Towards Automated Technical Analysis for Foreign Exchange Data

Fabian G. B. Schut[1], Jan N. van Rijn[1], and Holger H. Hoos[1,2]

[1] Leiden Institute of Advanced Computer Science, Leiden University, the Netherlands
[2] Department of Computer Science, University of British Columbia, Canada

Abstract. This work describes an initial study towards automating trading strategies for foreign currency pairs. Generally, a stream of bid and ask prices is converted in an unsupervised fashion into a pre-defined set of so-called technical indicators that allow machine learning algorithms to induce a model. This in itself is a data-wrangling task. We assembled a benchmark suite containing 32 currency pairs. We ran an initial experiment with a fixed set of technical indicators. The results show that we can outperform the majority classifier by a small margin, consistently across datasets. We outline how we intent to apply advances in automated machine learning to automate the process of generating technical indicators.

Keywords: Automated Machine Learning, meta-learning, Foreign Exchange, Automated Trading

1 Introduction

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 AI [10]. Related work has shown that the quality of stock price prediction can be more accurate with technical analysis [4, 12], i.e., statistical properties on recent intervals of the data.

There have been applications of machine learning to various well-defined subproblems of foreign exchange trading. In the following, we discuss several studies that inspired our research. Most notably, Dempster & Leemans [6] proposed a reinforcement learning technique. Their system deployed a proper trading strategy and obtained a profit of more than 26% per annum over two years with high-frequency data, on a single dataset. Baasher & Fakhr [2] study the binary classification problem whether the price of a currency will increase or decrease over a given time interval. They apply technical indicators and report that the natural prediction class, whether the closing price of an interval will rise or not, is not very predictable, therefore they define a task predicting whether the *high* price of a time interval will rise. Although their results seem solid, no viable trading strategy can be applied to such predictions. Sidehabi & Tandungan [15] view the problem as a regression task and train a support vector regressor to predict the actual value. Indeed, when working with numeric targets, regression is a natural fit for this problem. Although visual inspection suggests that the results are decent, the study is quite small in scale and does not implement any baseline methods for comparison, making it hard to assess the significance of the contribution.

We study the binary classification problem whether the currency price will increase or decrease over a specific period of time. We have assembled a benchmark suite with data obtained from Dukascopy[3], containing 32 currency pairs. For all currency pairs, we assembled datasets with daily, hourly and per-minute intervals. All currency pairs are recorded over the same period (daily intervals and hourly intervals: 2012-2018; per-minute intervals: 2018), so these can be merged to utilize meta-learning and transfer learning techniques. Our contributions are as follows: (i) We apply state-of-the-art hyperparameter optimization techniques (including Auto-sklearn) to the various datasets and report on performance compared to baselines; (ii) we outline how automated data science and meta-learning can be used to further improve predictive results; and (iii) we make the benchmark suite available on OpenML[4], for the community to participate in this scientific challenge on a homogeneous collection of datasets.

2 Machine Learning using Technical Indicators

For any given pair of currencies (A, B), trading data from a broker typically comes in a triplet of the follow information: timestamp, bid price and ask price. The ask price specifies what the trader wants to pay (expressed in currency B) for a unit of currency A, and the bid price in currency B the price at which the trader sells a unit of currency A. Note that the ask price is always higher than the bid price – this is how the market-maker makes profit. A natural machine learning problem is to predict whether the price increases or decreases within a specific time interval. The problem is often abstracted to fixed intervals – in the case of this study: days, hours and minutes. For each interval, for both the ask price and the bid price, the following information is collected: (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*: the liquidity at Dukascopy during the interval in millions. This results for each interval in 10 predictive features (5 based on the bid price, 5 based on the ask price). We denote the average of the bid and ask price at the open, close, high, low and volume in interval i as O_i, C_i, H_i, L_i and V_i, respectively. The high or low over multiple intervals is denoted by a range in subscript, e.g., $H_{[i-n,i]}$ denotes the highest price over the n intervals up to, and including i.

