UNIVERSITY OF WARSAW | FACULTY OF ECONOMIC SCIENCES

WORKING PAPERS
No. 15/2023 (422)

ENSEMBLED LSTM WITH WALK FORWARD OPTIMIZATION IN ALGORITHMIC TRADING

KAROL CHOJNACKI
ROBERT ŚLEPACZUK

WARSAW 2023

WORKING PAPERS 15/2023 (422)

Ensembled LSTM with Walk Forward Optimization in Algorithmic Trading

Karol Chojnacki[a], *Robert Ślepaczuk*[b]

[a] *University of Warsaw, Faculty of Economic Sciences*
[b] *University of Warsaw, Quantitative Finance Research Group, Department of Quantitative Finance, Faculty of Economic Sciences*
Corresponding author: rslepaczuk@wne.uw.edu.pl

Abstract: This study compares well-known tools of technical analysis (Moving Average Crossover MAC) with Machine Learning based strategies (LSTM and XGBoost) and Ensembled Machine Learning Strategies (LSTM ensembled with XGBoost and MAC). All models were compared to Buy and Hold benchmark and evaluated using Performance Metrics, that is Annualized Return Compounded, Maximum Drawdown, Maximum Loss Duration, and three types of Information Ratio. This research uses daily S&P 500 index data ranging from 2000 to 2023. Every strategy was optimized with novel walk forward approach consisting of numerous in sample and out of sample periods. MAC and best performing ML methods were subjected to sensitivity analysis. The results show that LSTM ensembled with XGBoost and MAC yields the most promising results in terms of risk-adjusted returns which suggest further research focused on ensembling of individual ML strategies. Finally, we show that classical methods of technical analysis (that is, MAC) are much less robust and indifferent to change in hyperparameters than machine learning based algorithms, especially LSTM.

Keywords: Algorithmic Investment Strategies, Machine Learning, Recurrent Neural Networks, Long Short-Term Memory, XGBoost, Walk Forward Optimization, Trading algorithms, Technical Analysis Indicators

JEL codes: C4, C14, C45, C53, C58, G13

Note: The article was prepared within the framework of a non-competitive project of the Ministry of Education "School of Eagles" based on the agreement between the Ministry of Education and the University of Warsaw, contract no. MNISW/2020/142/DIR/KH, signed on 26 May 2020. Project co-financed from the European Social Fund under the Operational Programme Knowledge Education Development 2014-2020.

1 Introduction

An attempt to reject the Efficient Market Hypothesis is a task often undertaken by theoreticians and market practicioners. An intermediate problem in examining market efficiency is aiming to answer the question: *Is there an investment strategy that, given a set of information Ω_τ at time τ, could bring above-average and statistically significant profits?* The answer to this question depends on the validity of the Efficient Market Hypothesis. If the Efficient Market Hypothesis is true, meaning that markets are efficient, then at any given time, the prices of securities fully reflect all available information about them. This means that there is no such investment strategy. In contrast, if the Efficient Market Hypothesis were not true, then a skillful investor would be able to find informational gaps and achieve above-average profits.

The aim of this study is to reject the Efficient Market Hypothesis in informational sense using a specifically designed investment strategy for the S&P 500 Index. The study will not only include an overview of the base case strategy, but also a very detailed sensitivity analysis in order to check the robustness of presented models, based on which additional conclusions will be drawn. During the research, the following research hypotheses will be verified:

- **Research Hypothesis 1** *The Efficient Market Hypothesis is true, and it is not possible to achieve above average risk-adjusted returns.*
- **Research Hypothesis 2** *Machine learning based algorithms yield higher risk-adjusted returns than technical analysis tools.*
- **Research Hypothesis 3** *Ensembling LSTM with other strategies yields better results than stand-alone machine learning or TA algorithms.*

The dataset used in this study comprises of the adjusted close of the S&P 500 index, obtained in a daily format from Yahoo Finance. The dataset used in this study comprises the adjusted close of the S&P 500 index, obtained in a daily format from Yahoo Finance. The S&P 500 index is a suitable choice for testing the trading algorithms LSTM, XGBoost, and Moving Average Crossover, as it represents a broad and diverse selection of 500 large-cap U.S. companies across various sectors. The index is highly liquid, allowing for easier trade execution and tighter bid-ask spreads. Additionally, the S&P 500 is considered one of the most efficient markets, making it an ideal benchmark for evaluating the performance of trading algorithms. Testing the algorithms on the S&P 500 can provide valuable insight into their effectiveness in capturing market trends and achieving above-average returns.

Moving averages, a widely recognized technical analysis tool for long-term investment, have been utilized to detect trends persisting over an extended period of time, typically spanning multiple days or months. This study addresses some issues often overlooked in scientific papers about algorithmic trading and investing, which highly increases its innovative contribution to this field of study. Firstly, results of this study are shown on the basis of the last 20 years of stock market data, which increases the informational content of this study. Secondly, the majority of papers that make use of machine learning methods, optimize them on a single in sample period and employ optimized strategy on a single out of sample period, which greatly increases the risk of backtest overfitting. Because of this, a walk forward approach is used for optimizing all of the strategies, resulting in generating a very long equity curve in the combined out of sample periods. Thirdly, many research studies do not check the robustness of their strategies, showing only base case results. Fourthly, most of the papers focus only on individual models without attempting to combine them and creating more powerful ensemble concepts.

Our research shows that using advanced machine learning (ML) algorithms makes it possible to achieve above average risk-adjusted returns compared to Buy And Hold (B&H) benchmark. Apart from that, ML based algorithms achieve higher scores compared to Moving Average Crossover technique, which is a well known technical analysis tool. Moreover, it was possible to achieve higher scores by applying ensembling methods to ML methods than using non-ensembled ML algorithms. The work is divided into several parts. The second part, after introduction, is devoted to the theoretical background and a review of the existing

literature related to the issues addressed in the work. The next section presents the research methodology and a description of the data used. The final section provides a summary of the empirical results, along with a detailed sensitivity analysis of the strategies used and the conclusions.

2 Literature Review

2.1 Historical Background and Market Efficiency

Algorithmic trading systems have become increasingly popular in recent decades due to their ability to analyze vast amounts of information and execute trades with minimal human intervention. By using these systems, investors can react quickly to new information, potentially leading to higher profits and reduced risk. However, the effectiveness of these systems depends on the efficiency of the markets in which they operate. Therefore, understanding the efficiency of financial markets is crucial for investors looking to develop and implement algorithmic trading strategies.

According to Sharpe (1994), the market is informationally efficient if all market participants have fast access to information and all available information is reflected in the prices of securities. This implies that the actual value of securities is always a reflection of their real value. According to Fama (1970), we can distinguish three types of Efficient Market Hypotheses, which assume different effects of access to the relevant type of information. Weak market efficiency assumes that investors relying on technical analysis cannot predict future prices of a financial instrument. Semi-strong efficiency assumes that investors relying on fundamental analysis cannot achieve above-average profits. Finally, strong efficiency assumes that all available information (public and confidential) is reflected in the prices of financial instruments. From the definition, it is implied that the set of information Ω_1 assumed in weak efficiency is included in the set of information Ω_2 assumed in semi-strong efficiency, and the latter is included in the set of information Ω_3 assumed in strong efficiency, i.e. $\Omega_1 \subset \Omega_2 \subset \Omega_3$. Additionally, semi-strong efficiency implies weak market efficiency, and strong efficiency implies semi-strong efficiency. However, it is important to note that these relationships are not bidirectional, and the implications do not hold in the opposite direction

The algorithmization of markets and stock exchanges began in the 1970s and 1980s (Conlan, 2016) when the New York Stock Exchange introduced the designated order turnaround (DOT) system for the first time. Since then, algorithmic trading has significantly developed. Over the last 30 years, scientific research on investment systems that uses changes in price trends (e.g. based on moving averages) has become extremely popular. Interest in this area of investment often stems from the simplicity of the implemented strategy and the relatively high risk-adjusted returns to the Buy and Hold strategy (Tapa et al., 2016, Castellano Gómez and Ślepaczuk, 2021). Many investors and researchers are interested in reducing investment risk. According to Markowitz (1952), optimal asset allocation allows for maximizing risk-adjusted returns. Therefore, in this work, we focus on the aggregated results obtained by S&P 500 index.

The attempt to construct a counterexample to the Efficient Market Hypothesis is also an interesting and popular research topic that will be undertaken in this work. Fama (1970) argued that markets are efficient reflecting the full set of information that could be used to predict future prices, i.e., abnormal profits would be achievable only by chance or by investing in risky assets. This was initially asserted by Cowles and Jones (1937), who claim that investors are unable to achieve results better than the market itself.

2.2 Moving Average Crossover

Moving Average Crossover is a technical indicator introduced by Appel (1979). It is one of the oldest and most commonly used trading indicators, still widely used by professional investors. The main advantage of Moving Average Crossover (MAC) is undoubtedly its ability to follow the upward or downward trend of prices, which offers investors the chance to identify emerging trends in prices. The issue of analyzing market efficiency and investment strategies based on technical indicators has been present in literature since at least the 1960s. James (1968) was one of the first to use technical analysis methods to predict price movements.

In his article, James described the possibilities of using moving averages of different lengths and weights with the aim of maximizing profits. The results showed that the benchmark Buy And Hold strategy achieved better results in most cases than various variations of exponential or simple moving averages. However, a rational investor will not analyze an investment strategy solely through the lens of earnings, but also the risk taken during investing. Huang and Huang (2020) in their work, based on empirical results, show that strategies based on moving averages generally correctly follow trends, bring positive returns, although usually smaller than the Buy And Hold strategy. On the other hand, they bring significantly higher risk-adjusted returns than Buy And Hold, which means that strategies based on moving averages are on average safer and carry less risk. Similarly, in the study of Gurrib (2016), it was tried to evaluate effectiveness of investment strategies based on Moving Average Crossover against the Buy And Hold (B&H) benchmark. It confirmed results stated by Huang. Using strategies based on moving averages on S&P 500 index, it is shown that the strategy based on MAs has both lower compounded returns, compounded risk and maximum drawdown, but higher Sharpe Ratio (1.351) than Buy And Hold (1.214). Moreover, Alajbeg et al. (2012) built an investment strategy based on Exponential Moving Averages on the S&P 500 index and achieved similar results. He concluded that the strategy achieved higher returns than B&H on S&P 500 index during the period 1990-2012, but not in the period 1950-2012. Additionally, the strategy had much lower maximum drawdown of 19.47% compared to B&H's drawdown of 56.77%.

Yet another way to test a strategy, is to use a profit factor defined as a ratio of gross profits divided by gross loses. Using this metric, Aycel and Santur (2022) tested a hybrid approach based on moving averages and compared it to strategies based on other technical indicators, such as Moving Average Crossover Divergence (MACD), Stochastic approach and RSI on BIST30 stocks in Turkey. Using this metric, it was concluded that the hybrid approach managed to achieve a higher PF than the other indicators widely used in trading and literature.

Apart from the metrics mentioned before, such as profit factor, compounded returns or risk, one can also measure the effectiveness of the strategy using derivatives of Sharpe Ratio, which take into account other factors, such as Maximum Loss Duration (MLD), Maximum Drawdown (MD) or Annualized Standard Deviation of rates of return (aSD). One such study that measured risk-adjusted returns using these metrics is (Castellano Gómez & Ślepaczuk, 2021). Among many other strategies, single and complex, based on a portfolio of them on S&P500 index, they focused on Moving Average Crossover strategy. Analyzing both base and and sensitivity analysis models (which covered different lengths of walk forward's in sample and out of sample periods) we can conclude that none of the models managed to achieve a higher return than B&H, but all models except one achieved higher IR^* metrics (which is defined as annualized rate of return ARC divided by annualized standard deviation of rates of return ASD) and all but two achieved higher IR^{**} (which is a metrics derived from IR^*, but additionally is multiplied by ARC and divided by MD).

2.3 XGBoost

Due to disagreement about the validity of Efficient Market Hypothesis, nowadays many researchers try to use machine learning techniques to construct such price-predicting algorithms that will be able to beat the market without high levels of risk. Decision Trees, Neural Networks, Support Vector Machines, Fuzzy Algorithms and so on. Such algorithms can be easily optimized and used to forecast future prices or rates of returns with great accuracy. One commonly mentioned machine learning algorithm in the literature is XGBoost. eXtreme Gradient Boosting is a machine learning algorithm, which gained traction in recent years. Since it's invention in 2016, it has been used in natural language processing, image recognition or time series forecasting. XGBoost is an ensemble algorithm that ensembles multiple decision trees in order to make a single prediction. The main idea is that by adding a new tree built atop of the old tree, we are able to significantly reduce errors.

In the spirit of trying to beat Buy And Hold benchmark, Drahokoupil (2022) tried to predict future Bitcoin BCT prices by optimizing XGBoost with Bayesian optimization techniques. In this paper, researcher managed to beat the Buy And Hold benchmark in both total profit and two risk measuring performance metrics that is Sharpe Ratio and Sortino Ratio. During sensitivity analysis, they also managed to prove

robustness of used algorithm and showed promising results by ensembling variants of XGBoost with special weight technique.

According to Yuan (2023), XGBoost outperforms other established machine learning algorithms, such as KNN (K Nearest Neighbours), only to be beaten by a newer version of itself that is LightGBM algorithm in the context of error metrics such as MSE, RMSE, MAE or MAPE. All the algorithms in this study were trained only on closing prices. Another study comparing XGBoost to other machine learning algorithms, such as Simple Decision Tree and Random Forest is (Mohammed et al., 2023), in which authors tried to predict prices of BAC, CVX and HD stocks. MSE and Accuracy metrics were measured, and comparing to the rest of strategies, XGBoost proved to bring the worst results out of the three. On the contrary, in the study conducted by Sekhar et al. (2023), XGBoost proved to be more accurate compared to LSTM based algorithm with enabled Early Stopping. XGBoost managed to have MAE equal to 0.023, where LSTM had MAE equal to 0.107. Both algorithms had only one out of sample and one in sample periods, as both were trained only on Adjusted Close Prices.

Another study, in which authors compare XGBoost to other established algorithms is (Chlebus et al., 2021). In this research, authors compare XGBoost to methods based on Support Vector Regression (SVR), K-Nearest Neighbours (KNN), Light Gradient Boosting Machine (LGBM) and Long Short-Term Memory (LSTM) on the Nvidia's stock from 2012 to 2018. It was shown that the SVR, based on stationary attributes, exhibited the most optimal performance under the scores of Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Median Absolute Error (MedAE). In general, it was demonstrated that models which rely solely on stationary variables outperformed models that incorporate both stationary and non-stationary variables. XGBoost based model achieved the highest loss function value in almost every scenario.

2.4 LSTM

Another example of machine learning algorithm widely used by researchers are Artificial Neural Networks. Recurrent Neural Networks (RNNs) are a sub-class of Artificial Neural Networks (ANNs) that can process sequential data. RNNs have a history in the field of natural language processing, predicting DNA sequences or algoritmic-trading. Unfortunately, RNNs can not manage the vanishing gradient problem (or exploding gradient problem). To solve this problem, the gate mechanism is introduced, which successfully replaces hidden layer in RNN. Long Short Term Memory (LSTM) is a type of RNN that is really effective in modelling long term dependencies, thus making them a popular choice for researchers to use as a tool to predict the stock prices or other financial time-series.

Since LSTM is a sub-class of Recurrent Neural Network group, it might be tempting to compare the two of them in practical use. For example, in (Lv et al., 2021), authors not only compared LSTM and RNNs, but also Ligth-LBGM-LSTM ensemble and GRU. Data used was Shanghai Composite Index and Shanghai Shenzhen 300 index. Comparing accuracy and F-score of the predictions, it was concluded that LightGBM-LSTM ensemble was the most accurate and precise.

