UNIVERSITY OF WARSAW

Faculty of Economic Sciences

Working Papers
No. 25/2022 (401)

DAILY AND INTRADAY APPLICATION OF VARIOUS ARCHITECTURES OF THE LSTM MODEL IN ALGORITHMIC INVESTMENT STRATEGIES ON BITCOIN AND THE S&P 500 INDEX

Katarzyna Kryńska
Robert Ślepaczuk

Warsaw 2022

WORKING PAPERS 25/2022 (401)

Daily and intraday application of various architectures of the LSTM model in algorithmic investment strategies on Bitcoin and the S&P 500 Index

Katarzyna Kryńska[a], Robert Ślepaczuk[b*]

[a] *Faculty of Economic Sciences, University of Warsaw, Quantitative Finance Research Group*
[b] *Faculty of Economic Sciences, University of Warsaw, Quantitative Finance Research Group, Department of Quantitative Finance*

Corresponding author: rslepaczuk@wne.uw.edu.pl

Abstract: This thesis investigates the use of various architectures of the LSTM model in algorithmic investment strategies. LSTM models are used to generate buy/sell signals, with previous levels of Bitcoin price and the S&P 500 Index value as inputs. Four approaches are tested: two are regression problems (price level prediction) and the other two are classification problems (prediction of price direction). All approaches are applied to daily, hourly, and 15-minute data and are using a walk-forward optimization procedure. The out-of-sample period for the S&P 500 Index is from February 6, 2014 to November 27, 2020, and for Bitcoin it is from January 15, 2014 to December 1, 2020. We discover that classification techniques beat regression methods on average, but we cannot determine if intra-day models outperform inter-day models. We come to the conclusion that the ensembling of models does not always have a positive impact on performance. Finally, a sensitivity analysis is performed to determine how changes in the main hyperparameters of the LSTM model affect strategy performance.

Keywords: machine learning, deep learning, recurrent neural networks, LSTM, algorithmic trading, ensemble investment strategy, intra-day trading, S&P 500 Index, Bitcoin

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

Aknowledgements: This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Introduction

Numerous studies have been conducted on the ongoing debate surrounding the predictability of financial markets, but no agreement has been established yet. With the rise and development of effective machine learning (ML) algorithms in recent years, it is not surprising that this topic has resurfaced. Today's ML algorithms complete jobs that until recently could only be undertaken by highly skilled people. This has resulted in both academia and practitioners seeking an innovative technology that will revolutionize how everyone invests for years.

The emergence of High-Frequency Trading during the previous two decades has caused a very rapid transition in the financial markets. Looking at daily closing prices does not provide us with a complete view of the market. This is why in our research, we attempt to develop a successful algorithmic investment strategy (AIS) not just utilizing daily data, but also using intra-day prices. Our study focuses on the application of the Long Short-Term Memory (LSTM) networks. The following are our research questions:

RQ1. Which type of LSTM model architecture generates the best buy/sell signals for Bitcoin and/or S&P 500 Index algorithmic trading?

RQ2. Does intra-day data improve the performance of transactional systems compared to systems using daily data?

RQ3. Does ensembling assets or signal frequencies improve the outcomes of investing strategies when compared to individual strategies?

RQ4. Does hyperparameter tuning help achieve better performance of the investment strategy?

RQ5. Do the results of the study change under different assumptions?

The assets under consideration are the S&P 500 Index (SPX) and Bitcoin (BTC). Our in-sample data begins on 2013-01-01 for daily data and 2014-01-01 for higher frequencies. We decided to limit the in-sample data for intra-day since in 2014 Bitcoin volatility was quite high and the trading volumes were low, and intra-day prices might have been unreliable. The out-of-sample data ranges from 2014-02-06 to 2020-11-27 for S&P 500 Index and 2014-02-01 to 2020-12-01 for Bitcoin. This difference was driven by the fact that S&P 500 Index trading hours are restricted, while Bitcoin trading hours are 24 hours per day, every day. The length of the out-of-sample period is the same across all frequencies to ensure comparability between them.

In order to answer the research questions, we undertake empirical research that involves developing AISes based on signals from different types of LSTM model architectures (called approaches). Each approach is tested on data collected at three different intervals: daily, hourly, and every 15 minutes. In addition, we aggregate signals from all three frequencies into an ensemble investment strategy by taking the majority vote of the models for price movement prediction. Having three predictors should result in a more robust outcome. The strategies' performance is then compared to one another.

To generate buy/sell signals, we use the LSTM model for price prediction, based on a walk-forward approach. We start with hyperparameters found in the literature and then adjust them with hyperparameter tuning. It is expected that the models based on hyperparameters that were chosen by the algorithm, will perform better.

Then we utilize LSTM to forecast price direction rather than the price itself in Approach #3 and #4. This model should be better suited for the needs of generating investment signals since the loss function makes more sense in this scenario - the algorithm is penalized for making a wrong decision. Finally, we conduct sensitivity analysis to see whether the study's findings alter under other assumptions.

This paper has the following structure: Section 1 includes a brief review of the literature, focusing on the use of recurrent neural networks in stock or cryptocurrency price prediction. Section 2 describes in detail the data and financial instruments used in this study. Section 3 outlines the study's methodology, which includes a description of the LSTM, a walk-forward approach, the construction of an equity line, an ensemble model, and performance metrics. It also contains a research description, which explains all of the approaches used in the research. Section 4 presents the empirical results in the form of a chart of equity lines and a table of performance metrics. The outcomes are compared to the benchmark - the buy-and-hold strategy. In section 5, a sensitivity analysis is performed to see how the most promising approach's outcomes change when the hyperparameters are altered. Finally, conclusions of the research are provided in section 6.

1 Literature review

Academics have been interested in the issue of financial market efficiency and, consequently, predicting stock price movements for decades now. The high interest has thus resulted in a lot of research that includes the usage of machine learning methods. The Efficient Market Hypothesis, also known as Random Walk Hypothesis, states that financial markets are unpredictable because current asset prices already include all publicly available information. The theory is known in three forms (the "weak", the "semi-strong" and the "strong"). All three versions imply that short-run movements in stock prices cannot be predicted. The weak form undermines the foundations of technical analysis by stating that stock prices behave like a random walk. The semi-strong version argues that all public information is already included in the stock price, diminishing fundamental analysis. The strong version goes even further, claiming that all information about the company is reflected in the price of the stock, even if it is not publicly available. (Malkiel, 1973).

Fama (1970) argues that there exists some statistically significant evidence for positive dependence in daily returns and it can be used to create profitable trading strategies but it is either insufficient or still consistent with the efficient market hypothesis. Malkiel (2005) goes as far as to say that active equity management is unlikely to obtain higher rates of return than passive investing. On the other hand, behaviouralists state that prices of equity shares can be away from their fundamental value and professional portfolio managers not beating benchmark does not necessarily mean that the market is efficient (Barberis and Thaler, 2002). The broad community of practicing technical analysts ("technicians") are also in opposition to the Random Walk Hypothesis, claiming that prices can be predicted based on historical price returns (Lo and Hasanhodzic, 2010). Finally, it is argued that the existence of stock market phenomena and serial correlations among economic figures influencing the market are evidence against the Efficient Market Hypothesis (Abu-Mostafa and Atiya, 1996).

1.1 Statistical approach

Due to the lack of agreement on the efficient market hypothesis, numerous studies have attempted to analyze and forecast asset prices, particularly stock prices. Many statistical techniques have been developed and tested for this purpose so far. A common smoothing method for time series data is called the exponential smoothing model (ESM), which effectively employs the exponential window function to smooth the data and analyze it (Billah et al. 2006). The adaptive ESM model and the ANN are compared by De Faria et al. (2009) for forecasting Brazilian stock indices. Their investigation demonstrates that both approaches yield comparable outcomes when it comes to forecasting index returns. It is argued that neural networks do better than the ESM at predicting the right sign of index return.

The Box-Jenkins Model for time series prediction was developed by two mathematicians: George Box and Gwilym Jenkins. They describe the concepts of their model, also known as ARIMA (autoregressive integrated moving average), in their publication "Time Series Analysis: Forecasting and Control" (1976). The method uses three principles: autoregression, differencing and moving average. This autoregressive model has been widely used for stock price prediction.

Ariyo et al. (2014) build a stock price predictive model using the ARIMA model on New York Stock Exchange and Nigeria Stock Exchange data. The results show that the ARIMA model demonstrates a capability for short-term price prediction and is able to compete with other forecasting techniques, namely Artificial Neural Networks (ANNs).

Azari (2018) demonstrates the utility of the ARIMA model in predicting the future value of Bitcoin by analyzing price time series over a three-year period. They find out that this method works well for short-term prediction, such as one day, when the time-series behavior is relatively constant. However, the ARIMA model fails to account for the sudden changes in price, such as the volatility towards the end of 2017. In general, it produces significant prediction errors for a long-term projection or when trained during a three-year period in which the Bitcoin price has changed substantially.

1.2 Machine Learning Approach

It is not surprising that many machine learning (ML) techniques have been researched for forecasting the direction of financial instruments' prices given the recent increase and development of powerful computers and efficient machine learning (ML) algorithms. Support Vector Machine (SVM), Decision Trees, and ANNs are examples of supervised learning approaches that can be trained to forecast asset prices and trends based on previous data and provide insightful historical price analysis.

