Universiteit Leiden
The Netherlands

Bachelor Computer Science & Economics

Machine Learning and Technical Analysis for Foreign Exchange Data with Automated Trading

Fabian Schut

Supervisors:
Jan N. van Rijn & Holger H. Hoos

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

08/07/2019

Abstract

The prediction of the foreign exchange market comes with great challenges and opportunities. This study describes the prediction whether the price of one currency will increase or decrease relative to another currency over a given time interval and how this can be exploited in a practical trading strategy. In order to generate a prediction, a stream of historical bid and ask prices is converted into a pre-defined set of so-called technical indicators. This provides the basis for machine learning algorithms to induce a model. More precisely, default random forest and gradient boosting were applied on three time intervals; daily, hourly and per-minute. Furthermore, we applied state-of-the-art hyperparameter optimization techniques on a daily interval, including automated machine learning with combined algorithm selection and hyperparameter optimization. The technical indicators are calculated based on five elements within any given time interval; open, high, low, close and volume. We will report on the performance of the models in respect to the majority and the no-change classifiers. The results reflect that most models outperform the majority and the no-change classifiers, often with statistical significance. The second part of this study describes the results of applying findings of the trend prediction into an offline real-world trading strategy. The results reflect that a 1.85% return on investment for an investment of €100.000 over 5 months can be achieved with the Euro to Norwegian Krone in a daily interval.

Contents

1 Introduction **1**
1.1 The situation . . . 1
1.2 Research question . . . 1
1.3 Thesis overview . . . 2

2 Definitions **3**
2.1 Bid and ask . . . 3
2.2 Open, high, low, close and volume . . . 3

3 Related Work **4**
3.1 Previous studies . . . 4
3.2 Relevance . . . 5

4 Methodology **6**
4.1 Trend prediction . . . 6
4.1.1 Features with technical indicators . . . 6
4.1.2 Target . . . 6
4.2 Backtesting . . . 8

5 Experimental Setup **9**
5.1 Trend prediction . . . 9
5.1.1 Data . . . 9
5.1.2 Implementation details . . . 9
5.1.3 Run . . . 9
5.2 Backtesting . . . 13

6 Results **15**
6.1 Trend prediction . . . 15
6.1.1 Default hyperparameters . . . 15
6.1.2 Optimized hyperparameters/algorithm selection . . . 19
6.1.3 Running time . . . 25
6.2 Backtesting . . . 28

7 Conclusions **31**

8 Further Research **32**

References **34**

1 Introduction

In the introduction we will explain the current situation, our research question and the thesis overview.

1.1 The situation

Trading on financial markets, such as the stock market and foreign exchange (Forex), comes with great challenges and opportunities. In order to deploy a trading strategy, many components need to work together, e.g., identifying trends, predicting whether the price of a certain financial asset will increase or decrease, assessing the risk and deciding on a proper action. The stock market often shows high volatility and has been identified as one of the most challenging applications of machine learning [13]. Research has shown that the quality of stock price prediction can be more accurate with technical analysis [4, 15], i.e., statistical properties on recent intervals of the data. However, technical analysis will only be useful if the market is not efficient, i.e., the current price of a financial asset does not contain all the current available information [4]. Related work has shown that there are many technical indicators available [1]. Before the upswing of artificial intelligence (AI), traders manually selected the most profitable indicators. Fortunately, machine learning algorithms can be applied on this task.

We will focus on the foreign exchange market. Foreign exchange, also known as Forex or FX, is the market in which currencies are traded. It is the most liquid financial market in the world and daily transactions are worth over trillions (10^{12}) of dollars. Foreign exchange currencies can be traded 24 hours per day and 5 days per week. Moreover, the Forex market is a decentralized network of banks, brokers, institutions and individual traders [19]. In general, there are three parties involved in trading on the Forex market; (i) *trader*: the party that buys and sells the actual currency; (ii) *stockbroker*: licensed professionals who have access to the Forex market. A trader is not able to directly buy from the Forex market and buys through a broker; and (iii) *market maker*: institutions that ensure there is enough liquidity on the financial market, so that stockbrokers are always able to buy and sell on the Forex market.

1.2 Research question

Despite the fact that technical indicators and machine learning models can be used for accurate prediction on the financial market, some aspects remain unclear. First of all, there are many different machine learning models available which can help solving this problem statement. How models perform in comparison to each other is unknown a priori. Another uncertainty is the performance of the models in an offline real-world trading strategy, i.e., a simulation of trading on the Forex market with historical data where transaction costs and market opening hours are taken into account. Hence, this raises the question: *'Which AI and machine learning approaches perform well in trading (algorithmically) on the foreign exchange market and how effective are these in terms of potential gains in profit and risk?'*. To answer this research question, the work is divided into two parts. In the first part, different machine learning approaches will be trained and tested on 32 currency pairs. We report on performance compared to baselines. In the second part, the best performing machine

learning model will be applied in an offline real-world trading strategy, i.e., backtest, to measure profit and risk.

1.3 Thesis overview

Following this introduction, Section 2 includes the definitions; Section 3 discusses related work; Section 4 continues with the methodology used; Section 5 explains experiments; Section 6 shows the results; Section 7 concludes this study; and Section 8 gives recommendations for further research. This is the bachelor thesis for the bachelor Computer Science & Economics at Leiden Institute of Advanced Computer Science and the thesis was supervised by Jan N. van Rijn and Holger H. Hoos.

2 Definitions

In the definitions we will introduce the terms we used throughout this paper.

2.1 Bid and ask

For any timestamp, any pair of currencies (A, B) is given the following information: bid and ask. The bid price specifies at which price (expressed in currency B) the trader sells a unit of currency A and the ask price specifies what the trader wants to pay in currency B for a unit of currency A [7]. The gap between the bid and ask prices is called the spread. Note that the ask price is always higher than the bid price – this is how the market maker makes a profit. Since this study is focused on the performance of an individual trader, the performance of the market maker is out of scope.

2.2 Open, high, low, close and volume

There are four prices and one volume indicator available for every bid and ask price in a time interval: (i) *open*: the price at the start of the interval; (ii) *close*: the price at the end of the interval; (iii) *high*: the highest price during the interval; (iv) *low*: the lowest price during the interval; and (v) *volume*: every price change and transaction comes with an amount of currencies provided by the market maker at that moment in time. For daily, hourly and per-minute intervals, volume is the sum of all currencies that is provided within the interval. Note that volume is not the number of currency pairs traded between two parties, since the forex market is decentralized and that information is not available.

We denote the average of the bid and ask price at the open, high, low, close and volume at interval i as O_i, H_i, L_i, C_i and V_i, respectively. The high or low over multiple intervals is denoted by a range in subscript, e.g., $H_{[i-n+1,i]}$ denotes the highest price over the n intervals up to and including i.

3 Related Work

There have been applications of machine learning to various well-defined sub-problems of foreign exchange trading. In the following section, we discuss several studies that inspired our research and we outline our contribution to the scientific community.

3.1 Previous studies

Baasher & Fakhr [3] have shown promising results in the prediction of the high price in a daily interval on historical Forex data. The study describes a binary classification problem whether the price of one currency will increase or decrease relative to another currency in a daily interval. They have used technical analysis and signal processing for their prediction on four currency pairs. The binary classification is as follows: (i) *true*: the high price will increase over a daily interval, compared to the high price of the day before; and (ii) *false*: the high price will decrease or stay the same over a daily interval, compared to the high price of the day before. Baasher & Fakhr used three supervised machine learning classifiers, namely, radial basis function neural network, multilayer perceptron neural network and support vector machine. To improve the classifiers, two feature selection and five feature extraction techniques are applied. With feature selection, a subset of the best features is selected. Feature extraction on the other hand still includes all the features. However, it reduces the number of features by grouping individual features into one. The results are then compared in terms of accuracy score, i.e., the percentage correctly predicted and percentage normalized profit (PNP), i.e., a profit criterion that appoints higher scores when important changes are predicted correctly. The highest accuracy score and PNP achieved are 79.8 and 67.7, respectively. Although their results seem solid, no viable trading strategy can be applied to such predictions.

Most notable is the work described by Dempster & Leemans [8]. They have used recurrent reinforcement learning for their automated trading system on the Forex market. Dempster & Leemans used a three-layer system: (i) *machine learning algorithm*: a recurrent single layer neural network, where the price of one currency is expressed in another; (ii) *risk management*: the evaluation whether the recommendation from the first layer is desirable in terms of risk as well; and (iii) *dynamic utility optimization*: the hyperparameter optimization of the first and second layer, in terms of utility and risk-return. Their system deployed a proper trading strategy based on an offline real-world simulation and obtained a profit of more than 26% per annum over two years with high-frequency data, on a single dataset.

Sidehabi & Tandungan [22] have shown that all four prices: open, high, low and close, can be proper predicted in an hourly interval, with an RMSE score between 0.000307 and 0.001084. Note that this is a prediction for the actual value of the prices, not a binary classification. They used three time periods of observation, i.e., 1 day, 5 days and 30 days. Sidehabi & Tandungan used both statistical and machine learning approaches. For the statistical method, they used adaptive spline treshold autoregression and for the machine learning method they used support vector machine and neural network.

Yang & Zhang [27] did not predict the direction or value of a financial asset, however, they have shown a new variance estimator, i.e., a predictor for volatility in the financial market. This new estimator can be used for risk analysis, namely, if the price of a financial asset increases or

decreases fast, this could be assigned as risky. In classical variance estimators of volatility, only closing prices were used. However, Yang & Zhang described an estimator based on opening, high, low and closing prices, which resulted in a more accurate estimator and independent of drift and opening price jumps.

3.2 Relevance

Baasher & Fakhr [3] predicted only the high price in a daily interval. Besides predicting the high price, our research will make a prediction for the opening, low and most importantly, the closing price as well. Additionally, we will compare the closing price with 3 time intervals, namely, daily, hourly and per-minute. Furthermore, we will show a spread of accuracy scores instead of a single value. Many machine learning algorithms use randomization to induce a model, which results in different models and different accuracy scores. A spread of values will therefore say more about the actual accuracy than a single value.

More generally, our contributions are as follows: (i) we apply state-of-the-art hyperparameter optimization techniques and automated algorithm selection to the various datasets and report on performance compared to baselines; (ii) we apply the best machine learning model in an offline real-world trading strategy based on historical data; (iii) we outline how automated data science and meta-learning can be used to further improve predictive results; and (iv) we make the benchmark suite available on OpenML[1], for the community to participate in this scientific challenge on a homogeneous collection of datasets.

[1]See: https://www.openml.org/s/219

4 Methodology

In the following, we will explain the theoretical background in order to define our features and target. We will first explain the trend prediction and then the backtesting, i.e., trading strategy.

4.1 Trend prediction

In the trend prediction we will discuss the defined features and the target.

4.1.1 Features with technical indicators

For technical analysis, there are five elements that can be used; open, high, low, close and volume. However, the Forex market provides the five elements twice, for both bid and ask. Therefore, we take our pre-defined elements O_i, H_i, L_i, C_i and V_i. An average between bid and ask is used, since sometimes there will be a sell signal and sometimes there will be a buy signal. With a sell signal, the trader sells the currency for the bid price and with a buy signal, the trader buys the currency for the ask price. In addition, Forex is a very liquid market which results in a relatively small spread [7]. However, the two values are added with volume instead of taking the average. Volume is an indicator of liquidity in the market and we calculated the technical analysis with the total liquidity.

The technical indicators were calculated based on O_i, H_i, L_i, C_i and V_i. Table 1 shows the technical indicators that we considered in this research. Note that several of these are parameterized (by size of the time window). For those, we followed the parameterization of Baasher & Fakhr [3].

In addition to the technical indicators from Table 1, more technical indicators are generates using a Python library. This is an open source technical analysis library build upon Pandas[2]. In total, this library provides 53 indicators. After removing the duplicates of indicators listed in Table 1, there are 41 new indicators left, which generates 41 features as well. The calculated and library indicators combined 86 features. In the first 51 time intervals, some of the technical indicators cannot be calculated, because they need a time window of 52 intervals. These rows are removed from the dataset. Note that the features are calculated over their time interval. This means that for the prediction in daily, hourly and per-minute intervals, the time window is in days, hours and minutes, respectively.

4.1.2 Target

Generally, machine learning models are induced based on a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, n\}$ to map an input $\mathbf{x}$ to output $f(\mathbf{x})$, which closely represents y. The $\mathbf{x}$-values are in this case the technical indicators plus the prices and volume indicator O_i, H_i, L_i, C_i and V_i. As such, the trained model predicts the y-value based on a small window (max 52) from the past.