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

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

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
Simple Moving Average	$SMA_i^c = \frac{C_{i-n+1}+C_{i-n+2}+\dots+C_i}{n}$ (can be defined on open, high and low as well)	[1]
Stochastic Oscillator	$\%K = \frac{C_i - L_{[i-n+1,i]}}{H_{[i-n+1,i]} - L_{[i-n+1,i]}}$, $\%D = \frac{\%K_{i-n+1}+\%K_{i-n+2}+\dots+\%K_i}{n}$	[1]
Momentum	$C_i - C_{i-n}$	[11]
Price Rate of Change	$\frac{C_i - C_{i-n}}{C_{i-n}}$	[1]
Williams %R	$\frac{H_{[i-n+1,i]} - C_i}{H_{[i-n+1,i]} - L_{[i-n+1,i]}}$	[1]
Weighted Closing Price	$\frac{C_i \times 2 + H_i + L_i}{4}$	[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]
Moving Average Convergence, Divergence	Exponential moving average, placing a greater weight on the most recent data points	[1]
Commodity Channel Index	The variation over the sum of H_i, L_i and C_i from its statistical mean	[1]
Bollinger Bands	Upper- and lowerband: two standard deviations shifted respectively up or down from SMA_i	[1]
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)	[16]
Mean Open & Close	$\frac{O_{i-n+1}+O_{i-n+2}+\dots+O_i+C_{i-n+1}+C_{i-n+2}+\dots+C_i}{2n}$	-
Variance Open & Close	Variance over $\{O_{i-n+1}, O_{i-n+2}, \dots, O_i, C_{i-n+1}, C_{i-n+2}, \dots, C_i\}$	-
High Price Average	Simple moving average over the high	-
Low Price Average	Simple moving average over the low	-
High, Low Average	Simple moving average over the sum of high and low	-
Trading Day Price Average	Simple moving average over the sum of the open, high, low and close	-

Generally, machine learning models are induced based on a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, \dots, n\}$ to map an input $\mathbf{x}$ to output $f(\mathbf{x})$, which closely represents y. In this study, we view this as a classification problem, where the y-values represent whether the average of the two closing price (bid and ask) at a given interval will increase within a specific time interval. Note that this is a binary classification problem, as the price will either (i) increase (positive class); or (ii) stay the same or decrease (negative class). The $\mathbf{x}$-values are in this case the technical indicators. As such, the trained model predicts the y-value based on a small window from the past. Note that this problem can generally be addressed as a data stream task [3]. In the data stream setting, the order of the data is relevant. After a certain interval has been processed, the information (and class-label) becomes available to the model and can be used for future predictions. For each observation, technical indicators [1] can be defined. Table 1 shows a subset of the technical indicators that we considered in this research. On top of this set, we also added a set of technical indicators from the 'Technical Analysis' library.[5] Note that several of these are parameterized (by size of the time window). For those, we followed the parameterization of [2]. However, a great challenge lies in properly tuning these parameters; this is considered future work.

[5] `https://github.com/bukosabino/ta`

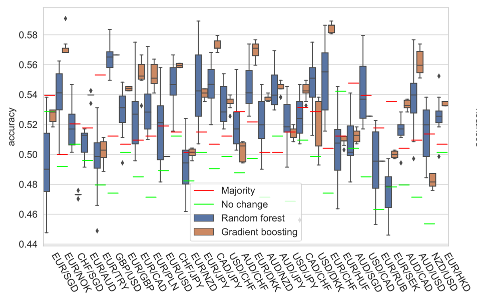

Fig. 1. Default hyperparameters, daily.

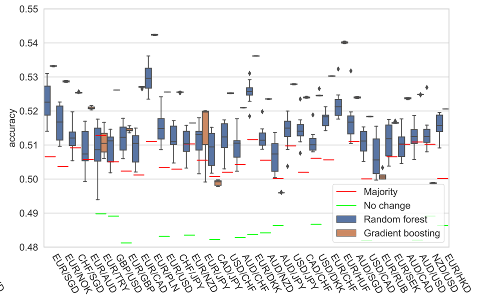

Fig. 2. Default hyperparameters, hourly.

3 Results

We present results for two experimental settings: (i) default hyperparameters, and (ii) optimized hyperparameters. For the first setting, we used random forests and gradient boosting with the default hyperparameters from Scikit-Learn [13]. The results reflect the different patterns in daily, hourly and per-minute intervals. Figure 1 and Figure 2 show the results of daily and hourly intervals, respectively. Result plots for per-minute intervals are omitted due to a lack of space, however, they can be found in [14]. For the second setting, we optimized random forests and gradient boosting using 100 iterations of random search (the recorded run time was always between 1,274 and 4,864 seconds). We also ran Auto-sklearn [8], using an overall wall-clock time budget of 3,600 seconds. This setup was run on a daily interval for the closing price. The results are shown in Figure 3. Note that due to the different parameterizations, it is hard to fairly compare the Auto-sklearn and random search results. Every experiment was run 10 times, with varying random seeds. Each figure plots the 32 datasets on the x-axis (currency pairs, abbreviated to their ISO-4217 code), and the respective accuracy scores on the y-axis. The boxes plot the spread across the random seeds. We used hold-out with the first 80% of the ordered observations for training and the final 20% of the ordered observations for testing. The hyperparameters are internally optimized using 5-fold cross-validation on the train set. The red line indicates the performance of the constant classifier that always predicts the majority class measured over the test set and the green line indicates the performance of a classifier that predicts the same direction as seen in the previous interval.

