# Predicting Prices of S&P 500 Index Using Classical Methods and Recurrent Neural Networks

***Mateusz Kijewski***
*University of Warsaw | QFRG*
*Warsaw, Poland* *mj.kijewski@student.uw.edu.pl*

***Robert Ślepaczuk***
*University of Warsaw | Dep. of Quantitative Finance and Machine Learning | QFRG*
*Warsaw, Poland* *rslepaczuk@wne.uw.edu.pl*

***Maciej Wysocki***
*University of Warsaw | Dep. of Quantitative Finance and Machine Learning | QFRG*
*Warsaw, Poland* *m.wysocki9@uw.edu.pl*

**Abstract**

This study implements algorithmic investment strategies based on classical methods and a recurrent neural network model, comparing their performance on the S&P 500 Index over 20 years (2000-2020). Our approach involves dynamic parameter optimization during backtesting using a rolling training-testing window. Each method's robustness to parameter changes was tested and evaluated by performance statistics such as the Information Ratio and Maximum Drawdown. While most of the presented strategies did not outperform the benchmark Buy&Hold strategy, combining signals from different methods proved stable, outperforming the Buy&Hold strategy by doubling its returns while maintaining the same level of risk. Classical methods with a rolling training-testing window were significantly more robust to parameter changes than the LSTM model.

**Keywords:** time series forecasting, algorithmic investment strategies, automated trading systems, long short-term memory model

## 1. Introduction

The S&P 500 Index is a widely recognized benchmark of the United States stock market, and forecasting its values remains a topic of interest for investors and financial analysts. Given investors' strong focus on risk management, optimizing trading decisions using returns weighted by risk is crucial, as it enables the maximum revenue potential from strategies. The recent growth of data mining techniques and related technologies has made it much more approachable and quicker to analyze large datasets and provide explanations for the results.

The analysis presented in this paper has enabled us to verify the following research hypotheses which were based on the current state of the literature review from the most recently published papers. The first hypothesis posits that the stock market is inefficient because it is possible to predict price movements based on historical data. The second hypothesis suggests that the combination of signals generated by different strategies will outperform each strategy individually. The third hypothesis proposes that classical methods exhibit greater robustness to changes in parameters compared to recurrent neural networks (RNNs). Lastly, the fourth hypothesis asserts that long short-term memory (LSTM) models predict price movements better than ARIMA models.

The main novelty of this paper is the proposition of ensemble methods that combine classical methods with RNNs in a single investment model, resulting in a diversified set of signals. This study operates under the assumption that there exists asymmetric information allowing for the analysis of volatility in the stock market, as well as the prediction of prices within available historical data, ultimately enabling the outperformance of the Buy&Hold (B&H) strategy. Therefore, the study encompasses multiple signal-generating methods, including the ARIMA model, macroeconomic factors, volatility breakout, momentum, and contrarian strategies, moving average crossover, and the RNN model. Each of these methods generated signals over 20 years, trading on the S&P 500 Index.

## 2. Literature Review

The application of time series analysis to forecast stock market prices is a topic widely discussed in scientific literature. Some researchers provide evidence that algorithmic trading brings many advantages to the market, such as lowering adverse selection, decreasing the amount of price discovery correlated with trading, and increasing the informativeness of quotes [8]. Others offer evidence that "algorithmic trading is associated with improved liquidity, improved efficiency, and elevated volatility" [2]. Additionally, there have been attempts to combine low liquidity and non-correlated investment strategies across various asset classes, resulting in significantly increased risk-adjusted returns of such combined systems [23]. Such research has been conducted not only on classical assets like equities, bonds, currencies, or commodities [27], but also on newly recognized asset classes such as volatility and cryptocurrency [11, 20, 27, 14].

The earliest articles were based on simple methodologies, employing moving averages, trend trading, and ARIMA class models. James [12] conducted research on the relationship between monthly future stock prices for a selected set of stocks from the NYSE from 1926 to 1960. The study indicates that combining various averages depending on market volatility would improve the quality of predictions. Brock et al. [3] verified the efficiency of the autoregressive model, GARCH-M, and random walk model using DJIA data for the years 1897-1986.

Holmberg et al. [9] utilized the volatility breakout rule to validate investment signals on US crude oil futures prices from March 30, 1983, to January 26, 2011. Market analysis is crucial for timely response to crises, as the repercussions of market collapses affect the entire economy, impacting not only current revenue but also investments, real wages, and unemployment rates [26]. Numerous measures of early indicators of crises are discussed, including National Reserves, GDP growth, National Debt, and Unemployment rate [15, 7, 5].

In addition to econometric models like ARIMA, VAR, and GARCH, more attention is being directed towards neural network applications due to their successes in pattern recognition. These patterns are also present in time series data such as stock prices. Roondiwala et al. [18] attempted to predict the returns of the NIFTY 50 index. They trained multivariate LSTM models using daily open, close, high, and low prices as features. After testing various combinations of epochs and features, the most accurate model in terms of Root Mean Square Error (RMSE) (utilized all four features and underwent 500 epochs of model training.

Other research implementing the LSTM model in a scientific approach was conducted by Chen et al. [4]. They collected data for China stocks and utilized the following specifications: a sequence length of 30 days, an RMSprop optimizer, and a learning rate of 0.001. Additionally, Bhandari et al. [1] presented an extensive study on different LSTM-based RNN architectures and additional inputs for S&P 500 Index forecasting. The authors demonstrated that a model with a single layer outperforms models with more layers in terms of forecasting accuracy.

Zhang et al. [25] introduced an AT-LSTM model, which combines LSTM with an attention-based model. The attention-based model introduces a new layer to LSTM to select weights of importance for each of the outputs generated by a simple LSTM model. The authors presented results from empirical studies using three indices: Russell 2000, DJIA, and NASDAQ. A slightly

different approach using the LSTM model was presented by Sang and Di Pierro [21]. Instead of using prices or returns for predicting stock price movements, the authors utilized well-known technical analysis trading strategy signals as features. Wang et al. [24] presented another novel model, the transformer, for predicting stock market indices and compared the results with different deep learning models, including LSTM. The authors provided results indicating that the transformer model outperforms others in terms of forecasting accuracy.