[2]See: https://github.com/bukosabino/ta

Table 1: Technical Indicators. The current interval is denoted with index i. Several technical indicators are parameterized by n, the size of the time window.

Name	Definition	ref	#features
Simple Moving Average	$SMA_i^c = \frac{C_{i-n+1}+C_{i-n+2}+...+C_i}{n}$ (can be defined on open, high and low as well)	[1]	0
Stochastic Oscillator	$\%K = \frac{C_i - L_{[i-n+1,i]}}{H_{[i-n+1,i]} - L_{[i-n+1,i]}}$, $\%D = \frac{\%K_{i-n+1}+\%K_{i-n+2}+...+\%K_i}{n}$	[1]	12
Momentum	$C_i - C_{i-n}$	[14]	6
Price Rate of Change	$\frac{C_i - C_{i-n}}{C_{i-n}}$	[1]	4
Williams %R	$\frac{H_{[i-n+1,i]} - C_i}{H_{[i-n+1,i]} - L_{[i-n+1,i]}}$	[1]	5
Weighted Closing Price	$\frac{C_i \times 2 + H_i + L_i}{4}$	[1]	1
Williams Accumulation Distribution Line	$WADL_i = WADL_{i-n} + \begin{cases} min(L_i, C_{i-n}) * V_i & \text{if } C_i > C_{i-n} \\ max(H_i, C_{i-n}) * V_i & \text{if } C_i < C_{i-n} \\ 0 & \text{otherwise} \end{cases}$	[1]	2
Moving Average Convergence, Divergence	Exponential moving average, placing a greater weight on the most recent data points	[1]	1
Commodity Channel Index	The variation over the sum of H_i, L_i and C_i from its statistical mean	[1]	1
Bollinger Bands	Upper- and lowerband: two standard deviations shifted respectively up or down from SMA_i	[1]	2
Heikin-Ashi	Candlestick bar $(O_{HAi}, C_{HAi}, H_{HAi}, L_{HAi})$ which eliminates irregularities from a normal bar (O_i, C_i, H_i, L_i)	[24]	4
Mean Open & Close	$\frac{O_{i-n+1}+O_{i-n+2}+...+O_i+C_{i-n+1}+C_{i-n+2}+...+C_i}{2n}$	-	1
Variance Open & Close	Variance over $\{O_{i-n+1}, O_{i-n+2}, \ldots, O_i, C_{i-n+1}, C_{i-n+2}, \ldots, C_i\}$	-	1
High Price Average	Simple moving average over the high	-	1
Low Price Average	Simple moving average over the low	-	1
High, Low Average	Simple moving average over the sum of high and low	-	2
Trading Day Price Average	Simple moving average over the sum of the open, high, low and close	-	1

We define our prediction target as the binary classification $Y_i < Y_{i+1}$, where Y can be O, H, L and C. The target attribute is $true$ when the price will increase over a specific time interval, and the class attribute is $false$ when the price will decrease or stay the same over a specific time interval.

The target prices – open, high and low – are only meant for research purposes and cannot be used in a trading strategy. The reason for the opening price to be unusable is as follows: the Forex market is open 24 hours per business day, which results in a situation that the closing price of the current interval and the opening price of the following interval are closely related to each other. The current closing price needs to be known for the calculation of the technical indicators, so there would be no time in between to trade. The gap between the current closing price and the following opening price is bigger in the weekends. Nevertheless, it is then not possible to trade. The reason for the high and low prices to be unusable is the same: it is not known when the high or low price will occur within a time interval, i.e., it is allocated afterwards. However, we still analyze the results, especially the high price to compare the results with the study of Baasher & Fakhr [3].

4.2 Backtesting

We will report on profit in quantitative terms, return on investment, annual return (in %) and the Sharpe ratio. The Sharpe ratio is a gain measurement that takes the risk of a return into account. Let R_{Ft} be the return on Forex F in period t and R_{Bt} the return on a benchmark in period t, i.e., a pre-defined return (in this case the S&P 500). D_t, the differential return, is then defined as:

$$D_t = R_{Ft} - R_{Bt} \tag{1}$$

Let D be the average of D_t for the period from $t = 1$ up until T:

$$D = \frac{1}{T} \sum_{t=1}^{T} D_t \tag{2}$$

Let σ_d be the standard deviation of D_t:

$$\sigma_D = \sqrt{\frac{\sum_{t=1}^{T} (D_t - D)^2}{T - 1}} \tag{3}$$

The Sharpe ratio is then defined as:

$$S = \frac{D}{\sigma_D} \text{ [21]} \tag{4}$$

5 Experimental Setup

In the following, we will explain what programs, data and models we have used in order to generate results. We will first explain the trend prediction and then the backtesting, i.e., trading strategy.

5.1 Trend prediction

In the trend prediction we will discuss the data used, the implementation details and the run of the experimental setup.

5.1.1 Data

There are several data providers for historical Forex data available, both free of charge and for a fee. We collected our data from Dukascopy, one of the free of charge data providers, which is commonly used by the scientific community [26, 17]. We were allowed to redistribute their data, which was important to us, since we uploaded our data to OpenML [25]. The data we used comes from their online data feed platform[3]. In total, we have used 32 currency pairs in order to compare the results. We have chosen the major forex pairs together with some random selection from Dukascopy. The complete list of currency pairs can be found in Table 2. Both the bid and ask prices are downloaded and then combined based on their timestamp. All currency pairs were recorded over the same period (daily and hourly intervals: 2012-2018; per-minute intervals: 2018), not involving the weekends. Table 3 reflects the data elements after collecting and merging bid and ask prices. The timestamp was adjusted to the format `YYYYmmdd HH:MM`, to fit the QuantConnect format, where the trading strategy is run.

5.1.2 Implementation details

Python (version 3.6) is an interpreted, object-oriented and high-level programming language [18]. It is thus not compiled and instructions are executed directly. In addition, Python comes with many good libraries for data structures (Pandas version 0.24 [16]) and machine learning (Scikit-Learn version 0.21 [20]). Pandas is build upon Python. It is fast and well structured for the manipulation and analysis of data. Scikit-Learn is build upon Python as well. With Scikit-Learn a machine learning model can be run with a few lines of code.

5.1.3 Run

With the features and target defined, the machine learning algorithms can induce a model. We will apply and report on three machine learning approaches in this research. The models are trained on 80% of the total data. The residual 20% is used to test the model and report the results. This is called hold-out validation. Note that the order of the data is relevant. The events in the testing set took place before the events in the training set.

[3]See: https://www.dukascopy.com/swiss/english/marketwatch/historical/

Table 2: All the Forex currency pairs we have analyzed in their ISO-4217 format.

abbreviation	short description
AUD/CAD	Australian Dollar vs. Canadian Dollar
AUD/CHF	Australian Dollar vs. Swiss Franc
AUD/JPY	Australian Dollar vs. Japanese Yen
AUD/NZD	Australian Dollar vs. New Zealand Dollar
AUD/SGD	Australian Dollar vs. Singapore Dollar
AUD/USD	Australian Dollar vs. US Dollar
CAD/CHF	Canadian Dollar vs. Swiss Franc
CAD/JPY	Canadian Dollar vs. Japanese Yen
CHF/JPY	Swiss Franc vs. Japanese Yen
CHF/SGD	Swiss Franc vs. Singapore Dollar
EUR/AUD	Euro vs. Australian Dollar
EUR/CAD	Euro vs. Canadian Dollar
EUR/CHF	Euro vs. Swiss Franc
EUR/DKK	Euro vs. Danish Krone
EUR/GBP	Euro vs. Pound Sterling
EUR/HKD	Euro vs. Honk Kong Dollar
EUR/HUF	Euro vs. Hungarian Forint
EUR/JPY	Euro vs. Japanese Yen
EUR/NOK	Euro vs. Norwegian Krone
EUR/NZD	Euro vs. New Zealand Dollar
EUR/PLN	Euro vs. Polish Zloty
EUR/RUB	Euro vs. Russian Rouble
EUR/SEK	Euro vs. Swedish Krona
EUR/GSD	Euro vs. Singapore Dollar
EUR/TRY	Euro vs. Turkish Lira
EUR/USD	Euro vs. US Dollar
GBP/USD	Pound Sterling vs. US Dollar
NZD/USD	New Zealand Dollar vs. US Dollar
USD/CAD	US Dollar vs. Canadian Dollar
USD/CHF	US Dollar vs. Swiss Franc
USD/DKK	US Dollar vs. Danish Krone
USD/JPY	US Dollar vs. Japanese Yen

Table 3: The collection of information for currency pairs; the data we used in our research.

name	short description
timestamp	`YYYYmmdd HH:MM`
bid open	price in currency B for currency A
bid high	price in currency B for currency A
bid low	price in currency B for currency A
bid close	price in currency B for currency A
bid volume	liquidity in millions
ask open	price in currency B for currency A
ask high	price in currency B for currency A
ask low	price in currency B for currency A
ask close	price in currency B for currency A
ask volume	liquidity in millions

Table 4: Config space for 100 iterations of random search with gradient boosting.

hyperparameter	lower bound	upper bound	default value	log
learning_rate	0.00001	0.1	0.001	*true*
n_estimators	64	2048	100	*true*
subsample	0.0	1.0	1.0	
min_samples_split	2	20	2	
min_samples_leaf	1	20	1	
min_weight_fraction_leaf	0.0	0.5	0.0	
max_depth	1	32	3	
min_impurity_decrease	0.0	1.0	0.0	
max_features	0.0	1.0	0.0	
validation_fraction	0	1	0.1	
n_iter_no_change	1	2048	200	
tol	$1e-5$	$1e-1$	$1e-4$	*true*
hyperparameter	choises			
criterion	$'friedman\_mse', 'mse', 'mae'$			

Table 5: Config space for 100 iterations of random search with random forest.

hyperparameter	lower bound	upper bound	default value
max_features	0.	1.	0.5
min_samples_split	2	20	2
min_samples_leaf	1	20	1
hyperparameter	value		
n_estimators	100		
min_weight_fraction_leaf	0.		
hyperparameter	choices	default	
criterion	$'gini', 'entropy'$	$'gini'$	
bootstrap	$true, false$	*true*	

Default hyperparameters The first approach reflects results of the models random forest [6] and gradient boosting [12], with default hyperparameters from Scikit-Learn [20]. This approach compares daily, hourly and per-minute intervals on the closing price.

Optimized hyperparameters The second approach uses random forest and gradient boosting as well. However, the hyperparameters of random forest and gradient boosting are optimized with 100 iterations of random search. The different settings which are used in the random search iterations can be found in the Table 4 for gradient boosting and in Table 5 for random forest. For every iteration, the internal cross-validation is fixed to 5. Cross-validation ensures that the hyperparameters are not fitted on one validation set and therefore aims to ensure that the models are performing well in general, i.e. it prevents overfitting of the hyperparameters. Random search executes on the training set, so the testing set is disregarded from the fitting process. For a complete overview of splitting the data in training, cross validation and testing sets, see Figure 1. This experiment is run in a daily interval, predicting all four prices; open, high, low and close.

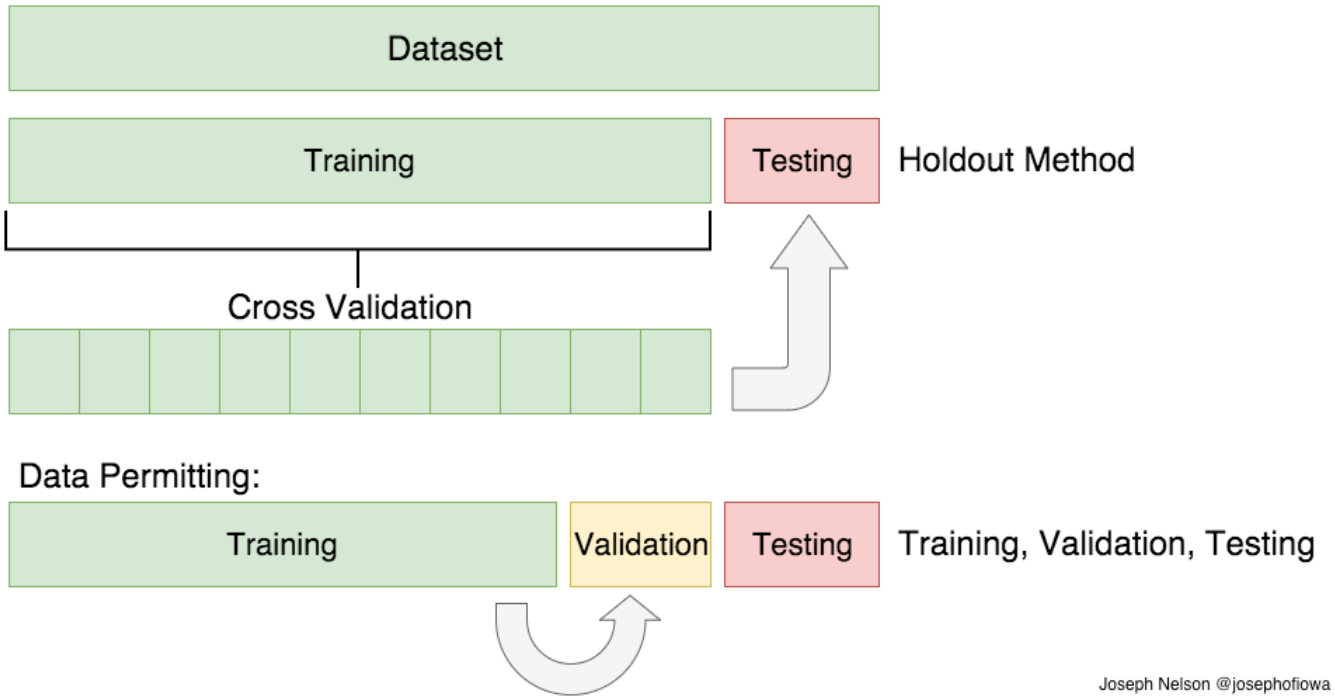


Figure 1: Overview splitting data in training, cross validation and testing sets[4].