In the study conducted by Nguyen and Ślepaczuk (2022), the authors decided to compare LSTM to well known methods of investing and technical analysis, such as Relative Strength Index (RSI), Moving Average Crossover Divergence (MACD) or Simple Moving Averages (SMA) on S&P 500 index. All of the models were assessed using derivative of Sharpe Ratio, mainly Information Ratio IR^* and Modified Information Ratio IR^{**}. In the study, it was shown that LSTM based model outperformed all of the models based on technical indicators in both IR^* and IR^{**}.

For example, Chena (2022) compared aforementioned XGBoost and LSTM with the tasks of predicting prices for gold and BTC. Researchers used error metrics and accuracy to measure how well these models work. Being trained just on adjusted close prices, LSTM managed to achieve 51.04% accuracy on BTC and 53.03% accuracy on gold, while XGBoost managed to achieve 57.19% and 44.15% accuracy on BTC and gold respectively.

Yet another study, where designed strategy managed to beat the benchmark with respect to risk-adjusted returns is (Mahfooz et al., 2022), in which authors proposed training LSTM on so called Unified Trading

Strategy (UTS) based on technical indicators. Later on, they compared effectiveness of the LSTM with benchmark Buy And Hold strategy and UTS using Sharpe Ratio and Profit Factor, defined as the ratio of winning to losing trades. Stocks used were AAPL, DIA, SPY, JNJ, MCD, NKE, MSFT, and PG. In this study, LSTM managed to achieve the highest Sharpe Ratio out of all strategies used.

Not only one can base entire strategy on LSTM prediction, but one can also use LSTM in hybrid, more advanced models. For example, in (Li et al., 2021) LSTM is used with Deep Q-Learning (DQN) model, meaning it is used instead of traditional Recurrent Neural Network in order to predict the Q-estimate and choose the action with the biggest predicted Q value. The LSTM-DQN ensemble was tested on data from CSI Nine Sector Index. Results show that in eight out of nine tested sectors model with LSTM managed to achieve higher Sharpe Ratio than Moving Average Crossover or Buy And Hold benchmarks.

2.5 Ensembling LSTM and XGBoost

Ensembling machine learning algorithms has become a popular technique for improving accuracy of predictions and overall robustness of a model. There are several ways to ensemble a ML model, in particular:

- *Bagging*
  Bagging involves optimizing models on different subsets of training data, where each models make a prediction independently of each other. Final prediction of the model is given by an average, democratic or weighted voting over all individual model predictions.
- *Boosting*
  Boosting involves training many models on the same dataset, where each and next model is fed errors or information about mistakes from the previous models. The global, final prediction usually is a weighted vote composed from all previous models.
- *Stacking*
  Stacking involves training multiple models on the same dataset and using outputs from one model as inputs to another, for example predictions from model A on day t for day $t+1$ can be used as inputs for model B that also makes a prediction for day $t+1$. The final meta model can be trained on the original raw data, just the outputs from other model or a concatenation of both mentioned.

Ensembling also can help to reduce overfitting of the model and smooth the predictions to reduce the impact of individual model's biases or peculiarities. In particular, ensembling LSTM with outputs from XGBoost may produce better signals. LSTM is well known for being able to model sequential data, simultaneously being able to capture existing long-term dependencies. XGBoost, one the other hand is well suited for handling high-dimensional data and making accurate predictions. By ensembling these two, there is a chance to achieve the best of both worlds and leverage the strengths of both models. Many researchers tried to ensemble machine learning algorithms with one another, achieving mixed results.

One such study that tried ensembling LSTM with outputs from XGBoost is (Yu et al., 2021). In this particular research, LSTM was compared to LSTM ensembled with XGBoost and with simple RNNs. The models were compared to each other using Root Mean Squared Error RMSE, Mean Absolute Error MAE, Accuracy and f_1 score, which is calculated using precision and recall. The data used were ES, YM, AAPL, SI and CL stocks. The stated conclusion was that there was no certain degree of stability in the forecast, but in all cases in sensitivity analysis the ensembled model managed to have lower MAE and both higher accuracy and f_1 than simple LSTM and RNNs.

In the study by Jiang (2022), five algorithms were compared using performance scores and error metrics, such as MSE, MAE and RMSE. Covered models were Attention based GRU - XGBoost, attention based BiLSTM, LSTM, CNN and ARIMA. Results have shown that ensembled model managed to achieve lower RMSE than all of the models, lower MAE than all models except Attention based BiLSTM and lower MSE than ARIMA.

Another example of ensembling XGBoost with other strategies that produces satisfying results is study conducted by Kumar et al. (2022) on Microsoft stock. Outputs from models were analyzed with MAE, MAPE and Accuracy metrics. It was shown that ensembled model brought better results in the context of all metrics used when compared to SARIMA and XGBoost.

2.6 Novelty of Results

Most of the authors during their work do not use walk forward optimization, but instead split the entire dataset into two (in case of ML algorithms three) parts and produce relatively short Equity line. In this work, we will investigate ensembling Machine Learning algorithms and optimizing them with walk forward procedure. Overall, in this study we will propose a model that is both optimized using walk forward optimization procedure, and is an ensemble of both machine learning and technical indicator based algorithms. Furthermore, we will analyze the results with many performance metrics that not only take into account rate of returns, but also risk factors such as Maximum Loss Duration, Maximum Drawdown and Annualized Standard Deviation of daily rates of return. All but one, newly introduced in this study IR^{***}, of mentioned Performance Metrics are derived from metrics used in (Kryńska & Ślepaczuk, 2022).

3 Methodology

This study focuses on creating an investing strategy, such that it is a counterexample to Efficient Market Hypothesis. We will explore topics of classical technical analysis tools, such as moving averages enhanced with more advanced Machine Learning algorithms, and at the very end we will try to combine these strategies into one using ensembling methods. All of the strategies were optimized with non-anchored walk forward optimization and compared to each other using performance metrics and focusing on risk-adjusted returns. In this study, we will introduce the following symbol to make equations more readable. For all natural numbers n let us define:

$$[n] := \Big\{ k \in \mathbb{Z} \;\Big|\; 1 \leq k \leq n \Big\} \tag{1}$$

3.1 Performance Metrics

In this study we will introduce methods for evaluating investment strategies that is Performance Metrics that take into account not only return rates, but also factors indicating safety of the investment. Many of the listed metrics are gathered from (Castellano Gómez & Ślepaczuk, 2021). Additionally, we will introduce a new metrics IR^{***}, which also takes into account the length of loss periods. In analyzing performance of investment strategies, one should not look only at returns, since it does not reflect how safe the strategy is. Therefore, we will introduce methods of measuring risk-adjusted returns by combining loss duration periods, percentage losses, standard deviation of return rates into three metrics: IR^{*}, IR^{**}, IR^{***}. These are the following:

- **Daily rate of return** r

$$r_{i+1} := \frac{p_{i+1} - p_i}{p_i} \tag{2}$$

  where p_i is the price of the asset on the $i-$th day. We will be using these notations extensively further on in the work.

- **Annualized Rate of Return Compounded** ARC

$$ARC := -1 + \prod_{i=1}^{n}(1 + r_i)^{\frac{252}{n}} \tag{3}$$

where n is the number of days the strategy has been running.

Annualized rate of return ARC is a measure of average yearly compounded wealth growth on a tested period. Two assumptions are made regarding ARC. First of all, average wealth growth from previous year is accumulated and added to next year capital. Second of all, on average stock markets are opened between 250 and 254 days every year, which for simplicity purposes we will approximate by 252 days in order to annualize the returns.

- **Annualized Standard Deviation** aSD

$$aSD := \sqrt{\frac{252}{n-1}\sum_{i=1}^{n}\left(r_i - \sum_{k=1}^{n}\frac{r_k}{n}\right)^2} \tag{4}$$

aSD is a yearly annualized standard deviation of daily returns $(r_i)_{i=1}^{n}$.

- **Maximum Drawdown** MD

$$MD := \sup_{(x,y)\in\left\{(t_1,t_2)\in\mathbb{R}^2_{\geq 0}\,:\,t_1<t_2\right\}} \frac{P_x - P_y}{P_x} \tag{5}$$

where P_t is overall equity held at the moment t in time.

Maximum Drawdown MD shows the greatest percentage capital loss across the entire investment horizon. Large values of MD that is large losses, may suggest that strategy is not stable. Optimal value of MD to be achieved is zero, since it implies that strategy has never lost any capital. The number is always non-negative and is shown in percentage terms.

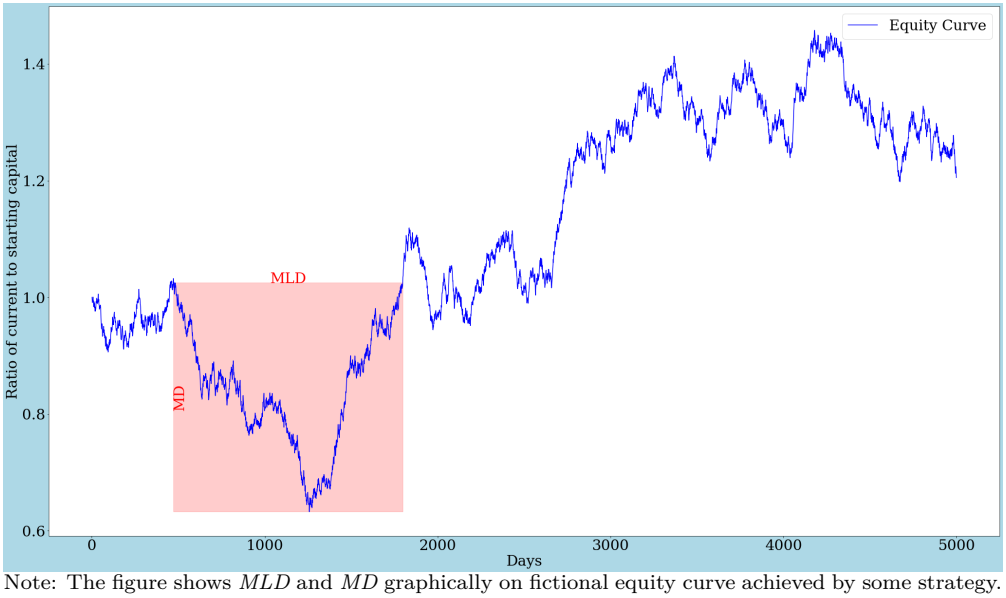
- **Maximum Loss Duration** MLD

$$MLD := \frac{y - x}{252} \tag{6}$$

where x and y are moments in time defined in equation (5).

Maximum Loss Duration MLD measures in years the longest loss period across the entire investment horizon when the value of our investments was on the lower level than the previous local maximum. Figure 1 presents graphically how MD and MLD are calculated.

Figure 1: Graphical representation of *MD* and *MLD*



Note: The figure shows *MLD* and *MD* graphically on fictional equity curve achieved by some strategy. The width represents the time period, in which the highest percentage loss occurs and goes back to the previous maximum. The height represents biggest percentage loss on the entire investment horizon.

Apart from the above measures of profits and risk, we also propose three additional metrics IR*, IR**, IR*** that accumulate information about returns and volatility of the strategy which combine mentioned metrics into one:

- **Information Ratio** IR*

$$IR^* := \frac{ARC}{aSD} \tag{7}$$

First iteration of *Information Ratio* is calculated as a ratio of annualized return rate to annualized standard deviation, so in a sense it is really similar to Sharpe Ratio, since the ratio is defined as:

$$\textit{Sharpe Ratio} := \frac{\text{return of portfolio} - \text{risk free returns}}{\text{standard deviation of rates of return}} = \frac{ARC - R_f}{aSD} = IR^* - \frac{R_f}{aSD} \tag{8}$$

and one only needs to assume that risk free returns R_f are equal to zero. The more profitable the strategy is (higher return rate ARC), or the safer the strategy is (lower standard deviation of daily returns), the higher information ratio is.

- **Information Ratio** IR**

$$IR^{**} := IR^* \cdot \frac{ARC}{MD} \cdot \text{sgn}(ARC) = \frac{\text{sgn}(ARC) \cdot ARC^2}{aSD \cdot MD} \tag{9}$$

Second iteration of information ratio takes into account the same parameters as IR*, but also the maximum loss of our investment (Maximum Drawdown *MD*). The lower *MD*, the higher IR** is. In the best, optimal scenario IR** is equal to infinity (when $MD = 0$). IR** has the same monotonicity characteristics as IR*.

- **Information Ratio** IR^{***}

$$IR^{***} := IR^{**} \cdot \frac{ARC}{MLD} = \frac{ARC^3}{aSD \cdot MD \cdot MLD} \tag{10}$$

Third and last iteration of information ratio is IR^{***}. Not only does it have the same parameters as IR^{**}, but it also takes into account *Maximum Loss Duration* MLD. IR^{***} is positively correlated with ARC and negatively with all of the metrics that reflect risks of an investment. Value of IR^{***} usually is very low, so for visualization aspects we will multiply this value by 10000.

From the point of our analysis, the most important metric is IR^{*}, and so most of the optimization criteria in the base case scenario are based on choosing a combination of hyperparameters with the highest IR^{*} value. The rest of the metrics are supplementary, and additional conclusions are drawn from analyzing their values.

3.2 Training inputs for LSTM

In this study, we tried to build investment algorithms, including LSTM based models. Some of the training inputs were technical indicators, those being *RSI* (Relative Strength Index) and *SD* (Standard Deviation) of returns rates of B&H, the definition of which can be found in many papers listed in literature review on the topic of technical indicators.

3.2.1 RSI

Relative Strength Index is one of the technical analysis indicators that determines the strength of a trend. It takes values in the range of $[0, 100]$ and is expressed by the formula defined in (Wilder, 1978):

$$RSI_n := 100 - \frac{100}{1 + RS_n} \tag{11}$$

$$RS_n := \frac{\text{average value of the closing price increase over last } n \text{ days}}{\text{average value of the closing price decrease over last } n \text{ days}} \tag{12}$$

where n is the number of observed days. If RSI is around 100, it indicates an increased probability of a trend reversal to a downtrend. If it is in the range of $(70, 100)$, it gives a sell signal. If it is in the range of $(0, 30)$, it gives a buy signal. If RSI is around 0, it indicates an increased probability of a trend reversal to an uptrend.