Long short-term memory (LSTM) networks were developed by Hochreiter and Schmidhuber in 1997. LSTM was found to be much more successful and learn much faster than other recurrent neural network (RNN) architectures (Hochreiter and Schmidhuber, 1997). The biggest advancement in comparison with previously designed RNNs is that LSTM is able to capture long-term relationships thanks to the lack of vanishing gradient problem. LSTM networks have revolutionized speech recognition but prove to be effective for other sequential tasks like stock price prediction.

Siami et al. (2018) examine if and how recently discovered deep learning-based time series forecasting algorithms, including LSTM, outperform more established ones. LSTM and other deep learning-based algorithms are found to perform better than more conventionally based algorithms like the ARIMA model. More specifically, the average error

rate reduction achieved by LSTM was between 84 and 87 percent lower than ARIMA, demonstrating LSTM's superiority over ARIMA. Furthermore, it was found that the trained forecast model behaved truly randomly and that the number of training cycles, or "epochs" as it is known in deep learning, had no impact on its performance.

Grudniewicz and Ślepaczuk (2021) apply several Machine Learning algorithms to technical analysis indicators for the WIG20, DAX, S&P 500, and a few selected CEE indices. The study's findings reveal that quantitative techniques beat passive strategies in terms of risk-adjusted returns, with the Bayesian Generalized Linear Model and Naive Bayes being the top models for the investigated indices.

Di Persio and Honchar (2017) use the price of Google stock to investigate the performance of three distinct recurrent neural network models: a basic RNN, the LSTM, and the Gated Recurrent Unit (GRU). The authors also describe and illustrate the hidden dynamics of RNN. It is clear from the data that the LSTM beat other versions with a 72 percent accuracy on a five-day horizon.

Kijewski and Ślepaczuk (2020) implement buy/sell signals by using algorithmic investment strategies based on traditional techniques and a recurrent neural network model (LSTM). The study evaluates the effectiveness of investment algorithms on S&P 500 index time series that span 20 years of data from 2000 to 2020. The method for dynamic parameter optimization throughout the backtesting process is presented in this study by employing a rolling training-testing window. The combination of signals from several methods performed well and doubled the returns on the same level of risk of the Buy & Hold strategy benchmark. LSTM model was found to be substantially less resistant to changes in parameters than conventional techniques, according to a thorough sensitivity study.

Additionally, studies attempt to use an ensemble or hybrid technique with LSTM. A deep learning-based hybrid model made up of the well-known DNN architectures LSTM and GRU is developed by Hossain et al. (2018). The S&P 500 time series, which spans over 66 years, is used by the authors to train a prediction model (1950 to 2016). In this method, the input data is passed to the LSTM network to produce a first-level prediction, and the output of the LSTM layer is passed to the GRU layer to provide the final prediction. The suggested network outperforms earlier neural network techniques, achieving a Mean Squared Error (MSE) of 0.00098 in prediction.

Non-stationarity is not taken into consideration in the majority of current studies. An excellent illustration of how an LSTM-RNN-based model may provide exceptional predictions on non-stationary data is Shah et al. (2018). The study of Shah et al. (2018) demonstrates that the LSTM model not only provides excellent results for daily forecasts, i.e., predictions made one day in advance, but also provides more than satisfactory outcomes for predictions made seven days in advance using simply the daily price as a feature. The authors purposefully use a larger training dataset (price data spanning 20 years), as that time span includes numerous market ups and downs. Authors claim that the LSTM RNN exhibits potential for identifying underlying trends and producing longer-term forecasts on volatile stock data sets with the addition of more features.

Cross-validation of ML models is another issue in financial time series forecasting. To predict future asset values, Baranochnikov and Ślepaczuk (2022) present a walk-forward procedure that is in charge of training models and choosing the best one. They test the algorithms on four financial assets (Bitcoin, Tesla, Brent Oil, and Gold) and discover that LSTM outperforms GRU in the vast majority of cases.

Stock price and return are two of the most frequent input elements in a directional

forecasting model. The decision between the former and latter variables is frequently arbitrary. Kamalov et al. (2021) compare the efficiency of stock price and return as input features in directional forecasting models, using historical data from ten high-cap US corporations spanning 10 years. They use four well-known categorization techniques as the foundation for the forecasting models in the investigation. As an independent input feature, stock price performs better than return, according to the findings. When technical indicators are included in the input feature set, the difference diminishes. Authors come to the conclusion that when predicting the direction of price movement, price is typically a more powerful input characteristic than return value.

Due to the notable increase in cryptocurrency trading on digital blockchain platforms, Machine Learning techniques are increasingly being used for reliable prediction of highly nonlinear and noisy data. Suhwan et al. (2019) investigate and examine numerous cutting-edge deep learning techniques for predicting Bitcoin prices, including deep neural networks (DNN), long short-term memory (LSTM) models, convolutional neural networks, deep residual networks, and their combinations. Experimental results reveal that while DNN-based models outperform the other models for the direction of price movement prediction, LSTM-based models surpassed the other models for price prediction. Additionally, profitability evaluation revealed that classification models were superior to regression models for algorithmic trading.

The current developments in high-frequency data estimation are linked not only to technical progress and big data's expanding processing power but also to the need to comprehend and forecast the behavior of variables across shorter time horizons. Lahmiri and Bekiros (2020) use three different types of ML models in high-frequency trading of Bitcoin: (i) algorithmic models like regression trees, (ii) statistical ML approaches like support vector regressions (SVR), and (iii) ANN topologies like feedforward (FFNN) or Bayesian regularization (BRNN). Their findings indicate an overall superiority of artificial neural networks in noisy signal environments.

Michańków et al. (2022) forecast the Bitcoin and S&P 500 index values using data from 2013 to the end of 2020 and the following frequency ranges: daily, 1 h, and 15 min. They create their own loss function, which enhances the LSTM model's predictive power in algorithmic investing strategies. They find that the main factors influencing the effectiveness of LSTM in algorithmic investment strategies are the method used to tune the hyperparameters, the model's architecture, and the estimation process.

In the spot and futures markets for the S&P 500, Schulmeister (2009) examines how technical trading strategies can use momentum and reversal effects. Based on daily statistics, 2580 technical models' profitability has continuously decreased from 1960 and has been negative since the early 1990s. The same models, however, yield an average gross return of 7.2% each year between 1983 and 2007 when based on 30-minutes-data. This outcome may be the consequence of recent improvements in stock market efficiency or a change in stock price trends from 30-minute prices to higher frequency pricing.

To summarize, many studies indicate that the LSTM model outperforms other techniques, particularly statistical approaches (e.g., ARIMA models), especially for non-stationary data. It has been proposed that models perform better when the input value is asset price rather than return. It has further been suggested that intra-day data-driven strategies should outperform inter-day data-driven strategies. In our study, we aim to enhance the research by comparing different types of LSTM models to develop successful trading strategies using prices as input data. We will also investigate whether intraday data can increase the strategy's profitability.

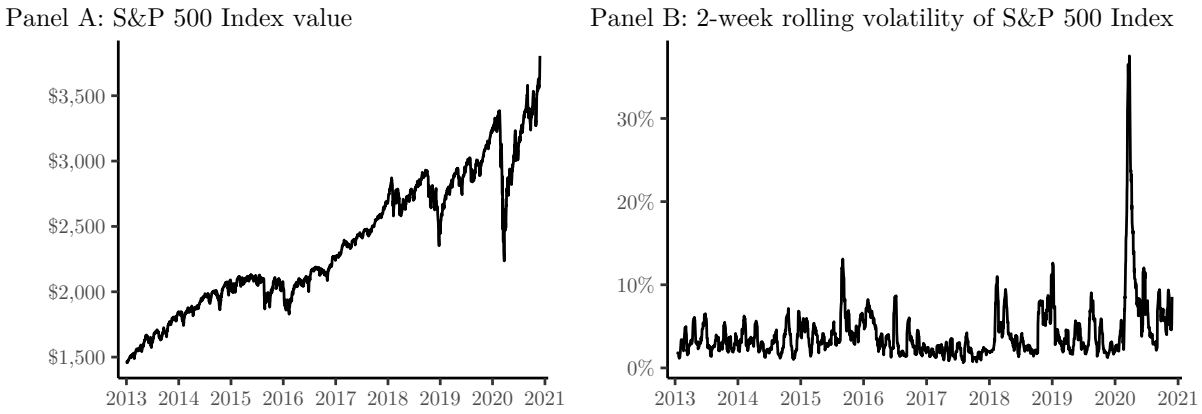
2 Data description

In our study the trading algorithm is tested on two assets: Bitcoin (BTC) and S&P 500 Index (SPX). The choice of these two assets has been dictated by the availability of intraday data. Historical intraday data from the securities trading markets is rarely available for free, especially for a longer period, which is essential to backtest the investment strategy.