Algorithm selection and optimized hyperparameters As third approach, we apply Auto-sklearn [10] for a daily interval for the four prices; open, high, low and close. Auto-sklearn is an automated machine learning system that applies meta-learning and black box optimization to search for the best combination of machine learning models with their best-performing hyperparameters. This is called combined algorithm selection and hyperparameter optimization (CASH) [23]. Auto-sklearn may seem to always perform better than the first two approaches, because Auto-sklearn searches over algorithms with their best-performing hyperparameters by itself. It is therefore at least able to find the same models as the default hyperparameters and the random search hyperparameters. Nonetheless, it seems natural to assume that it finds better performing models. However, there is one critical setting for this approach: running time. If there is an unlimited amount of time available, this approach is probably always better than the first two approaches. Since we do not have an unlimited amount of time available, we have set a time budget.

The first time limitation is on the total run, i.e., one complete Auto-sklearn model fitting and is set to 3600 seconds wall-clock time (we used a single core). The second time limitation is on a single call to try out one machine learning model, which is set to 360 seconds. The cross validation is set to 5-fold[5]. To compare the results between optimized hyperparameters and algorithm selection with optimized hyperparameters more fairly, cross validation is used in both approaches.

Note that the two approaches with hyperparameter optimization both make a prediction for the opening, high, low and closing price in a daily interval. Therefore, we show their results

[4]See: https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6

[5]Note that Auto-sklearn normally uses holdout in stead of cross validation.

side by side. We do note that the results of the two approaches have a different setup, namely, hyperparameter optimization with random search has fixed budget on the number of iterations and combined algorithm selection and hyperparameter optimization with Auto-sklearn has a fixed budget on running time. We have run every experiment 10 times for all approaches, with varying random seeds (0 to 9). We used 10 different seeds, since our algorithms used randomization to induce a model. Setting a fixed random seed ensures that our research can be reproduced.

Significance To give proper conclusions, we calculated the significance of the results. We took the mean of the 10 achieved accuracy scores on every dataset and ran a statistical test to significantly compare all the models. We used the Friedman test, where every machine learning model gets a different rank on every dataset, based on their accuracy score [9]. The Friedman test compares different models with the null-hypothesis that every model is equivalent. The best performing model is ranked first, the worst performing model is ranked last. Ties get an average rank. We then took the average rank for every model. A critical distance was calculated based on these average ranks. When the average ranks of two classifiers are not at least the critical distance removed from each other, the models do not significantly differ. This post-hoc analysis is called the Nemenyi test [9]. An example of a figure that a Nemenyi test induces can be found in Figure 3. Models that not significantly differ are connected with the tick line. The critical distance is dependent on the significance level α. When α increases, the critical distance decreases. We used a standard setting of 0.05 for α.

5.2 Backtesting

With backtesting, an offline real-world trading strategy is simulated and reports reflect performance in profit and risk. We have used the backtesting platform Lean, from QuantConnect[6], which is an open source platform. We have chosen QuantConnect for its open source, the active community, and for the clean and informative reports. Note that QuantConnect, in backtesting mode, simulates a real-world trading environment where transaction costs and opening-hours are taken into account. Another advantage of QuantConnect is that the backtested trading strategy can be easily deployed to the live market. Since trading through a broker with live market data is out of the scope of this research, this feature is neither analyzed nor reported.

The trading strategy uses the same data as the trend prediction, however, with a slightly different time range. The trading strategy uses the dataset with the highest accuracy score achieved at the trend prediction. By seeing the results of the trend prediction, the trend prediction testing set did become part of the training set for backtesting. It would have been unfair to use the same testing set for the trading strategy, because future information would have been available. The data range is therefore from 2012 up to and including 5/2019 in daily intervals. Only the last 5 months will be used for testing, since this data has not been used for trend prediction. The backtesting part will only be in daily intervals.

Furthermore, we will calculate the technical indicators on the training set and fit the model. The testing set enters one by one, where the technical indicators are calculated for a single row and

[6]See: https://www.quantconnect.com/lean/

a prediction is made with the machine learning model. If the prediction is assigned the positive class, i.e., the closing price increases within a specific interval, a long position will be taken (currency A is bought and currency B is sold). If the prediction is assigned the negative class, i.e., the closing price decreases or stays the same within a specific interval, a short position is taken (currency B is bought and currency A is sold). Due to the short position it is possible to make a profit while the closing price of the currency pair is decreasing over a specific time interval. Note that this simple strategy will invest the total capital with every decision and a short or long position is taken every interval.

This strategy will be compared with the no-change strategy, which makes the same decision as the previous interval, e.g., if the price over the previous interval increased, the no-change strategy predicts it will increase within the following interval as well and takes a long position. If the price in the previous interval decreased or stayed the same, it will take a short position.

The start currency will be the base currency, i.e., the first currency A of the currency pair (A, B). Note that it is not possible in QuantConnect to start your capital in any other currency than US dollars. As a consequence, QuantConnect tries to buy euros with US dollars at the currency rate EUR/USD. At the end of the trading sessions, the euro's must be exchanged back to US dollars. To bypass the exchange from euro's to US dollars and back, the name of EUR/NOK was changed to USD/NOK. Furthermore, to simulate the real-world with transaction fees, the brokerage model InteractiveBrokers is used.

We imported our own data in QuantConnect. Since we did not download the weekend data from Dukascopy, QuantConnect fills this missing data with the last available data: Friday. This gave us three duplicate days per week in the data to train the machine learning model. Therefore, we removed the duplicate rows. The duplicate rows will not be available in the testing set, since every day enters one by one. Thus, we checked if the pricing and volume information was not equal to the last row, otherwise no decision and no trade is made. This did not influenced the technical indicator values. Because the technical indicators WADL and MACD did not returned the expected value with the data stream, they were left out of the trading strategy.

6 Results

In the following, we will show the accuracy scores in comparison with baselines and reflect on notable patterns. We will first show the trend prediction and then the backtesting, i.e., trading strategy.

6.1 Trend prediction

The results of the machine learning approaches will be shown in box plots, e.g., Figure 2. The title shows the time interval, which target price was predicted and what approach was run. The x-axis of the box plot show the 32 different currency pairs from Table 2. In Figure 2, two bars belong to one currency pair, in contrary to the box plots of the optimized hyperparameters/algorithm selection, where three bars belong to one currency pair (with default hyperparameter only two approaches are compared and with optimized hyperparameters/algorithm selection three approaches are compared). The y-axis shows the accuracy of the algorithm used, i.e., the number of correct predictions divided by the total number of predictions. The red line indicates the performance of the constant classifier that always predicts the majority class. Note that this constant classifier could be $true$ or $false$. The majority is measured over the testing set and could be different from the majority in the training set. Therefore, it is always ≥ 0.50. The lime (green) line indicates the performances of the no-change classifier that always follows the direction of the previous time interval.

The colored center part of the box plot represents 50% of the accuracy scores and is called the interquartile. In this case the interquartile consists of 5 achieved accuracy scores from one model, since every box plot bar is made out of 10 runs, with the median shown as black line in between. The maximum (25%) and the minimum (25%) of the accuracy scores are indicated by the vertical lines attached respectively above and below the interquartile. Some models have outliers, represented as a diamond shaped dot. An accuracy score is considered an outlier if it is 1.5 times the interquartile spread removed from the lower or upper bound of the interquartile.

6.1.1 Default hyperparameters

Daily Figure 2 shows the box plot where the models had the task to predict the direction of the closing price within a daily interval, with default hyperparameters. Results show that random forests outperform the constant classifier for 20 of the 32 datasets and gradient boosting for 18 of the 32 datasets. Furthermore, random forests have a wider spread of values than gradient boosting and gradient boosting outperforms random forests on average in 24 of the 32 datasets. In addition, random forests improve the constant classifier on average by 0.68% and gradient boosting with 1.41%. However, if we look at the significance test of the models (see Figure 3), we see that random forests and gradient boosting do not significantly differ from the majority nor from each other. However, the models do significantly outperform the no-change classifier. The highest accuracy score achieved in this run is 0.591. It is an outlier of gradient boosting on the currency pair EUR/NOK. The highest accuracy score of gradient boosting without an outlier can be found on the currency pair EUR/CHF (0.589). Since the majority of this dataset on this task is 0.504, this is on average the biggest gain in accuracy score (8.06%) as well. Therefore, we consider this the best

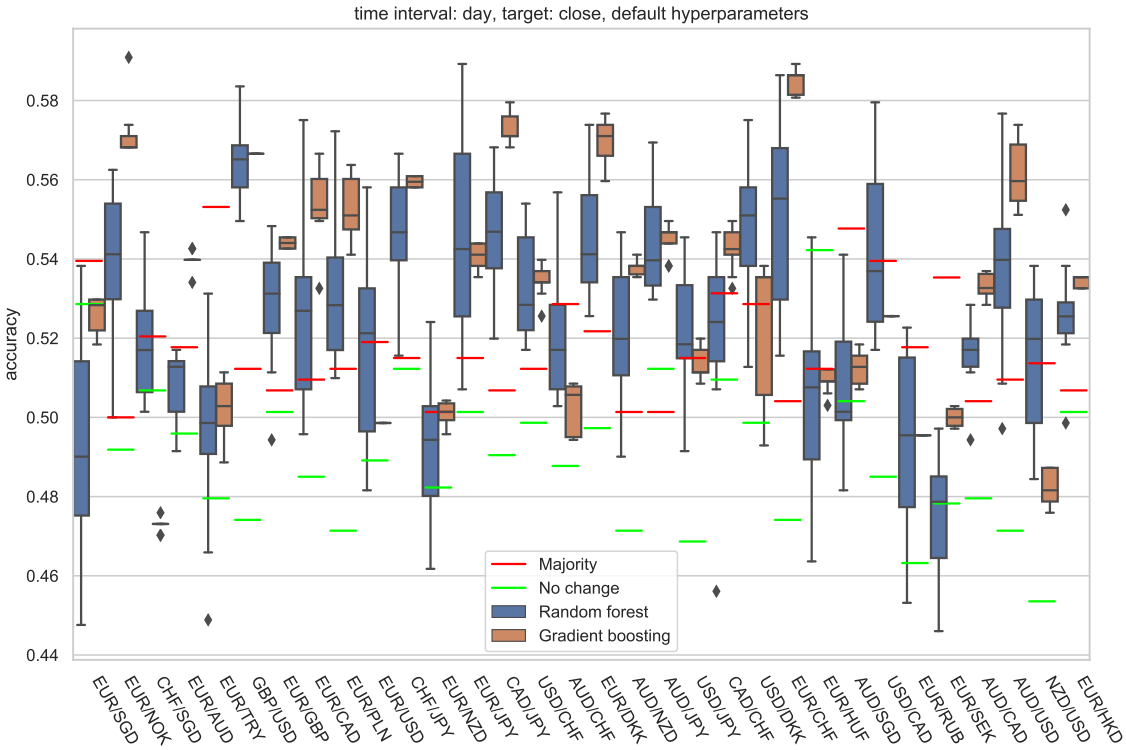


Figure 2: Box plot of machine learning prediction with default hyperparameters, daily time interval and closing price.

predictable currency pair, over the measured time frame. The majority outperforms the currency pair EUR/SEK the most (5.37%). This currency pair has the lowest accuracy score in this task (0.446) as well. Therefore, we consider this the worst predictable currency pair, over the measured time frame.