In the results we see different patterns in the prediction task. Firstly, the hourly interval can be best predicted, improving over the constant classifier on average by 0.67% (random forest) and 1.47% (gradient boosting). The daily interval improves over the constant classifier on average by 0.68% (random forest) and 1.41% (gradient boosting). The per-minute interval is hard to predict, gradient boosting outperforms the constant classifier on average by 0.47%, whereas the constant classifier outperforms the random forests by 0.56%. We speculate that the minute data is too volatile to predict with the current resolution of the data. Although these numbers seem small on average, we note that im-

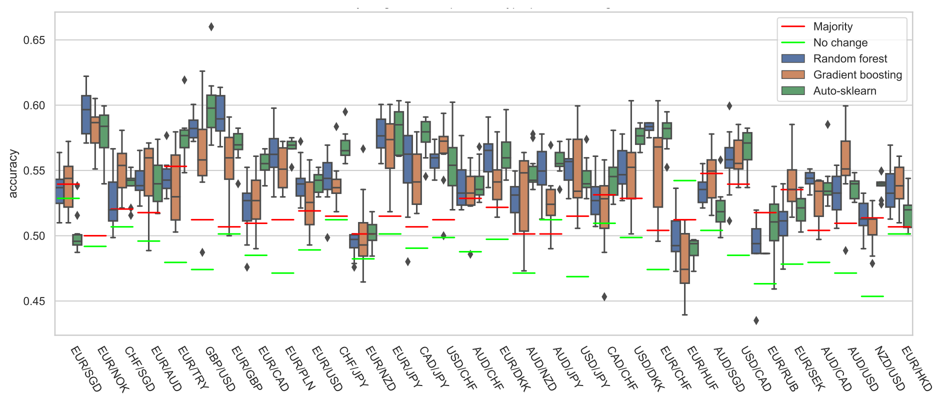

Fig. 3. Optimized hyperparameters, daily interval.

provements are consistent across datasets. A Nemenyi test [7] showed significant difference between the classifiers and the no-change. Moreover, gradient boosting is significantly better than the constant classifier in the hourly interval and Auto-sklearn and random forests are significantly better than the constant classifier in the daily interval with optimized hyperparameters.[6] Furthermore, some currency pairs can be very well predicted, e.g., Euro to Norwegian Krone. Secondly, gradient boosting obtains higher accuracy scores than random forests with default hyperparameters, i.e., 0.80% for daily intervals, 0.73% for hourly intervals and 1.03% for per-minute intervals. On the contrary, random forest obtains a higher accuracy score than gradient boosting when hyperparameters are optimized, i.e., 0.47% for daily intervals. 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 forest (7 currency pairs). Naturally, we see that the hyperparameter optimization and Auto-sklearn obtain better accuracy than default hyperparameters; random forest improves by 1.89%, gradient boosting by 0.69%. Further experiments showed that a profit can be made from these improvements [14].

4 Future Work: Improvements and Automation

Clearly, automating data science can be used in several ways to further improve these results. In future work, we propose to use (i) meta-learning, where trends in historic datasets are used to make predictions for new datasets [5, 9]. Unlike the current work, 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. (ii) Optimizing the technical features. Many of the technical indicators have one

[6] Omitted for space reasons, for full details see [14].

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 AutoML methods to optimize in this parameter space. Whether this should be considered a two-stage or joint optimization problem is an open research question. 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. Baasher, A.A., Fakhr, M.W.: Forex trend classification using machine learning techniques. In: 11th WSEAS international conference on Applied computer science. pp. 41–47. WSEAS (2011)
3. Bifet, A., Gavaldà, R., Holmes, G., Pfahringer, B.: Machine learning for data streams: with practical examples in MOA. MIT Press (2018)
4. Blume, L., Easley, D., O'hara, M.: Market statistics and technical analysis: The role of volume. The Journal of Finance 49(1), 153–181 (1994)
5. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: Metalearning: Applications to data mining. Springer Science & Business Media (2009)
6. Dempster, M.A., Leemans, V.: An automated fx trading system using adaptive reinforcement learning. Expert Systems with Applications 30(3), 543–552 (2006)
7. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7(Jan), 1–30 (2006)
8. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28, pp. 2962–2970. Curran Associates, Inc. (2015)
9. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 1126–1135. JMLR. org (2017)
10. Hamed, I.M., Hussein, A.S., Tolba, M.F.: An intelligent model for stock market prediction. International Journal of Computational Intelligence Systems 5(4), 639–652 (2012)
11. Kim, K.j.: Financial time series forecasting using support vector machines. Neurocomputing 55(1-2), 307–319 (2003)
12. Lo, A.W., Mamaysky, H., Wang, J.: Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. The journal of finance 55(4), 1705–1765 (2000)
13. 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., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
14. Schut, F.: Machine Learning and Technical Analysis for Foreign Exchange Data with Automated Trading (2019)
15. 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. pp. 63–68. IEEE (2016)
16. Valcu, D.: Using The Heikin-Ashi Technique. Technical analysis of stocks and commodities -magazine edition- 22(2), 16–29 (2004)