In case of Bitcoin, we downloaded raw transactional data from Bitstamp - the world's longest-standing cryptocurrency exchange - and grouped it into OHLC (open-high-low-close) data, with different frequencies (15 minutes, 1 hour, and 1 day). In this manner, we obtained closing prices, which we used in the study, Regarding S&P 500 Index, we gathered data from CBOE, which contained bid and ask prices at a 1-minute frequency. We combined them into mid prices and again derived OHLC data for frequencies given above. In our analysis for S&P 500 Index, as for Bitcoin, we only use the closing price.

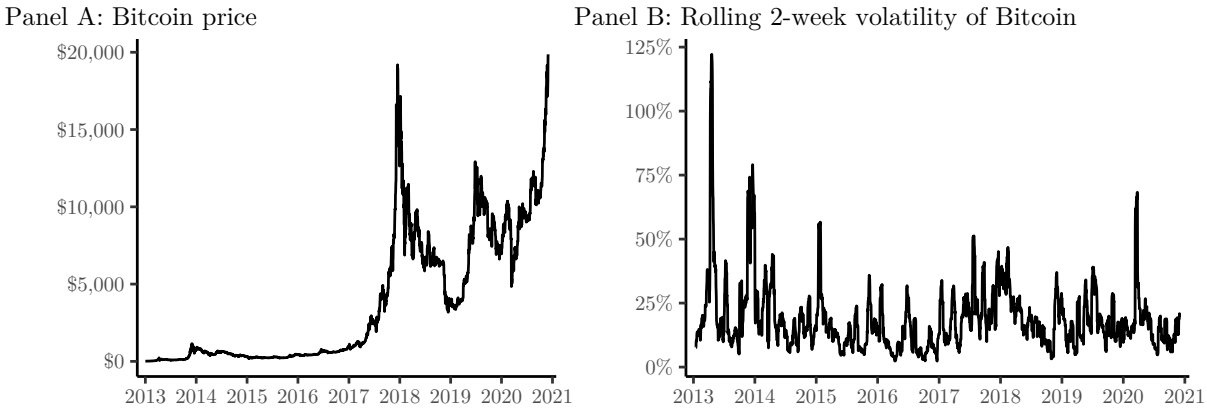
These two financial instruments differ significantly in terms of volatility. Figures 1 and 2 depict S&P 500 Index and Bitcoin prices as well as rolling 2-week volatility across the examined period. It is apparent that Bitcoin prices are characterized by very high volatility, especially in the period until 2014. Our sample is restricted in such a way that for all frequencies the out-of-sample period starts on 2014-01-01. The sample was restricted for both of the assets to ensure comparability of results. For S&P 500 Index we removed observations for which there was no trading - namely holidays and prices after 1 pm at partial holidays (days when the stock exchange closes early).

Figure 1: S&P 500 Index value and rolling 2-week volatility



Note: Daily data from 2013-01-02 until 2020-11-27. Calculated using bid-ask prices from CBOE. 2-week rolling volatility means the Standard Deviation of the last 10 daily returns, annualized.

Figure 2: Bitcoin price and its rolling 2-week volatility



Note: Daily data from 2013-01-01 until 2020-12-01. Calculated using transactional prices from Bitstamp. 2-week rolling volatility means the Standard Deviation of the last 14 daily returns, annualized. access: https://api.bitcoincharts.com/v1/csv/

The descriptive statistics obtained for all financial instruments at all frequencies are shown in Table 1. We can notice that the average return across all frequencies and for both instruments is positive. We cannot compare the Standard Deviation across frequency and assets since it is not annualized.

We can observe that the distributions of all the time series under study exhibit the negative skewness and leptokurtosis that characterize time series in finance. The distribution's negative skewness suggests that an investor could anticipate frequent small gains and occasional heavy losses. With a leptokurtic distribution, there is a larger chance of receiving exceptionally low or high returns due to wider fluctuations. A leptokurtic distribution with negative skewness (left-tailed distribution) suggests a higher risk due to the increased likelihood of negative outliers.

Table 1: Descriptive statistics for the log returns series

Frequency	Mean	Standard deviation	Maximum	Minimum	Skewness	Kurtosis	Jarque-Bera p-value
S&P Index							
daily	0.0005	0.0109	0.0900	-0.1280	-1.0317	25.6865	0
hourly	0.0001	0.0041	0.0563	-0.0720	-1.2678	43.7967	0
15-minute	0.0000	0.0021	0.0503	-0.1066	-4.6923	284.1831	0
Bitcoin							
daily	0.0025	0.0467	0.3375	-0.6639	-1.4953	27.9631	0
hourly	0.0001	0.0086	0.2055	-0.1809	-0.4622	37.9786	0
15-minute	0.0000	0.0046	0.1796	-0.1563	-0.5600	73.1581	0

Note: The statistics for the log-returns series are not annualized. The null hypothesis of the Jarque-Bera test is that the distribution is normal.

The Jarque-Bera test measures how well sample data fit a normal distribution in terms of skewness and kurtosis. The statistic can be applied to determine if the data are representative of a normal distribution. We can reject the null hypothesis that the distribution is normal since the p-value of the test for all time series is less than 0.

3 Methodology

3.1 LSTM

Artificial neural networks (ANN) loosely mimic the primary actions of biological learning systems (Jain et al., 1996). Simple units, called neurons, process a vector of input values and provide a single output value. Feed-forward neural networks (FFNNs) are the most frequent form of conventional neural network. In this system, neurons are structured in layers, where each neuron calculates a weighted sum of its inputs. The input layer receives signals from the environment, whereas the output layer transmits signals to it. Hidden neurons are connected to other neurons but are not directly linked to the environment. Feed-forward neural networks are loop-free, meaning that neurons forward signal to the next layer, never returning it back to the previous one. Consequently, only static classification tasks can be handled by feed-forward neural networks. As a result, they can only offer a static matching between input and output. A so-called dynamic classifier is required to model sequential prediction tasks.

Feed-forward neural networks can be expanded to support dynamic classification. We must feed signals from earlier timesteps back into the network in order to gain this capability. The term "Recurrent Neural Networks" refers to these networks with recurrent connections (RNN). RNNs have an internal state at each classification time step. Circular connections between neurons in higher and lower layers as well as connections for optional self-feedback are responsible for this. RNNs can convey data from earlier events to current processing steps thanks to these feedback links. RNNs develop a memory of time series events as a result. However, RNNs can only look back around ten timesteps at a time (Gers et al., 2000). The reason for this is that the gradient that carries information tends to either vanish or explode. This means that the process of learning long data sequences is hampered.

Long Short-Term Memory Recurrent Neural Networks (LSTM-RNN) were used to overcome this problem of either vanishing or exploding gradient. The solution uses a special structure of the repeating module, which has four neural network layers that interact in a unique way. Multiplicative gate units control access to the cells and learn when to allow it. Depending on the size of the created network, LSTM networks are able to learn more than 1,000 timesteps (Sepp et al., 1997).

The LSTM's core architecture is as follows. Let x_t stand for the input at the present time step t and h_{t-1} stand for the output of the hidden layer at step $t-1$. The cell state C_t is the LSTM's central component. The LSTM solves the vanishing gradient issue by allowing the gradient to be propagated effectively even after the cell state (C) was passed many steps before.

The first element of the LSTM cell is the forget gate f_t. Then the input gate i_t defines the information to update. Before a new cell state value is changed, the new candidate value $\tilde{C}_t$ is momentarily stored. h_{t-1} and x_t are the inputs at each step, and the output is generated using the parameters of the weights, biases, and the activation function (e.g. tahn or ReLU). Following is the calculation for the gates:

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f) \tag{1}$$

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i)) \tag{2}$$

$$\tilde{C}_t = tanh(W_C * [h_{t-1}, x_t] + b_C) \tag{3}$$

The forget gate f_t is then multiplied element-wise with the cell state C_{t-1} and the input gate i_t is multiplied element-wise with the potential candidate value $\tilde{C}_t$. The cell state C_t is determined by the sum of these two.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C} \tag{4}$$

The output gate o_t is the final component of the cell. This gate decides what information will be in the output.

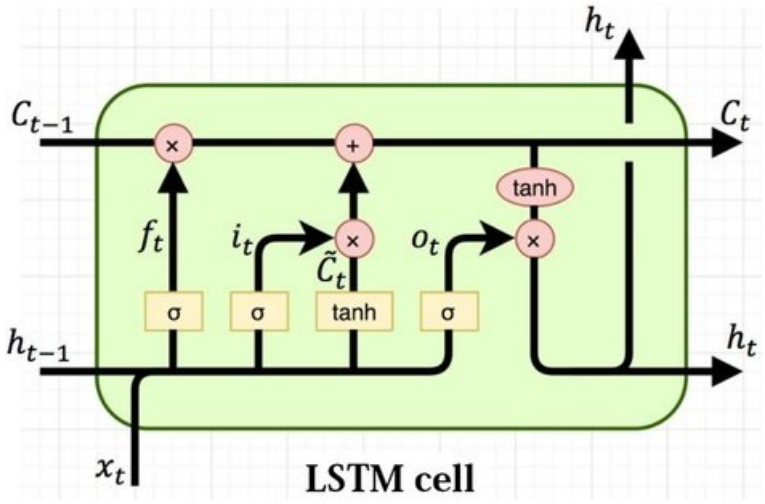
$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o) \tag{5}$$

The final step is to calculate the hidden state h_t.

$$h_t = o_t * tanh(C_t) \tag{6}$$

The construction of an LSTM cell is seen in Figure 3.