Hourly The results from the prediction on the closing price within an hourly interval and default hyperparameters, can be found in Figure 4a and Figure 4b. In Figure 4a, two datasets, EUR/HUF and EUR/RUB, have a very high majority accuracy score. Consequently, the spread of the different models are visually difficult to compare and they are left out in Figure 4b. These two data sets get a majority score of 0.52 when comparing average improvements and when running the Friedman test. The results reflect that the accuracy score of the majority classifier is on average 0.517. This means that the closing price volatility at an hourly interval is high, i.e., the price direction fluctuates often. Furthermore, random forests and gradient boosting perform on average better than the majority on 28/32 and 26/32 datasets, respectively. In addition, the models in an hourly interval have on average a slightly better performance than in a daily interval (0.03%). Random forests achieve on this task again a wider accuracy spread and gradient boosting (1.47% improvement relative to majority) outperforms random forests (0.67% improvement relative to majority) on average by 0.80%. Different from the results for a daily interval, we see that gradient boosting significantly differ from the majority classifier (see Figure 5). However, random forests is still not significantly better than the majority. When we compare gradient boosting and random forests, we

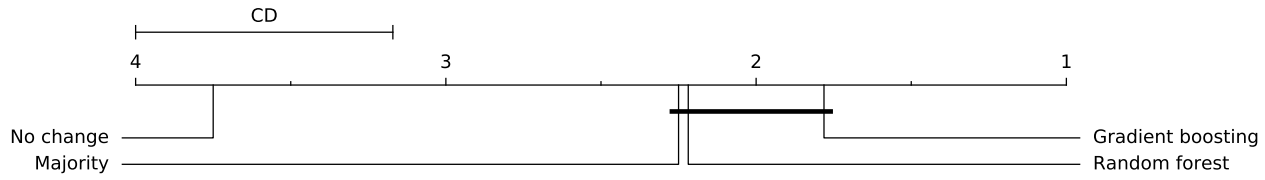
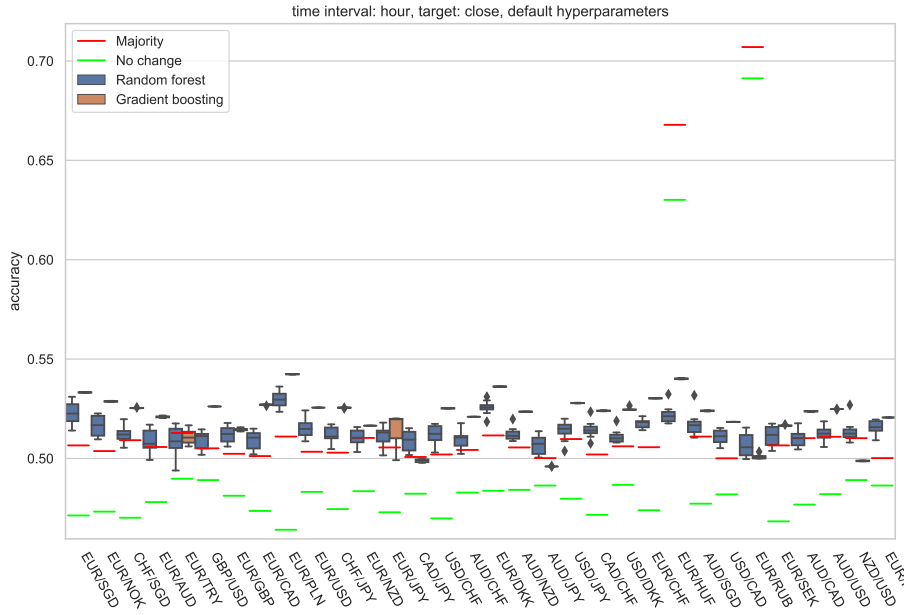


Figure 3: Comparison of all default hyperparameter machine learning models and baselines against each other with the Nemenyi test, in daily time interval and on closing price. Models that do not significantly differ (distance smaller than CD; critical distance) are connected.

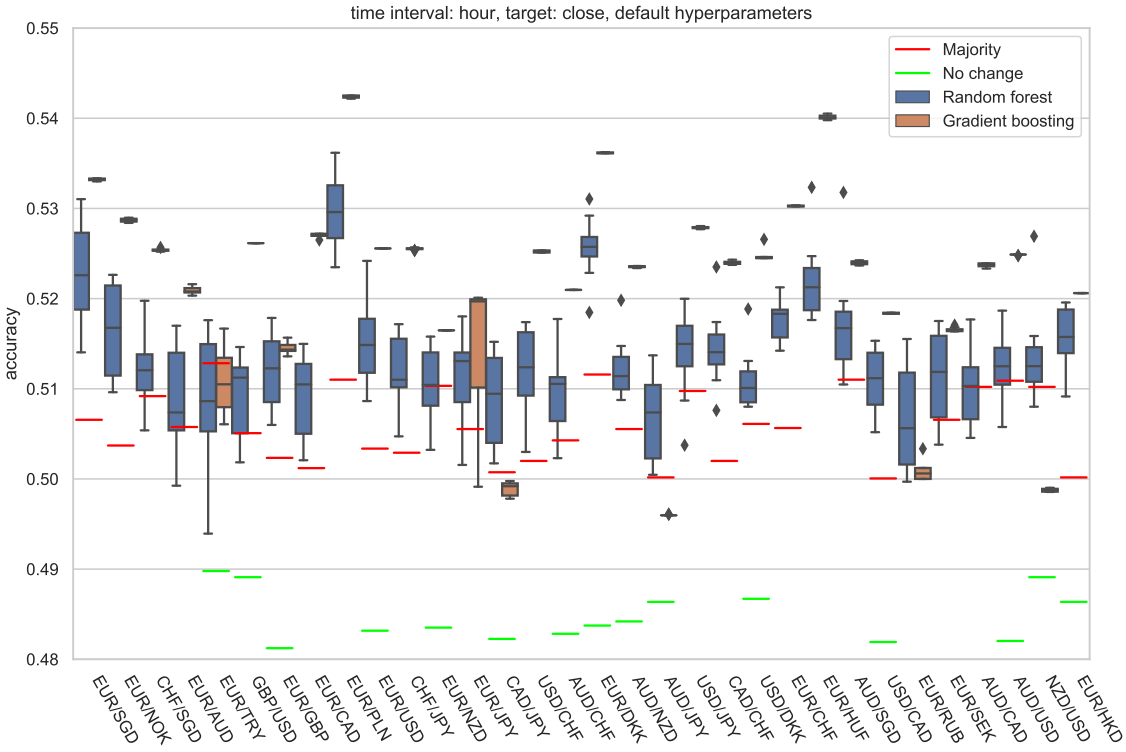
see no significant difference as well. However, the no-change classifier is significantly worse than the models. The highest accuracy at an hourly interval is lower than at a daily interval and can be found on the currency pair EUR/PLN, with a score of 0.543. This is achieved with gradient boosting and outperforms the majority with 3.14%. EUR/PLN is therefore considered the best predictable currency pair, over the measured time frame. The worst accuracy score compared to the majority is on the currency pair EUR/RUB, where the majority is 0.707. This is thus considered the worst predictable currency pair, over the measured time frame.

Per-minute Figure 6a shows the results for the per-minute interval with the prediction on the closing price. Both random forests (0/32) and gradient boosting (8/32) are not often performing better than the majority. This means that the features are not good predictors for the direction of the closing price within the per-minute interval with default hyperparameters. To obtain better insights between the datasets that have a more equal distribution in price direction, Figure 6b has a y-axis with accuracy scores between 0.50 and 0.56. EUR/HUF and EUR/RUB get a majority score of 0.53 when comparing average improvements and when running the Friedman test. In addition to daily and hourly intervals, gradient boosting (0.47% improvement relative to majority) outperforms random forests (0.56% decreased relative to majority) on average by 1.03% and has a smaller accuracy spread than random forests. The significance test in Figure 7 shows that the accuracy score achieved by random forests is significantly different from that observed for the majority and gradient boosting, however, significantly worse. Gradient boosting on the other hand, is not significantly worse than the majority. The highest accuracy score is achieved on the currency pair EUR/DKK, with a score of 0.693. Since the majority is very high as well, we did not consider this as an improvement. Gradient boosting outperforms the majority the most on the currency pair EUR/SGD, with an improvement of 0.75% and is therefore considered the best predictable currency pair, over the measured time frame. The majority outperforms the currency pair EUR/HUF on average by 21.49%, which is more than on any other currency pair. Therefore, we classify this currency pair as worst predictable, over the measured time frame.

When we compare the three time intervals with each other, we see that the hourly interval can be best predicted, followed by the daily interval and the per-minute interval is hard to predict. We did see that the average performance difference between daily and hourly intervals was not big (0.03%). However, note that gradient boosting in the hourly interval was the only significant better performing model. We speculate that the technical indicators in per-minute intervals have a low level of granularity, i.e., it adds no information whether the price increases or decreases within a



(a)



(b)

Figure 4: Box plot of machine learning prediction with default hyperparameters, hourly time interval and closing price. 4a with no fixed y-axis and 4b with fixed y-axis).

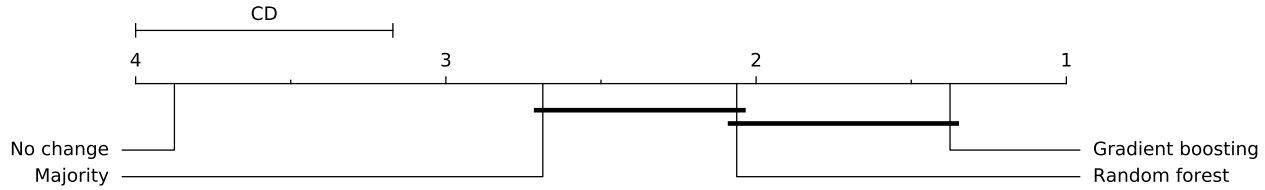


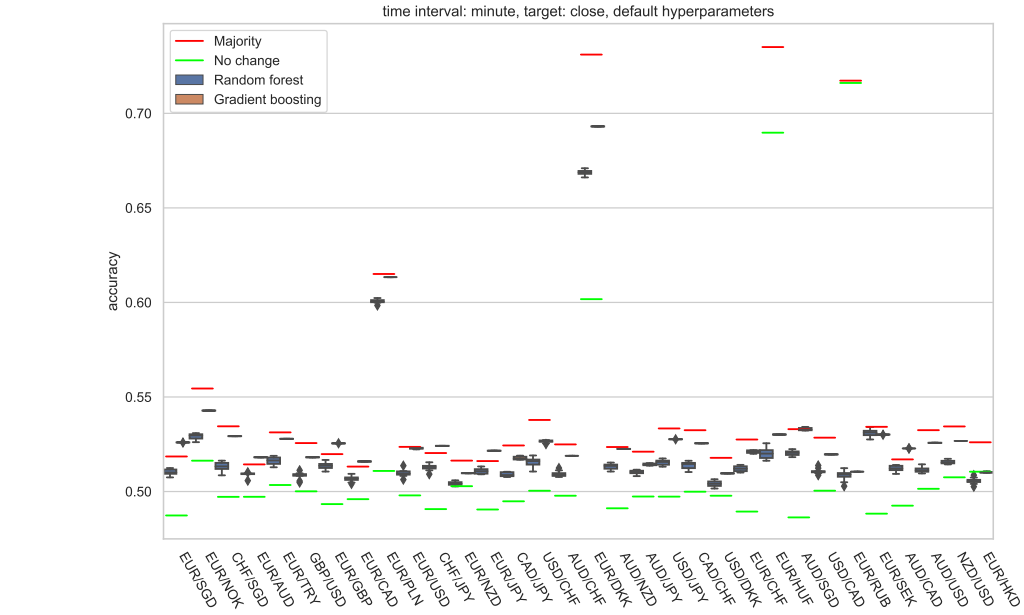
Figure 5: Comparison of all default hyperparameter machine learning models and baselines against each other with the Nemenyi test, in hourly time interval and on closing price. Models that do not significantly differ (distance smaller than CD; critical distance) are connected.

per-minute interval. We suspect that adding higher resolution data, e.g., seconds, will increase the prediction for the per-minute interval. In a daily and hourly interval, the level of granularity is higher and this results in significant higher accuracy scores. The no-change classifier is always the worst performing classifier, compared to the models with default hyperparameters. In daily and hourly intervals it is significantly worse than all classifiers, however, it is significantly equal in the per-minute interval with random forests.