3.3 Moving Average Crossover

The idea of moving averages is well-known to investors and often used in technical analysis. Moving Average Crossover method does not predict future prices, but indicates current market trend. For MAC to work, one needs to employ them on a market that has both long- and short-term trends. The strategy consists of calculating and looking for crossovers of two averages that is one slow, and one fast. Buy signal is created, when fast moving average is below slow moving average, and then we can expect the prices to grow in short term. Analogously, sell signal is created when slow moving average is below fast moving average, which indicates that we expect the prices to fall. If we denote the value of fast moving average on day t by FMA_t and slow moving average by SMA_t, we can write:

$$\textit{signal generated on the day } t := \begin{cases} Buy, \text{ if } FMA_t > SMA_t \\ Sell, \text{ otherwise} \end{cases} \tag{13}$$

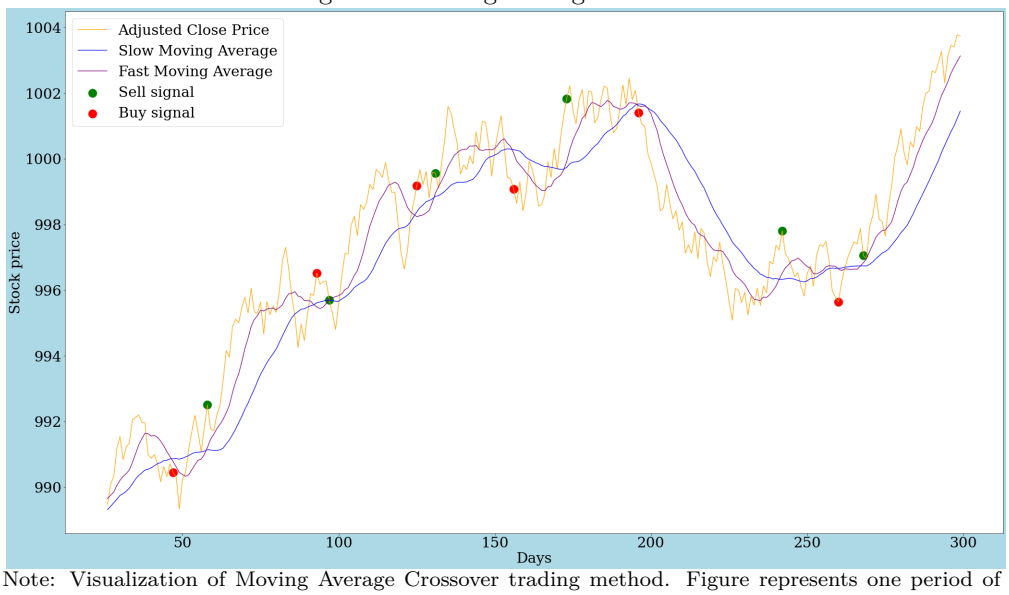
In this study, we chose hyperparameter space defined as:

$$\text{Length of Fast Moving Average} \in \Omega^{fast} := \{1, 3, 5, ..., 39, 41\} \tag{14}$$

$$\text{Length of Slow Moving Average} \in \Omega^{slow} := \{40, 50, 60, ..., 140\} \tag{15}$$

Thus, in each iteration we need to calculate $\#\Omega^{fast} \times \#\Omega^{slow}$ combinations of parameters. In walk forward procedure, a combination of fast and slow moving averages that gives the best value of chosen optimization criterion is employed in out of sample period. Additionally, we define a Moving Average of length one as a index price from previous day. Figure 2 presents graphically how MAC strategy works in practice.

Figure 2: Moving Average Crossover



Note: Visualization of Moving Average Crossover trading method. Figure represents one period of using the method, with fixed lengths of both slow and fast moving averages. OX axis represents trading days, and OY axis represents both moving averages and stock price in dollars.

In the base case scenario, we chose optimization criteria to be IR*, the in sample period length of 3 years and out of sample period length of 1 year. In the sensitivity analysis, we calculate more combinations of optimization criteria, *OOS* and *IS* period lengths.

Table 1: Hyperparameters for moving averages

	Optimization Criteria	IS Length	OOS Length	Ω^{fast}	Ω^{slow}
Base Case	IR*	3 years	1 year	{1,3,...,41}	{40,50,...,140}
Sensitivity analysis	{*ARC*, IR*, IR**, IR***}	{1,2,3,4} years	1 year	{1,3,...,41}	{40,50,...,140}

Note: Note: Curly brackets denote possible combinations of the hyperparameter. The in sample period starts in 1996 and ends in 2023. The out of sample period starts on 2000.01.01 and ends on 2023.05.11. Both base case and sensitivity analysis are optimized on space of hyperparameters defined by $\#\Omega^{fast} \times \#\Omega^{slow}$.

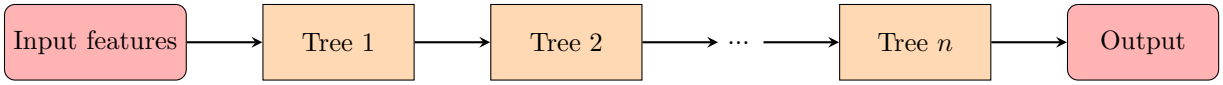
All in all, thirteen models of moving averages are calculated, since there is one base case model and twelve possible combinations (four possible optimization criteria timer three possible lengths of in sample period) of optimization criteria and IS length in sensitivity analysis.

3.4 XGBoost

3.4.1 Theory behind XGBoost

XGBoost (eXtreme Gradient Boosting), introduced in (Chen & Guestrin, 2016), is a machine learning algorithm that is widely used for various tasks, including classification or regression. It is a boosted tree ensemble model, meaning it iteratively trains weak decision trees and combines their output to make predictions. The algorithms works by minimizing loss function that measures, how far apart were predicted values and the actual value. XGBoost works as shown on Figure 3.

Figure 3: Visualization of XGBoost



Note: Figure shows standard visualization of XGBoost. In our study XGBoost is optimized with grid search, defined in Table 2, based on walk forward optimization.

Let D be a training set with n samples and m features, in which y is the target variable. XGBoost constructs an ensemble of K decision trees f_k, such that each one maps an input feature vector $x \in \mathbb{R}^m$ to a scalar output $\forall_{k\in[K]}\ f_k(x) \in \mathbb{R}$. Output of XGBoost model is given by the formula:

$$\hat{y}(x) := \sum_{k\in[K]} f_k(x) \tag{16}$$

Let us denote a differentiable loss function by $l(y_i, f_k(x_i))$, measuring the difference between predicted value $\hat{y}(x_i)$ and actual value y_i for $i \in [n]$. Also, let $\Omega(f_k)$ be a regularization term that prevents trees from overfitting. Every decision tree is optimized by minimizing objective function $\mathcal{L}$ defined by:

$$\mathcal{L}(f_k) := \sum_{i=1}^{n} l(y_i, f_k(x_i)) + \Omega(f_k) \tag{17}$$

Regularization term $\Omega(f_k)$ controls complexity (thus, prevents overfitting) of $k-th$ tree. Typically, it is defined as:

$$\Omega(f_k) := \gamma \cdot T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{18}$$

where T is the number of terminal nodes in the tree, γ and λ control the degree of regularization and w_j is the output of $j-th$ node.

XGBoost uses gradient boosting to iteratively improve the ensemble of decision trees, meaning at each iteration t the algorithm adds a new decision tree f_t, such that it minimises objective function:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}^{(t-1)}(x_i) + f_t(x_i)) + \Omega(f_t) \tag{19}$$

where $\hat{y}^{(t-1)}(x_i)$ is the current prediction of the ensemble model for sample i at iteration $t-1$.

3.4.2 XGBoost in practice

For our needs, we will use XGBoost with walk forward optimization and grid search to predict future prices. In every in sample window, we will optimise the algorithm with grid search and employ the found combination of parameters on out of sample period. Algorithm makes prediction for one day only, so after each day we add previous day's price to data vector and make predictions on new dataset. If predicted price is greater than today's price, a buy signal is created. Otherwise, a sell signal is created. Hyperparameter search space is defined as follows:

Table 2: Hyperparameter space for XGBoost optimization

# of Estimators	Learning Rate	Max Depth	Gamma
{100, 200, 300, 400}	{0.001, 0.005, 0.01, 0.05}	{8, 10, 12, 15}	{0.001, 0.005, 0.01, 0.02}

Note: The table shows all possible parameters used in grid search optimization for XGBoost. Values were taken from research conducted by Zhang Y. (2022). The number of tested combinations in grid search is equal to $4 \cdot 4 \cdot 4 \cdot 4 \cdot 4 = 64$.

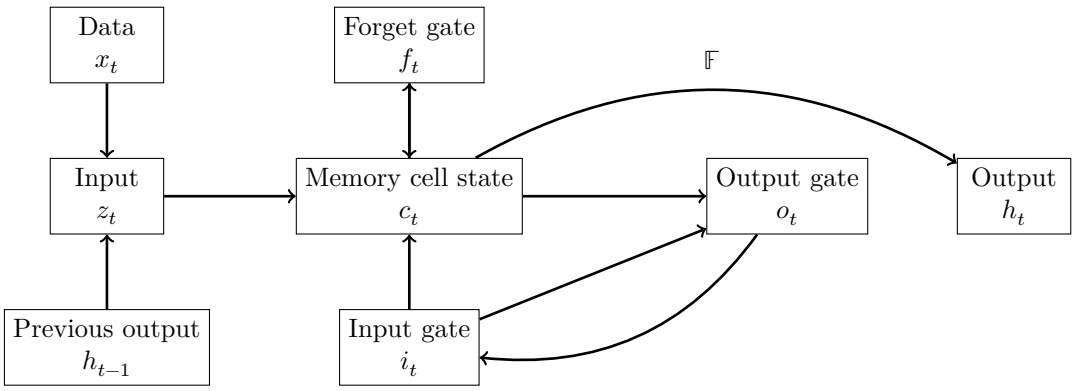
3.5 LSTM

3.5.1 Theory behind LSTM

Long Short-Term Memory (LSTM) model, introduced in (Hochreiter & Schmidhuber, 1997), is particularly useful in finance and algorithmic trading, because it can effectively capture complex patterns and dependencies in financial time series data. By using LSTM networks, traders and investors can gain an edge by identifying and acting on emerging market trends and investment opportunities.

LSTM is a type of Recurrent Neural Network RNN that additionally addresses the problem of vanishing gradient present in standard Recurrent Neural Networks. In standard RNNs the gradient of loss function can become incredibly small as it propagates backward, making it difficult for network to work effectively. This problem can be especially acute in investing or algorithmic trading, where time series data may have long term dependencies and complex patterns. LSTM networks have a dedicated memory cell that allow them to selectively remember (or not) information from past inputs. The memory cell is controlled by three gates: input gate, forget gate and output gate that control the flow of information. By selectively gating information, systems based on LSTM can effectively capture long term dependencies in the data provided and minimise the vanishing gradient problem. LSTM works as shown on Figure 4:

Figure 4: Visualization of how LSTM works



Note: Figure shows standard representation of LSTM. In our study, LSTM is optimized with grid search defined in Table 3, which uses with walk forward optimization.

Let us denote output of the LSTM cell at time t by h_t, and input by x_t. Generalized input z_t to the LSTM cell is a concatenation of the input at time t and output from the previous time step h_{t-1} that is:

$$z_t := [h_{t-1}, x_t] \tag{20}$$

The input gate i_t determines how much of the input should be stored in the memory cell. Let us assume that W_i and b_i are learnable parameters representing weights and biases of the network. Then it is calculated by applying sigmoid function σ to a sum containing input and previous output:

$$i_t := \sigma(W_i \cdot z_t + b_i) \tag{21}$$

The forget gate f_t determines how much of the previous memory cell state should be saved. Analogically to input gate i_t, sigmoid function is applied to a sum containing input and previous output:

$$f_t := \sigma(W_f \cdot z_t + b_f) \tag{22}$$

The output gate o_t determines how much of the memory cell state will be in output, or simply what information will be in the output:

$$o_t := \sigma(W_o \cdot z_t + b_o) \tag{23}$$

The memory cell state c_t is updated by selectively adding and removing information based on the input and forget gates:

$$c_t := f_t \cdot c_{t-1} + i_t \cdot \mathbb{F}(W_c \cdot z_t + b_c) \tag{24}$$

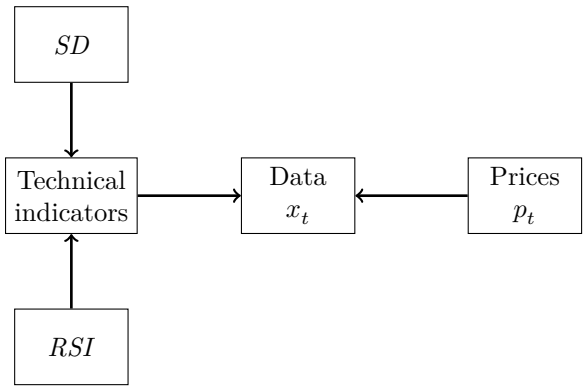
where $\mathbb{F}$ is an activation function (usually hyperbolic tangent function tanh or ReLU). The output itself is calculated by selectively selecting information from the memory cell based on the output gate:

$$h_t := o_t \cdot \mathbb{F}(c_t) \tag{25}$$

3.5.2 LSTM in practice

In this study, it was decided to train LSTM on current market price of an index and technical indicators based on market price. At every day t, the LSTM model has access to the price from day t and all indicators calculated based on prices up to (and including) day t. This way, Data x_t in Figure 5 consists of the newest, but not future data (to avoid look-ahead bias, which is very common in many studies). LSTM models in our case base their decisions on the last 15 days of available data that is on days t, $t-1$, ..., $t-14$ when making decision on day $t \geq 15$.

Figure 5: Visualization of base case LSTM model on the input layer



Note: Figure shows construction of Data x_t used in calculation process by LSTM algorithm in base case scenario. Logic behind calculations is the same as in Figure 4. *SD* stands for standard deviation.

Each and every one of the models listed below was subjected to grid search in order to find best-fitting parameters in every walk forward period. Since grid search is very time consuming, we used predefined combination of parameters chosen from literature. At every test period in in sample, algorithm performed grid search and optimized all of the ten models, then chose the best one on the validation set with respect to the highest value of optimization criteria, this being the highest value of IR*. Proposed combination of parameters are summarized in Table 3.

Table 3: Chosen hyperparameters for LSTM optimization

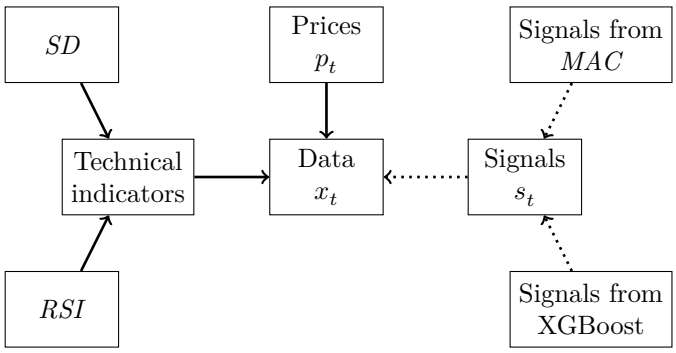
Model	Neuron Dropout	# of Neurons	# of Hidden Layers	Learning Rate	Loss function	# of Epochs
1.	0.3	64	1	0.001	BC	80
2.	0.2	30	0	0.01	BC	80
3.	0.1	10	1	0.0001	BC	80
4.	0.1	25	0	0.001	BC	30
5.	0.2	32	0	0.001	BC	80
6.	0.2	12	1	0.003	BC	80
7.	0	128	1	0.001	BC	80
8.	0.2	32	0	0.002	BC	80
9.	0.2	120	0	0.001	BC	80
10.	0.25	50	0	0.001	BC	80

Note: BC stands for Binary Cross-Entropy. Each of the models had a lag (decision period) of 15. Optimization criterion was IR^*. $\#\Omega$ means the cardinality of set Ω, or the number of elements in the set, or simply 'the number of'.

3.5.3 LSTM Ensembles

Furthermore, we will ensemble models with signals produced by Moving Average Crossover and XGBoost. At every day t, LSTM will not only base it's prediction on technical indicators and prices explained in the previous section, but also signals generated on day t for day $t+1$. Overall, Ensembled LSTM data and models are presented as follows:

Figure 6: Visualization of LSTM Ensembles



Note: Figure shows construction of Data x_t used in calculation process by LSTM algorithm in an ensemble model. Logic behind calculations is the same as in Figure 4. *SD* stands for standard deviation, *MAC* stands for Moving Average Crossover.

Logic behind generating signals is similar as in previous chapters. Four LSTM models for each Short Allowed and Long Only format will be covered in base case scenario, so all in all eight models will be tested. We will change the ranges of technical indicators and see, how the model performs in both Long Only and Short Allowed scenario. Sensitivity analysis will not only cover lengths of inputs to technical indicators, but also ratios pf partitions of train, validation and test periods. Overall, thirteen models will be calibrated and presented for each LSTM model.