Previous research has primarily focused on selecting appropriate methods and testing their profitability without adjusting parameters based on previous results. However, there is a growing recognition of the potential benefits of merging multiple methodologies or dynamically adjusting model parameters to achieve better outcomes. This approach acknowledges the complexity of financial markets and the importance of adaptability in response to changing conditions. By integrating diverse methodologies and continuously refining model parameters, researchers can enhance the robustness and effectiveness of predictive models in financial analysis.

## 3. Data and Methodology

### 3.1. Data description

This study utilizes S&P 500 Index close prices from January 1, 1994, to May 2, 2020, with the trading period starting from January 1, 2000. The data was sourced from the Yahoo Finance API. Additionally, the study incorporates a time series consisting of initial jobless claims in weekly intervals for the same period, downloaded from the Federal Reserve Bank Economic Dataset (fred.stlouisfed.org).

### 3.2. Description of used classical methods

Each presented strategy will adhere to the same general rules during the investment process, encompassing three possible signals: buy, sell, and stop. A buy signal indicates that the strategy will utilize all available capital to purchase the asset, while a sell signal signifies that the strategy will allocate all available capital for deposit in the short transaction, which requires a 100% deposit. A stop signal indicates that the strategy will not maintain any position on the following day. Additionally, every trade will incur a transactional fee equivalent to 0.05% of the investing capital.

Except for the ARIMA and LSTM models, each presented strategy will optimize parameters using a rolling training-testing window. This iterative process consists of the following steps:

1. Select the first 504 days as a training window to estimate the model and calculate indicators. During this period, the strategy generates trades for every combination of tested parameters.
2. Select the next 504 days as a testing window. During this period, the strategy utilizes every calculated statistic and signal based on that. The algorithm then calculates a combination of tested parameters and selects parameters with the best information ratio (IR).
3. Select the next 63 days as a trading period and utilize the selected parameters from step 2 to generate signals for each day.
4. Shift each window by 63 days.
5. Repeat steps 1-4 until the last trading period reaches the end of the time series.

#### *ARIMA*

The Autoregressive Integrated Moving Average (ARIMA) model is represented by the following equation (1):

$$y_t = a_1 y_{t-1} + ... + a_p y_{t-p} + e_t + b_1 e_{t-1} + ... + b_q e_{t-q} \tag{1}$$

Where $p$ is the number of autoregressive lags, $q$ is the number of error term lags, $y_t$ is the value of the variable at time $t$ and $e_t$ is the error term at time $t$.

The detailed procedure for forecasting the future values of the S&P 500 Index is presented below.

1. Set the training sample as the first 504 days in the dataset.
2. Identify optimal values of $p$, $d$, and $q$ from all combinations of $p : (0 - 5)$, $d : (0 - 3)$, and $q : (0 - 5)$.
3. Estimate the model with the chosen $p$, $d$, and $q$ using a maximum likelihood method.
4. Forecast one step ahead.
5. If the forecasted value is higher than the last observation plus fees, generate a buy signal. If the forecasted value is lower than the previous observation minus fees, generate a sell signal. Otherwise, generate a stop signal.
6. Repeat steps 1-5, moving the training sample one day ahead, until the end of the dataset.

### *Moving average crossover*

Moving average crossover, based on short and long moving averages, has been extensively studied in the literature [10]. Many researchers have found this method to be profitable and capable of outperforming the B&H strategy in out-of-sample periods [6, 17]. A simple moving average is given by the following equation (2):

$$SMA_t(k) = \frac{y_{t-k+1} + y_{t-k+2} + ... + y_t}{k} = \frac{1}{k} \sum_{i=t-k+1}^{t} y_i \tag{2}$$

Where $k$ is the number of time series values considered.

Considering the findings of previous studies, this research employs a 504-day rolling training-testing window, as described at the beginning of this section, and optimizes the moving average crossover method within the following parameters: days for the short moving average: $\{5, 10, 15, 20, 25, 30\}$, days for the long moving average: $\{50, 100, 150, 175, 200, 225, 250, 300\}$. In each iteration, the algorithm explores 48 combinations of parameters, selects the one with the highest information ratio during the testing period, and utilizes the chosen combination for the subsequent 63 days.

### *Momentum and contrarian*

The momentum approach operates under the assumption that asset prices follow long-term trends, while short-term prices tend to randomly fluctuate around zero returns [16, 22]. On the other hand, the contrarian approach assumes that short-term price trends frequently change between upward and downward movements.

The first signal-generating system is based on a momentum approach. The algorithm calculates historical returns for each of the following lags $\{7, 21, 63, 126\}$ on the training window. Subsequently, on the testing window, it selects weights for each of the historical returns, with possible weights of 0 or 1.

The second signal-generating system is based on a contrarian approach. The algorithm calculates historical returns for each of the following lags $\{1, 2, 5, 10\}$. Then, on the testing window, it chooses weights for each of the historical returns, with possible weights of 0 or -1.

***Volatility breakout***

The volatility breakout strategy is rooted in the volatility stylized facts. Whenever the current volatility level surpasses its historical $q_1$ quantile ($q_1 > 0.5$), signifying an impending high volatility period, the strategy generates a sell signal. Conversely, if the current volatility level falls below the $q_2$ quantile ($q_2 < 0.5$), indicating a low volatility period, the strategy generates a buy signal. Otherwise, a stop signal is generated. The key advantage of this method lies in staying out of the market during periods of mixed volatility and capitalizing on consistent market behavior. Using a rolling training-testing window, the following parameters are optimized: days for calculating the distribution of volatility $\{63, 126, 252, 504, 1008\}$, quantile level $q_1 : \{0.6, 0.7, 0.8, 0.9\}$, quantile level $q_2 : \{0.1, 0.2, 0.3, 0.4\}$.