Figure 3: The LSTM cell's structure.



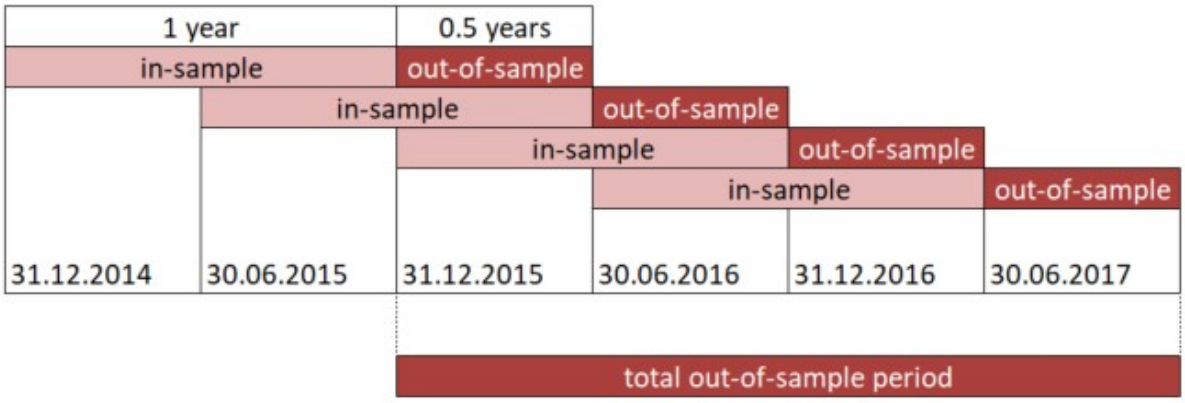
source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

3.2 Walk-forward approach

Cross-validation (CV) is a technique used to avoid overfitting by measuring out an ML algorithm's generalization error. CV is simply another example of how traditional ML approaches to financial issues go wrong. CV's hyper-parameter adjustment will increase overfitting. Because observations cannot be considered to be produced out of an IID process, CV fails in financial time series analysis and forecasting.

Instead, what financial researchers often use is a walk-forward approach. First, in-sample data is used to optimize the trading strategy. Following the in-sample data, a small portion of the rest of the data is evaluated, and the results are stored. Then, the period covered by the out-of-sample test is added to the in-sample time frame, and the procedure is repeated. Finally, by employing such a walk-forward optimization technique, we are able to generate a relatively long total out-of-sample duration made up of numerous shorter OOS periods. This prolonged total OOS is used to assess the performance metrics of tested investing strategies. Figure 4 provides an illustration of the walk forward optimization.

Figure 4: Walk forward optimization chart with 1-year in-sample periods with 0.5 years out-of-sample periods.



Walk-forward approach has two major benefits: (1) WF has a direct historical interpretation. Its success can be explained by paper trading - in other words, we simulate purchasing and selling decisions without placing actual orders. (2) Since history acts as a filter, employing sequential data ensures that there is no leakage between testing and out-of-sample datasets.

The walk-forward method may also be used to tune hyperparameters. A validate period follows the in-sample and is before the out-of-sample in this scenario. The walk-forward model training with the hyperparameter adjustment procedure is analogous to the process described above.

3.3 Construction of equity line

In order to show how LSTMs may be used in algorithmic trading, a straightforward buy-sell trading strategy is chosen based on whether the instrument price is anticipated to rise or fall over the next time period. For simplicity, we assume that the orders we place will not have an effect on the market and that they are executed instantly, at the last close price. As both S&P 500 Index and Bitcoin markets are very liquid, this assumption seems not far from the truth. If our model sends a "buy" signal, the strategy closes out a short position and takes a long position. If the long position was already taken, it leaves the position open. If the model sends a "sell" signal the algorithm takes a short position. To calculate the cumulative unrealized P&L the following assumptions are used:

1. The account is opened with $1.000;
2. Positions can be opened in any amount, they do not have to be full units;
3. Transaction costs are calculated for each opening and closing of the position, which means changing position from short to long will incur double costs. Transaction cost for S&P 500 Index amounts to 0.005%[1], for Bitcoin it is 0.1%[2];

[1]The transactional cost is the sum of the nominal cost and half of the bid-ask spread, divided by the value of the investment (e.g. for a very liquid futures contract on E-mini S&P 500 Index quoted on CME and Globex). We assume the nominal cost at the level of 1.5 USD, a contract multiplier of 50 USD, the level of the index of 3-4 thousand USD on average, and a minimum spread of 12.5 USD per contract (=0.25*$50). Therefore, finally we have TC%=($1.5+0.5*$12.50)/($3000*50)=0.005%

[2]Average transaction cost in percentage terms of the largest platforms (Binance, Coinbase, Bitstamp) over the last 10 years.

4. If the leverage is used, the cost of leveraging and the required margin are ignored. Additionally, if the leverage is lower than 1 (where leverage equal to 1 means no leverage), the cash account is assumed to accrue zero interest;

5. No risk mitigation strategies, such as setting stop-loss orders, are used;

6. The benchmark "buy-and-hold" strategy is created using the same assumptions. The long position is opened in the first period and held throughout the whole period.

The assumptions above provide a foundation for the practical application of this research, simulating the P&L of our strategy in conditions as close to the market ones.

3.4 Ensemble models

We also test the efficacy of a strategy based on a mix of signals from models based on daily, hourly, and 15-minute data in our study. We collect the votes from all three models every 15 minutes, so if the hourly model delivers a "buy" signal, it will create four 15-minute "buy" signals. The majority vote is then used to determine our buy-sell signal. As a result, we should see fewer changes in positions compared to a strategy based only on 15-minute signals. An illustration of the ensemble signal combination is presented below, in Figure 5.

Figure 5: A demonstration of an ensemble model for different frequencies.

	signal			
	15-minutes	hourly	daily	**ensemble**
18.03.2015 14:45	buy	buy	buy	**buy**
18.03.2015 15:00	sell			**buy**
18.03.2015 15:15	sell	sell		**sell**
18.03.2015 15:30	sell			**sell**
18.03.2015 15:45	buy			**buy**
19.03.2015 16:00	sell			**sell**
19.03.2015 09:45	buy	sell	sell	**sell**

Additionally, we employ ensembling for the combination of assets as well as frequencies. We put together a portfolio with the following weights: 50% for S&P 500 Index and 50% for Bitcoin. The account is opened with $1,000, financial instruments are totally divisible, and transaction charges are 0.005% for S&P 500 Index and 0.1% for Bitcoin. This is done to allow comparability with single strategies. The portfolio is rebalanced every three months, which means the asset allocation proportions are brought back to the starting points. During the rebalancing operations, the transaction fees are paid in accordance with the amount that has to be bought or sold.

3.5 Performance metrics

For each strategy and asset, a number of indicators are computed in order to evaluate profitability and performance. When evaluating portfolio performance, it is critical to

consider not just the return but also the risk of the strategy. In the study we utilize performance metrics from Michańków et al. (2022) and Ryś and Ślepaczuk (2018).

3.5.1 Annualized Return Compounded (ARC)

The Annualized Return Compounded (ARC), is the constant rate of annual return over the whole period of investment, so that:

$$V(t_n) = V(t_0) * (1 + ARC)^n \tag{7}$$

where:
$V(t_0)$ - the initial value of the investment
$V(t_n)$ - the value of the investment at the end of the period
$t_n - t_0$ - number of years
ARC may be simply derived using the equation above.

$$ARC(t_0, t_n) = (\frac{V(t_n)}{V(t_0)})^{\frac{1}{t_n - t_0}} - 1 \tag{8}$$

3.5.2 Annualized Standard Deviation (ASD)

Volatility is a statistical indicator of the variation of returns. Most of the time, a security is riskier the more volatile it is. Volatility may be expressed as either the standard deviation or variation between returns from the same securities or market index. Volatility might be easily switched to annualized values by multiplying the standard deviation of the returns by the square root of the number of observations in a year (e.g. 252 for daily data of the S&P 500 Index and 365 for daily data of Bitcoin prices). In our research we use Annualized Standard Deviation (ASD) as a measure of volatility:

$$ASD = \sqrt{\frac{1}{N-1}\sum_{t=1}^{N}(R_t - \bar{R})^2} * \sqrt{n_{year}} \tag{9}$$

where:
$\bar{R}$ - the average simple return (e.g. daily for daily data) of the given instrument
R_t - the simple return during period t
n_{year} - number of observations in a year

3.5.3 Information Ratio*

Sharpe ratio, created by Nobel Prize winner William F. Sharpe, aids investors in determining the return on investment relative to its risk. The ratio is the average return over the risk-free rate for each unit of volatility or overall risk. Because we assume a zero-rate risk-free rate, instead of Sharpe Ration we will define IR*:

$$IR^* = \frac{ARC}{ASD} \tag{10}$$

3.5.4 Max Drawdown (MDD)

A portfolio's maximum drawdown (MDD) is the largest loss that could be recorded between a portfolio's peak and bottom before a new high is reached. Maximum drawdown

serves as a gauge for the potential loss over a certain time frame. Maximum drawdown (MDD), a major concern for most investors, is a tool used to compare the relative riskiness of different investment strategies.

$$MDD(T) = max_{\tau \in [0,T]}(max_{t \in (0,\tau])}V_t - V_\tau), \tag{11}$$

3.5.5 Maximum Loss Duration (MLD)

Maximum Loss Duration (MLD) is the worst (the greatest/longest) period of time between peaks that the investment has experienced. It is expressed in a number of years:

$$MLD = \max \frac{m_j - m_i}{S} \tag{12}$$

for which $Val(m_j) > Val(m_i)$ and $j > i$. $Val(m_j)$ and $Val(m_i)$ are the values of the local maximums in days m_j and m_i respectively. m_j and m_i are the numbers of days indicating local maximums of the equity line. The scale parameter S denotes the number of trading sessions in a year.