Table 4: Proposed LSTM models

	Slow MA	Fast MA	Slow SD	Fast SD	RSI	Signals from MAC	Signals from XGBoost	Train	Val	Test
Base Case	60	15	30	10	14			7	1	2
	60	15	30	10	14	Yes		7	1	2
	60	15	30	10	14		Yes	7	1	2
	60	15	30	10	14	Yes	Yes	7	1	2
Sensitivity Analysis	60	15	30	10	14	Yes	Yes	4	1	2
	60	15	30	10	14	Yes	Yes	12	1	2
	60	15	30	10	14	Yes	Yes	7	2	2
	60	15	30	10	14	Yes	Yes	7	4	2
	60	15	30	10	14	Yes	Yes	7	1	1
	60	15	30	10	14	Yes	Yes	7	1	3
	60	15	30	10	14	Yes	Yes	7	1	4
	120	30	60	20	28	Yes	Yes	7	1	2
	30	8	15	5	7	Yes	Yes	7	1	2

Note: By *Slow* a bigger value of hyperparameter was meant, by *Fast* a smaller, since it could adapt and represent current market conditions faster. Four models are described in base case, two models in sensitivity analysis, each in both Short-Allowed and Long-Only format. Numbers in Train, Val and Test columns represent a ratio in which the sizes of the sets are. Given the entire window is 1000 days, for example in the base case scenario train set is 700 days long, validation set is 100 days long and test set is 200 days long.

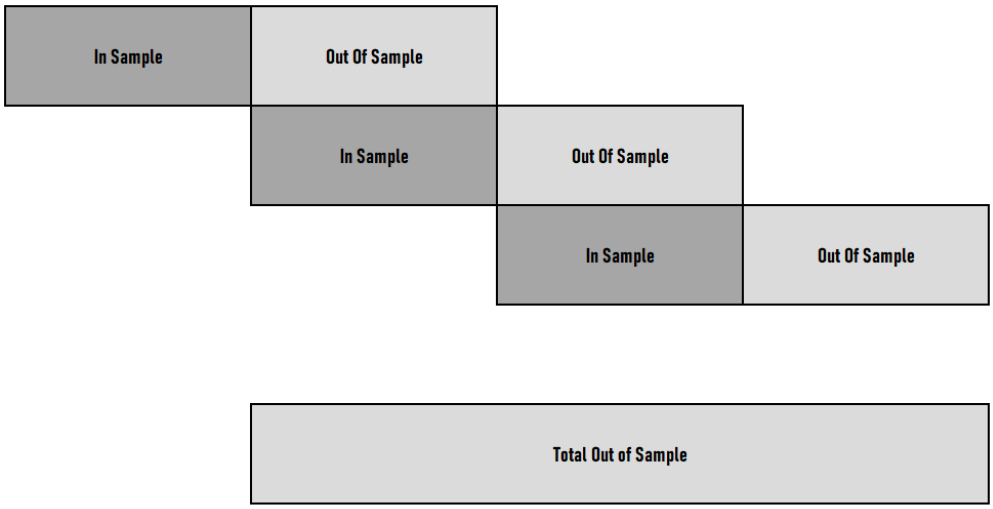
3.6 Walk Forward Optimization

3.6.1 General idea of Walk Forward Optimization

Every strategy mentioned is optimized with walk forward optimization. In this study, we chose a walk forward optimization instead of anchored walk forward or having one OOS and IS period for a few reasons. First of all, we allow algorithm to face different market regimes in separate in sample periods, which leads to better hyperparameter optimization. Second of all, chances of over-fitting are much lower, due to larger number of shorter training periods. Methodology of walk forward in general is as shown on Figure 7.

1. Divide all of the available data into equal parts, containing n of IS and OOS periods
2. Choose In Sample (IS) and out of sample (OOS) period lengths
3. Optimise strategy on $i-$th IS period, where $i \in [n]$
4. Employ optimized strategy on $i-$th OOS period
5. Repeat for periods IS_{i+1} and OOS_{i+1}, until we've covered all partitions

Figure 7: Walk forward Optimization in general



Note: Visualization of walk forward optimization in general. Total out of sample is constructed by concatenating the results of all out of sample periods chronologically.

3.6.2 Walk Forward Optimization for Moving Average Crossover

Specifically for moving averages, walk forward process is as follows:

1. Divide all of the available data into equal parts, containing n IS and OOS periods.
2. Choose in sample (IS) and out of sample (OOS) period lengths.
3. Optimise strategy on $i-$th IS period, where $i \in [n]$. In the case of Moving Average Crossover, by that we mean choose such combination of parameters denoting lengths of FMA and SMA that maximises chosen optimization criteria (IR*, IR**, IR*** and ARC). Performing grid search, we have to check exactly $\#\ \Omega^{fast} \cdot \#\ \Omega^{slow}$ combinations, in our case $21 \cdot 11 = 231$ combinations.
4. Use chosen combination of SMA and FMA lengths in $i-$th OOS period.
5. Repeat for periods IS_{i+1} and OOS_{i+1}, until we've covered all partitions.

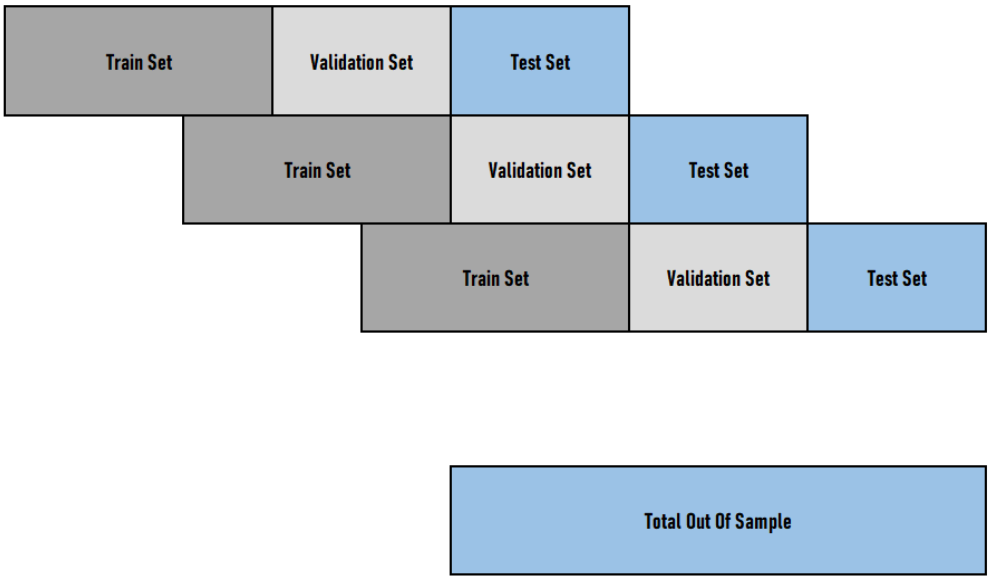
3.6.3 Walk Forward Optimization for Machine Learning Algorithms

Similarly to moving averages, ML strategies are also optimized with walk forward approach. The difference between WFO for MAs and Machine Learning based strategies is that we introduce a new period, i.e. validation period. For a literature chosen space of hyperparameteres Ω, the optimization process is as shown on Figure 8.

1. Divide all of the available data into n equal parts $T_i\big|_{i\in[n]}$, each T_i consists of three nonoverlapping periods: train T_i^{train}, validation T_i^{val}, test T_i^{test} such that $\forall_{i\in[n]}\ T_i^{train} \cup T_i^{val} \cup T_i^{test} = T_i$.
2. Train the model on period T_i^{train}.
3. Validate and choose the best set of hyperparameters $\omega^* \in \Omega$ from available parameters $\omega \in \Omega$ on T_i^{val}.
4. Employ algorithm with chosen set of hyperparameters ω^* on T_i^{test} and save the results.

5. Repeat for T_{i+1}, until we've covered all indexes $i \in [n]$.

Figure 8: Walk forward Optimization for Machine Learning Algorithms



Note: Visualization of walk forward optimization especially for Machine Learning Algorithms. Logic stays the same, as in general case. Total out of sample is constructed by concatenating the results of all out of sample periods chronologically.

3.7 Assumptions and Descriptions of Strategies

During investing process, some general assumptions are needed to be made. For simplicity purposes, for all the strategies listed in this research, we assumed that:

- We could trade any fraction or quantity of an index
- During any trading day t, only one action could be made that is do nothing, buy or sell
- We are always able to perform an action on the market
- Transaction costs (0.02% of the value of an order) were included in every transaction
- We invest all the available capital (according to signal) and calculate the returns using percentage rates
- When buying or selling on trading day t, we assumed index close price from day t

Each strategy was tested in two variants: Long Only and Short Allowed. In general, all strategies produce two signals, those being 1 and τ, where $\tau = 0$ for Long Only and $\tau = -1$ for Short Allowed. We denote price of SP500 index at the end of the day t by p_t, by $\hat{p}_t$ a price prediction for day t, and by $signal_t$ a signal generated by the investment algorithm. Transaction cost at 0.02% level are a reasonable assumption, since they are average of bid-ask spread typical range of 0.01% or less to 0.05% or more. Such spread heavily depends on trading volume, liquidity and market volatility, thus making 0.02% a conservative estimate for transaction costs, which includes reasonable assumptions about bid-ask spread costs.. For specific strategies, following assumptions were made:

- **Buy And Hold**

  The index was bought at the beginning of out of sample period and sold at the end of it. B&H is a benchmark strategy and represents all the changes in stock market over the years.

- **Moving Average Crossover**

  We enter a long position, when Fast Moving Average (FMA) is above Slow Moving Average (SMA) and analogically short position, when SMA is above FMA. Then:

$$\text{signal}_t = \begin{cases} 1 \text{ if } FMA_t > SMA_t \\ \tau \text{ otherwise} \end{cases} \tag{26}$$

- **LSTM Vanilla, Stacking, Voting** and **XGBoost**

  At the end of every day t, we make a prediction $\hat{p}_{t+1}$ what tommorow's (day $t+1$) price will be equal to. If it's higher than today's (day t), we enter a long position. Otherwise, we enter position τ. Then:

$$\text{signal}_t = \begin{cases} 1 \text{ if } \hat{p}_{t+1} > p_t \\ \tau \text{ otherwise} \end{cases} \tag{27}$$

- **Voting**

  Having n systems that generate signals $(s_i)_{i\in[n]}$, such that $\forall_{i\in[n]}\ s_i \in \{\tau_i, 1\}$ at the end of every day t we define a vote (later denoted by Voting) as:

$$\text{signal}_t = \frac{1}{n}\sum_{i\in[n]} s_i \tag{28}$$

  By this definition, we allow fractional signals to occur. They can be interpreted as investing only a portion of capital.

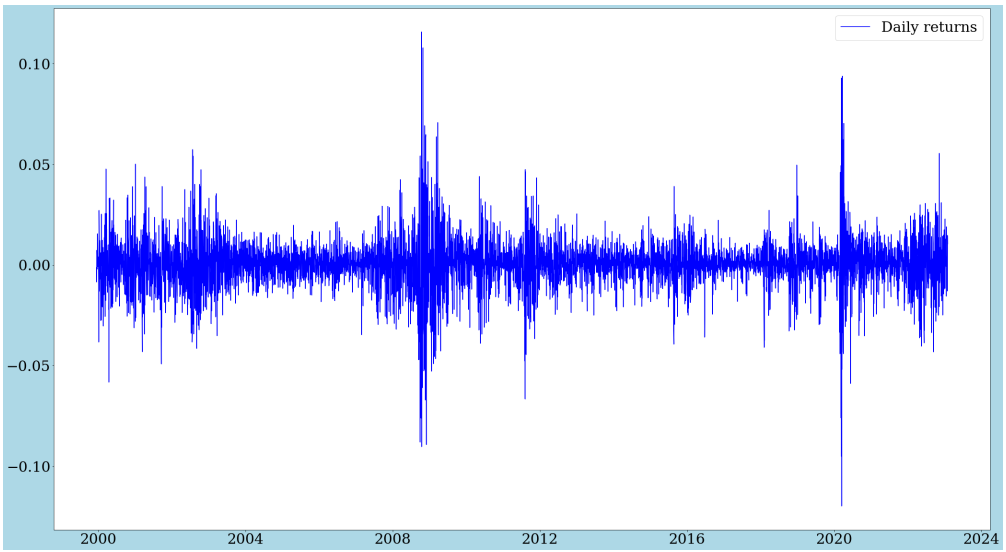
3.8 Data Description

In the study all the algorithms were tested on S&P 500 index. The data, ranging from 01.01.1996 to 11.05.2023, was imported in daily format from Yahoo Finance using *yfinance* package. Figure 9 shows the S&P 500's adjusted close price, and Figure 10 shows the S&P 500's rates of return in the covered period.

Figure 9: S&P 500 index

Note: The figure shows S&P 500 index adjusted close value in American dollars over the years 1996-2023.

Figure 10: Rates of return of S&P 500 index



Note: The figure shows S&P 500 index absolute rates of return over the years 1996-2023.

The data downloaded could not only be the period covering outputs of the models, mainly post 2000, because it had to be trained and optimized in the years prior. We wanted to show the models performance starting in the year 2000, and the longest optimization period was for the Moving Average Crossover with in sample period length of four years, as shown in the sensitivity analysis. The Table 5 shows some descriptive statistics for the data used, including arithmetic mean, standard deviation, minimum, maximum and 25%, 50%, 75% percentiles.

Table 5: Descriptive statistics S&P 500

Statistics	SP 500 index	Daily returns
Mean	1880.2144	0.0003
Standard deviation	970.8993	0.0125
Minimum	676.5300	-0.1198
Maximum	4796.5601	0.1158
P.25%	1184.4500	-0.0049
P.50%	1433.1899	0.0006
P.75%	2374.7300	0.0059
Skewness	1.2492	-0.1529
Kurtosis	0.6143	9.9866
Normal test	1007.9678	1043.5912
p-value	0.0000	0.0000

Note: P. is short for Percentile. Descriptive statistics calculated on data ranging from 1996.01 to 2022.01. All the prices are denoted in inflation adjusted American dollars.

4 Empirical results - base case

Every investment strategy is presented in two ways: Long Only and Short Allowed.

- **Long Only**
  This strategy does not allow for shorting our positions, i.e. such a strategy produces only two types of signals that is Buy Signal (1) and Market Neutral (0). Long only strategies are often used by investors, who have a more conservative approach to investing and want to avoid risks associated with short selling. Such investors buy assets with expectation that their value will increase over time.
- **Short Allowed**
  This strategy allows for shorting our positions, and similarly to *Long Only* it produces two signals that is Buy Signal (1) and a sell signal (-1). By adding the possibility of short selling, investors can profit from falling prices as well as rising prices. This allows them to take advantage of market inefficiencies and to potentially generate higher returns.

In this chapter base case empirical results the following models will be displayed and analyzed and compared to benchmark Buy And Hold strategy. The models are divided into following subclasses:

- **Benchmark**
  The benchmark strategy used is simple B&H, since we focus on Efficient Market Hypothesis.
- **Vanilla**
  By *vanilla* we denote models that are used to derive more complicated algorithms from. These are:
  - *Moving Average Crossover*
    To optimize base case Moving Average Crossover method, we use both grid search and walk forward optimization with hyperparameters defined in Table 1.
  - *XGBoost*
    To optimize base case XGBoost method, we use grid search with hyperparameters defined in Table 2 and walk forward optimization with walk forward window equal to 1000 days, consisting of train, validation and test set with lengths of the mentioned sets being equal to $7 : 1 : 2$ respectively.