***Macro factor***

The S&P 500 Index serves as a measure of the performance of 500 companies with the highest capitalization from the NYSE and Nasdaq, making it a partial reflection of the condition of the US economy. In the literature, commonly cited indicators include National Reserves, GDP growth, National Debt, and the unemployment rate [7, 5]. This study specifically focuses on the unemployment rate indicator, which can be proxied by Initial Jobless Claims.

Every time initial claims exceed their historical $q_1$ ($q_1 > 0.5$) quantile – economic growth is expected to decrease and strategy generates a sell signal. However, if the current number of initial claims is lower than the $q_2$ ($q_2 < 0.5$) quantile – the economic growth is expected to increase, and the strategy generates a buy signal, otherwise it produces a stop signal. The following parameters are optimized during rolling training-testing windows: days for calculating the distribution of initial jobless claims $\{63, 126, 252, 504, 1008\}$, quantile level $q_1 : \{0.6, 0.7, 0.8, 0.9\}$, quantile level $q_2 : \{0.1, 0.2, 0.3, 0.4\}$.

### 3.3. Recurrent neural network

The simple extension of the neural network that facilitates information flow in a cyclic manner is known as the Recurrent Neural Network (RNN). However, RNNs encounter a significant challenge with optimization algorithms, particularly gradient descent. Due to their cyclic architecture, RNNs often encounter the issue of gradient vanishing, as discussed in [13]. This phenomenon entails that changes in the weights of past observations diminish exponentially over time. Consequently, simple RNNs struggle to capture long-term dependencies effectively.

***Long short-term memory***

Long Short-Term Memory (LSTM) networks are specifically designed to tackle the issue of gradient vanishing by modifying the architecture of Recurrent Neural Networks (RNNs). While RNNs utilize a hidden matrix of parameters passed between sequential inputs, LSTM networks introduce mechanisms to retain and forget information over long sequences. In an RNN, parameters in this matrix are optimized to minimize the loss function, and information is transferred between inputs through the hidden state. Consequently, RNNs may yield different results for the same input depending on different past contexts, as reflected in the varying weights of the hidden state.

The training data will be segmented into smaller sequences consisting of the last $n$ days' values of the S&P 500 Index, with the output being the next day's value of the index. This setup is analogous to the simple $AR(n)$ process. To generate signals, this research adopts a rolling training window of 252 days. The model is trained on this window and then used to predict values each day on the testing window. After the testing window period concludes, the model is refitted on the training window, and the process repeats until the final testing window reaches

the end of the time series.

Generating signals is similar to the ARIMA-based strategy if the predicted value is higher than the current value plus fees – generate a buy signal. If the predicted value is lower than the current value plus fees – generate a sell signal, otherwise generate a stop signal.

***Hyperparameters***

One of the main concerns when using neural networks is the number of hyperparameters and their optimization problems due to the high complexity of compiling the model. There are two ways of solving this issue, the first one is to select a training sample and use cross-validation to select the best hyperparameters on the training sample and use them for the entire testing period. The second one is to use heuristic methods and literature to select optimal hyperparameters and for that reason gain time to refit the model several times which enables training on the rolling window. In this study, a second method was chosen, because to get reliable results using the first method, the training sample needs to be significantly bigger than the testing sample. To test this method on the last 20 years, the model should be tested on the last 80 years which could be time-consuming and market relations could move importantly over such a long period. For that reason, it is expected to obtain more accurate results specifying hyperparameters at the start and refit the model during our sample several times. While the generalization of model-based trading systems might be a problematic issue, our method of training and testing allowed a robust check of the LSTM-based strategy performance, also testing for generalization across years. This approach ensured that the models were not only well-tuned but also capable of maintaining their predictive power over different periods.

The hyperparameters for our LSTM model are chosen using heuristic methods and literature, but results for combinations of other parameters will be presented in the sensitivity analysis chapter. Selected hyperparameters are as follows: number of units in hidden layers - 30, length of a single input (sequence) - 15, activation function for hidden layer - ReLU, loss function - Mean Squared Error, optimizer - Adam. epochs - 100, dropout rate - 0.2, starting learning rate - 0.01.

### 3.4. Signal combination

According to portfolio theory, when comparing two portfolios with the same expected returns, the riskier portfolio will have a higher asset correlation, indicating less diversification. Similarly, a portfolio of diversified signals generated by various strategies is expected to have lower risk than individual strategies. In this study, signals from all strategies are combined by aggregating signals from six classical strategies. Each buy signal is assigned a value of 1, each sell signal a value of -1, and each stop signal a value of 0. The sum of signals is normalized by dividing by the number of strategies, which is 6 in this case. If the composite signal is not an integer, it is treated as a ratio of capital invested in the strategy and held as cash (e.g., a ratio of 3/6 indicates that 50% of available capital is actively invested in the strategy while the other 50% is held as cash).

This ensemble method is likely to yield lower annual returns compared to strategies employing 100% of capital consistently. However, it is expected to significantly reduce risk, as measured by standard deviation and maximum drawdowns. Consequently, this strategy should offer higher risk-adjusted returns.

### 3.5. Performance statistics

Appropriate performance statistics presented in Table 1 were selected based on [19], which extensively describes the process of creating automated trading systems and the evaluation of strategies' performance.