3.5.6 Information Ratio**

Kość et al. (2018) in their study use an additional measure to assess the effectiveness of the strategy, which is a modification of the Information Ratio measure. This measure also takes into account the sign of the portfolio's rate of return and the maximum drawdown:

$$IR^{**} = \frac{ARC^2 * sign(ARC)}{ASD * MDD} \tag{13}$$

3.5.7 Position changes

Position changes, expressed as a percentage, indicates the relative number of position changes to all trading days. For instance, if the position changes for daily data are equal to 20%, then our trading strategy includes a position change every fifth trading day on average.

3.6 Research description

3.6.1 Description of approaches used in the study

The difficulty to optimize the model's hyperparameters due to its high computational complexity is one of the foremost challenges with neural networks. Our first approach is to choose a set of hyperparameters using heuristic techniques and existing research, which allows us to refit the model more than once and execute training on a rolling window. The exact values of hyperparameters used are based on the research of Kijewski and Ślepaczuk (2020) and presented in Table 2.

Table 2: The model frameworks used in the algorithmic investing strategy

	Approach #1	Approach #2	Approach #3	Approach #4
Input	Price	Price	Price	Price
Output	Price	Price	Price direction	Price direction
Units	30	{20,40,60,80,100}	30	n/a**

Table 2: The model frameworks used in the algorithmic investing strategy *(continued)*

	Approach #1	Approach #2	Approach #3	Approach #4
Activation	tahn*	tahn	tahn	tahn
Loss method	mse	mse	log-loss	log-loss
Epochs	50*	50	50	n/a**
Learning rate	0.01	{0.1, 0.01, 0.001}	0.01	n/a**
Dropout rate	0.2	{0.1,0.2,0.3,0.4,0.5}	0.2	n/a**
# hidden layers	1	{1,2,3}	1	n/a**
Description	Regression problem	Regression problem	Classification problem	Classification problem
Source	Kijewski and Ślepaczuk (2020)	Approach #1 with hyperparameter tuning	Approach #1 but classification problem	Voting of pre-defined 10 LSTM models

Note: The hyperparameters are shown in curly brackets if they are calibrated during the research phase (hyperparameter tuning). The Adam optimizer is used to train each model. The sequence length in each approach is 15 and the batch size (the number of samples per gradient update) is 32.
* Number of epochs and the activation function were modified to reduce computing time
** The hyperparameters for Approach 4 are different for each of the LSTM models employed.

The hyperparameters are optimized in Approach #2 using the walk-forward method. Because this is a computationally intensive problem, we utilize RandomSearch rather than GridSearch on each window, with 15 trials to identify the best hyperparameters. The criterion metric for determining the best set of hyperparameters for regression issues is MSE. It is worth noting that the total out-of-sample duration for this approach is reduced because it needs a validation period to fine-tune the hyperparameters.

Approach #3 employs the same LSTM model hyperparameters as Approach #1, but this time on a classification problem. Instead of forecasting prices, we predict price direction. This entails modifying the loss function to one that is suitable for binary classification - in our instance, the log-loss function.

Finally, Approach #4 employs an ensemble (voting) model-inspired strategy. We incorporate the predictions of ten models published in the literature and utilized by Baranochnikov and Ślepaczuk (2022) to produce a final voting model. We do not tune hyperparameters in this procedure since fitting ten models to each frequency and asset is highly computationally intensive, especially for 15-minute Bitcoin data. Unlike in other approaches, we generate hold signals as well as buy/sell indications. Table 3 lists all of the model architectures employed in the voting model.

It is worth mentioning that we employ all four approaches at each frequency for both S&P 500 Index and Bitcoin. For each approach, we also derive ensemble models. Eventually, for each approach, we produce 12 equity lines (4 types of frequency - daily, hourly, 15-minute, and ensemble for 3 types of portfolios - pure Bitcoin, pure S&P 500 Index, and ensemble BTC+SPX).

Table 3: Model architectures used in the voting (Approach #4) model.

	Model #1	Model #2	Model #3	Model #4	Model #5
# neurons 1st layer	64	30	4	25	32
# neurons 2nd layer	128	-	10	-	-
Dropout rate	0.3	0.2	0.1	0.1	0.2
Epochs	100	100	100	30	100
Learning rate	0.001	0.01	1e-04	0.001	0.001
Source	Sethia and Raut (2019)	Kijewski and Ślepaczuk (2020)	Lim and Lundgren (2019)	Ghosh et al.(2021)	Zou and Qu (2020)

	Model #6	Model #7	Model #8	Model #9	Model #10
# neurons 1st layer	12	128	32	120	50
# neurons 2nd layer	12	64	16	-	-
Dropout rate	0.2	0	0.2	0.2	0.25
Epochs	100	100	100	100	100
Learning rate	0.003	0.001	0.002	0.001	0.001
Source	Du et al. (2019)	Roondiwala et al. (2015)	Site et al. (2019)	Shahi et al. (2020)	Girsang et al. (2020)

Note: The Adam optimizer is used to train each model. The sequence length in each approach is 15 and the batch size (the number of samples per gradient update) is 32. '-' by the number of neurons in the 2nd layer means that the model does not include a 2nd hidden layer.

3.6.2 Loss functions

The loss function we deploy for regression problems (price prediction) is the Mean Squared Error (MSE). The mean squared error is determined by averaging the squared discrepancies between the expected and actual figures. Regardless of the sign of the anticipated and actual numbers, the outcome is always positive, and a perfect value is 0.0. Because of the squaring, bigger mistakes cost the model more in terms of error than smaller ones, which means that bigger mistakes are penalized more.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{14}$$

where:

Y_i - observed values of the variable being predicted

$\hat{Y}_i$ - the predicted values

However, MSE might be inappropriate for algorithmic investment strategy development because it penalizes both good and bad decisions. To address this issue, we switch from regression to classification problem (price direction prediction) and use one of the loss functions appropriate for such tasks, namely log-loss, also known as binary cross-entropy loss:

$$LogLoss = -(y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)) \tag{15}$$

where:

y_i - the real (binary) value, where 1 represents the “buy” signal and 0 the “sell” signal,

$\hat{y}_i$ - the predicted probability of a “buy” signal"

We can also present this equation in a conditional form:

$$\begin{cases} LogLoss = -\log(1-\hat{y}_i) & \text{if } y_i = 0 \\ LogLoss = -\log(\hat{y}_i) & \text{if } y_i = 1 \end{cases} \tag{16}$$

The logarithmic function $y = \log(x)$, which is only defined for positive x, rises from $-\infty$ to 0 as x increases to 1. Thanks to this property, entropy loss severely penalizes predictions that are confident yet inaccurate.

3.6.3 Sensitivity analysis

Sensitivity analysis is a method used to examine how the test results change when specific parameters are altered. This is one technique to assess the robustness of a strategy. A sensitivity analysis, sometimes referred to as a what-if analysis or a what-if simulation exercise, is used to forecast how various independent variable values impact a certain dependent variable in a given set of circumstances.

The following variables will be altered throughout the sensitivity analysis:

1. Training period’s duration,
2. Testing period’s duration,
3. Type of optimizer,
4. Type of loss function,
5. Type of input variable normalization,
6. Length of the sequence,
7. Number of epochs,
8. Transaction costs.

4 Results

With the use of daily, hourly, and 15-minute data frequencies, we test our investment algorithm. The S&P 500 Index out-of-sample data spans from 2014-02-06 to 2020-11-27 and the Bitcoin out-of-sample data spans from 15.01.2014 to 2020-12-01. The Adjusted Information Ratio (IR**), which is determined by formula 13, will serve as our primary performance criterion. By including the return volatility and the maximum drawdown, this metric incorporates not only the strategy’s profitability but also its riskiness. A buy-and-hold scenario will serve as our benchmark for evaluating how well our approach performed.

The lengths of the walk-forward process unit periods vary with data frequency. In all cases, the training and testing durations are the same. The daily data in-sample size is one year, which equals 252 observations for S&P 500 Index and 365 for Bitcoin. The training period for hourly data is one month, with 126 observations for S&P 500 Index

and 720 for Bitcoin. Finally, for 15-minute data the period equals 260 observations for S&P 500 Index and 1344 for Bitcoin throughout a two-week period.