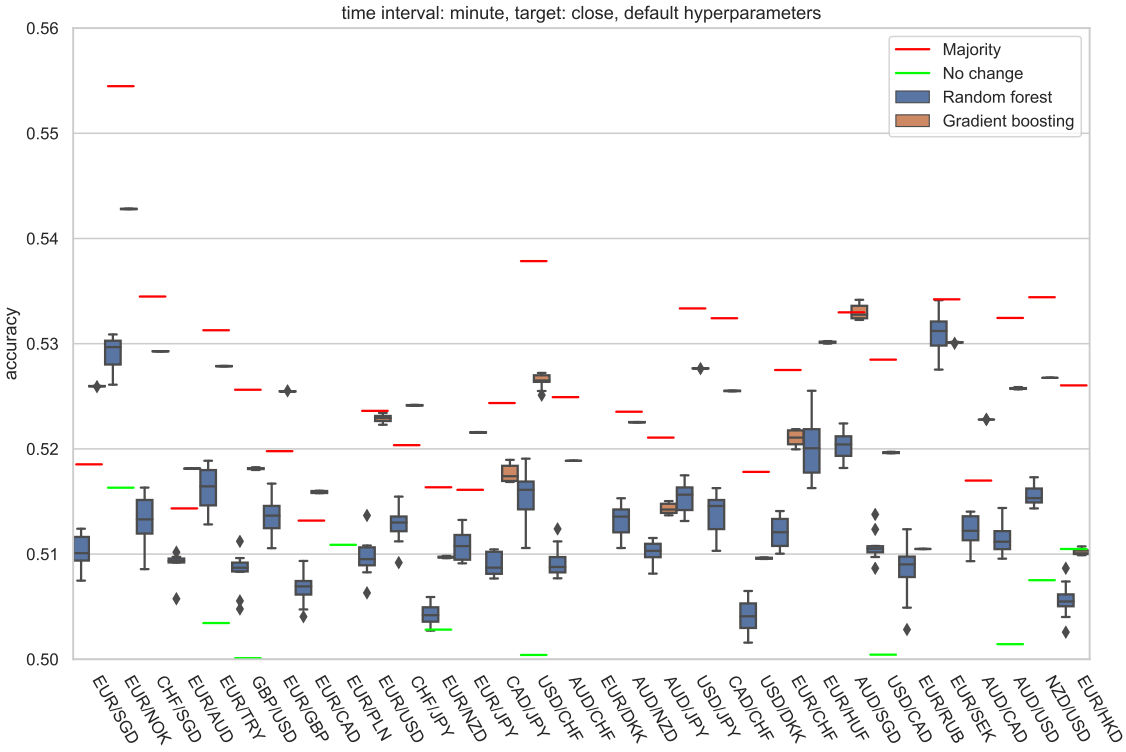
6.1.2 Optimized hyperparameters/algorithm selection

Close In the following, we present the results for optimized hyperparameters/algorithm selection. In Figure 8, the results of the closing price prediction in a daily interval is shown. This is the same setup as showed in Figure 2 and can therefore be compared well. First of all, random forests (23/32), gradient boosting (26/32) and Auto-sklearn (27/32) perform better than the majority classifier. It is not a surprise that this is an improvement in comparison with the daily interval with default hyperparameters (Figure 2). In addition, the models perform better than the no-change classifier as well. Moreover, the model Auto-sklearn outperforms the majority class on average by 3.06%, random forests with optimized hyperparameters by 2.57% and gradient boosting with optimized hyperparameters by 2.10%. Random forests outperform gradient boosting, in contrast with the findings for default hyperparameters. We saw that the spread of random forests became smaller than gradient boosting as well. Furthermore, we see that Auto-sklearn (16 currency pairs) often achieves the highest average accuracy score, followed by gradient boosting (9 currency pairs) and then random forests (7 currency pairs). Figure 9 shows that random forests and Auto-sklearn reflect significant difference with the majority classifier, while gradient boosting does not. However, there is no significant difference between the three models. Note that optimized random forests are significant compared to default random forests. On the currency pair GBP/USD, the highest accuracy score, apart from the outliers, is achieved by gradient boosting (0.626). Since the spread of this model is wide, we speculate that this score would not be achieved often. Therefore, the currency pair EUR/NOK seems to be the best predictable currency pair since on average, random forests outperform the majority class with 9.46%, gradient boosting with 8.18% and Auto-sklearn with 7.87%. The worst predictable currency pair is EUR/HUF, because the majority classifier outperforms this dataset with 2.35%.

We see that Auto-sklearn outperforms random forests and gradient boosting with hyperparam-



(a)



(b)

Figure 6: Box plot of machine learning prediction with default hyperparameters, per-minute time interval and closing price. 6a with no fixed y-axis and 6b with fixed y-axis.

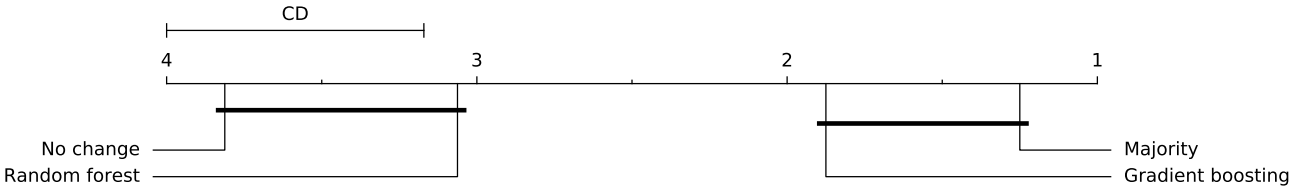


Figure 7: Comparison of all default hyperparameter machine learning models and baselines against each other with the Nemenyi test, in per-minute time interval and on closing price. Models that do not significantly differ (distance smaller than CD; critical distance) are connected.

eter optimization by 0.49% and 0.96%, respectively. This tells us that Auto-sklearn, with an overall wall-clock time budget of 3600 seconds, is able to find a better predictor than using random forests and gradient boosting after 100 iterations of random search, where the recorded running time was always between 1274 and 4864 seconds.

Open As mentioned in Section 5, we research the opening, high and low prices as well. In Figure 10, the opening price prediction is shown for a daily interval. The first pattern that stands out are the high accuracy scores, while the majority is on average 0.520. The spreads of all box plots are small, i.e., the achieved accuracy scores are relatively constant, and the models with optimized hyperparameters/algorithm selection outperform the constant classifier on average by 42.18%, which is a lot higher than seen in previous results. The reason why the models perform so well on the opening price is as follow: the closing price of the current interval needs to be known in order to calculate the technical indicators. Since the Forex market is open 24 hours per day and 5 days per week, the difference between the closing price of the current interval and the opening price of the following interval tends to be small. Only on weekends the gap is bigger and we assume that this weekend gap is the reason that the accuracy score is on average 0.942 and not very close to 1.0.

Results show that Auto-sklearn and random forests perform both best on average (16/32), followed by gradient boosting (8/32). Note that the total number of wins adds up to more than 32. This tells us that the average of the models are equal for some of the currency pairs. It seems natural to assume that these results are significantly better than the majority and the no-change classifiers and a significance test confirmed this (see Figure 11). The highest accuracy score (0.980) is achieved on the currency pair AUD/USD, with all three classifiers. The lowest (0.831) accuracy score is achieved on the currency pair EUR/RUB, with gradient boosting.

High The high price comparisons in a daily interval with optimized hyperparameters/algorithm selection can be found in Figure 12. The models perform always better than the majority (a few outliers not taken into account), improving on average by 18.18%. For this prediction, random forests (15/32) outperform Auto-sklearn (14/32) and gradient boosting (3/32). In addition, random forests outperform the majority on average by 18.45%, Auto-sklearn by 18.27% and gradient boosting by 17.81%. The significance test shows, according to expectations, that all three classifiers perform significantly better than the majority and the no-change classifiers (see Figure 13). On the contrary of the other price comparisons, the no-change classifier performs better than the majority,

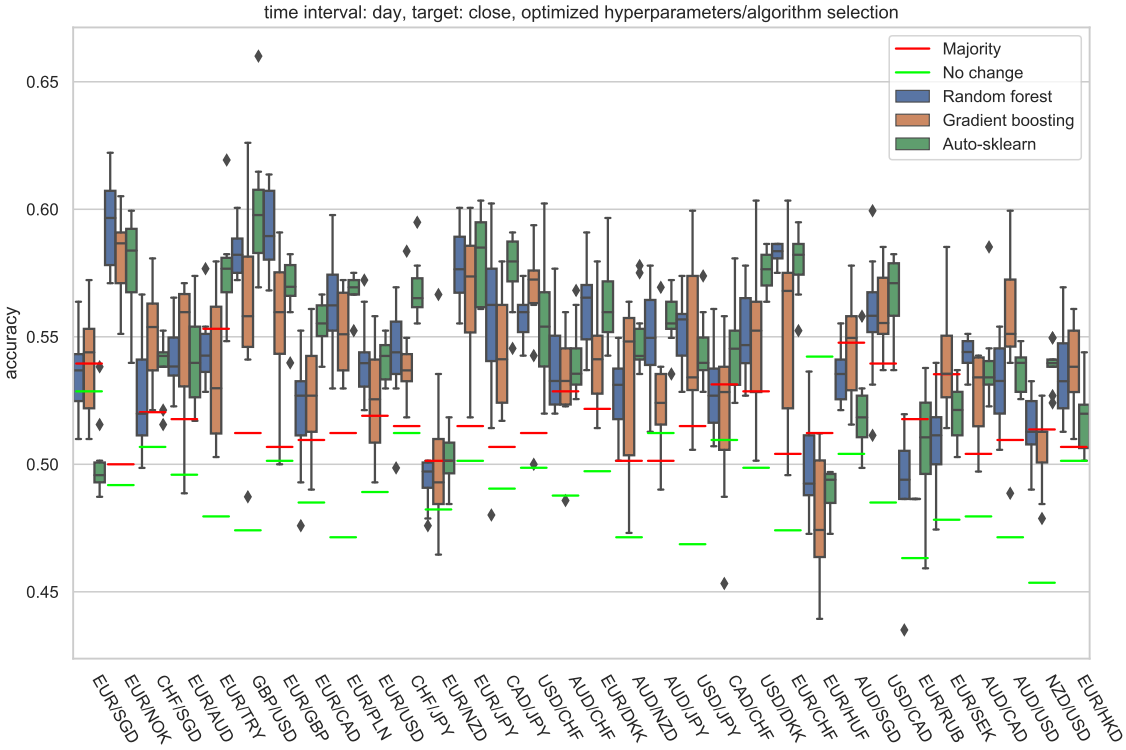


Figure 8: Box plot of machine learning prediction with optimized hyperparameters/algorithm selection, daily time interval and closing price.

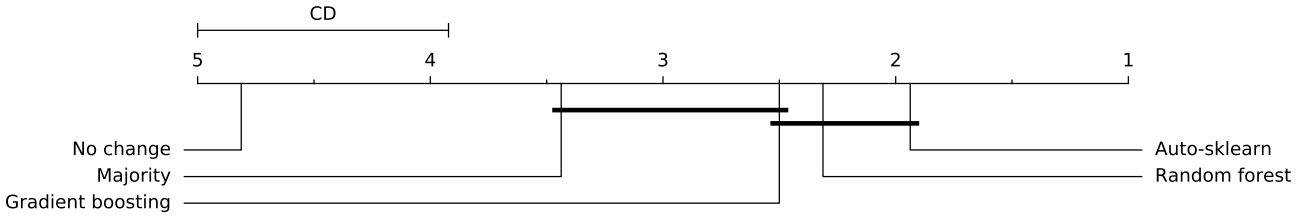


Figure 9: Comparison of all hyperparameter optimized machine learning models and baselines against each other with the Nemenyi test, in daily time interval and on closing price. Models that do not significantly differ (distance smaller than CD; critical distance) are connected.