  - *LSTM*
    To optimize base case LSTM method, we use grid search with pre-defined combinations of hyper-parameters defined in Table 3 and walk forward optimization with window equal to 1000 days, consisting of train, validation and test set with lengths of the mentioned sets being equal to 7 : 1 : 2 respectively as defined in Table 4. Also, in this table are specified information about training inputs (besides prices) used to calibrate the model.

- **Stacking**
  By *stacking* we mean ensembling algorithms on the input level, so in case of the methods used in this work we feed LSTM outputs (generated signals) from both XGBoost and Moving Average Crossover. We also denote stacking algorithm $\mathbb{X}$ with outputs from algorithm $\mathbb{Y}$ by $\mathbb{X}$ x $\mathbb{Y}$. We obtain the following:

  - *LSTM x MA*
    We optimize LSTM the same way, as vanilla LSTM. Apart from the vanilla LSTM training inputs, we add signals generated by base case Moving Average Crossover.
  - *LSTM x XGBoost*
    We optimize LSTM the same way, as vanilla LSTM. Apart from the vanilla LSTM training inputs, we add signals generated by base case XGBoost.
  - *LSTM x MA x XGBoost*
    We optimize LSTM the same way, as vanilla LSTM. Apart from the vanilla LSTM training inputs, we add signals generated by both base case MAC and base case XGBoost.

- **Voting**
  By *voting* we mean the procedure of generating a signal described in equation (28) by vanilla LSTM, MAC and XGBoost. We denote ensembling algorithms $\mathbb{X}$ and $\mathbb{Y}$ by voting as $\mathbb{X}$ $\mathbb{Y}$. This way, the following is obtained:

  - *LSTM MA*
  - *LSTM XGBoost*
  - *LSTM MA XGBoost*

All of the results of strategies described above are compared with ARC, MD, MLD, IR*, IR** and IR*** metrics.

4.1 Long Only strategies

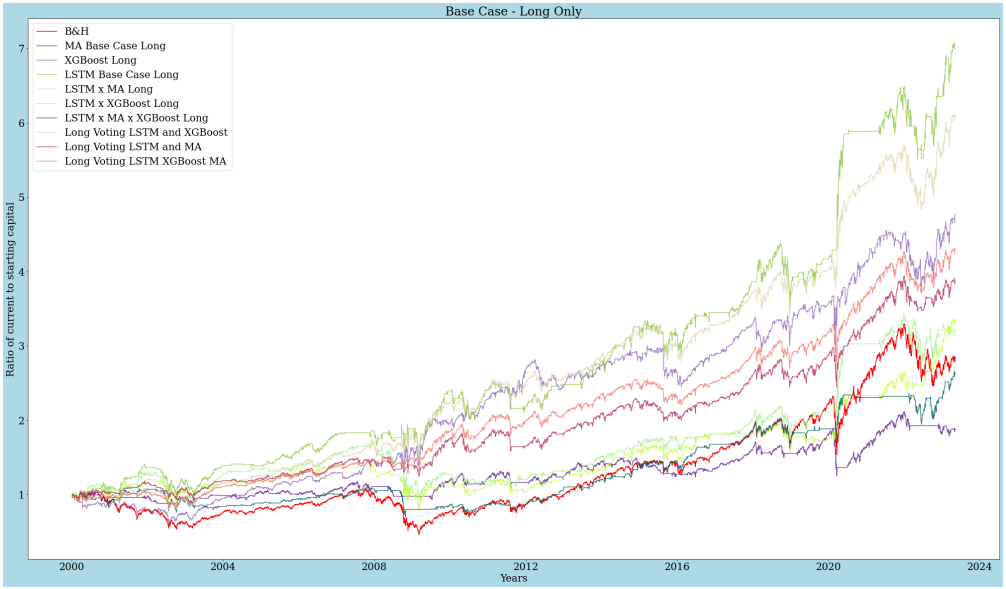
Table 6 and Figure 11 show results of the benchmark, Vanilla, Stacking and Voting base case strategies for S&P 500 on daily frequency using Long Only format.

Table 6: Base Case - Long Only

Type	Model	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***	No. trades
Benchmark	B&H	4.56	19.79	0.23	56.78	0.02	1.35	6.27	2
Vanilla	MA	2.73	12.75	0.21	33.92	0.02	1.36	3.45	142
	XGBoost	6.91	14.80	0.47	37.43	0.09	1.37	43.54	2606
	LSTM	**8.57**	14.19	0.60	31.61	0.16	1.35	103.68	411
Stacking	LSTM x MA	5.02	15.96	0.31	47.07	0.03	7.19	2.34	405
	LSTM x XGBoost	5.22	14.77	0.35	43.12	0.04	6.17	3.63	641
	LSTM x MA x XGBoost	4.19	13.28	0.32	36.42	0.04	2.68	5.66	489
Voting	LSTM MA	5.88	11.23	0.52	21.24	0.14	**0.27**	**315.82**	631
	LSTM XGBoost	7.95	12.69	**0.63**	26.57	**0.19**	1.23	121.39	2796
	LSTM MA XGBoost	6.35	**11.18**	0.57	**21.37**	0.17	1.43	74.94	2845

Note: Performance metrics for all base case strategies in Long Only scenario calculated from 2000.01.01 to 2023.05.11. Bold font means that the strategy is the best in given metric. Length of the entire walk forward window was 1000 days, and the ratio of train to validation to test set is equal to 7 to 1 to 2.

Figure 11: Base case strategies results - Long Only



Note: The figure shows Equity Curves of the strategies in the base case Long Only format. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

Analyzing the results of Long Only strategies presented in Figure 11 and Table 6, we note that all but two of the strategies managed to obtain higher ARC than simple Buy And Hold strategy. Vanilla LSTM proved to be the best under ARC metric (ARC=8.57). Voting in general seems to have the best scores, distributed among its iterations. Voting LSTM MA XGBoost had the lowest ASD (ASD=11.18) and MD (MD=21.37) metric. Voting LSTM XGBoost obtained the best IR* and IR** scores, respectively equal to 0.63 and 0.19. Voting LSTM MA managed to have the shortest loss period (MLD=0.27), thus having the highest IR*** score of 315.82. Stacking, that is one of two analyzed types of ensembling machine learning algorithms, was not the best under any criterion.

4.2 Short Allowed strategies

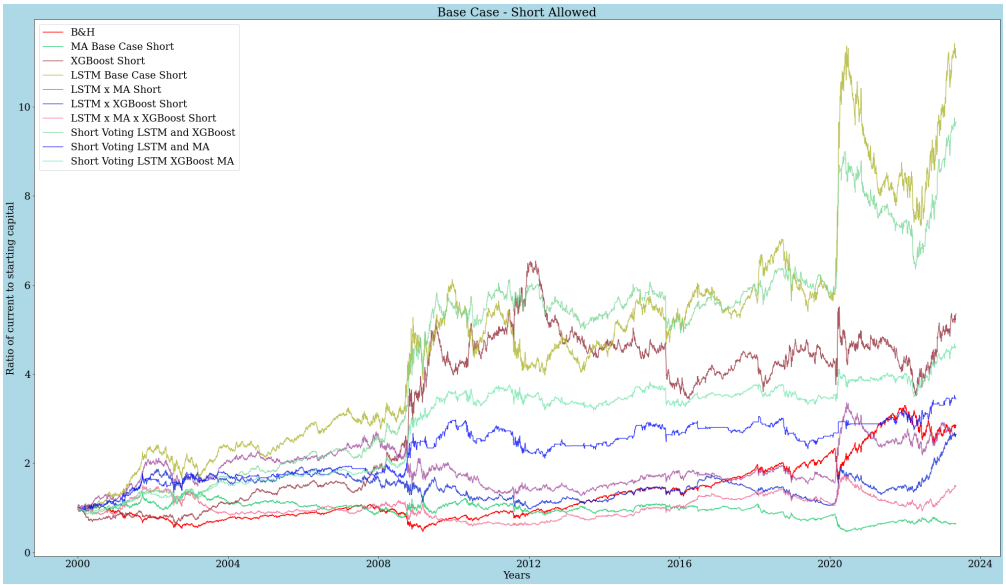
Table 7 and Figure 12 show results of the benchmark, Vanilla, Stacking and Voting base case strategies for S&P 500 on daily frequency using Short Allowed format.

Table 7: Base Case - Short Allowed

Type	Model	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***	No. trades
Benchmark	B&H	4.56	19.79	0.23	56.78	0.02	**1.35**	6.27	2
Vanilla	MA	-1.92	19.80	-0.10	66.91	-0.00	20.34	-0.03	156
	XGBoost	7.44	19.79	0.38	47.53	0.06	11.16	3.92	2606
	LSTM	**10.73**	19.65	0.55	35.56	0.16	3.19	55.44	411
Stacking	LSTM x MA	4.16	19.66	0.21	55.38	0.02	12.32	0.54	405
	LSTM x XGBoost	4.19	19.66	0.21	49.65	0.02	16.86	0.45	641
	LSTM x MA x XGBoost	1.72	19.66	0.09	61.22	0.00	18.48	0.02	489
Voting	LSTM MA	5.37	13.03	0.41	28.84	0.08	8.05	5.10	640
	LSTM XGBoost	10.07	13.97	**0.72**	29.27	0.25	3.02	**82.78**	2797
	LSTM MA XGBoost	6.68	**10.80**	0.62	**15.19**	**0.27**	2.57	70.71	2858

Note: Performance metrics for all base case strategies in Short Allowed scenario calculated from 2000.01.01 to 2023.05.11. Bold font means that the strategy is the best in given metric. Length of the entire walk forward window was 1000 days, and the ratio of train to validation to test set is equal to 7 to 1 to 2.

Figure 12: Base Case Strategies Results - Short Allowed



Note: The figure shows Equity Curves of the strategies in the base case Short Allowed format. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

Analyzing results in Table 7 and Figure 12, we can see that similarly to Long Only results (Table 6), Vanilla LSTM is the best strategy considering its rate of return (ARC=10.73). Voting LSTM XGBoost MA proved

to be the best under ASD, MD, IR^{**}, obtaining 10.80, 15.19 and 0.27 respectively metrics. Voting LSTM XGBoost managed to have the highest IR^{*} of 0.72 and IR^{***} of 82.78. Many models, except for Stacking, managed to scored significantly higher than the benchmark B&H under all metrics. All Voting models scored at least 0.2 higher IR^{*}, at least four times higher IR^{**}, and in case of models, where XGBoost could take a vote, at least 10 times higher IR^{***} than benchmark.

Based on the results of both Long Only and Short Allowed base case models, it can be said that there are reasons to reject Research Hypothesis 1 (RH1), since in Table 6 and Table 7 several strategies are shown, which managed to outperform benchmark strategy of B&H. Additionally, there are no reasonable grounds to reject (RH2), since many ML base algorithms performed better than Moving Average Crossover strategy. Finally, there are no grounds to reject (RH3), since Voting showed promising results in both Long Only and Short Allowed format. In order to further verify research hypotheses, an extensive sensitivity analysis is conducted.

5 Sensitivity analysis

Sensitivity analysis is a crucial tool for evaluating the robustness of machine learning models in algorithmic investment strategies. By varying input parameters and analyzing the impact on output, sensitivity analysis can identify the key drivers of an algorithm's performance and assess its reliability under different market conditions. This information is essential for algorithmic traders to make informed decisions about which algorithms to use in real-world trading scenarios. Additionally, the sensitivity analysis can help to identify areas where improvements can be made to enhance the algorithm's performance, thereby increasing its profitability and reducing the risk of losses.

5.1 Moving averages

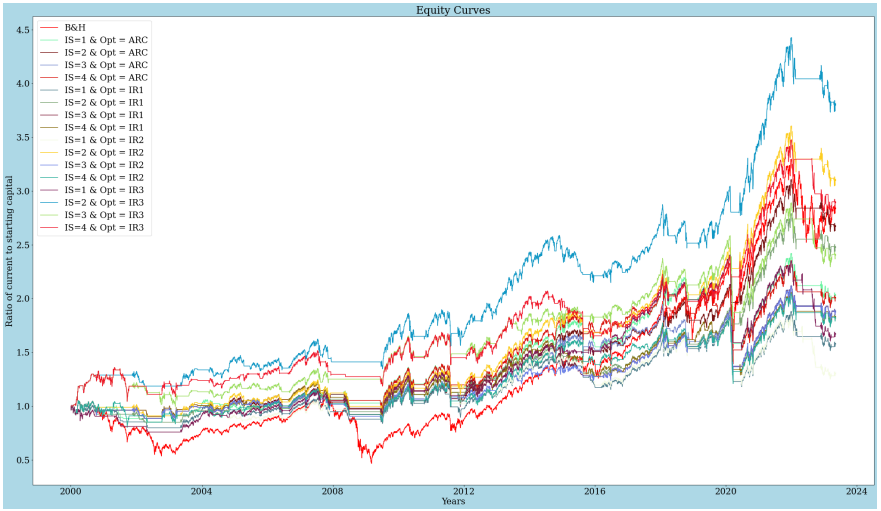
Sensitivity analysis for Moving Average Crossover trading strategy is performed according to Table 1. Ceteris paribus, both optimization criterion and length of in sample period are changed. Results are shown in Table 8 and Table 9.

5.1.1 Long Only strategies

From the results in Table 8 and Figure 13, we can clearly see that in the majority of the cases the base case strategy, that is in sample length of three years and optimization criterion IR^{*}, scored less in IR^{*}, IR^{**} and IR^{***} metrics than sensitivity analysis models. Combination of optimization criterion IR^{***} and in sample length of two years proved to be the best in every category.

It does not seem, in the case of Moving Average Crossover that there exists a correlation between in sample period length and monotonicity of acquired results. For every optimization criterion, usually in sample length of two years yields best results in every metric used, while in sample of one year yields the worst results. Base case model in Long Only format proved to be worse that benchmark strategy of B&H in all but MD metrics, whereas in performed significantly worse only in ARC and IR^{***} metrics. Twelve out of fifteen models in sensitivity analysis performed better under ARC metrics than base case, nine models performed better under IR^{*} metric, six in IR^{**} metric and nine in IR^{***} metric.

Figure 13: Moving Average Crossover Sensitivity Analysis - Long Only



Note: The figure shows Equity Curves of the Moving Average Crossover strategy during sensitivity analysis. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

Table 8: Sensitivity analysis for moving averages - Long Only

Model	IS_{len}	Optimization criterion	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***
Benchmark	-	-	4.58	19.79	0.23	56.78	0.02	**1.35**	6.33
Moving Averages	1	ARC	3.10	12.61	0.25	36.50	0.02	1.86	3.48
	2		4.31	12.07	0.36	26.39	0.06	**1.35**	18.63
	3		2.76	12.86	0.21	35.61	0.02	3.48	1.32
	4		3.04	12.49	0.24	35.59	0.02	3.48	1.81
	1	IR*	1.99	12.27	0.16	36.50	0.01	1.86	0.94
	2		3.98	11.56	0.34	22.72	0.06	**1.35**	17.69
	3		*2.73*	*12.75*	*0.21*	*33.92*	*0.02*	*1.36*	*3.45*
	4		2.63	11.96	0.22	29.57	0.02	**1.35**	3.83
	1	IR**	1.15	12.18	0.09	36.50	0.00	3.23	0.11
	2		4.98	11.73	0.42	23.48	0.09	**1.35**	33.18
	3		2.74	12.77	0.21	33.92	0.02	1.36	3.49
	4		2.61	13.38	0.19	33.92	0.01	1.36	2.87
	1	IR***	2.23	12.22	0.18	36.50	0.01	3.23	0.77
	2		**5.91**	**11.00**	**0.54**	**17.90**	**0.18**	**1.35**	**77.28**
	3		3.85	13.07	0.29	33.92	0.03	1.36	9.45
	4		4.67	11.80	0.40	23.26	0.08	**1.35**	27.45

Note: Performance Metrics on MAC in Long Only format. Out of sample period covers 2000.01.01 to 2023.05.11. Base case scenario is italicized, and the best results in each column is bolded.