4.1 Approach #1 - Regression problem

In the first approach, we use an LSTM model with a set of hyperparameters that have already been tested in the literature. We use the instrument's price as an output. When the price forecast for the next time step is greater than the present price, a buy signal is created. The outcomes of our trading algorithm for all instruments are displayed in Table 4. The Adjusted Information Ratio is included in the table along with all other relevant performance measures.

Moreover, Figure 6 displays how the results of strategies have been developing throughout time in the form of an equity line. For S&P 500 Index only the daily strategy does not outperform the benchmark in terms of IR**, with the ensemble model performing the best. However, the model does not provide profitable buy/sell signals for Bitcoin. All of the strategies lag behind buy-and-hold investing. This lowers the performance of the SPX+BTC portfolio, which does not beat the benchmark.

Table 4: Performance metrics for strategies based on signals generated by Approach 1

	Cumulative Return	ARC%	ASD	IR*	MDD	MLD	IR**	Position changes
S&P 500 Index								
daily	61.27%	7.27%	17.97%	0.4	-26.24%	3.21	0.11	26.41%
hourly	114.48%	11.86%	16.08%	0.74	-19.07%	1.69	0.46	23.14%
15-minute	94.59%	10.27%	16.81%	0.61	-22.44%	3.59	0.28	19.99%
ensemble	**223.16%**	**18.79%**	**16.81%**	**1.12**	**-16.3%**	**2.0**	**1.29**	**11.51%**
buy-and-hold	114.57%	11.86%	17.96%	0.66	-33.97%	1.65	0.23	-
Bitcoin								
daily	64.06%	7.51%	75.31%	0.1	-92.75%	3.06	0.01	13.3%
hourly	-99.67%	-56.69%	80.33%	-0.71	-99.82%	9.84	-0.4	18.02%
15-minute	-100.00%	-100.0%	86.46%	-1.16	-100.0%	6.8	-1.16	18.83%
ensemble	-100.00%	-97.98%	86.1%	-1.14	-100.0%	6.75	-1.12	10.74%
buy-and-hold	**2,340.49%**	**59.58%**	**75.27%**	**0.79**	**-83.43%**	**2.96**	**0.57**	**-**
SPX+BTC								
daily	176.93%	16.13%	37.95%	0.43	-68.42%	3.05	0.1	-
hourly	-77.96%	-19.91%	43.39%	-0.46	-91.59%	6.79	-0.1	-
15-minute	-100.00%	-86.4%	32.11%	-2.69	-100.0%	6.79	-2.32	-
ensemble	-99.98%	-72.25%	36.71%	-1.97	-99.99%	6.74	-1.42	-
buy-and-hold	**1,017.18%**	**42.52%**	**41.32%**	**1.03**	**-61.42%**	**2.86**	**0.71**	**-**

Note: Trading S&P 500 Index starts on 2014-02-06 and ends on 2020-11-27, while trading Bitcoin starts on 2014-02-01 and ends on 2020-12-01. The period for SPX+BTC portfolio was limited to S&P 500 Index trading period. Transaction costs are 0.005% for S&P 500 Index and 0.1% for Bitcoin. Ensemble model means combining daily, hourly and 15-minutes signals into one strategy. SPX+BTC assumes quarterly rebalancing, with weights for both assets equal to 50%. Both training and testing periods contain a year of observations for daily frequency, one month for hourly and two weeks for 15-minute. For each portfolio (S&P 500 Index, Bitcoin, or SPX+BTC), the strategy with the greatest Adjusted Information Ratio is shown in bold.

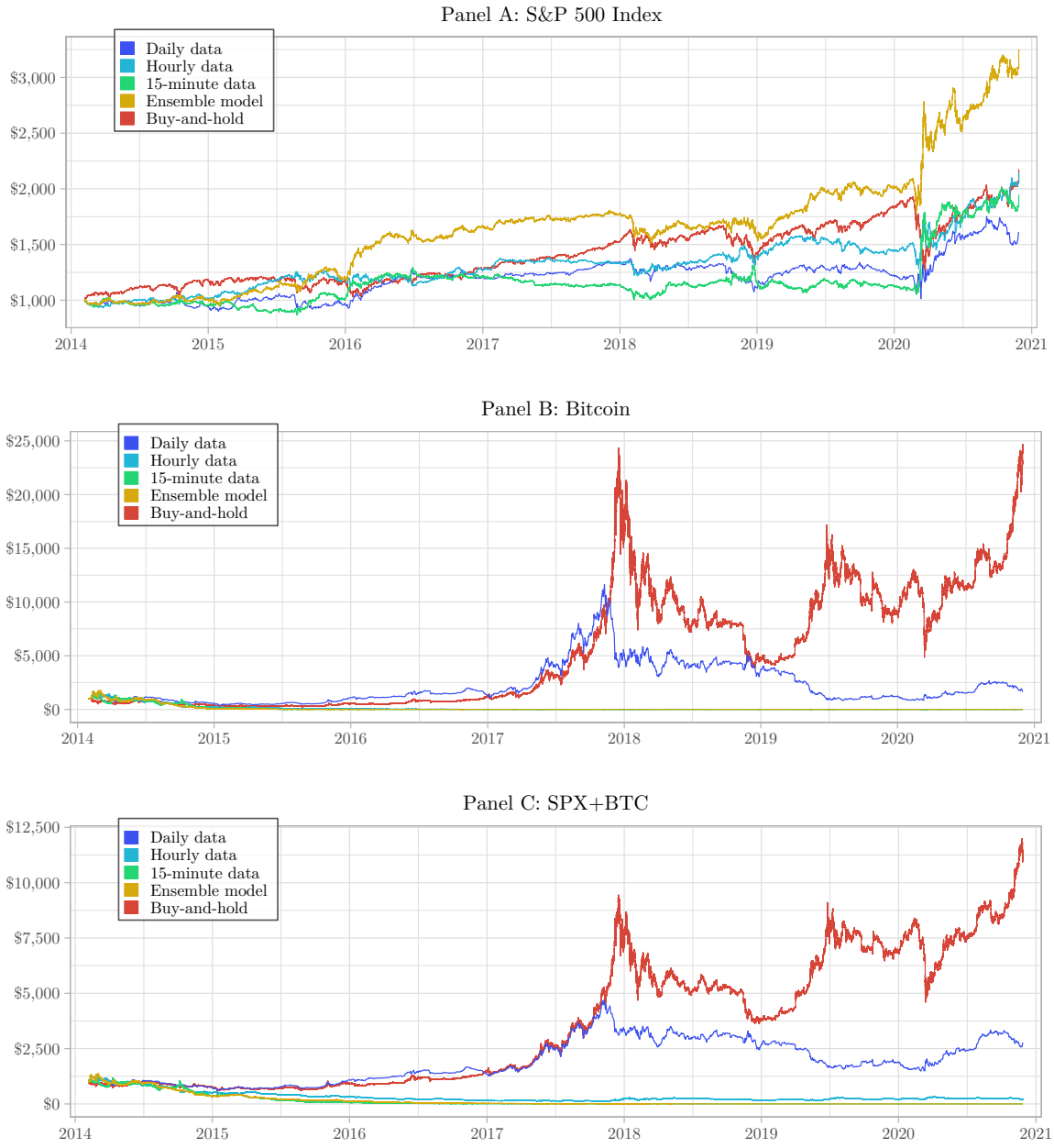
The variations in relative position changes between the strategies based on different frequencies are minor. However, if we convert position changes from percentages to nominal values (how many transactions have been made in total, regardless of the frequency), it appears that the daily strategy makes the fewest transactions and the 15-minute approach makes the most. The ensemble frequency method creates fewer position changes than the straight 15-minute data strategy, as expected.

Figure 6 shows that our approach demonstrates a sharp increase in value for S&P 500 Index when the market was moving downhill, notably in March 2020, when the covid crisis

rattled the markets. It appears to be investing in the instability as a contrarian. However, Bitcoin models for all frequencies except daily data see a value decline from 2014 to 2015 and are unable to recover.

In general, we can observe that for Approach #1, intra-day data strategies outperform inter-day data methods for S&P 500 Index but do substantially worse for Bitcoin. Poor performance of intra-day strategies for Bitcoin brings down the performance of the intra-day SPX+BTC portfolio, resulting in the daily model outperforming the inter-day and ensemble models.

Figure 6: S&P Equity line for strategies generated by Approach #1.



Note: The charts show the strategies' net value in USD over time. Trading S&P 500 Index starts on 2014-02-06 and ends on 2020-11-27, while trading Bitcoin starts on 2014-02-01 and ends on 2020-12-01. For SPX+BTC portfolio trading period was restricted to S&P 500 Index trading period. Transactional costs are 0.005% for S&P 500 Index and 0.1% for Bitcoin. Ensemble model means combining daily, hourly and 15-minute signals into one strategy. SPX+BTC assumes quarterly rebalancing, with weights for both assets equal to 50%. Both training and testing periods contain a year of observations for daily frequency, one month for hourly, and two weeks for 15-minute.

4.2 Approach #2 - Hyperparameter tuning

In this technique, we employ an LSTM model structure comparable to Approach #1, but we tune the hyperparameters. We adjust the number of hidden layers, the number of neurons in each layer, the dropout rate, and the learning rate. The trials are conducted throughout a validation period that has the same duration as the training and testing periods.