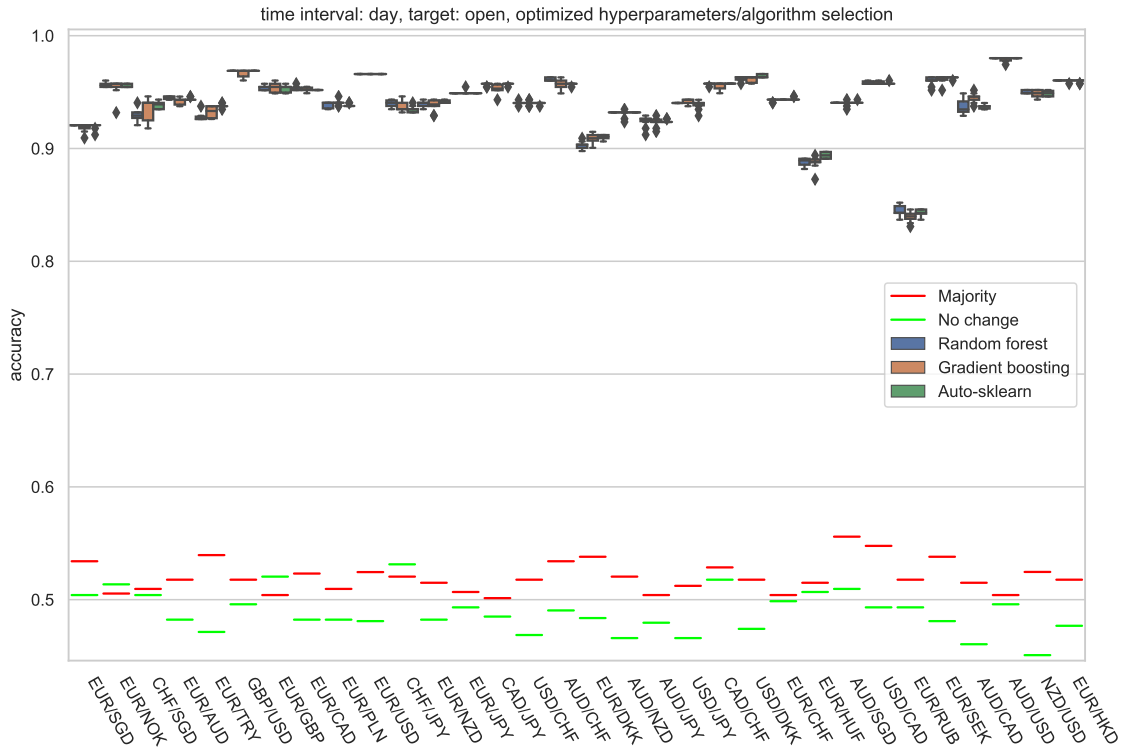


Figure 10: Box plot of machine learning prediction with optimized hyperparameters/algorithm selection, daily time interval and opening price.

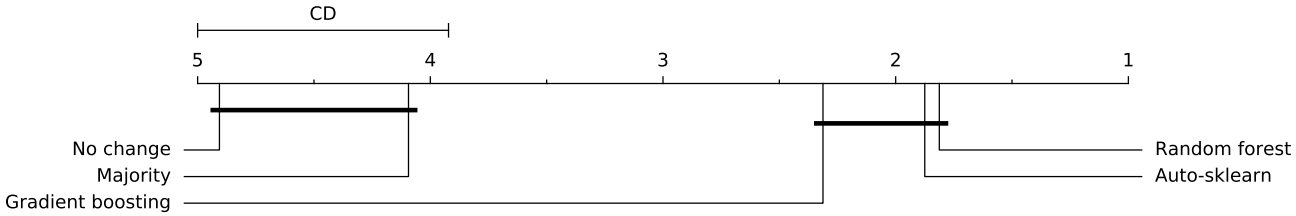


Figure 11: Comparison of all hyperparameter optimized machine learning models and baselines against each other with the Nemenyi test, in daily time interval and on opening price. Models that do not significantly differ (distance smaller than CD; critical distance) are connected.

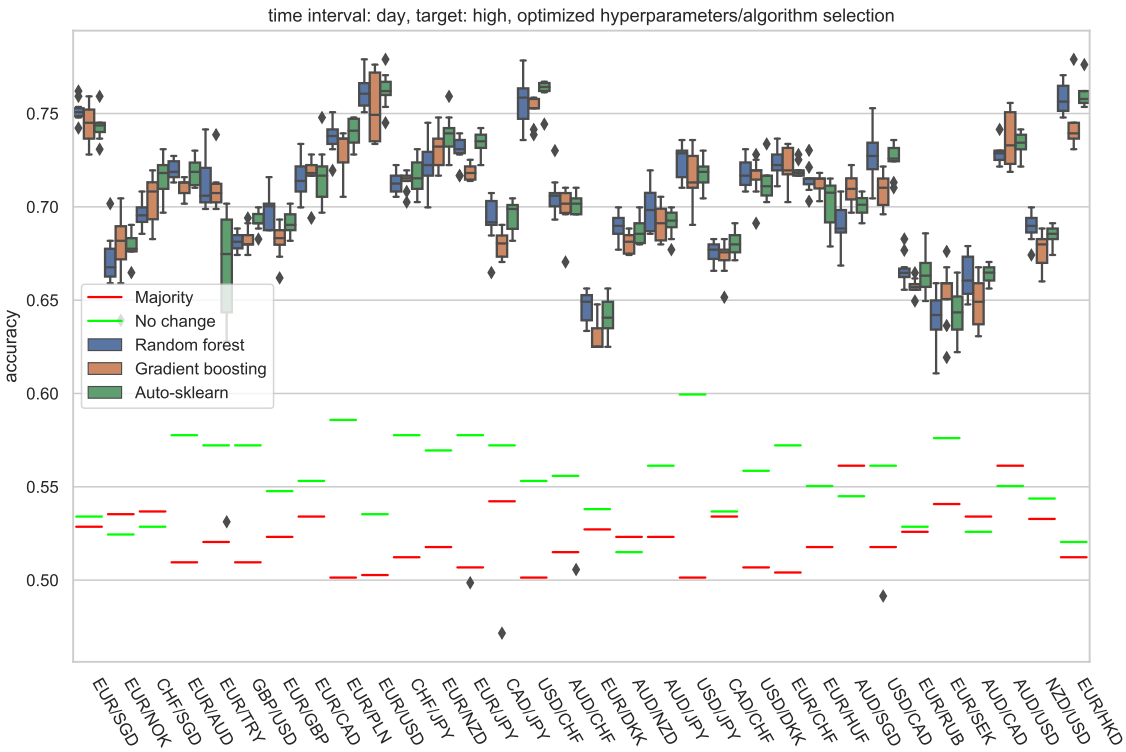


Figure 12: Box plot of machine learning prediction with optimized hyperparameters/algorithm selection, daily time interval and high price.

however, with an insignificant difference. Two currency pairs, EUR/USD and EUR/HKD, have the highest accuracy score (0.779). However, Auto-sklearn outperforms the majority of currency pair USD/CHF on average the most, with an improvement of 26.11%. With the lowest accuracy score (0.611) and one of the worst in respect to outperforming the majority (10.55%), EUR/SEK is the hardest to predict currency pair, over the measured target.

One particular important accuracy score to notice is the score achieved on the currency pair USD/JPY. Baasher & Fakhr [3] used the same classification task, namely, the closing price in a daily interval. Their study included the currency pair USD/JPY as well, and they observed a score between 0.690 and 0.778, depending on the machine learning technique used. As seen in Figure 12, their highest score (0.778) was not achieved in our experiments; our results show a maximum score of 0.736. We do note that many accuracy scores with optimized hyperparameters/algorithm selection fall within the same spread. One explanation for their higher accuracy score could be the misinterpretation and missing of technical indicators. Unfortunately, the calculation of the technical indicators was not explained well in their work, which led to own interpretation and less reproducibility. Even after contacting the authors, the technical indicators could not be clarified. Furthermore, Baasher & Fakhr [3] used three currency pairs which are not available in Dukascopy, which led to comparison on a single currency pair. Therefore, the comparison between our and their research could not be made fairly; still, there is no evidence that our results fall significantly short of theirs.

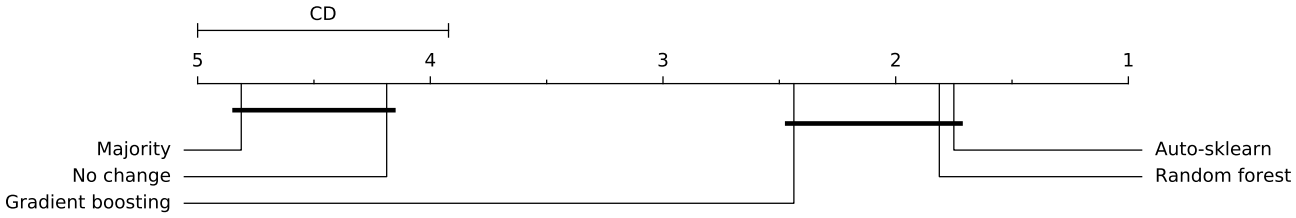


Figure 13: Comparison of all hyperparameter optimized machine learning models and baselines against each other with the Nemenyi test, in daily time interval and on the high price. Models that do not significantly differ (distance smaller than CD; critical distance) are connected.

Low The final box plot reflects the results of the prediction task for the low price in a daily interval with optimized hyperparameters/algorithm selection. In terms of accuracy scores, this is comparable with the high prediction task, with an average improvement of the majority class by 16.25%. We see that random forests (12/32) often achieve the highest average accuracy score, followed by gradient boosting (11/32) and Auto-sklearn (9/32). Figure 15 shows, according to expectations, significant differences between the three models and the majority/no-change classifier. The highest accuracy is achieved by the Auto-sklearn model on the currency pair NZD/USD, with an accuracy of 0.796. Comparing to the high price, this is an improvement of 1.70%. The biggest on average out-performance in respect to the majority is accomplished on the currency pair USD/DKK, with gradient boosting (24.70%). EUR/TRY is the hardest currency pair to predict, where the average improvement of the models in respect to the majority is just 8.73%. Furthermore, we see that the distribution of the majority scores is high (0.531) in comparison with the three other time intervals.

We see similarities in performance of models predicting high and low prices. The average accuracy scores are close to each other, which implies that the high and low price can be predicted equally good with our technical indicators. We speculate that the reason for better performance of the high and low in respect to the closing price is as follows: the closing price needs to be known in order to calculate the technical indicators. The closing price is future information in respect to the high and low prices, since the closing price occurs later (or with a small possibility at the same time) within a specific time interval. This future information could say something about the following high or low price, e.g., if the closing price is close to the high, this could indicate that the high price will increase within the following time interval, since there is a rising trend. For the low price, it would be exactly the other way around; if the closing price is close to the low, this could indicate a falling trend where the low price will decrease within the following time interval.

6.1.3 Running time

An important consideration is running time. When two models have the same accuracy score, whereas one is faster than the other, it can be argued that the faster model is preferable. Figure 16 shows the running time of the machine learning models. The y-axis shows the running time in seconds on a logarithmic scale.

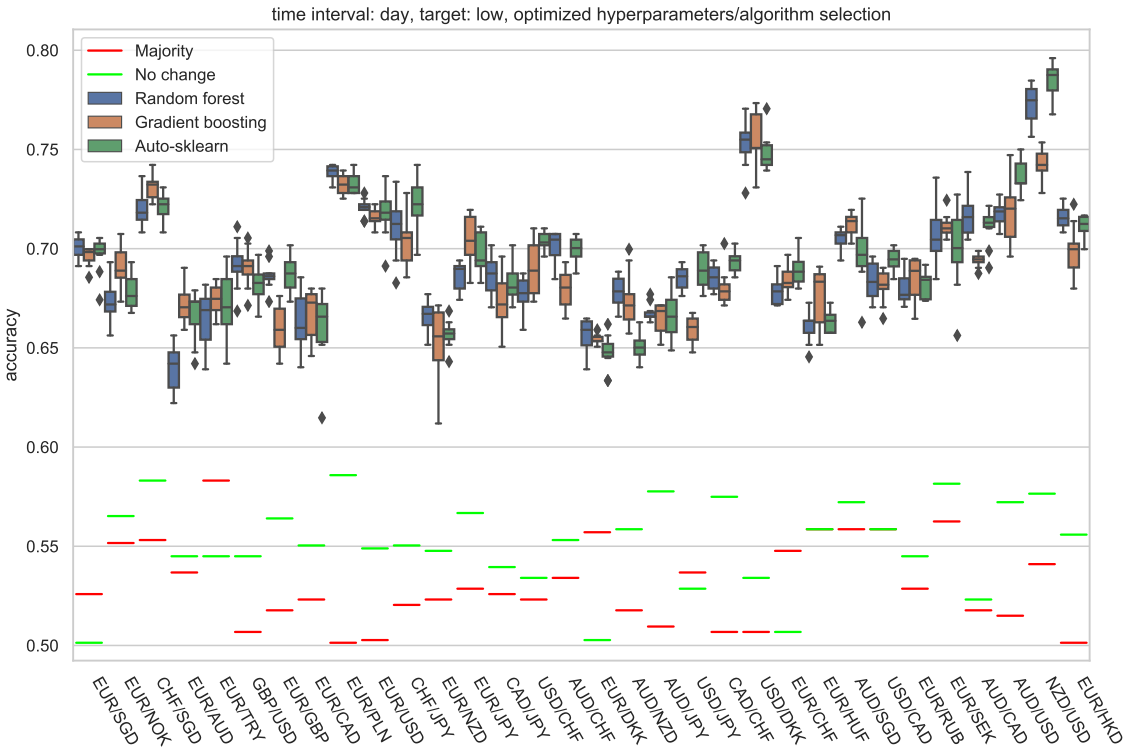


Figure 14: Box plot of machine learning prediction with optimized hyperparameters/algorithm selection, daily time interval and low price.