5.1.2 Short Allowed strategies

From the results in Table 9 and Figure 14, we can clearly see that in nearly all of the cases, the benchmark strategy of B&H scored higher under all metrics than strategies based on moving averages, with a single exception of Moving Average Crossover optimized with IR** on two year in sample period. Twelve strategies

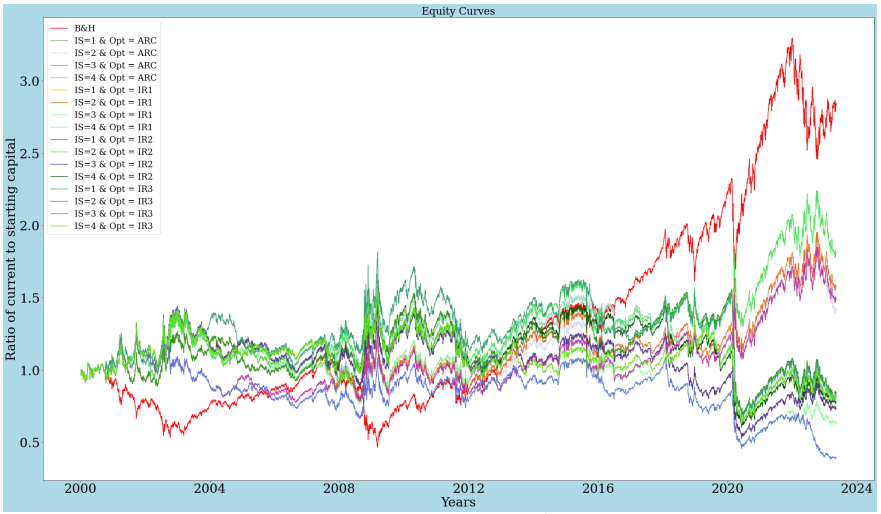
gained negative returns, resulting in negative IR*, IR** and IR***, making the results uninterpretable. In Table 8 it was observed that models optimized with two year in sample period brought the overall best results, and the same holds true for Table 9.

Table 9: Sensitivity analysis for moving averages - Short Allowed

Model	IS_{len}	Optimization criterion	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***
Benchmark	-	-	**4.58**	**19.79**	**0.23**	56.78	**0.02**	**1.35**	**6.33**
Moving Averages	1	ARC	-0.71	19.80	-0.04	58.10	-0.00	14.16	-0.00
	2		1.51	19.80	0.08	42.03	0.00	17.18	0.02
	3		-1.30	19.80	-0.07	61.82	-0.00	20.35	-0.01
	4		-0.85	19.80	-0.04	62.08	-0.00	14.16	-0.00
	1	IR*	-0.71	19.80	-0.04	58.10	-0.00	14.16	-0.00
	2		1.99	19.80	0.10	42.03	0.00	12.13	0.08
	3		*-1.92*	*19.80*	*-0.10*	*66.91*	*-0.00*	*20.34*	*-0.03*
	4		-0.85	19.80	-0.04	62.08	-0.00	14.16	-0.00
	1	IR**	-0.71	19.80	-0.04	58.10	-0.00	14.16	-0.00
	2		2.59	**19.79**	0.13	**35.40**	0.01	6.20	0.40
	3		-1.29	19.80	-0.06	63.65	-0.00	14.16	-0.01
	4		-1.06	19.80	-0.05	62.08	-0.00	14.16	-0.01
	1	IR***	-0.79	19.80	-0.04	63.28	-0.00	14.16	-0.00
	2		1.75	19.80	0.09	41.80	0.00	17.62	0.04
	3		-3.91	19.80	-0.20	71.33	-0.01	20.78	-0.20
	4		-0.89	19.80	-0.05	60.58	-0.00	14.16	-0.00

Note: Performance Metrics on MAC in Short Allowed format. Out of sample period covers 2000.01.01 to 2023.05.11. Base case scenario is italicized, and the best results in each column is bolded.

Figure 14: Moving Average Crossover Sensitivity Analysis - Short Allowed



Note: The figure shows Equity Curves of the Moving Average Crossover strategy during sensitivity analysis. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

5.2 LSTM Ensembled

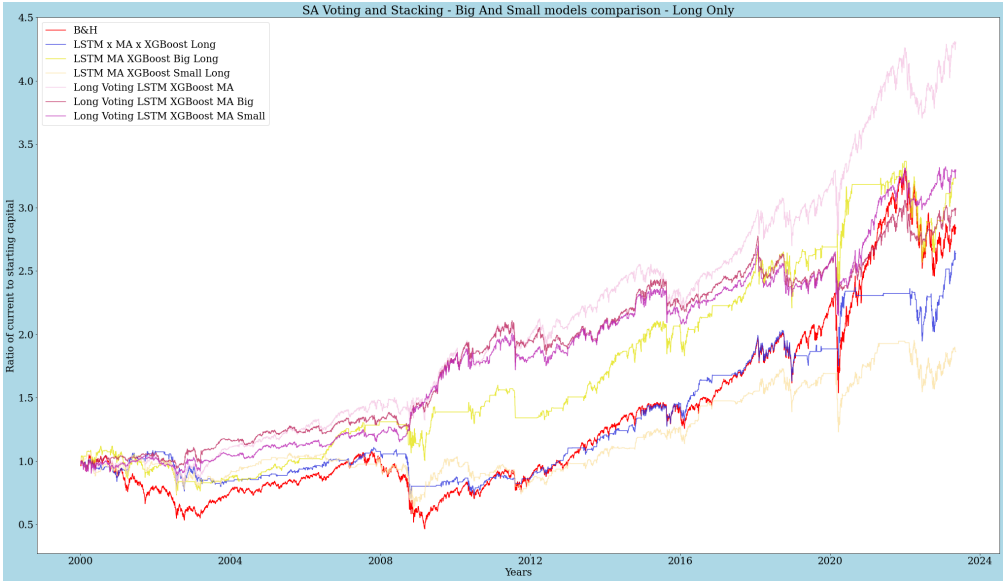
Similarly to Moving Average Crossover, sensitivity analysis will also be conducted in case of LSTM. After analyzing of the base case results in Table 6 and Table 7, we can see that the most advanced model (Stacking LSTM with both XGBoost and MAC) did perform poorly in comparison to the other, less advanced models. Because of this, the sensitivity analysis will be performed for this model to find reasons for such performance. Also, the same sensitivity analysis is performed for Voting LSTM MA XGBoost, since it showed promising results in both long only and short allowed scenarios. We change the range of technical indicators, on which the model was trained and, ceteris paribus, lengths of train, validation and test periods. Overall, eighteen new models are shown. For simplicity purposes "Big" and "Small" both refer to lengths of technical indicators fed to the model, where LSTM x XGBoost x MA is the base case ensemble (stacking) LSTM with XGBoost and MAC and LSTM XGBoost MA is the base case ensemble (voting) LSTM with XGBoost and MAC.

5.2.1 Long Only strategies

Looking at results in Table 10 and Figure 15, we can see that changing the lengths of technical indicators helped the model perform better almost under every criterion. 'Big' Stacking model outperformed base case Stacking model, 'Small' Stacking and Buy And Hold, whereas 'Small' Stacking scored lower than base case on every metric except Maximum Loss Duration. Changes between base case and 'Big' Stacking are not as significant, as changes between 'Big' and 'Small' models, since differences in between base case and 'Big' models are 0.02, 0.01, 9.54 respectively in IR*, IR** and IR*** metrics. Changes in the same metrics for 'Big' and 'Small' are 0.17, 0.04 and 13.92.

In the case of Voting, 'Big' and 'Small' models scored better than B&H under every metrics except MLD. When it comes to comparing 'Big' and 'Small' Voting models to base case Voting, the results are not improved except for ASD and MD metrics. Also, the changes in metrics between models are not as large as between Stacking models.

Figure 15: Sensitivity analysis Stacking and Voting - Changes in lengths of technical indicators - Long Only



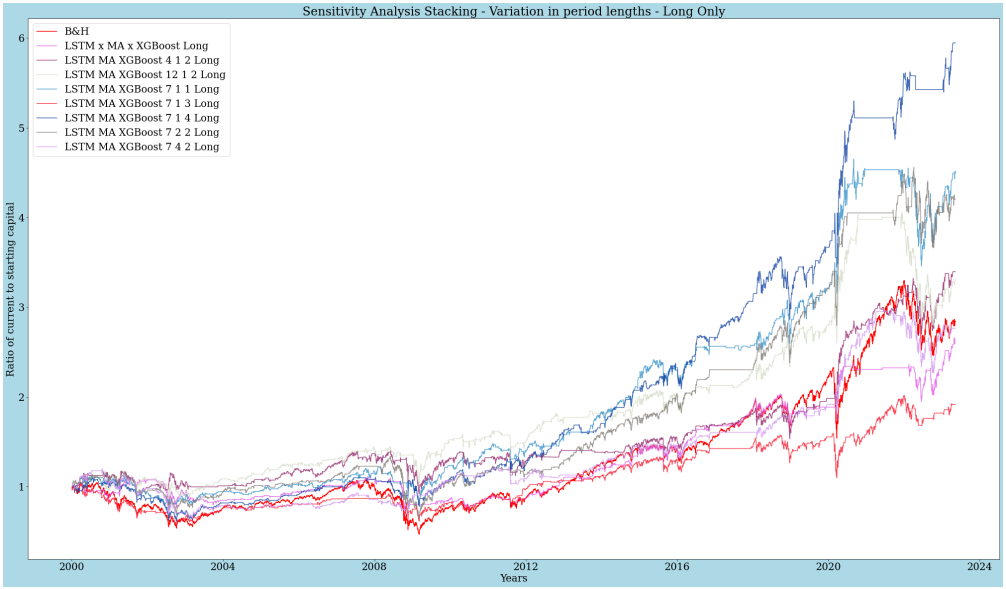
Note: The figure shows Equity Curves of the strategies in the Sensitivity analysis Long Only format. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

Table 10: Sensitivity analysis Stacking and Voting - Changes in lengths of technical indicators - Long Only

Model	Type	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***	No. trades
B&H	Benchmark	4.56	19.79	0.23	56.78	0.02	**1.35**	6.27	2
Stacking	Base Case	4.19	13.28	0.32	36.42	0.04	2.68	5.66	489
	Big	5.09	14.82	0.34	34.52	0.05	1.69	15.20	581
	Small	2.67	15.61	0.17	44.46	0.01	2.13	1.28	728
Voting	Base Case	**6.35**	11.18	**0.57**	21.37	**0.17**	1.43	**74.94**	2845
	Big	4.73	**9.83**	0.48	21.81	0.10	3.25	15.18	2834
	Small	5.16	10.05	0.51	**20.36**	0.13	2.79	23.99	2900

Note: Sensitivity analysis for Ensemble (both Voting and Stacking) of LSTM with XGBoost and MAC in Long format. "Big" and "Small" both refer to models proposed in Table 4 in Sensitivity analysis from top to bottom respectively. All of the models were optimized on 7:1:2 ratio of lengths of train to validation to test sets. Out of sample period covers 2000.01.01 to 2023.05.11.

Figure 16: Sensitivity analysis Stacking - Changes in lengths of walk forward periods - Long Only



Note: The figure shows Equity Curves of the strategies in the Sensitivity analysis Long Only format. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

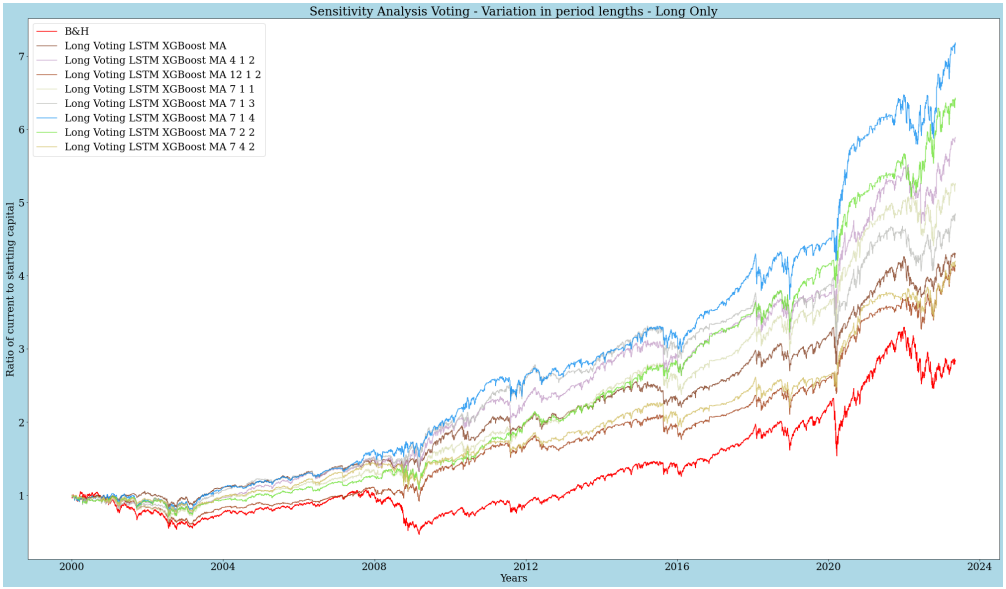
Changing ceteris paribus possible lengths of training, validation and test periods yields interesting results shown in Table 11 and Figure 16. Modifying length of any period resulted in higher ARC than the base case (except combination 7:1:3), thus five out of seven models in SA scored higher IR* than base case. Model 7:1:4 managed to score the highest ARC, IR*, IR** and IR***. The model seems robust to changes in train, validation and test periods, since the differences in values of performance metrics between the models are not large. The biggest change though is observed while changing duration of test period, since the difference between base case model and models with change test periods are nearly 0.2 in IR* and almost doubled in IR** metrics.

Table 11: Sensitivity analysis Stacking - Changes in lengths of walk forward periods - Long Only

Model	Train	Val	Test	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***	No. trades
B&H	-	-	-	4.56	19.79	0.23	56.78	0.02	1.35	6.27	2
Base Case	7	1	2	4.19	**13.28**	0.32	36.42	0.04	2.68	5.66	489
Sensitivity Analysis	4	1	2	5.28	14.07	0.38	34.53	0.06	1.24	24.45	690
	12	1	2	5.22	13.66	0.38	**31.00**	0.06	1.51	22.32	655
	7	2	2	6.13	15.69	0.39	51.82	0.05	1.77	15.97	576
	7	4	2	4.44	14.13	0.31	46.04	0.03	2.20	6.11	530
	7	1	1	6.59	14.28	0.46	35.53	0.09	2.91	19.41	709
	7	1	3	2.71	15.54	0.17	46.36	0.01	**0.54**	5.06	522
	7	1	4	**7.63**	13.57	**0.56**	43.87	**0.10**	1.10	**68.12**	712

Note: Sensitivity analysis for Stacking of LSTM with XGBoost and MAC in Long format. Out of sample period covers 2000.01.01 to 2023.05.11.

