when a threshold is imposed on the target variable. We conclude that the algorithms become more conservative as trades with small margins become unprofitable once transaction costs are considered, leaving the models to predict an upwards movement only when a relatively large price increase is expected.

Several limitations have been identified. Namely, this study was not able to deploy an

exhaustive grid search over the hyperparameters of each model due to the accompanied computational complexity, nor was the possibility of a strategy based on a 1-minute frequency explored for the same reason. However, a 1-minute strategy would likely not be profitable when transaction costs are considered, as the profit margins for a 1-minute frequency would be even smaller than for a 5-minute strategy, which is already unprofitable with the inclusion of transaction costs. Moreover, the cryptocurrency market had an increasing trend in both testing periods, indicated by the high returns of the B&H strategy. Smaller subsets of data with varying market conditions might be considered to find out how robust the trading strategies are when being tested in a bearish market. For the SVM model, non-linear kernels have not been considered due to the increased computational complexity, exploring the performance of these non-linear kernels might be of future research interest. Lastly, the soft-majority voting approach was only applied to the 15-minute strategies. However, the increase in performance shows that it could prove promising for future research.

References

- Baek, C., & Elbeck, M. (2015). Bitcoins as an investment or speculative vehicle? a first look. *Applied economics letters*, 22(1), 30–34.
- Baur, D. G., & Dimpfl, T. (2018). Asymmetric volatility in cryptocurrencies. *Economics letters*, 173, 148–151.
- Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1), 289–300.
- Biais, B., Woolley, P. et al. (2011). High frequency trading. *Manuscript, Toulouse University, IDEI*.
- Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.
- Borges, T. A., & Neves, R. F. (2020). Ensemble of machine learning algorithms for cryptocurrency investment with different data resampling methods. *Applied soft computing*, 90, 106187–.
- Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2), 123–140.
- Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5–32.
- Brogaard, J., Hendershott, T., & Riordan, R. (2014). High-frequency trading and price discovery. *The Review of financial studies*, 27(8), 2267–2306.
- Chen, Z., Li, C., & Sun, W. (2020). Bitcoin price prediction using machine learning: An approach to sample dimension engineering. *Journal of Computational and Applied Mathematics*, 365, 112395.
- Cheng, E. (2017, June 13). *Just 10% of trading is regular stock picking, jpmorgan estimates*. <https://www.cnbc.com/2017/06/13/death-of-the-human-investor-just-10-percent-of-trading-is-regular-stock-picking-jpmorgan-estimates.html>
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273–297.
- Dietterich, T. G. (2000). Ensemble methods in machine learning. *International workshop on multiple classifier systems*, 1–15.
- Dwyer, G. P. (2015). The economics of bitcoin and similar private digital currencies. *Journal of financial stability*, 17, 81–91.
- Dyhrberg, A. H. (2016). Bitcoin, gold and the dollar – a garch volatility analysis. *Finance research letters*, 16, 85–92.