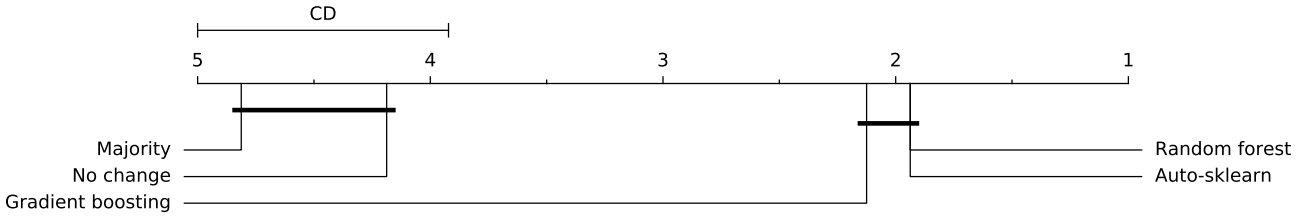


Figure 15: Comparison of all hyperparameter optimized machine learning models and baselines against each other with the Nemenyi test, in daily time interval and on the low price. Models that do not significantly differ (distance smaller than CD; critical distance) are connected.

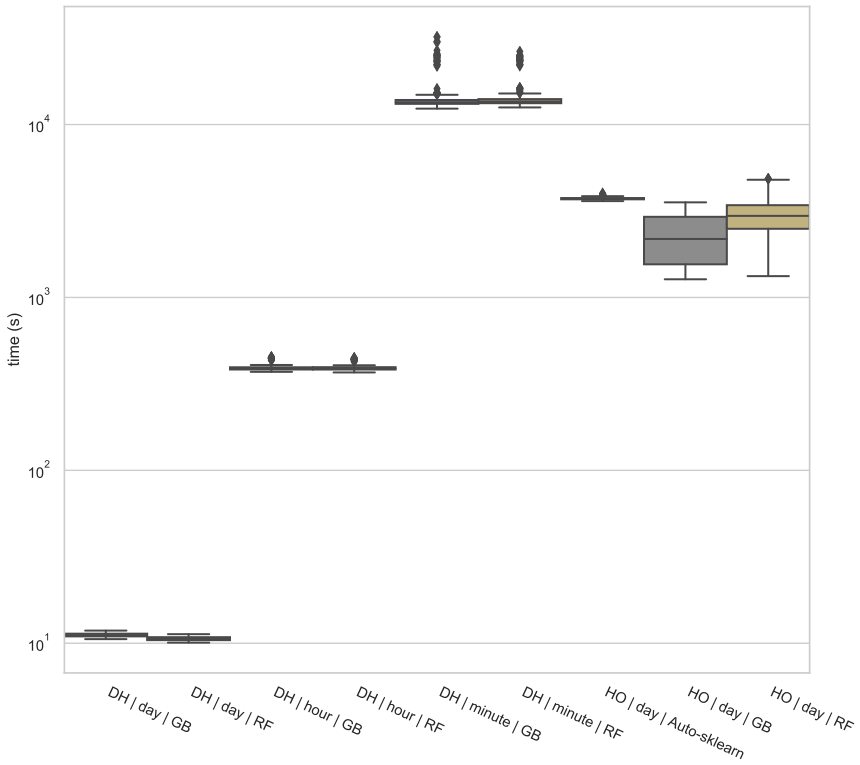


Figure 16: Box plot of running time for individual machine learning methods. DH stands for default hyperparameters, HO stands for hyperparameter optimization, RF stands for random forest and GB stands for gradient boosting. The y-axis is in a logarithmic scale.

This figure reflects that random forests with default hyperparameters are more or less as fast as gradient boosting. In contrast, the differences in running time between the different time intervals are remarkable. While the number of rows in the dataset in an hourly interval has a multiplier of 24 relative to the daily interval (since there are 24 hours in a day, and the Forex is traded 24 hours a day), the running time multiplier is on average almost 36. This remarkable pattern is found between hourly and per-minute intervals as well. The number of rows multiplier is 8.5 since there are 60 minutes in one hour, however, only one year is taken for the minute, while the day and hour have 7 years, so 60/7. Nevertheless, the running time multiplier is on average more than 37.

Auto-sklearn has the highest average running time of the three models with hyperparameter optimization. However, we see that gradient boosting and random forests have a wider spread in running time. The upper bound of 100 iterations of random search exceeds the upper bound of Auto-sklearn. Moreover, the average running time of Auto-sklearn is 3726. Since the model fitting of Auto-sklearn is fixed to 3600 seconds, it takes on average 126 seconds to calculate the technical indicators and test the performance.

Table 6: The best performing hyperparameters we have used in our trading strategy.

hyperparameter	value
random_state	6
bootstrap	*True*
criterion	'*gini*'
max_features	0.9341811650828181
min_samples_leaf	17
min_weight_fraction_leaf	0.0
n_estimators	100

6.2 Backtesting

The results of trend prediction (Section 6.1) showed that the most promising model is random forest with hyperparameter optimization on the currency pair EUR/NOK. The backtesting results reflect the performance of this model on this currency pair. The hyperparameters of this model are shown in Table 6.

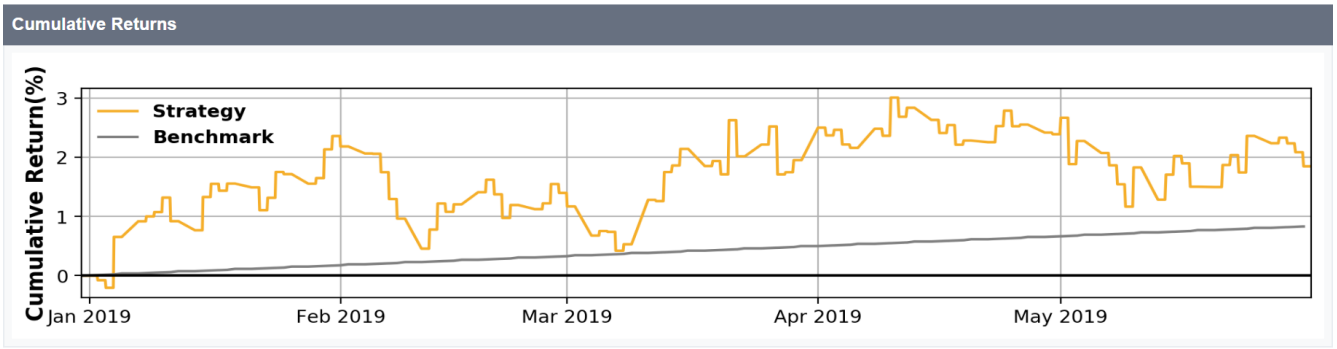
QuantConnect generates various statistics. We will discuss the most important ones. We will start with the actual profit. In less than five months time, a net profit of €1,846 has been made in 108 daily trades with a starting capital of €100,000. This comes down to a 1.85% return on investment and on annual basis this will be 4.45%. The transaction fees were a total of €228.

Another important statistic is the Sharpe ratio, which was 0.841 in this strategy. As explained in Section 4.2, the Sharpe ratio is dependent on the benchmark as well. Therefore, we have made the comparison between the strategy and the benchmark visible in Figure 17a. This shows us that the strategy is performing better than the benchmark.

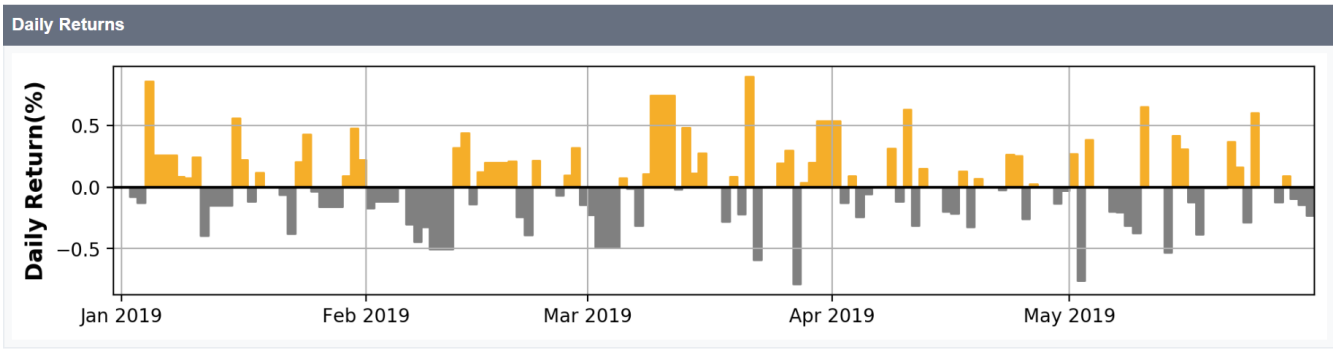
Figure 17b clearly shows that the right decision is not made everyday. The grey bars indicate the days where a wrong decision is made between taking a long or a short position. If we then look at Figure 17c, we see that the strategy almost always chose for a short position. Unfortunately, we did not have time to check whether this is in agreement with the results of the trend prediction, although it does raise some concern.

The no-change strategy achieved some different results. The same number of trades are made, however, instead of a profit a loss of €8,345 is made. This strategy achieves a -8.35% return on investment in less than five months. On annual basis, this will be a loss of 19.0%, according to QuantConnect. The Sharpe ratio is -3.359, which is much lower than our strategy. Due to the loss, this strategy performs worse than the benchmark, which is visible in Figure 18a. The figure reflects that the no-change strategy was not performing extremely bad in the first 2.5 months. In this period, the no-change strategy did not made a profit, however, it did not made a big loss as well.

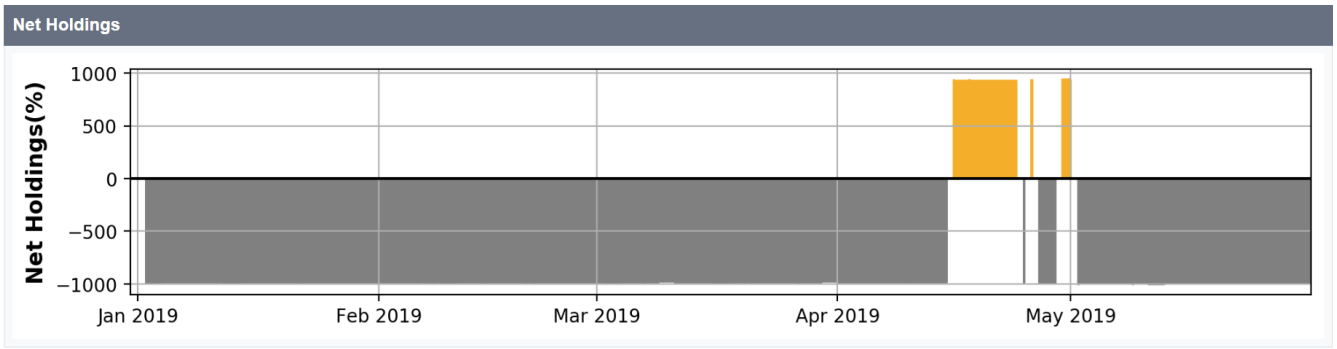
Figure 18b shows what profit and loss is made on a daily basis with the no-change strategy. The total volume of the grey bars is more than the orange bars, which is in line with the negative return on investment. Since the no-change strategy will follow the distribution, the long/short ratio should be close to 1. This can be seen in Figure 18c, where about half of the bars indicate a long position and the other half indicate a short position.



(a) Returns of the machine learning strategy in comparison with the benchmark.