Figure 17: Sensitivity analysis Voting - Changes in lengths of walk forward periods - Long Only



Note: The figure shows Equity Curves of the strategies in the Sensitivity analysis Long Only format. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

Analyzing results of Table 12 and Figure 17, all but two models achieved higher returns than base case strategy, and all models achieved higher returns that benchmark. Base case scored less than models in SA in every metrics but ASD and MD. Model 7:1:4 achieved highest ARC, IR* and IR**, while 7:4:2 achieved lowest MLD and highest IR***. The model seems robust to changes in parameters, since the changes in performance metrics are not large between base case and sensitivity analysis models. Biggest changes in SA and base case model are 0.13 in IR*, 0.09 in IR** and a significant change of 180.17 in IR***.

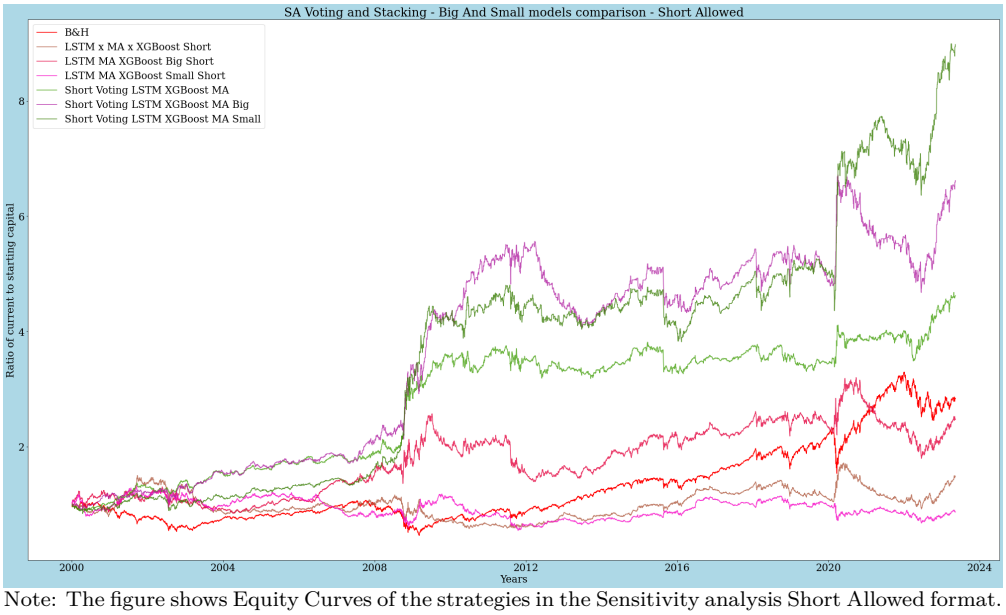
Table 12: Sensitivity analysis Voting - Changes in lengths of walk forward periods - Long Only

Model	Train	Val	Test	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***	No. trades
B&H	-	-	-	4.56	19.79	0.23	56.78	0.02	1.35	6.27	2
Base Case	7	1	2	6.35	**11.18**	0.57	**21.37**	0.17	1.43	74.94	2845
Sensitivity Analysis	4	1	2	7.74	12.58	0.61	29.11	0.16	1.23	102.47	2811
	12	1	2	6.24	12.76	0.49	39.85	0.08	0.50	96.30	2770
	7	2	2	8.00	12.91	0.62	29.22	0.17	1.00	136.18	2833
	7	4	2	6.31	13.04	0.48	26.06	0.12	**0.29**	**255.11**	2736
	7	1	1	7.29	12.96	0.56	35.25	0.12	0.91	92.72	2778
	7	1	3	6.68	13.04	0.51	29.53	0.12	0.52	150.17	2846
	7	1	4	**8.45**	12.12	**0.70**	25.28	**0.23**	1.68	117.33	2934

Note: Sensitivity analysis for Voting of LSTM with XGBoost and MAC in Long format. Out of sample period covers 2000.01.01 to 2023.05.11.

5.2.2 Short Allowed strategies

Figure 18: Sensitivity analysis Stacking and Voting - Changes in lengths of technical indicators - Short Allowed



Note: The figure shows Equity Curves of the strategies in the Sensitivity analysis Short Allowed format. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

Analyzing results in Table 13 and Figure 18, we can see that contrary to long only strategies, short allowed seem to have worse risk-adjusted returns that base case strategies. When it comes to Stacking, 'Small', as one of the only ones in this study, achieved negative ARC, thus resulting in negative IR*, IR** and IR*** scores. 'Big' Stacking achieved higher metric scores than base case, except for ASD. For Voting, 'Small' model achieved the best ARC, IR*, IR** and IR*** scores in the entire Table 13. The Stacking model seems

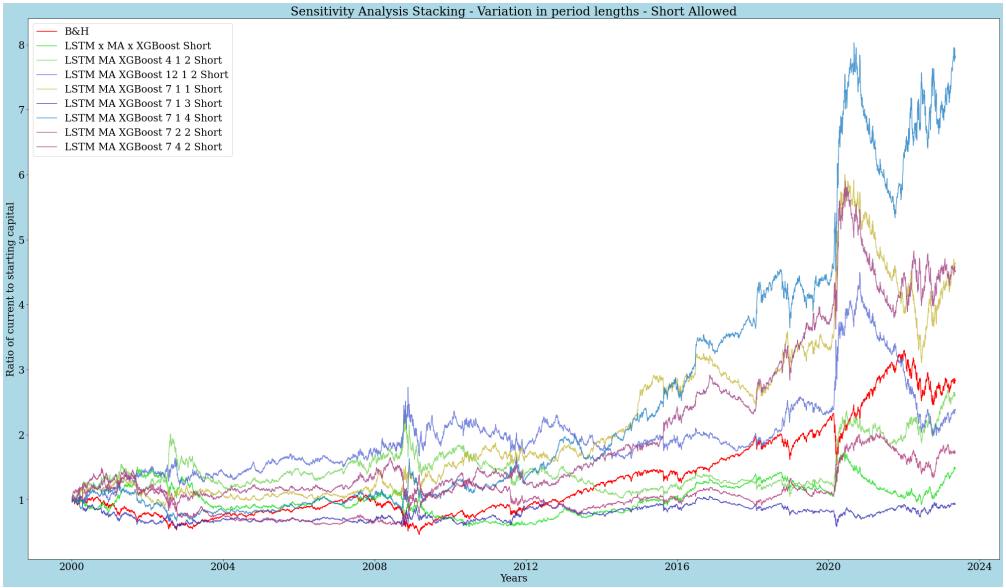
not to be robust to changes, since diversity in changes of performance metrics values can be easily seen, unlike in Voting model.

Table 13: Sensitivity analysis Stacking and Voting - Changes in lengths of technical indicators - Short Allowed

Model	Type	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***	No. trades
B&H	Benchmark	4.56	19.79	0.23	56.78	0.02	**1.35**	6.27	2
Stacking	Base Case	1.72	19.66	0.09	61.22	0.00	18.48	0.02	489
	Big	3.93	19.66	0.20	45.77	0.02	2.96	2.28	581
	Small	-0.56	19.66	-0.03	58.62	-0.00	21.95	-0.00	728
Voting	Base Case	6.68	**10.80**	0.62	**15.19**	**0.27**	2.57	70.71	2858
	Big	8.30	13.99	0.59	30.37	0.16	3.43	39.18	2752
	Small	**9.70**	14.01	**0.69**	24.71	**0.27**	1.65	**159.85**	2820

Note: Sensitivity analysis for Ensemble (both Voting and Stacking) of LSTM with XGBoost and MAC in Short Allowed format. "Big" and "Small" both refer to models proposed in Table 4 in Sensitivity analysis from top to bottom respectively. All of the models were optimized on 7:1:2 ratio of lengths of train to validation to test sets. Out of sample period covers 2000.01.01 to 2023.05.11.

Figure 19: Sensitivity analysis Stacking - Changes in lengths of walk forward periods - Short Allowed



Note: The figure shows Equity Curves of the strategies in the Sensitivity analysis Short Allowed format. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

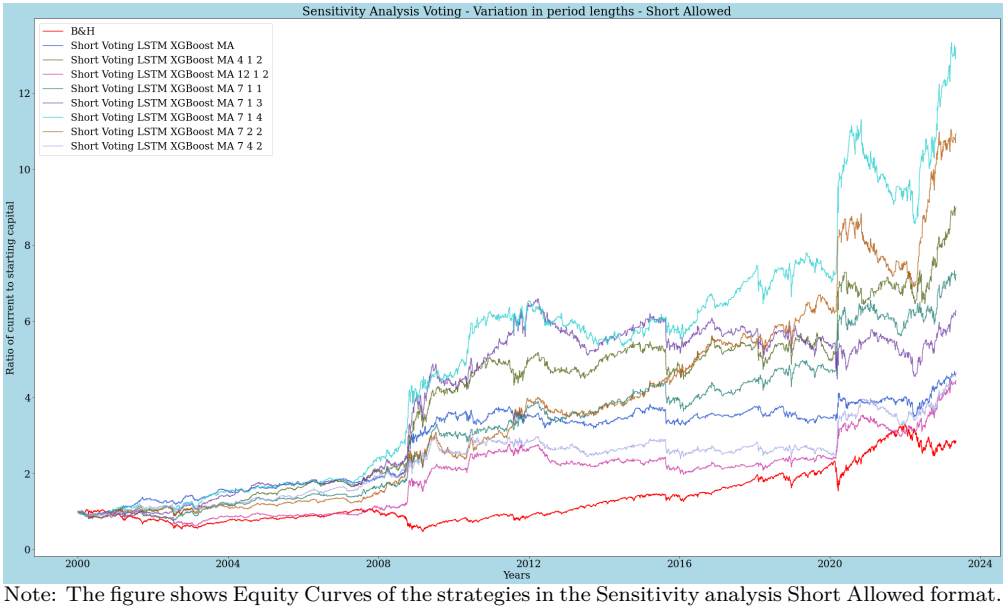
Analyzing results in Table 14 and Figure 19, we can clearly see that 7:1:2 partition wasn't the best partition to begin with, since model 7:1:4 managed to obtain higher ARC, IR*, IR** and IR*** scores. Base case proved to be the best only under MLD score. The model seems to be stable to changes in train period lengths, but is not robust to changes in test and validation period length. Between base case and models with changed train period lengths, changes in IR* and IR** are less than 0.12 and 0.02 respectively. Changes between base case and models with changed test and validation period length are significant, since they are at least 0.3 in IR* metric (apart from model 7:4:2 and uninterpretable model 7:1:3, since its IR* is negative).

Table 14: Sensitivity analysis Stacking - Changes in lengths of walk forward periods - Short Allowed

Model	Train	Val	Test	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***	No. trades
B&H	-	-	-	4.56	19.79	0.23	56.78	0.02	**1.35**	6.27	2
Base Case	7	1	2	1.72	19.66	0.09	61.22	0.00	18.48	0.02	489
Sensitivity Analysis	4	1	2	4.12	19.61	0.21	56.76	0.02	11.64	0.54	690
	12	1	2	3.77	19.75	0.19	55.79	0.01	2.62	1.86	655
	7	2	2	6.46	19.46	0.33	51.82	0.04	3.71	7.20	576
	7	4	2	2.36	19.76	0.12	60.51	0.00	19.37	0.06	530
	7	1	1	6.72	19.70	0.34	48.28	0.05	3.15	10.13	709
	7	1	3	-0.28	**19.38**	-0.01	48.27	-0.00	24.12	-0.00	522
	7	1	4	**8.86**	19.40	**0.46**	**48.12**	**0.08**	3.61	**20.64**	712

Note: Sensitivity analysis for Stacking of LSTM with XGBoost and MAC in Short Allowed format. Out of sample period covers 2000.01.01 to 2023.05.11.

Figure 20: Sensitivity analysis Voting - Changes in lengths of walk forward periods - Short Allowed



Note: The figure shows Equity Curves of the strategies in the Sensitivity analysis Short Allowed format. The vertical axis shows ratio of current to starting capital, and horizontal axis shows sequential years. Out of sample period covers 2000.01.01 to 2023.05.11.

Analyzing results in Table 15 and Figure 20, one can see that model 7:1:4 achieved the highest ARC, IR*, IR** (and almost IR***) scores. Base case proved to be the best under ASD and MD metrics. Model 7:2:2 managed to obtain the IR*** score of 164.17. The model seems stable to changes in all train, validation and test period lengths, since the changes in metrics between base case and said models are not large. It seems that the smaller train, validation and test periods, the better ARC, IR*, IR** and IR*** (excluding IR** in model 7:1:1).

Table 15: Sensitivity analysis Voting - Changes in lengths of walk forward periods - Short Allowed

Model	Train	Val	Test	ARC (%)	ASD (%)	IR*	MD (%)	IR**	MLD (years)	IR***	No. trades
B&H	-	-	-	4.56	19.79	0.23	56.78	0.02	**1.35**	6.27	2
Base Case	7	1	2	6.68	**10.80**	0.62	**15.19**	0.27	2.57	70.71	2858
Sensitivity Analysis	4	1	2	9.68	13.78	0.70	21.68	0.31	2.08	145.73	2811
	12	1	2	6.55	14.06	0.47	40.54	0.08	8.71	5.66	2771
	7	2	2	10.41	13.76	0.76	23.04	0.34	2.17	**164.17**	2834
	7	4	2	6.78	13.98	0.49	24.08	0.14	9.72	9.53	2736
	7	1	1	8.75	14.13	0.62	31.63	0.17	1.79	83.97	2779
	7	1	3	7.83	13.71	0.57	32.11	0.14	12.13	8.99	2847
	7	1	4	**11.21**	13.40	**0.84**	24.17	**0.39**	2.84	153.15	2935

Note: Sensitivity analysis for Voting of LSTM with XGBoost and MAC in Short Allowed format. Out of sample period covers 2000.01.01 to 2023.05.11.

6 Statistical significance of results

The regression analysis of rates of return of a benchmark to those of a strategy serves as a vital tool in analysis for performance evaluation, benchmark comparison, risk assessment, factor analysis, and understanding the strategy's sensitivity to market movements. By quantifying the relationship between the strategy's returns and the benchmark's returns, the regression enables the assessment of the strategy's performance relative to the benchmark and the identification of any additional factors contributing to its returns. Moreover, it facilitates the measurement of systematic risk through the beta coefficient, providing insight into the strategy's risk exposure. Overall, this regression analysis plays a crucial role in evaluating the strategy's performance, risk profile, and its relative positioning in the market. A regression is performed:

$$R_t = \alpha + \beta \cdot r_t + \epsilon_t \tag{29}$$

where, for a trading day t, r_t is the daily return of the benchmark, R_t is the daily return from the strategy and ϵ_t is random interference. Table 16 and Table 17 and show results of regression on all models used in this study. Assuming level of significance equal to 10%, we can say that 35 out of 56 models in this study bear statistically significant results.