- Fang, F., Ventre, C., Basios, M., Kong, H., Kanthan, L., Li, L., Martinez-Regoband, D., & Wu, F. (2020). Cryptocurrency trading: A comprehensive survey. *arXiv preprint arXiv:2003.11352*.
- Hawkins, D. M. (2004). The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1), 1–12.
- Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55–67.
- Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression* (Vol. 398). John Wiley & Sons.
- Kursa, M. B., Jankowski, A., & Rudnicki, W. R. (2010). Boruta – a system for feature selection. *Fundamenta informaticae*, 101(4), 271–285.
- La Roche, J. (2021, February 16). *Bitcoin breaks \$50,000 for the first time*. <https://finance.yahoo.com/news/bitcoin-breaks-50000-130116150.html>
- Madan, I., Saluja, S., & Zhao, A. (2015). Automated bitcoin trading via machine learning algorithms. URL: <http://cs229.stanford.edu/proj2014/Isaac%20Madan, 20>.
- McNally, S., Roche, J., & Caton, S. (2018). Predicting the price of bitcoin using machine learning. *2018 26th euromicro international conference on parallel, distributed and network-based processing (PDP)*, 339–343.
- Nakamoto, S. (2019). *Bitcoin: A peer-to-peer electronic cash system* (tech. rep.). Manubot.
- Nakano, M., Takahashi, A., & Takahashi, S. (2018). Bitcoin technical trading with artificial neural network. *Physica A: Statistical Mechanics and its Applications*, 510, 587–609.
- Platt, J. et al. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3), 61–74.
- Sebastião, H., & Godinho, P. (2021). Forecasting and trading cryptocurrencies with machine learning under changing market conditions. *Financial Innovation*, 7(1), 1–30.
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267–288.
- Vo, A., & Yost-Bremm, C. (2020). A high-frequency algorithmic trading strategy for cryptocurrency. *Journal of Computer Information Systems*, 60(6), 555–568.
- Wigglesworth, R. (2019, January 9). *Volatility: How 'algorithms' changed the rhythm of the market*. <https://www.ft.com/content/fdc1c064-1142-11e9-a581-4ff78404524e>
- Wilcoxon, F. (1992). Individual comparisons by ranking methods. *Breakthroughs in statistics* (pp. 196–202). Springer.

Yaga, D., Mell, P., Roby, N., & Scarfone, K. (2018). Blockchain technology overview. <https://doi.org/10.6028/nist.ir.8202>

Yermack, D. (2015). Is bitcoin a real currency? an economic appraisal. *Handbook of digital currency* (pp. 31–43). Elsevier.

Appendix

Table 6: Results of all strategies using a 5-minute frequency and no transaction fee

Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses

<i>Algorithm</i>	B&H	RF	LR	SVM	EV
<i>Metric</i>					
Accuracy	49.90% (50.21%)	52.93% (52.52%)	51.71% (51.81%)	51.82% (51.76%)	52.19% (51.84%)
F_1 -score (Buy)	66.58% (66.85%)	47.48% (51.03%)	47.92% (54.41%)	45.14% (53.90%)	46.35% (54.20%)
F_1 -score (Sell)	0.00% (0.00%)	57.36% (53.92%)	55.00% (48.89%)	57.04% (49.41%)	56.88% (49.24%)
Precision (Buy)	49.90% (50.21%)	53.57% (52.91%)	51.89% (51.82%)	52.27% (51.81%)	52.67% (51.87%)
Annualized returns (%)	804.23% (398.22%)	1149.4% (322932%)	339.5% (12720%)	505.0% (7772%)	653.1% (13893%)
Annualized Sharpe ratio (%)	3.52 (1.98)	5.50 (9.92)	3.44 (6.06)	4.15 (5.58)	4.65 (6.17)
Win rate (%)	N/A N/A	65.79% (59.49%)	62.47% (59.36%)	62.98% (60.29%)	64.11% (59.58%)
Maximum Drawdown (%)	30.51% (69.59%)	22.02% (27.51%)	26.52% (42.06%)	23.01% (39.74%)	22.83% (43.32%)
Time in market (%)	100% (100%)	39.72% (46.76%)	42.80% (50.86%)	37.93% (49.79%)	39.21% (50.29%)

Table 7: Results of all strategies using a 5-minute frequency and 0.1% transaction fee

Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses

<i>Algorithm</i>	B&H	RF	LR	SVM	EV
<i>Metric</i>					
Accuracy	24.18% (33.91%)	75.82% (66.38%)	74.15% (59.17%)	75.38% (59.14%)	75.39% (59.21%)
F_1 -score (Buy)	38.95% (50.65%)	7.02% (8.84%)	16.59% (29.31%)	10.73% (29.87%)	11.11% (29.19%)
F_1 -score (Sell)	0.00% (0.00%)	86.11% (79.39%)	84.71% (71.29%)	85.72% (71.17%)	85.72% (71.36%)
Precision (Buy)	24.18% (33.91%)	50.13% (54.94%)	37.75% (35.49%)	43.57% (35.73%)	43.90% (35.49%)
Annualized returns (%)	795.46% (396.92%)	-38.57% (48.82%)	-62.76% (-17.73%)	-50.46% (-44.73%)	-42.82% (-5.18%)
Annualized Sharpe ratio (%)	3.52 (1.98)	-2.22 (1.28)	-3.83 (-0.08)	-2.52 (-0.75)	-2.03 (0.17)
Win rate (%)	N/A N/A	51.14% (54.60%)	38.88% (52.26%)	47.45% (50.53%)	49.05% (52.36%)
Maximum Drawdown (%)	30.45% (69.58%)	42.20% (38.98%)	60.90% (57.38%)	53.63% (68.23%)	51.04% (56.22%)
Time in market (%)	100% (100%)	1.82% (2.97%)	6.81% (19.21%)	3.40% (19.71%)	3.50% (19.05%)

Table 8: Results of all strategies using a 30-minute frequency and no transaction fee

Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses					
<i>Algorithm</i>	B&H	RF	LR	SVM	EV
<i>Metric</i>					
Accuracy	51.13% (50.90%)	52.81% (52.00%)	52.39% (52.35%)	53.17% (52.22%)	53.15% (52.23%)
F_1 -score (Buy)	67.67% (67.46%)	49.24% (49.93%)	48.32% (54.31%)	56.88% (53.60%)	52.69% (54.40%)
F_1 -score (Sell)	0.00% (0.00%)	55.92% (53.90%)	55.86% (50.20%)	48.76% (50.75%)	53.61% (49.84%)
Precision (Buy)	51.13% (50.90%)	54.72% (53.22%)	54.30% (53.04%)	53.74% (52.99%)	54.48% (52.90%)
Annualized returns (%)	793.42% (409.72%)	500.35% (711.56%)	597.96% (541.47%)	1017.95% (192.21%)	912.21% (250.54%)
Annualized Sharpe ratio (%)	3.51 (2.00)	4.10 (2.77)	4.79 (2.60)	4.65 (1.69)	4.83 (1.88)
Win rate (%)	N/A N/A	71.57% (63.21%)	73.16% (68.73%)	72.33% (68.56%)	72.99% (68.55%)
Maximum Drawdown (%)	28.92% (69.53%)	20.48% (52.37%)	15.78% (54.70%)	24.98% (52.13%)	21.76% (56.60%)
Time in market (%)	100% (100%)	41.82% (44.97%)	40.99% (53.39%)	53.15% (52.08%)	47.87% (53.85%)

Table 9: Results of all strategies using a 30-minute frequency and 0.1% transaction fee

Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses					
<i>Algorithm</i>	B&H	RF	LR	SVM	EV
<i>Metric</i>					
Accuracy	39.04% (44.39%)	61.61% (55.95%)	57.02% (55.34%)	58.32% (55.21%)	58.30% (55.34%)
F_1 -score (Buy)	56.15% (61.49%)	16.88% (32.55%)	32.89% (36.11%)	31.65% (35.47%)	31.60% (35.67%)
F_1 -score (Sell)	0.00% (0.00%)	75.04% (67.30%)	68.39% (65.68%)	70.02% (65.70%)	70.01% (65.80%)
Precision (Buy)	39.04% (44.39%)	54.51% (50.83%)	42.12% (49.49%)	43.98% (49.21%)	43.94% (49.48%)
Annualized returns (%)	793.24% (408.40%)	-12.28% (-55.24%)	-9.64% (-8.19%)	-3.67% (-52.04%)	-7.66% (-19.26%)
Annualized Sharpe ratio (%)	3.51 (2.00)	-0.43 (-1.15)	-0.19 (0.17)	0.01 (-0.80)	-0.13 (-0.03)
Win rate (%)	N/A N/A	56.71% (55.22%)	48.07% (56.29%)	47.31% (53.66%)	47.04% (56.02%)
Maximum Drawdown (%)	28.93% (69.52%)	26.82% (67.39%)	30.14% (41.23%)	29.75% (51.28%)	29.77% (42.98%)
Time in market (%)	100% (100%)	7.15% (20.91%)	25.00% (25.50%)	21.95% (25.01%)	21.92% (25.01%)