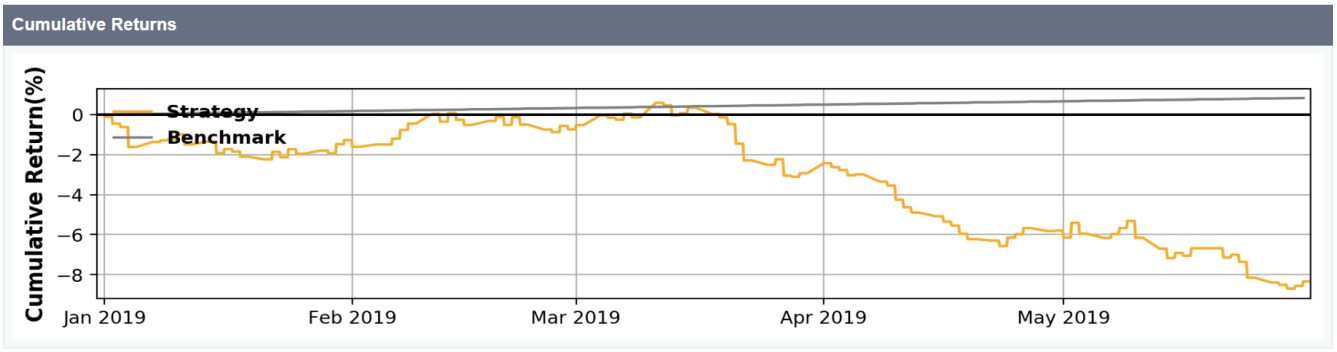


(b) Daily returns of the machine learning strategy in %. An orange bar indicates the right decision for that day, a grey bar indicates a wrong decision.

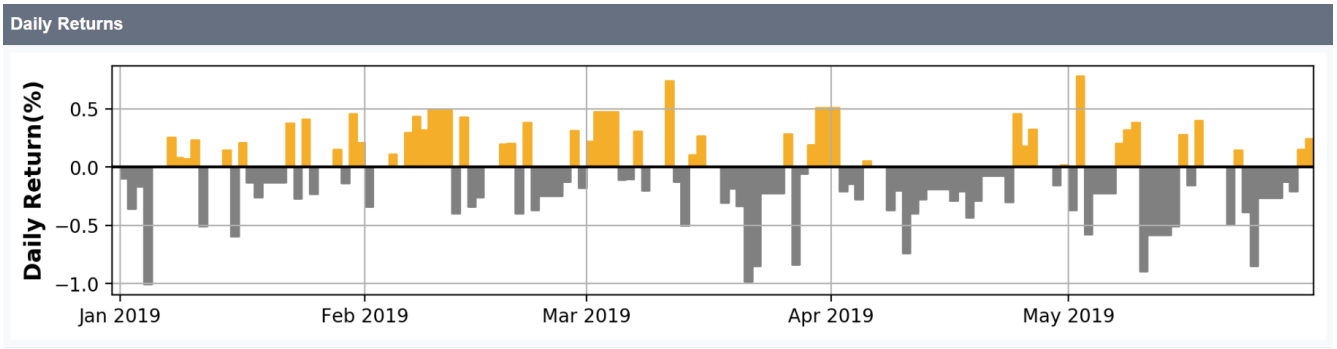


(c) Net holdings of the machine learning strategy, which indicate whether a strategy took a long or short position. A long position was taken if the net holdings are above zero, otherwise a short position is taken.

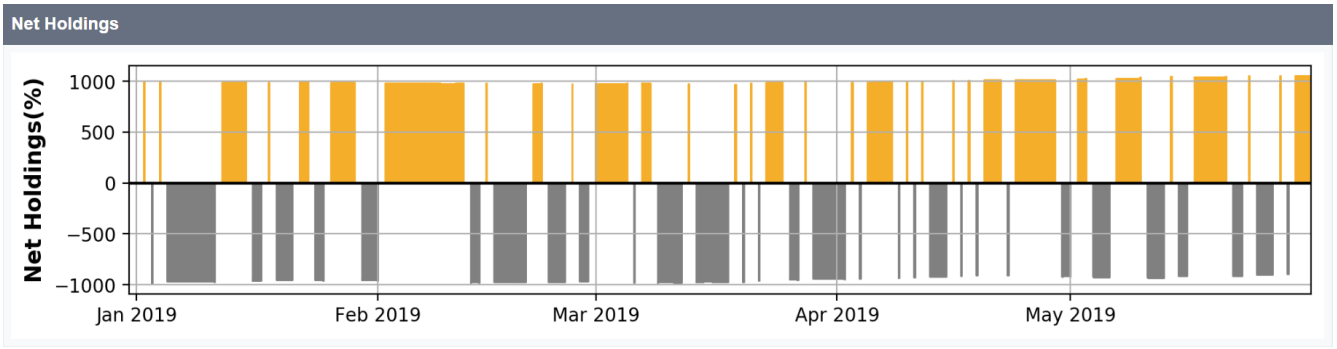
Figure 17



(a) Returns of the no-change strategy in comparison with the benchmark.



(b) Daily returns of the no-change strategy in %. An orange bar indicates the right decision for that day, a grey bar indicates a wrong decision.



(c) Net holdings of the no-change strategy which indicate whether a strategy took a long or short position. A long position was taken if the net holdings are above zero, otherwise a short position is taken.

Figure 18

7 Conclusions

We have researched the predictability of 32 currency pairs in the Forex market. For this purpose, we have collected open, high, low, close and volume attributes from Dukascopy and calculated different technical indicators. We defined a binary classification problem whether the price of one currency increased or decreased/stayed the same relative to another currency over a given time interval. Machine learning was applied to the technical indicators to induce a model and predict the target. We have used three different machine learning approaches for this task: default hyperparameters, 100 iterations of random search for hyperparameter optimization and a combination of algorithm selection and hyperparameter optimization. The data was split into a training and testing set containing 80% and 20% respectively.

In order to significantly compare the different models, majority and no-change classifiers, we have run a Friedman test. The results showed that the hourly interval can be best predicted with default hyperparameters, closely followed by the daily interval. We saw that the per-minute interval was the hardest to predict in our setup. Gradient boosting performed better than random forests with default hyperparameters. Moreover, most of the machine learning models did not significantly outperform the majority. In addition, the differences between the models were insignificant.

From the results with optimized hyperparameters/algorithm selection, we saw that the opening price can be best predicted. However, we did note that this cannot be exploited in a trading strategy. The models on the high and low prices achieved high accuracy scores as well, however, they can also not be exploited. We did not achieved the highest accuracy score of Baasher and Fakhr [3], however, we noted that the comparison could not be made under equal conditions. Furthermore, we saw that random forests outperformed gradient boosting with optimized hyperparameters. We concluded from both the default hyperparameters and optimized hyperparameters that gradient boosting needed less optimization, however, random forests achieved the best results when optimization was applied. We were most interested in the closing price and results reflected that the optimized models perform in general significantly better than the majority and the no-change classifiers.

In addition to the accuracy scores, we saw differences in running time. Firstly, we concluded that the relation between the number of rows in a dataset and the running time is not proportionate, the running time increased more than the number of rows. When combining running time and model performance, we saw that random search is on average faster than Auto-sklearn. If running time is (highly) critical, it may be better to use random forests with random search than Auto-sklearn; however, the upper bound of random search still exceeds Auto-sklearn. If time is not of concern, Auto-sklearn will be the better option, according to our results.

We tested our results in an offline real-world trading strategy with QuantConnect. We took the most promising model on the best predictable currency pair and exploited it in a trading strategy. The testing set was therefore different than in the trend prediction part. We compared our trading strategy with the no-change classifier. The results reflected that the strategy made an actual profit, where transaction fees were taken into account as well. Furthermore, our strategy performed better than the no-change strategy. Unfortunately, the Sharpe ratio was not high, which indicated that the risk of our strategy was relatively high. In addition, the yearly growth rate of 4.45% was not spectacular.

8 Further Research

Notably was the difference between the daily and hourly interval runs, with default hyperparameters. The models in the hourly interval performed better than in a daily interval; however, we did not optimize the hyperparameters in the hourly interval. The comparison between these two time intervals could be very interesting, as well as in terms of profit in the trading strategy.

Furthermore, the current results can be optimized. Therefore, we came up with two different techniques. The first technique is meta-learning, where trends in historic datasets are used to make predictions for new datasets [5, 11]. In contrast to our research, meta-learning takes into consideration correlations between datasets to improve predictions. It seems natural to assume that exchange rates between various currency pairs are correlated and this can be exploited. This can especially be useful when two currency pairs share the same currency, e.g., EUR/USD and USD/JPY. Another meta-learning technique could be exploited if more information than only currency pairs is used, e.g. the gold price to predict the Australian Dollar. The gold price and the Australian Dollar are correlated [2], which could be exploited.

Another technique for further improvement is optimizing the technical analysis. Many of the technical indicators have one or multiple parameters, representing the width of the time window the indicator is based on. In most current studies, this window is fixed. However, there is a great opportunity for automated machine learning methods to optimize in this parameter space. Whether this should be considered a two-stage or joint optimization problem is an open research question.

This research could ultimately lead to a trading strategy with low risk and high returns. It can also be extended with other financial data, e.g., stocks and bonds. Moreover, the features can be extended with non-financial data, e.g., news and Twitter.

By making the assembled benchmark suite publicly available, we encourage further study of this scientific challenge.

References

[1] Achelis, S. B. *Technical Analysis from A to Z*. McGraw Hill New York, 2001.

[2] Apergis, N., and Papoulakos, D. The australian dollar and gold prices. *The Open Economics Journal 6* (2013), 1–10.

[3] Baasher, A. A., and Fakhr, M. W. Forex trend classification using machine learning techniques. In *11th WSEAS International Conference on Applied Computer Science* (2011), WSEAS, pp. 41–47.

[4] Blume, L., Easley, D., and O'hara, M. Market statistics and technical analysis: The role of volume. *The Journal of Finance 49*, 1 (1994), 153–181.

[5] Brazdil, P., Giraud-Carrier, C., Soares, C., and Vilalta, R. *Metalearning: Applications to data mining*. Springer Science & Business Media, 2009.

[6] Breiman, L. Random forests. *Machine Learning 45*, 1 (2001), 5–32.

[7] Choi, J. Y., Salandro, D., and Shastri, K. On the estimation of bid-ask spreads: Theory and evidence. *Journal of Financial and Quantitative Analysis 23*, 2 (1988), 219–230.

[8] Dempster, M. A., and Leemans, V. An automated fx trading system using adaptive reinforcement learning. *Expert Systems with Applications 30*, 3 (2006), 543–552.

[9] Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research 7*, Jan (2006), 1–30.

[10] Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. Efficient and robust automated machine learning. In *Advances in neural information processing systems* (2015), pp. 2962–2970.

[11] Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (2017), JMLR. org, pp. 1126–1135.

[12] Friedman, J. H. Stochastic gradient boosting. *Computational statistics & data analysis 38*, 4 (2002), 367–378.

[13] Hamed, I. M., Hussein, A. S., and Tolba, M. F. An intelligent model for stock market prediction. *International Journal of Computational Intelligence Systems 5*, 4 (2012), 639–652.

[14] Kim, K.-j. Financial time series forecasting using support vector machines. *Neurocomputing 55*, 1-2 (2003), 307–319.

[15] Lo, A. W., Mamaysky, H., and Wang, J. Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *The Journal of Finance 55*, 4 (2000), 1705–1765.

[16] McKinney, W., et al. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference* (2010), vol. 445, Austin, TX, pp. 51–56.

[17] Ntakaris, A., Magris, M., Kanniainen, J., Gabbouj, M., and Iosifidis, A. Benchmark dataset for midprice forecasting of limit order book data with machine learning methods. *Journal of Forecasting 37*, 8 (2018), 852–866.

[18] Oliphant, T. E. Python for scientific computing. *Computing in Science & Engineering 9*, 3 (2007), 10–20.

[19] Ozturk, M., Toroslu, I. H., and Fidan, G. Heuristic based trading system on forex data using technical indicator rules. *Applied Soft Computing 43* (2016), 170–186.

[20] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[21] Sharpe, W. F. The sharpe ratio. *Journal of portfolio management 21*, 1 (1994), 49–58.

[22] Sidehabi, S. W., Tandungan, S., et al. Statistical and machine learning approach in forex prediction based on empirical data. In *2016 International Conference on Computational Intelligence and Cybernetics* (2016), IEEE, pp. 63–68.

[23] Thornton, C., Hutter, F., Hoos, H. H., and Leyton-Brown, K. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (2013), ACM, pp. 847–855.

[24] Valcu, D. Using the Heikin-Ashi technique. *Technical analysis of stocks and commodities -magazine edition- 22*, 2 (2004), 16–29.

[25] Vanschoren, J., van Rijn, J. N., Bischl, B., and Torgo, L. Openml: Networked science in machine learning. *SIGKDD Explorations 15*, 2 (2013), 49–60.

[26] Villa, S., and Stella, F. A continuous time bayesian network classifier for intraday fx prediction. *Quantitative Finance 14*, 12 (2014), 2079–2092.

[27] Yang, D., and Zhang, Q. Drift-independent volatility estimation based on high, low, open, and close prices. *The Journal of Business 73*, 3 (2000), 477–492.