**Table 1.** Performance statistics definitions.

| Metric | Abbreviation | Formula | Additional Information |
|---|---|---|---|
| Daily Returns | r | $r_i := \frac{p_i - p_{i-1}}{p_t}$ | $p_t$ is asset price on day $i$ of backtest |
| Annualized Compounded Returns | aRC | $aRC := 252 * \frac{1}{N} \sum_{i=1}^{N} r_t$ | $N$ is the number of days in the whole backtesting period |
| Annualized Standard Deviation | aSD | $aSD := \sqrt{252} * \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (r_t - \overline{r})^2}$ | $\overline{r}$ is the average of daily returns |
| Maximum Drawdown | MD | $MD := \min_{\substack{i=1,\dots,t \\ t=1,\dots,N}} \sum_{j=i}^{t} R_j$ | |
| Maximum Loss Duration | MLD | $MLD := \frac{y-x}{252}$ | $x, y$ are days defined in formula MD formula and indicate days of consecutive local maxima of the equity values |
| Average Maximum Drawdown | AMD | $AMD = \frac{1}{N} \sum_{i=1}^{n} MD_i^{yearly}$ | $MD_i^{yearly}$ is the MD calculated on a one-year period |
| Information Ratio | IR | $IR := \frac{aRC}{aSD}$ | |
| allRisk | allRisk | $allRisk = \frac{aSD * MD * MLD * AMD}{1000}$ | |
| Annual Return Compounded/Maximum Drawdown | aRCMD | $aRCMD := \frac{aRC}{MD}$ | |
| Annual Return Compounded/Average Maximum Drawdown | aRCAMD | $aRCAMD = \frac{aRC}{AMD}$ | |

# 4. Empirical results

## 4.1. Classical methods

Based on the analysis of the six classical methods applied to S&P 500 data over the last 20 years, the following observations were made. ARIMA performed exceptionally well during periods of high volatility and significant downward trends, particularly during the financial crisis of 2008-2009 and the 2020 COVID crisis. However, it did not consistently outperform the B&H strategy, achieving an IR of 0.13 compared to 0.16 for B&H. While MA Crossover initially outperformed the B&H strategy, it underperformed in the latter part of the trading period. It did not surpass the benchmark throughout the entire period, with an IR of 0.06 compared to 0.16 for B&H. Contrarian strategy demonstrated notable performance during the financial crises of 2008-2009 and 2020 COVID, achieving even higher gains during these periods. It outperformed the benchmark with an IR of 0.35 compared to 0.16 for B&H. Momentum strategy experienced a gradual decline in value over the trading period, likely due to transaction costs. Consequently, it did not surpass the benchmark, yielding an IR of -0.19 compared to 0.16 for B&H. Volatility breakout designed to reduce risk during uncertain market conditions, aimed to stay out of the market unless there were strong downward trends or low volatility upward trends. However, it did not outperform the benchmark, achieving an IR of 0.10 compared to 0.16 for B&H. The macro factor strategy effectively generated correct buy signals for most of the trading period, except for the 2020 COVID crisis, where the market panic led to a deviation from its predictions. Despite this, it outperformed the benchmark with an IR of 0.25 compared to 0.16 for B&H. Overall, while some strategies demonstrated strong performance during specific market conditions, none consistently outperformed the B&H strategy over the entire trading period.

## 4.2. LSTM

The LSTM-based strategy exhibited superior performance compared to the B&H strategy, particularly during periods of market turmoil. Similar to the ARIMA and Contrarian strategies, the LSTM strategy performed exceptionally well during the 2008-2009 financial crisis. It outperformed the benchmark strategy, achieving an IR of 0.27 compared to 0.16 obtained by the B&H strategy.

## 4.3. Combination of signals

The final step in implementing the strategies in this research was to combine them into an ensemble system based on a combination of single signals. The results of such an ensemble method are presented in Figure 1. One of the main advantages of combining various signals is that they tend to align when the market trend is strong while neutralizing each other during periods of uncertainty about the market's future direction. This approach resulted in significantly reduced risk, despite achieving returns similar to those of the B&H strategy. By using a combination


**Fig. 1.** LSTM-based strategy in comparison to benchmarks

of signals, it becomes possible to adjust leverage levels, thereby enabling similar risk levels to the B&H strategy. Figure 1 depicts the results for the combined strategy with leverage equal to 200%, clearly demonstrating that this leveraged ensemble strategy has outperformed all other methods.

Table 2 presents statistics in the base case scenario for every strategy and the combination of them used in this study. The B&H strategy obtained 3.23% for aRC and aSD equal to 19.49%, resulting in a 0.16 for IR. It also took more than 7 years to achieve a new local maximum after the 2001 downward trend. The MD was almost 57%. The momentum strategy performed the worst, achieving -3.87% for aRC with almost the same aSD as the benchmark. On the other hand, the best-performing strategy was contrarian, with 6.95% aRC and 0.35 IR, which outperforms the B&H strategy, doubling its results. Despite huge losses for macro factors in the 2020 COVID crisis, this strategy managed to obtain the second-best IR (0.25) and the second-best allRisk measure (5.5). Volatility breakout strategy statistics perfectly represent the purpose of risk-averse strategies, obtaining the lowest MD (28.5%) and the lowest aSD, which leads to the lowest allRisk metric (1.44). There is a positive correlation between the number of days out of the market and the risk measure, which is obvious because there is no risk of losing capital when being out of the market. Thus, adding any filtering to the strategy, which will reduce the number of transactions and investing days, will improve risk statistics. The combined strategy outperforms every strategy in almost every risk measure, which is a great achievement compared to diversifying the risk of using only one strategy instead of combined signals. At the same time, the aRC of the combined strategy is slightly higher than the aRC for the benchmark strategy. However, the strategy with increased leverage achieved 6.52 aRC compared to 3.23 of B&H with the same allRisk measure.

## 5. Sensitivity analysis

### 5.1. Classical methods

ARIMA has only one optimizable parameter, which was chosen manually—the size of the rolling training window. Therefore, only this parameter was changed and tested in the sensitivity analysis. Table 3 presents results for three additional ARIMA models with a rolling training window equal to 126 days, 504 days, and a hooked training window in which the starting point remains unchanged, but new data is added to the already selected training window. Except for a reduced rolling window to 126 days, results are stable with similar risk and return statistics.