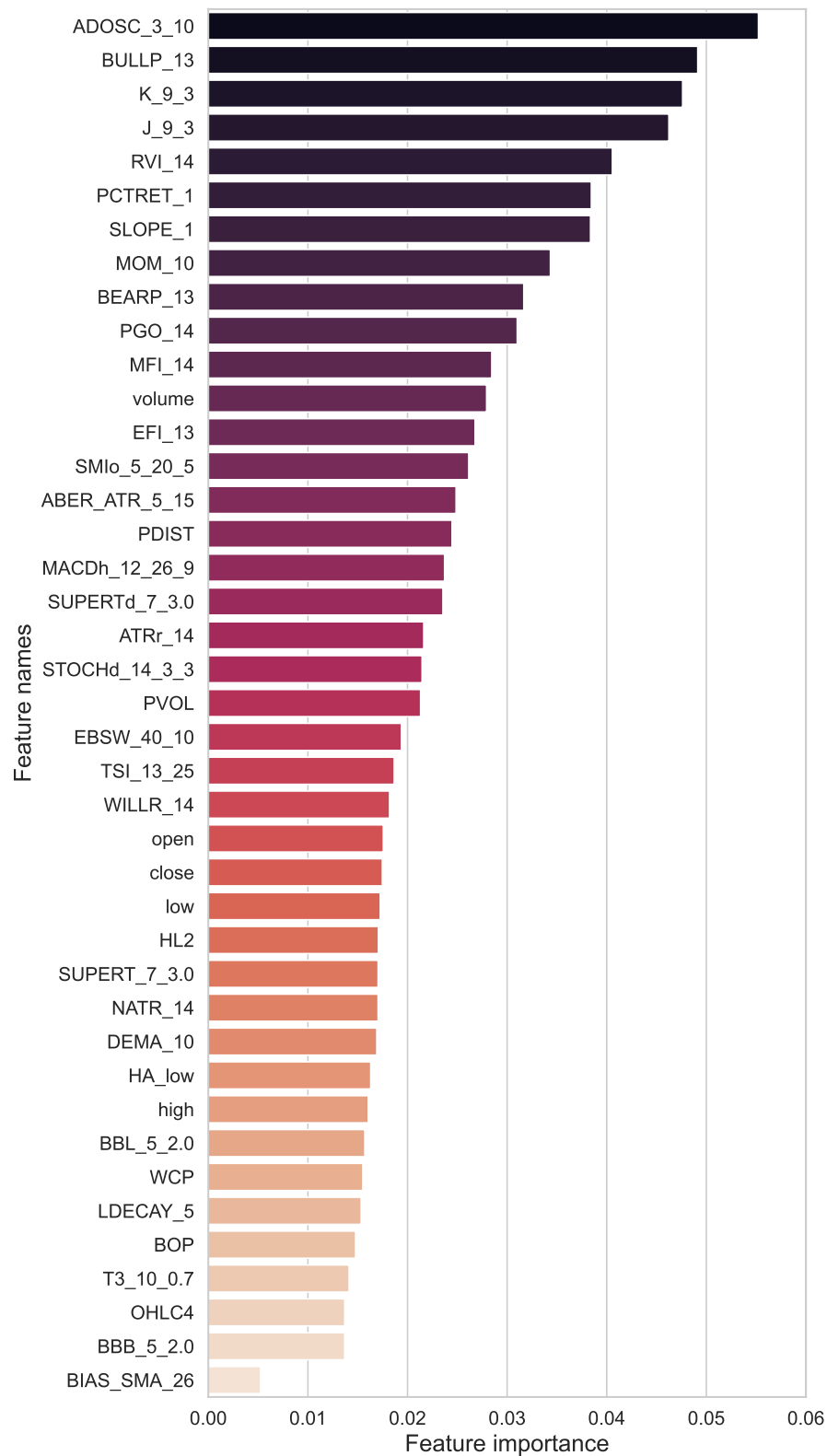
Table 10: Results of all strategies using a 15-minute frequency with soft-majority voting and no transaction fee

Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses					
<i>Algorithm</i>	B&H	RF	LR	SVM	EV
<i>Metric</i>					
Accuracy	50.54% (50.65%)	52.27% (52.77%)	51.87% (52.42%)	52.52% (52.28%)	52.37% (52.85%)
F_1 -score (Buy)	67.15% (67.24%)	50.45% (54.27%)	50.10% (54.18%)	49.25% (54.16%)	49.79% (54.57%)
F_1 -score (Sell)	0.00% (0.00%)	53.97% (51.17%)	53.51% (50.52%)	55.40% (50.24%)	54.69% (51.00%)
Precision (Buy)	50.54% (50.65%)	53.07% (53.25%)	52.62% (52.88%)	53.56% (52.74%)	53.28% (53.29%)
Annualized returns (%)	795.86% (408.34%)	571.52% (3642.87%)	295.11% (1004.26%)	130.90% (1210.68%)	362.16% (3739.55%)
Annualized Sharpe ratio (%)	3.51 (2.00)	4.43 (4.58)	3.33 (3.30)	2.18 (3.61)	3.77 (4.64)
Win rate (%)	N/A N/A	69.52% (64.51%)	70.38% (70.49%)	66.80% (69.77%)	71.52% (70.64%)
Maximum Drawdown (%)	29.97% (69.54%)	20.26% (38.09%)	25.35% (38.48%)	24.96% (40.70%)	24.35% (35.37%)
Time in market (%)	100% 100%	45.67% (52.61%)	45.92% (53.16%)	43.01% (53.42%)	44.33% (53.12%)

Table 11: Results of all strategies using a 15-minute frequency with soft-majority voting and a 0.1% transaction fee