Table 5 displays the performance metrics for the strategies developed by Approach #2. We can observe that none of the strategies outperformed the benchmark in terms of ARC%, while the daily strategy for SPX+BTC portfolio outperformed in terms of IR**, owing to (i) a lower drawdown, and (ii) limiting the trading time of SPX+BTC portfolio to match the S&P 500 Index. In days 2015-01-15 to 2015-01-26 the buy-and-hold strategy gained over 30%, while our algorithm based on daily signals lost 15%. Then, in days 2020-11-27 to 2020-12-01, the buy-and-hold gained 16%, while our strategy lost 14%. All in all, if we had restricted Bitcoin trading period to the S&P 500 Index trading period, the IR** for our daily strategy would equal 1.88 and it would beat the benchmark, which would have IR** at 1.72.

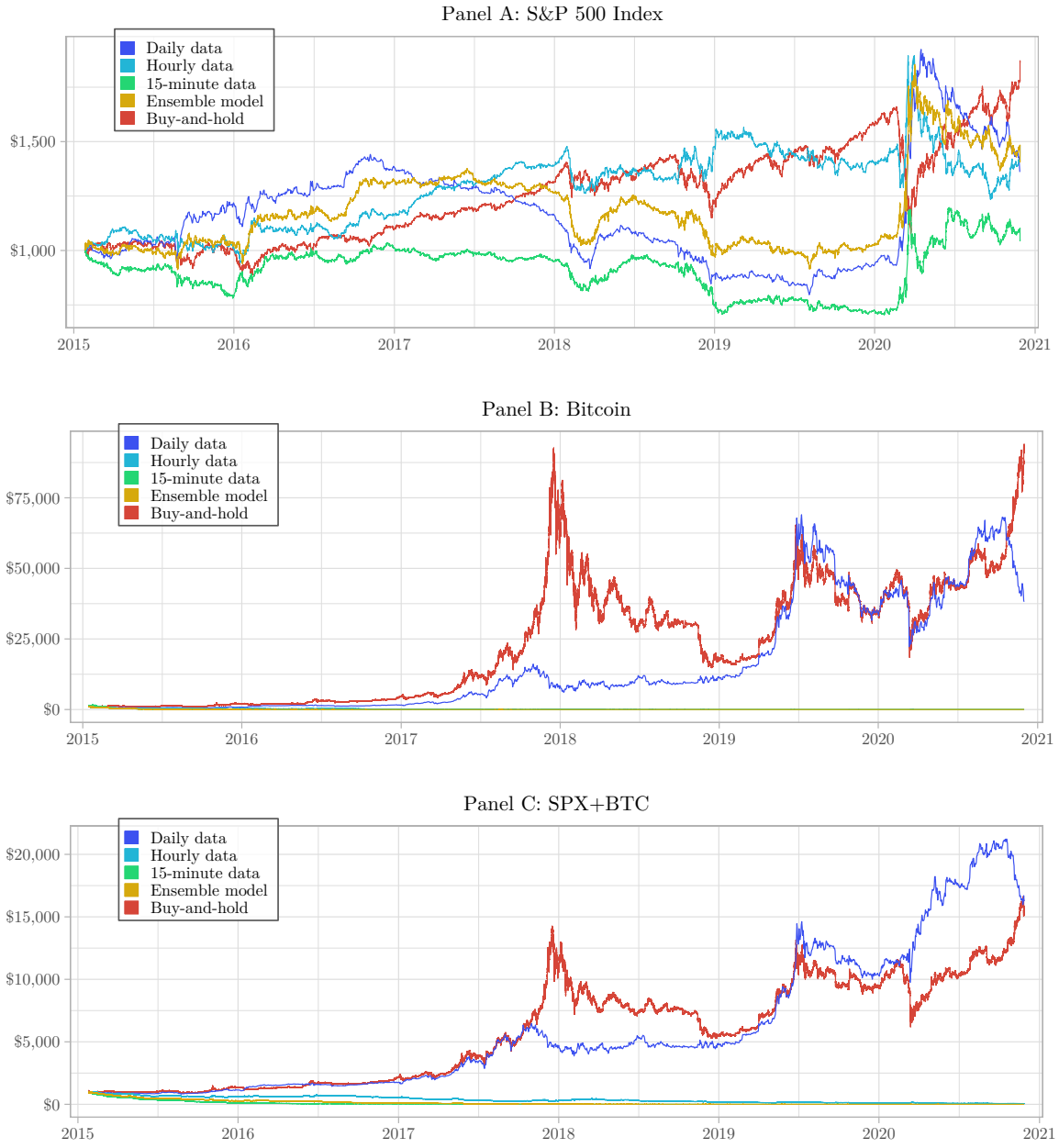
Table 5: Performance metrics for strategies based on signals generated by Approach #2

	Cumulative Return	ARC%	ASD	IR*	MDD	MLD	IR**	Position changes
S&P 500 Index								
daily	36.06%	5.41%	18.85%	0.29	-44.75%	4.86	0.03	22.83%
hourly	48.12%	6.96%	16.84%	0.41	-34.85%	1.46	0.08	22.99%
15-minute	4.92%	0.83%	17.61%	0.05	-31.9%	4.71	0.0	20.12%
ensemble	41.27%	6.09%	17.61%	0.35	-33.56%	3.98	0.06	12.52%
buy-and-hold	**84.97%**	**11.1%**	**18.84%**	**0.59**	**-33.97%**	**1.65**	**0.19**	**-**
Bitcoin								
daily	3,718.34%	85.75%	73.72%	1.16	-67.75%	1.41	1.47	16.29%
hourly	-99.95%	-72.23%	78.98%	-0.91	-99.95%	8.52	-0.66	16.14%
15-minute	-100.00%	-100.0%	82.94%	-1.21	-100.0%	5.85	-1.21	16.04%
ensemble	-100.00%	-96.74%	82.32%	-1.18	-100.0%	5.88	-1.14	8.68%
buy-and-hold	**9,306.96%**	**116.52%**	**73.71%**	**1.58**	**-83.43%**	**2.96**	**2.21**	**-**
SPX+BTC								
daily	**1,525.06%**	**61.18%**	**40.88%**	**1.5**	**-40.29%**	**1.43**	**2.27**	**-**
hourly	-93.97%	-38.18%	40.41%	-0.94	-94.56%	5.84	-0.38	-
15-minute	-100.00%	-88.91%	30.29%	-2.94	-100.0%	5.83	-2.61	-
ensemble	-99.93%	-71.06%	36.61%	-1.94	-99.93%	5.84	-1.38	-
buy-and-hold	1,420.44%	59.35%	42.8%	1.39	-62.05%	2.89	1.33	-

Note: Trading S&P 500 Index starts on 2015-01-26 and ends on 2020-11-27, while trading Bitcoin starts on 2015-01-15 and ends on 2020-12-01. The period for SPX+BTC portfolio was limited to S&P 500 Index trading period. Transaction costs are 0.005% for S&P 500 Index and 0.1% for Bitcoin. Ensemble model means combining daily, hourly and 15-minutes signals into one strategy. SPX+BTC assumes quarterly rebalancing, with weights for both assets equal to 50%. Both training and testing periods contain a year of observations for daily frequency, one month for hourly and two weeks for 15-minute. For each portfolio (S&P 500 Index, Bitcoin, or SPX+BTC), the strategy with the greatest Adjusted Information Ratio is shown in bold.

For S&P 500 Index, the hourly model beats the inter-day model, however for Bitcoin, none of the intra-day models perform better than the daily model. Because of the poor performance of intraday Bitcoin strategies, the intraday SPX+BTC portfolio performs very poorly.

Figure 7: S&P Equity line for strategies generated by Approach #2.



Note: The charts show the strategies' net value in USD over time. Trading S&P 500 Index starts on 2015-01-26 and ends on 2020-11-27, while trading Bitcoin starts on 2015-01-15 and ends on 2020-12-01. For SPX+BTC portfolio trading period was restricted to S&P 500 Index trading period. Transactional costs are 0.005% for S&P 500 Index and 0.1% for Bitcoin. Ensemble model means combining daily, hourly and 15-minute signals into one strategy. SPX+BTC assumes quarterly rebalancing, with weights for both assets equal to 50%. Both training and testing periods contain a year of observations for daily frequency, one month for hourly, and two weeks for 15-minute.

Similar conclusions may be drawn from Figure 7's equity lines. Only the daily strategy for the SPX+BTC portfolio matches buy-and-hold investing. Bitcoin models perform very poorly.

4.3 Approach #3 - Binary (buy-sell) prediction

Approach #3 entails utilizing the same model with the same hyperparameters as in Approach #1, but this time doing a classification forecast - predicting the direction of

the price rather than the price itself. The model structure differs only in the output layer activation function - sigmoid activation is required for classification problems. In this manner, the model's outputs will be a probability of class membership - that is, the likelihood of the signal belonging to the "buy" or "sell" class. Then we set the classification threshold to 0.5, which means that if the output probability of being a "buy" class is more than 50%, we take it as a "buy" signal.