As presented in Table 3, the following changes in MA Crossover strategy parameters were

**Table 2.** Performance statistics of strategies

| Name | aRC | IR | aSD | MD | AMD | MLD | allRisk | aRCMD | aRCAMD | Trades | Out Signal |
|---|---|---|---|---|---|---|---|---|---|---|---|
| BuyHold | 3.23 | 0.16 | 19.95 | 56.78 | 17.34 | 7.155 | 15.92 | 0.05 | 0.18 | 1 | 0 |
| ARIMA | 2.06 | 0.13 | 15.44 | 57.74 | 12.36 | 11.02 | 12.14 | 0.04 | 0.17 | 1548 | 2786 |
| Contrarian | 6.95 | 0.35 | 19.82 | 68.38 | 15.88 | 10.79 | 23.31 | 0.10 | 0.44 | 2705 | 3 |
| Momentum | -3.87 | -0.19 | 19.94 | 79.39 | 18.67 | 17.76 | 52.47 | -0.05 | -0.21 | 399 | 0 |
| MA Crossover | 1.25 | 0.06 | 19.95 | 53.26 | 17.49 | 11.14 | 20.70 | 0.02 | 0.07 | 57 | 0 |
| Macro factor | 4.36 | 0.25 | 17.21 | 40.22 | 13.27 | 6.00 | 5.51 | 0.11 | 0.33 | 352 | 1620 |
| Volatility breakout | 0.90 | 0.11 | 8.32 | 28.56 | 7.62 | 7.95 | 1.44 | 0.03 | 0.12 | 1014 | 3620 |
| Classical methods | 1.36 | 0.15 | 9.20 | 26.91 | 8.05 | 9.18 | 1.83 | 0.05 | 0.17 | 3603 | 561 |
| LSTM | 5.14 | 0.27 | 19.20 | 50.00 | 15.87 | 7.72 | 11.76 | 0.10 | 0.32 | 447 | 566 |
| Combined | 3.82 | 0.54 | 10.64 | 27.38 | 8.83 | 8.62 | 2.22 | 0.14 | 0.43 | 3688 | 98 |
| Combined leverged | 6.52 | 0.31 | 21.23 | 50.38 | 17.32 | 8.64 | 16.01 | 0.13 | 0.38 | 3688 | 98 |

**Note**: Performance statistics of every method used in this research.

tested: 126 and 252 testing window days, 126 and 21 trading window days, changed periods: SMA $\{5, 10, 15, 20\}$; LMA $\{50, 100, 150, 170\}$. Reducing the size of the testing and trading window resulted in worse performance. However, extending the trading window to 126 days from the default 63 days improved all measured statistics. The sensitivity analysis results showed that reducing longer moving averages decreases the performance.

The momentum and contrarian strategy utilized a rolling training-testing window, thus three parameter changes were tested: testing window length, trading window length, and periods for calculating momentum returns (only momentum). As presented in Table 3, the contrarian and momentum strategies are robust to changes in parameters and all performance metrics remain stable.

The volatility breakout strategy utilized a rolling testing-training window, thus changes were tested in the testing window size, trading window size, and the number of quantiles (reducing middle quantiles and leaving $\{0.1, 0.2, 0.8, 0.9\}$). Results presented in Table 3 suggest that this strategy's performance is highly dependent on manually chosen parameters. Any change significantly decreases the results. Therefore, the method is not robust and should be reconsidered.

The macro factor strategy also utilized a rolling training-testing window, thus changes were tested in the testing window size, trading window size, and the number of quantiles used for optimization (reducing middle quantiles, leaving $\{0.1, 0.2, 0.8, 0.9\}$). As presented in Table 3, this strategy is not sensitive to parameter changes.

## 5.2. LSTM

The neural networks require defining many hyperparameters before the model optimization process. Moreover, the estimation process is highly time-consuming. Thus, the LSTM model strategy in this research did not optimize any of the hyperparameters, and all parameters were chosen manually before the estimation process using heuristic methods and literature. For that reason, it is crucial to test all selected settings in sensitivity analysis and check whether the results are stable.

As presented in Table 4, decreasing/increasing the number of days to 126/504 in the training window negatively/positively affected performance. However, the increase to 756 & 1008 days worsened the results significantly, making the LSTM results not robust to this parameter.

Analyzing results presented in Table 4 for sequence, epochs, and dropout we can notice that the LSTM model is not robust to changes in these parameters, i.e. the change in their values always affects the performance negatively.

The influence of learning rate on LSTM results is not robust as well but in a different way (Table 4). It can increase or decrease the performance depending on the exact change.

Three possibilities presented in Table 4 provided similar results, with the worst performing tanh activation function. LSTM might be robust to changes in the activation function. The