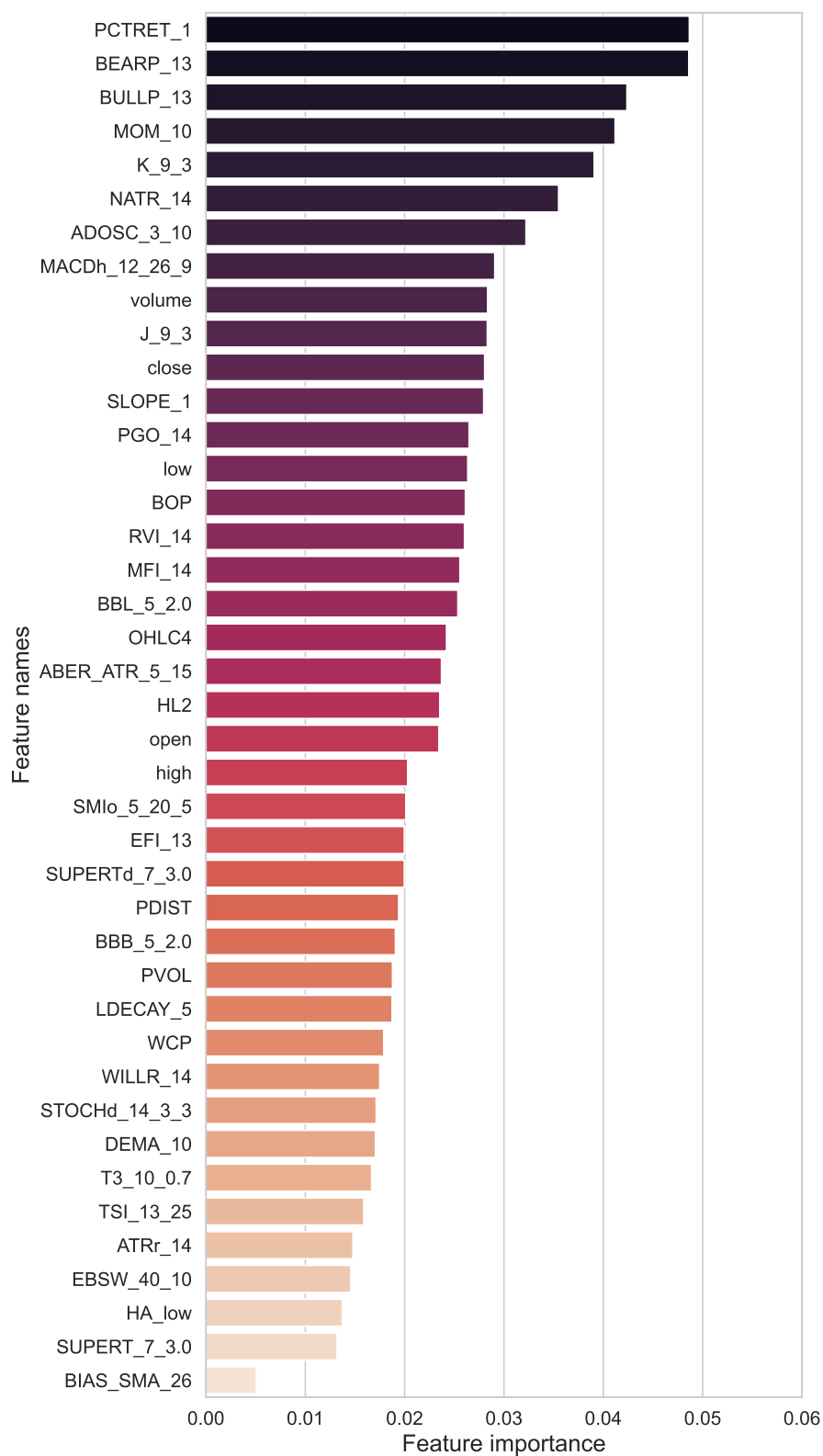
Results from 2017-2018 in (parentheses) and results from 2020-2021 in no parentheses					
<i>Algorithm</i>	B&H	RF	LR	SVM	EV
<i>Metric</i>					
Accuracy	34.14% (41.20%)	66.15% (59.01%)	62.99% (58.05%)	65.21% (58.08%)	65.71% (58.29%)
F_1 -score (Buy)	50.90% (58.35%)	11.93% (24.23%)	28.97% (28.83%)	4.21% (14.37%)	13.20% (22.71%)
F_1 -score (Sell)	0.00% (0.00%)	79.05% (71.91%)	74.97% (70.26%)	78.74% (72.25%)	78.63% (71.44%)
Precision (Buy)	34.14% (41.20%)	53.34% (50.80%)	42.00% (47.87%)	35.06% (45.34%)	48.58% (48.00%)
Annualized returns (%)	795.67% (407.03%)	20.90% (-33.68%)	-67.47% (-64.17%)	-19.58% (19.59%)	-0.16% (118.07%)
Annualized Sharpe ratio (%)	3.51 (2.00)	0.90 (-0.48)	-3.44 (-1.61)	-1.95 (0.65)	0.07 (1.72)
Win rate (%)	N/A N/A	53.90% (57.05%)	44.36% (51.87%)	28.77% (55.24%)	50.96% (59.84%)
Maximum Drawdown (%)	29.97% (69.53%)	16.42% (53.42%)	62.07% (70.66%)	20.72% (37.16%)	18.76% (38.65%)
Time in market (%)	100% 100%	4.30% (12.90%)	17.97% (17.75%)	2.15% (7.76%)	5.37% (12.76%)

Figure 3: RF feature importance scores for 15-minute strategy with no transaction fee



Note: feature definitions can be found at <https://github.com/twopirlc/pandas-ta/blob/main/README.md>

Figure 4: RF feature importance scores for 15-minute strategy with 0.1% transaction fee



Note: feature definitions can be found at <https://github.com/twopirllc/pandas-ta/blob/main/README.md>