Table 16: Linear regression on all models in Long Only format

Model	α	Std. error α	t_α	p_α	β	Std. error β	t_β	p_β
MA Base Case	0.0000	0.0001	0.42	0.6734	0.4150	0.0064	64.53	0.0000***
XG	0.0002	0.0001	2.06	0.0395*	0.5592	0.0065	86.33	0.0000***
L Base Case	0.0002	0.0001	2.92	0.0035**	0.5105	0.0065	78.85	0.0000***
L x MA	0.0001	0.0001	1.06	0.2888	0.6453	0.0062	104.10	0.0000***
L x XG	0.0001	0.0001	1.32	0.1878	0.5523	0.0064	85.72	0.0000***
L x MA x XG	0.0001	0.0001	1.05	0.2936	0.4467	0.0064	69.33	0.0000***
Voting L and XG	0.0002	0.0001	3.48	0.0005***	0.5288	0.0046	114.03	0.0000***
Voting L and MA	0.0001	0.0001	2.54	0.0111*	0.4582	0.0043	106.98	0.0000***
Voting L XG MA	0.0001	0.0000	3.21	0.0013**	0.4878	0.0036	134.18	0.0000***
L MA XG Big	0.0001	0.0001	1.24	0.2162	0.5565	0.0064	86.44	0.0000***
L MA XG Small	-0.0000	0.0001	-0.04	0.9682	0.6170	0.0063	97.93	0.0000***
Voting L XG MA	0.0003	0.0001	3.18	0.0017**	-0.0024	0.0076	-0.35	0.6369
Voting L XG MA Big	0.0001	0.0001	2.00	0.0454*	0.3237	0.0048	66.84	0.0000***
Voting L XG MA Small	0.0001	0.0001	2.19	0.0287*	0.3067	0.0052	58.86	0.0000***
L MA XG 4 1 2	0.0001	0.0001	1.48	0.1391	0.5004	0.0065	77.42	0.0000***
L MA XG 12 1 2	0.0001	0.0001	1.41	0.1589	0.4749	0.0065	73.03	0.0000***
L MA XG 7 1 1	0.0002	0.0001	2.03	0.0424*	0.5216	0.0065	80.42	0.0000***
L MA XG 7 1 3	0.0000	0.0001	0.12	0.9026	0.6075	0.0062	97.38	0.0000***
L MA XG 7 1 4	0.0002	0.0001	2.81	0.0050**	0.4607	0.0064	72.26	0.0000***
L MA XG 7 2 2	0.0001	0.0001	1.80	0.0721*	0.6158	0.0062	98.70	0.0000***
L MA XG 7 4 2	0.0001	0.0001	0.99	0.3206	0.5103	0.0065	78.38	0.0000***
Voting L XG MA 4 1 2	0.0002	0.0001	3.44	0.0006***	0.5233	0.0046	113.82	0.0000***
Voting L XG MA 12 1 2	0.0001	0.0001	2.27	0.0231*	0.5371	0.0046	116.12	0.0000***
Voting L XG MA 7 1 1	0.0002	0.0001	3.05	0.0023**	0.5505	0.0046	119.02	0.0000***
Voting L XG MA 7 1 3	0.0002	0.0001	2.79	0.0053**	0.5456	0.0046	118.49	0.0000***
Voting L XG MA 7 1 4	0.0002	0.0001	4.23	0.0000***	0.4907	0.0046	107.77	0.0000***
Voting L XG MA 7 2 2	0.0002	0.0001	3.67	0.0002***	0.5355	0.0046	115.66	0.0000***
Voting L XG MA 7 4 2	0.0001	0.0001	2.34	0.0192*	0.5590	0.0045	122.88	0.0000***

Note: The table represents results of linear regression $R_t = \alpha + \beta \cdot r_t + \epsilon_t$, where r_t is daily return from a benchmark, and R_t is the daily return from the strategy. Regression was calculated on the entire out of sample period 2000.01.01 - 2023.05.11. MA is short for Moving Average Crossover, L is short for LSTM and X is short for XGBoost, BC is short for Base Case. Asterisks *, ** and *** denote statistical significance at 10%, 1% and 0.1% respectively.

Table 17: Linear regression on all models in Short Allowed format

Model	α	Std. error α	t_α	p_α	β	Std. error β	t_β	p_β
MA BC	0.0000	0.0002	0.28	0.7782	-0.1692	0.0129	-13.15	0.0000***
XG	0.0003	0.0002	2.06	0.0395*	0.1184	0.0130	9.14	0.0000***
L BC	0.0005	0.0002	2.93	0.0034**	0.0426	0.0128	3.33	0.0009***
L x MA	0.0002	0.0002	1.05	0.2932	0.3123	0.0122	25.68	0.0000***
L x XG	0.0002	0.0002	1.31	0.1915	0.1263	0.0127	9.94	0.0000***
L x MA x XG	0.0002	0.0002	1.03	0.3027	-0.0850	0.0128	-6.65	0.0000***
Voting L and XG	0.0004	0.0001	3.52	0.0004***	0.0792	0.0090	8.76	0.0000***
Voting L and MA	0.0003	0.0001	2.42	0.0155*	-0.0615	0.0085	-7.26	0.0000***
Voting L XG MA	0.0003	0.0001	3.18	0.0015**	-0.0024	0.0070	-0.34	0.7369
L MA XG Big	0.0002	0.0002	1.23	0.2206	0.1346	0.0127	10.60	0.0000***
L MA XG Small	-0.0000	0.0002	-0.07	0.9439	0.2557	0.0124	20.65	0.0000***
Voting L XG MA Big	0.0003	0.0001	2.84	0.0046**	0.1583	0.0089	17.81	0.0000***
Voting L XG MA Small	0.0004	0.0001	3.36	0.0008***	0.1073	0.0090	11.88	0.0000***
L MA XG 4 1 2	0.0002	0.0002	1.43	0.1520	0.0288	0.0127	2.26	0.0240*
L MA XG 12 1 2	0.0002	0.0002	1.46	0.1442	-0.0434	0.0130	-3.35	0.0008***
L MA XG 7 1 1	0.0003	0.0002	2.01	0.0447*	0.0505	0.0129	3.91	0.0001***
L MA XG 7 1 3	-0.0000	0.0001	-0.01	0.9921	0.2702	0.0119	22.65	0.0000***
L MA XG 7 1 4	0.0004	0.0002	2.66	0.0079***	-0.0204	0.0124	-1.65	0.0999*
L MA XG 7 2 2	0.0003	0.0002	1.70	0.0884*	0.2845	0.0119	23.83	0.0000***
L MA XG 7 4 2	0.0002	0.0002	1.01	0.3135	0.0235	0.0130	1.81	0.0704*
Voting L XG MA 4 1 2	0.0004	0.0001	3.46	0.0006***	0.0746	0.0089	8.38	0.0000***
Voting L XG MA 12 1 2	0.0003	0.0001	2.36	0.0184*	0.0811	0.0092	8.84	0.0000***
Voting L XG MA 7 1 1	0.0003	0.0001	3.03	0.0025***	0.1084	0.0092	11.82	0.0000***
Voting L XG MA 7 1 3	0.0003	0.0001	2.80	0.0051***	0.1465	0.0086	17.06	0.0000***
Voting L XG MA 7 1 4	0.0004	0.0001	4.16	0.0000***	0.0396	0.0086	4.62	0.0000***
Voting L XG MA 7 2 2	0.0004	0.0001	3.68	0.0002***	0.1239	0.0087	14.27	0.0000***
Voting L XG MA 7 4 2	0.0003	0.0001	2.37	0.0178*	0.1209	0.0091	13.34	0.0000***

Note: The table represents results of linear regression $R_t = \alpha + \beta \cdot r_t + \epsilon_t$, where r_t is daily return from a benchmark, and R_t is the daily return from the strategy. Regression was calculated on the entire out of sample period 2000.01.01 - 2023.05.11. MA is short for Moving Average Crossover, L is short for LSTM and X is short for XGBoost, BC is short for Base Case. Asterisks *, ** and *** denote statistical significance at 10%, 1% and 0.1% respectively.

7 Summary

The last decades have brought a very rapid growth in the quantity and quality of automated investment systems. Investment algorithms are commonly used by both private investors, large investment firms and hedge funds. If the Efficient Market Hypothesis in an informational sense were true, according to the (Malkiel, 1973), no investment strategy could generate abnormal profits. As a result, the entire process of investing and a whole host of experts working on Wall Street could be replaced by a "monkey throwing darts at a board", which in the long run would make decisions at least as good as the supposed experts. The main aim of this paper was to create and test an automated investment strategy in order to provide a counterexample to the Efficient Market Hypothesis. The work thoroughly explained the methodology, the process of walk forward optimization, discussed the results of both base case and extended sensitivity analysis. The data for S&P 500 index used in this research was downloaded from Yahoo Finance in a daily format from 1996 to 2022, thus covering various market and characteristics of returns.

At the beginning of the study, three Research Hypotheses were stated and then verified during the work. There are reasonable grounds to reject *Research Hypothesis 1*, since we were able to suggest many strategies that were able to 'beat the market' regarding risk-adjusted returns. *Research Hypothesis 2* was not rejected,

because LSTM and it's ensembles were able to achieve higher risk-adjusted returns that MAC. Last but not least, *Research Hypothesis 3* was not rejected as well, since Voting LSTM with XGBoost and MAC and many another yielded better risk-adjusted returns that simple LSTM.

The sensitivity analysis showed that Moving Average Crossover algorithm is not robust to changes in hyperparameters, while LSTM is. MAC strategy is not robust in changes to both optimization criterion and length of in sample period. While performing sensitivity analysis for LSTM ensembles, it was discovered that increasing lengths of technical indicators in the inputs yielded overall better results. Decreasing the lengths of technical indicators yielded opposite results. Changing proportions on train, validation and test periods resulted in higher risk-adjusted returns in most cases. It cannot be said, whether ceteris paribus increasing length of any train, validation of test period on LSTM yields better results or not.

This study discussed topics oftentimes omitted by researchers. First of all, many studies focus solely on vanilla machine learning models. In this study, we both compared vanilla machine learning based strategies to technical analysis based models (Moving Average Crossover) and their ensembles (Voting and Stacking). Also, many studies optimize their models on a single window, that is one train, one validation and one test period, exposing themselves to the risks of overfitting. This problem can be avoided using walk forward optimization, which was used in this study. Lastly, an often overlooked issue is robustness of the model. An extended sensitivity analysis is performed to check robustness of the models.

Within this research new perspectives on investment strategies are offered, by demonstrating effectiveness of combining LSTM, XGBoost and Moving Average Crossover in a trading algorithm. These perspectives can inform and suggest potential benefits to traders and investment practitioners. By utilizing said strategies, investors can have an advantage over other market participants. In the context of Efficient Market Hypothesis, this study challenges the notion that markets are fully efficient, meaning that prices reflect all the information available. This implies that there are opportunities for traders to exploit market inefficiencies and generate above average returns. In light of risk management, the extended sensitivity analysis offers insight into robustness and stability of the trading strategy. These findings may help manage potential risks associated with using such trading algorithms. Finally, the findings may have implications for regulatory policies on financial markets. Since this research shows profitability of proposed algorithms, it could spark discussions on regulations surrounding algorithmic trading.

The study can be extended in many ways. First and foremost, having more time and powerful PC on hands, one could perform better optimization in terms of searching for optimal parameters not chosen from literature, for example by large grid search for an every model described in the paper. Moreover, one could think of other Machine Learning methods to ensemble LSTM with, since ensembling LSTM with both MAC and XGBoost showed promising results in sensitivity analysis.

References

Alajbeg, D., Bubaš, Z., & Vukas, J. (2012). The effectiveness of the 50/200 dual exponential moving average crossover on the s&p 500. *ASBBS E-Journal*, *8*(1), 8.

Appel, G. (1979). The moving average convergence-divergence method. *Great Neck, NY: Signalert*, 1647–1691.

Aycel, Ü., & Santur, Y. (2022). A new moving average approach to predict the direction of stock movements in algorithmic trading. *Journal of New Results in Science*, *11*(1), 13–25.

Castellano Gómez, S., & Ślepaczuk, R. (2021). *Robust optimisation in algorithmic investment strategies*.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 785–794.

Chena, Y., Tongb, L., & Chenc, R. (2022). *Combined trading strategy of bitcoin and gold*.

Chlebus, M., Dyczko, M., & Woźniak, M. (2021). Nvidia's stock returns prediction using machine learning techniques for time series forecasting problem. *Central European Economic Journal*, *8*(55), 44–62.

Conlan, C. (2016). *Automated trading with r: Quantitative research and platform development*. Apress.

Cowles 3rd, A., & Jones, H. E. (1937). Some a posteriori probabilities in stock market action. *Econometrica, Journal of the Econometric Society*, 280–294.

Drahokoupil, J. (2022). Application of the XGBoost algorithm and bayesian optimization for the bitcoin price prediction during the COVID-19 period. *FFA Working Papers*.

Faber, M. (2007). A quantitative approach to tactical asset allocation. *The Journal of Wealth Management, Spring*.

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, *25*(2), 383–417.

Frattini, A., Bianchini, I., Garzonio, A., & Mercuri, L. (2022). Financial technical indicator and algorithmic trading strategy based on machine learning and alternative data. *Risks*, *10*(12), 225.

Gurrib, I. (2016). Optimization of the double crossover strategy for the s&P500 market index. *Optimization*, *7*(1), 92–107.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Huang, J.-Z., & Huang, Z. J. (2020). Testing moving average trading strategies on ETFs. *Journal of Empirical Finance*, *57*, 16–32.

James, F. (1968). Monthly moving averages—an effective investment tool? *Journal of Financial and Quantitative Analysis*, *3*(3), 315–326.

Jiang, Z. (2022). An attention GRU-XGBoost model for stock market prediction strategies. *Proceedings of the 4th International Conference on Advanced Information Science and System*, 1–5.

Kryńska, K., & Ślepaczuk, R. (2022). *Daily and intraday application of various architectures of the LSTM model in algorithmic investment strategies on bitcoin and the s&p 500 index*.

Kumar, D. S., Thiruvarangan, B., Vishnu, A., Devi, A. S., & Kavitha, D. (2022). Analysis and prediction of stock price using hybridization of SARIMA and XGBoost. *2022 International Conference on Communication, Computing and Internet of Things (IC3IoT)*, 1–4. IEEE.

Li, Y., Zhou, P., Li, F., & Yang, X. (2021). An improved reinforcement learning model based on sentiment analysis. *arXiv Preprint arXiv:2111.15354*.

Lv, J., Wang, C., Gao, W., & Zhao, Q. (2021). An economic forecasting method based on the LightGBM-optimized LSTM and time-series model. *Computational Intelligence and Neuroscience*, *2021*, 1–10.

Mahfooz, S., Ali, I., & Khan, M. N. (2022). Improving stock trend prediction using LSTM neural network trained on a complex trading strategy. *International Journal for Research in Applied Science and Engineering Technology*, *10*(7), 4361–4371.

Malkiel, B. G. (1973). *A random walk down wall street [by] burton g. malkiel*. Norton.

Markowitz, H. (1952). Modern portfolio theory. *Journal of Finance*, *7*(11), 77–91.

Mohammed, S., Krishna, S. H., Mudalkar, P. K., Verma, N., Karthikeyan, P., & Yadav, A. S. (2023). Stock market price prediction using machine learning. *2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 823–828.

Nguyen, T. T. G., & Ślepaczuk, R. (2022). *The efficiency of various types of input layers of LSTM model in investment strategies on s&P500 index*.

Sekhar, P. C., Padmaja, M., & Sarangi, B. (2022). Prediction of cryptocurrency using LSTM and XGBoost. *2022 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS)*, 1–5. IEEE.

Sharpe, W. F. (1994). The sharpe ratio, the journal of portfolio management. *Stanfold University, Fall*.

Tapa, A., Yean, S. C., & Ahmad, S. N. (2016). Modified moving-average crossover trading strategy: Evidence in malaysia equity market. *International Journal of Economics and Financial Issues*, *6*(7), 149–153.

Wilder, J. W. (1978). *New concepts in technical trading systems*. Trend Research.

Yu, S., Tian, L., Liu, Y., & Guo, Y. (2021). LSTM-XGBoost application of the model to the prediction of stock price. *Artificial Intelligence and Security: 7th International Conference, ICAIS 2021, Dublin, Ireland, July 19–23, 2021, Proceedings, Part i 7*, 86–98. Springer.

Yuan, Z. (2023). Gold and bitcoin price prediction based on KNN, XGBoost and LightGBM model. *Highlights in Science, Engineering and Technology*, *39*, 720–725.

University of Warsaw
Faculty of Economic Sciences
44/50 Długa St.
00-241 Warsaw
www.wne.uw.edu.pl