**Table 3.** Classical methods sensitivity analysis

| Classical methods sensitivity analysis. | aRC | IR | aSD | MD | AMD | MLD | allRisk | aRCMD | aRCAMD | Trades | Out Signal |
|---|---|---|---|---|---|---|---|---|---|---|---|
| BuyHold | 3.23 | 0.16 | 19.95 | 56.78 | 17.34 | 7.155 | 15.92 | 0.05 | 0.18 | 1 | 0 |
| ARIMA | 2.06 | 0.13 | 15.44 | 57.74 | 12.36 | 11.02 | 12.14 | 0.04 | 0.17 | 1548 | 2786 |
| ARIMA 504 training days | 1.68 | 0.10 | 16.44 | 68.98 | 13.79 | 11.37 | 17.78 | 0.02 | 0.12 | 1773 | 2389 |
| ARIMA 126 training days | -2.88 | -0.19 | 14.94 | 67.60 | 12.20 | 8.78 | 10.83 | -0.04 | -0.24 | 1005 | 3393 |
| ARIMA hooked | 2.85 | 0.16 | 17.88 | 61.98 | 14.58 | 9.03 | 14.59 | 0.05 | 0.19 | 1999 | 1822 |
| MA Crossover | 1.25 | 0.06 | 19.95 | 53.26 | 17.49 | 11.14 | 20.70 | 0.02 | 0.07 | 57 | 0 |
| MA Crossover 126 testing days | -1.52 | -0.08 | 19.99 | 68.53 | 19.27 | 11.14 | 29.40 | -0.02 | -0.08 | 61 | 0 |
| MA Crossover 252 testing days | 0.99 | 0.05 | 19.98 | 55.16 | 18.04 | 9.79 | 19.46 | 0.02 | 0.06 | 57 | 0 |
| MA Crossover 21 trading days | 0.60 | 0.03 | 19.95 | 53.90 | 18.22 | 11.14 | 21.83 | 0.01 | 0.03 | 59 | 0 |
| MA Crossover 126 trading days | 2.08 | 0.10 | 19.95 | 47.09 | 16.86 | 5.61 | 8.88 | 0.04 | 0.12 | 47 | 0 |
| MA Crossover changed periods | -4.30 | -0.22 | 19.95 | 76.47 | 20.90 | 17.76 | 56.62 | -0.06 | -0.21 | 81 | 0 |
| Contrarian | 6.95 | 0.35 | 19.82 | 68.38 | 15.88 | 10.79 | 23.23 | 0.10 | 0.44 | 2705 | 3 |
| Contrarian 252 testing days | 8.60 | 0.43 | 19.82 | 50.06 | 14.82 | 9.11 | 13.39 | 0.17 | 0.58 | 2737 | 0 |
| Contrarian 126 testing days | 7.06 | 0.36 | 19.82 | 62.19 | 15.77 | 10.79 | 20.98 | 0.11 | 0.45 | 2545 | 0 |
| Contrarian 21 trading days | 7.20 | 0.36 | 19.83 | 67.93 | 15.80 | 10.79 | 22.97 | 0.11 | 0.46 | 2717 | 0 |
| Contrarian 126 trading days | 6.29 | 0.32 | 19.83 | 70.37 | 16.15 | 10.79 | 24.33 | 0.09 | 0.39 | 2695 | 0 |
| Momentum | -3.87 | -0.19 | 19.94 | 79.39 | 18.67 | 17.76 | 52.47 | -0.05 | -0.21 | 399 | 0 |
| Momentum 252 testing days | -2.91 | -0.15 | 19.94 | 72.61 | 19.05 | 20.26 | 55.87 | -0.04 | -0.15 | 383 | 0 |
| Momentum 126 testing days | -3.56 | -0.18 | 19.94 | 75.99 | 18.72 | 20.26 | 57.46 | -0.05 | -0.19 | 388 | 0 |
| Momentum 21 trading days | -1.96 | -0.10 | 19.94 | 63.25 | 18.05 | 20.26 | 46.11 | -0.03 | -0.11 | 362 | 0 |
| Momentum 126 trading days | -1.18 | -0.06 | 19.93 | 56.39 | 18.52 | 9.82 | 20.44 | -0.02 | -0.06 | 353 | 0 |
| Momentum changed periods | -8.95 | -0.45 | 19.93 | 91.89 | 20.52 | 20.29 | 76.25 | -0.10 | -0.44 | 523 | 0 |
| VB | 0.90 | 0.11 | 8.32 | 28.56 | 7.62 | 7.95 | 1.44 | 0.03 | 0.12 | 1014 | 3620 |
| VB 252 testing days | -1.93 | -0.17 | 11.29 | 52.83 | 8.35 | 19.59 | 9.76 | -0.04 | -0.23 | 944 | 3843 |
| VB 126 testing days | -0.79 | -0.07 | 10.47 | 35.32 | 8.43 | 17.54 | 5.47 | -0.02 | -0.09 | 1053 | 3751 |
| VB 21 trading days | 0.29 | 0.03 | 8.97 | 31.19 | 7.74 | 10.31 | 2.23 | 0.01 | 0.04 | 984 | 3547 |
| VB 126 trading days | -0.62 | -0.07 | 8.45 | 38.30 | 7.98 | 18.16 | 4.69 | -0.02 | -0.08 | 1086 | 3540 |
| VB less quantiles | -3.24 | -0.26 | 12.65 | 62.13 | 10.64 | 20.29 | 16.97 | -0.05 | -0.30 | 1015 | 3819 |
| Macro factor | 4.36 | 0.25 | 17.21 | 40.22 | 13.27 | 5.99 | 5.51 | 0.11 | 0.33 | 352 | 1620 |
| Macro factor 252 testing days | 4.02 | 0.24 | 17.04 | 37.75 | 13.11 | 5.98 | 5.04 | 0.11 | 0.31 | 391 | 1859 |
| Macro factor 126 testing days | 4.08 | 0.25 | 16.08 | 43.05 | 12.24 | 4.59 | 3.89 | 0.10 | 0.33 | 394 | 1841 |
| Macro factor 21 trading days | 4.50 | 0.27 | 16.80 | 40.04 | 12.88 | 5.93 | 5.14 | 0.11 | 0.35 | 381 | 1764 |
| Macro factor 126 trading days | 3.70 | 0.22 | 17.20 | 37.92 | 13.51 | 5.99 | 5.28 | 0.10 | 0.27 | 339 | 1715 |
| Macro factor less quantiles | 3.51 | 0.21 | 16.39 | 37.92 | 13.21 | 7.06 | 5.80 | 0.09 | 0.26 | 364 | 2121 |

**Note**: Sensitivity analysis results for all classical methods testing for robustness in changing strategies' parameters.

reason for this may be that LSTM models already have multiple activation functions in a single unit, thus adding one more might not have a huge impact on the optimization process.

The two alternative optimizers presented in Table 4 provided almost the same results, but significantly different from the default ADAM optimizer. The optimization process is crucial in terms of achieving accurate results. Changes in important parameters like the optimizer can affect results in every way; hence, the LSTM model is not robust to changes in the optimizer.

There were two alternative loss functions presented in Table 4, namely MAE and logcosh. The performances of these two options are significantly worse than the default MSE loss function. Similarly to the optimizer, loss functions are important in optimization processes, and changing them will alter the result of optimization.

Three alternatives presented in Table 4 performed comparably, and the best-performing setting was 10 units. Results show that a smaller number of units could improve the final outcome. The LSTM model was not robust to changes in this parameter.

## 6. Conclusions

Most of the presented strategies were not able to outperform the benchmark B&H strategy; however, this research introduced a method based on combining signals from different strategies to diversify the risk of wrong predictions by a single strategy. This method outperformed B&H, doubling its compounded returns on the same level of risk. The sensitivity analysis showed that a rolling training-testing window with dynamic optimization of parameters made the performance robust to changes in any parameters chosen at the beginning of this study. However, LSTM model results were extremely volatile to changes made for crucial parameters.

At the beginning of this paper, we stated four hypotheses, which were verified during the empirical part. The first hypothesis stated that the market is inefficient and it is possible to obtain

**Table 4.** LSTM sensitivity analysis

| Name | aRC | IR | aSD | MD | AMD | MLD | allRisk | aRCMD | aRCAMD | Trades | Out Signal |
|---|---|---|---|---|---|---|---|---|---|---|---|
| BuyHold | 3.23 | 0.16 | 19.95 | 56.78 | 17.34 | 7.155 | 15.92 | 0.05 | 0.18 | 1 | 0 |
| LSTM | 5.14 | 0.27 | 19.20 | 50.00 | 15.87 | 7.72 | 11.76 | 0.10 | 0.32 | 447 | 566 |
| LSTM 126 training days | 0.24 | 0.01 | 19.41 | 57.28 | 16.46 | 9.77 | 17.88 | 0.00 | 0.02 | 45 | 417 |
| LSTM 504 training days | 6.26 | 0.31 | 19.95 | 42.96 | 16.01 | 6.20 | 8.51 | 0.15 | 0.39 | 328 | 159 |
| LSTM 756 training days | -6.23 | -0.32 | 19.29 | 90.29 | 18.76 | 17.76 | 58.02 | -0.07 | -0.33 | 103 | 527 |
| LSTM 1008 training days | -5.71 | -0.29 | 19.81 | 88.40 | 18.35 | 17.76 | 57.05 | -0.06 | -0.31 | 221 | 47 |
| LSTM 20 days sequence | 1.00 | 0.05 | 19.30 | 52.43 | 16.31 | 8.72 | 14.39 | 0.02 | 0.06 | 334 | 495 |
| LSTM 10 days sequence | -3.15 | -0.16 | 19.44 | 78.52 | 17.91 | 17.76 | 48.54 | -0.04 | -0.18 | 103 | 284 |
| LSTM 50 epochs | -4.71 | -0.24 | 19.45 | 85.98 | 18.65 | 20.07 | 62.59 | -0.05 | -0.25 | 50 | 497 |
| LSTM 150 epochs | -8.37 | -0.43 | 19.45 | 90.79 | 19.56 | 20.06 | 69.33 | -0.09 | -0.43 | 118 | 353 |
| LSTM 0.1 dropout rate | 0.47 | 0.02 | 19.75 | 62.05 | 16.60 | 11.14 | 22.66 | 0.01 | 0.03 | 66 | 315 |
| LSTM 0.3 dropout rate | -7.97 | -0.40 | 19.72 | 90.12 | 20.19 | 20.07 | 72.00 | -0.08 | -0.39 | 120 | 214 |
| LSTM 0.05 learning rate | -8.55 | -0.43 | 19.89 | 89.09 | 20.53 | 19.32 | 70.30 | -0.10 | -0.42 | 35 | 7 |
| LSTM 0.001 learning rate | 7.43 | 0.39 | 19.02 | 39.19 | 14.84 | 5.43 | 6.00 | 0.19 | 0.50 | 672 | 920 |
| LSTM 0.005 learning rate | -1.29 | -0.06 | 19.54 | 66.35 | 17.52 | 6.21 | 14.11 | -0.02 | -0.07 | 320 | 614 |
| LSTM 0.0001 learning rate | 4.34 | 0.23 | 18.54 | 50.22 | 14.92 | 6.25 | 8.68 | 0.08 | 0.29 | 946 | 996 |
| LSTM tanh activation | 1.66 | 0.09 | 19.02 | 55.23 | 15.36 | 7.31 | 11.79 | 0.03 | 0.11 | 1065 | 534 |
| LSTM elu activation | 5.43 | 0.29 | 18.73 | 52.11 | 14.85 | 6.29 | 9.11 | 0.10 | 0.37 | 1159 | 542 |
| LSTM selu activation | 1.56 | 0.08 | 19.39 | 57.39 | 17.35 | 7.75 | 14.98 | 0.03 | 0.09 | 886 | 424 |
| LSTM sigmoid activation | 6.74 | 0.35 | 19.45 | 52.23 | 14.24 | 6.18 | 8.95 | 0.13 | 0.47 | 691 | 311 |
| LSTM SGD | -1.26 | -0.07 | 18.02 | 72.11 | 16.17 | 11.14 | 24.26 | -0.02 | -0.08 | 922 | 1363 |
| LSTM AdaDelta | -0.62 | -0.03 | 17.54 | 65.53 | 16.71 | 11.13 | 21.38 | -0.01 | -0.04 | 936 | 1307 |
| LSTM MAE | 0.67 | 0.03 | 19.87 | 66.57 | 16.93 | 17.54 | 39.28 | 0.01 | 0.04 | 119 | 21 |
| LSTM LogCosh | -1.62 | -0.08 | 19.64 | 79.05 | 18.33 | 11.14 | 31.70 | -0.02 | -0.09 | 166 | 286 |
| LSTM 40 units | 3.82 | 0.19 | 19.70 | 44.78 | 15.59 | 7.18 | 9.87 | 0.08 | 0.24 | 174 | 312 |
| LSTM 20 units | 6.44 | 0.33 | 19.26 | 58.32 | 15.10 | 3.74 | 6.34 | 0.11 | 0.43 | 395 | 538 |
| LSTM 10 units | 8.53 | 0.44 | 19.41 | 31.97 | 13.39 | 3.12 | 2.59 | 0.27 | 0.64 | 149 | 302 |
| LSTM 5 units | -3.85 | -0.22 | 17.58 | 76.51 | 16.92 | 20.07 | 45.67 | -0.05 | -0.23 | 74 | 261 |

**Note**: Sensitivity analysis results for all LSTM strategies testing for robustness in changing strategies' parameters.

abnormal returns. As it was presented, the combination of strategies outperformed the market significantly, and sensitivity analysis showed it could be done consistently. The second hypothesis about the efficiency of signal combination also cannot be rejected because the combination of signals gave the best results by diversifying the risk of a single strategy. Sensitivity analysis did not allow the rejection of the third hypothesis as the LSTM model is not robust to changes in most of the parameters. Finally, the LSTM with selected hyperparameters outperformed the ARIMA model, as was stated in the fourth hypothesis.

## References


1. Bhandari, H.N., Rimal, B., Pokhrel, N.R., Rimal, R., Dahal, K.R., Khatri, R.K.: Predicting stock market index using LSTM. Machine Learning with Applications 9, 100320 (2022)
2. Boehmer, E., Fong, K., Wu, J.: Algorithmic Trading and Market Quality: International Evidence. Journal of Financial and Quantitative Analysis 56 (8), 2659–2688 (2021)
3. Brock, W., Lakonishok, J., LeBaron, B.: Simple Technical Trading Rules and the Stochastic Properties of Stock Returns. The Journal of Finance 7 (5), 1731–1764 (1992)
4. Chen, K., Zhou, Y., Dai, F.: A LSTM-based method for stock returns prediction: A case study of China stock market. in 2015 IEEE International Conference on Big Data (Big Data), pp. 2823–2824 (2015)
5. Frankel, J., Saravelos, G.: Can leading indicators assess country vulnerability? Evidence from the 2008–09 global financial crisis. Journal of International Economics, 87 (2), 216–231 (2012)
6. Gunasekarage, A., Power, D.M.: The profitability of moving average trading rules in South Asian stock markets. Emerging Markets Review 2 (1), 17–33 (2001)
7. Hawkins, J., Klau, M.: Measuring potential vulnerabilities in emerging market economies. (2000)

8. Hendershott, T., Jones, C.M., Menkveld, A.J.: Does Algorithmic Trading Improve Liquidity? The Journal of Finance 66 (1), 1–33 (2011)
9. Holmberg, U., Lönnbark, C., Lundström, C.: Assessing the profitability of intraday opening range breakout strategies. Finance Research Letters 10 (1), 27–33 (2013)
10. Irwin, S., Park, C.-H.: What do we know about profitability of technical analysis. Journal of Economic Surveys 21, 786–826 (2007)
11. Jabłecki, J., Kokoszczyński, R., Sakowski, P., Ślepaczuk, R., Wójcik, P.: Volatility as an Asset Class, Peter Lang GmbH (2015)
12. James, F.E.: Monthly Moving Averages—An Effective Investment Tool? Journal of Financial and Quantitative Analysis, 3 (3), 315–326 (1968)
13. Kolen J.F., Kremer, S.C.: A Field Guide to Dynamical Recurrent Networks, John Wiley & Sons (2001)
14. Kosc, K., Sakowski, P., Ślepaczuk, R.: Momentum and contrarian effects on the cryptocurrency market. Physica A: Statistical Mechanics and its Applications 523, 691–701 (2019)
15. McQueen G., Roley, V.: Stock Prices, News, and Business Conditions. Review of Financial Studies 6, 683–707 (1993)
16. Miffre, J., Rallis, G.: Momentum strategies in commodity futures markets. Journal of Banking & Finance 31 (6), 1863–1886, (2007)
17. Olson, D.: Have trading rule profits in the currency markets declined over time? Journal of Banking & Finance 28 (1), 85–105 (2004)
18. Roondiwala, M., Patel, H., Varma, S.: Predicting Stock Prices Using LSTM. International Journal of Science and Research (IJSR) 6 (2017)
19. Ryś, P., Ślepaczuk, R.: Machine Learning Methods in Algorithmic Trading Strategy Optimization – Design and Time Efficiency. Central European Economic Journal 5 (52), 206–229 (2018)
20. Sakowski, P., Ślepaczuk, R., Wywiał, M.: Can we invest on the basis of equity risk premia and risk factors from multi-factor models ? Economics and Business Review 2(16)(3), 78–98 (2016)
21. Sang C., Di Pierro, M.: Improving trading technical analysis with TensorFlow Long Short-Term Memory (LSTM) Neural Network. The Journal of Finance and Data Science 5 (1), 1–11 (2019)
22. Schiereck, D., De Bondt, W., Weber, M.: Contrarian and Momentum Strategies in Germany', Financial Analysts Journal 55 (6), 104–116 (1999)
23. Ślepaczuk, R., Sakowski, P., Zakrzewski, G.: Investment Strategies that Beat the Market. What Can We Squeeze from the Market? e-Finanse 14, 36–55 (2018)
24. Wang, C., Chen, Y., Zhang, S., Zhang, Q.: Stock market index prediction using deep Transformer model. Expert Systems with Applications 208, 118128 (2022)
25. Zhang, X., Liang, X., Li, A., Zhang, S., Xu, R., Wu, B.: AT-LSTM: An Attention-based LSTM Model for Financial Time Series Prediction. IOP Conference Series: Materials Science and Engineering 569, 052037 (2019)
26. Zoega, G.: The Financial Crisis: Joblessness and Investmentlessness. Capitalism and Society 5 (2), (2010)
27. Ślepaczuk, R., Zenkova, M.: Robustness of Support Vector Machines in Algorithmic Trading on Cryptocurrency Market. Central European Economic Journal 5 (52), 186–205 (2018)