Table 6: Performance metrics for strategies based on signals generated by Approach #3

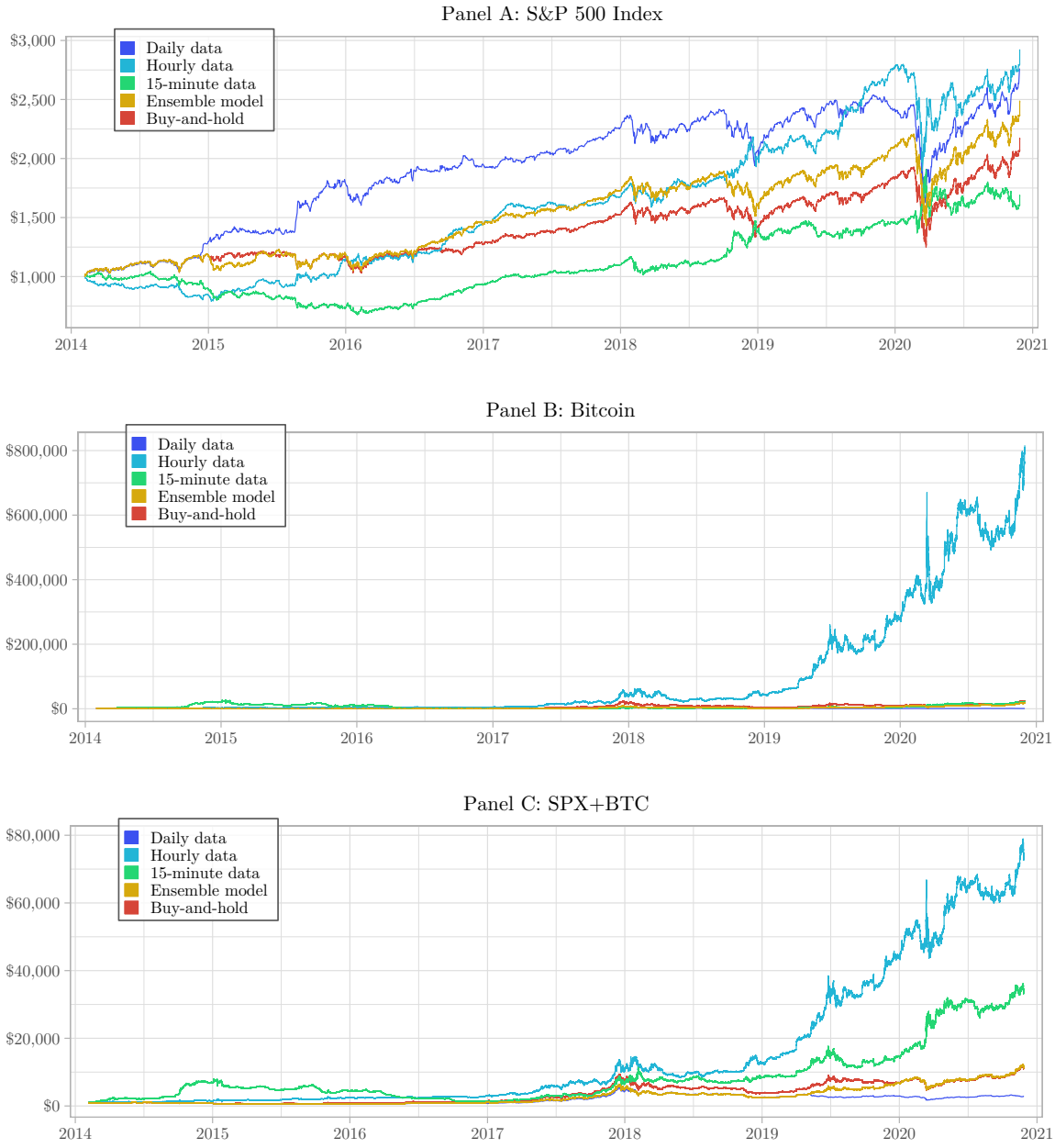
	Cumulative Return	ARC%	ASD	IR*	MDD	MLD	IR**	Position changes
S&P 500 Index								
daily	176.26%	16.09%	17.95%	0.9	-36.05%	1.18	0.4	1.28%
hourly	**192.43%**	**17.06%**	**16.08%**	**1.06**	**-32.25%**	**2.23**	**0.56**	**3.35%**
15-minute	68.06%	7.92%	16.81%	0.47	-35.25%	4.26	0.11	1.1%
ensemble	147.5%	14.23%	16.81%	0.85	-35.23%	1.45	0.34	0.19%
buy-and-hold	114.57%	11.86%	17.96%	0.66	-33.97%	1.65	0.23	-
Bitcoin								
daily	-27.86%	-4.67%	75.37%	-0.06	-91.44%	3.53	-0.0	0.6%
hourly	**81,526.51%**	**166.67%**	**80.05%**	**2.08**	**-65.71%**	**1.88**	**5.28**	**1.0%**
15-minute	2,161.42%	57.81%	85.61%	0.68	-99.54%	5.89	0.39	3.33%
ensemble	1,836.27%	54.27%	85.48%	0.63	-86.9%	3.3	0.4	0.39%
buy-and-hold	2,340.49%	59.58%	75.27%	0.79	-83.43%	2.96	0.57	-
SPX+BTC								
daily	193.48%	17.13%	39.99%	0.43	-69.13%	2.95	0.11	-
hourly	**7,402.18%**	**88.5%**	**46.7%**	**1.9**	**-42.52%**	**0.85**	**3.95**	**-**
15-minute	3,336.28%	68.09%	49.28%	1.38	-84.51%	2.94	1.11	-
ensemble	1,060.74%	43.33%	45.43%	0.95	-63.11%	2.59	0.65	-
buy-and-hold	1,017.18%	42.52%	41.32%	1.03	-61.42%	2.86	0.71	-

Note: Trading S&P 500 Index starts on 2014-02-06 and ends on 2020-11-27, while trading Bitcoin starts on 2014-02-01 and ends on 2020-12-01. The period for SPX+BTC portfolio was limited to S&P 500 Index trading period. Transaction costs are 0.005% for S&P 500 Index and 0.1% for Bitcoin. Ensemble model means combining daily, hourly and 15-minutes signals into one strategy. SPX+BTC assumes quarterly rebalancing, with weights for both assets equal to 50%. Both training and testing periods contain a year of observations for daily frequency, one month for hourly and two weeks for 15-minute. For each portfolio (S&P 500 Index, Bitcoin, or SPX+BTC), the strategy with the greatest Adjusted Information Ratio is shown in bold.

The results of our trading system for all instruments are shown in Table 6. The method based on hourly frequency clearly beats other strategies, particularly for Bitcoin, getting the greatest IR**. Except for the 15-minute model, all S&P 500 Index models surpass the benchmark, however only the hourly model outperforms the benchmark for Bitcoin. Both the hourly and 15-minute models outperform the buy-and-hold investment for the SPX+BTC portfolio.

As position changes are far less frequent, the classification model appears to provide more solid predictions than the regression model. As noted previously, the position changes for ensemble frequency are the smallest.

Figure 8: S&P Equity line for strategies generated by Approach #3.



Note: The charts show the strategies' net value in USD over time. Trading S&P 500 Index starts on 2014-02-06 and ends on 2020-11-27, while trading Bitcoin starts on 2014-02-01 and ends on 2020-12-01. For SPX+BTC portfolio trading period was restricted to S&P 500 Index trading period. Transactional costs are 0.005% for S&P 500 Index and 0.1% for Bitcoin. Ensemble model means combining daily, hourly and 15-minute signals into one strategy. SPX+BTC assumes quarterly rebalancing, with weights for both assets equal to 50%. Both training and testing periods contain a year of observations for daily frequency, one month for hourly, and two weeks for 15-minute.

Equity lines are shown in Figure 8 for each strategy produced by Approach #3. We can observe that for S&P 500 Index, the equity lines for all frequencies have a fairly similar form, with the majority of them (except for S&P 500 Index 15-minute) seeing a large dip followed by a swift rebound after Covid-19 jolted the markets. However, for Bitcoin, the approach based on hourly signals begins to decisively outperform other methods in late 2017, with the gap only widening until 2021. The 15-minute approach beats other strategies in late 2014, when volatility was high, but it quickly loses all of its lead.

In Approach #3, there is no indication that intra-day data models outperform inter-day

strategies in the case of S&P 500 Index. However, for Bitcoin, intra-day and ensemble algorithms outperform the daily strategy. As a result, the portfolio SPX+BTC performs the best when based on intraday, particularly hourly indications.

4.4 Approach #4 - Voting model

The ensemble model-inspired voting approach #4 is used. To create a final voting model, we combine the predictions from 10 models that have been published in the literature. Unlike other approaches, we produce buy/sell signals as well as hold signals. This situation happens when 5 of the models vote for the "buy" class and the other 5 for the "sell" class. Hold signals indicate that the position should be kept open without trading.

Table 7 displays the outcomes of our trading strategies for all instruments. When we look at IR**, we can see that for S&P 500 Index, the daily, hourly, and ensemble techniques outperformed the buy-and-hold investment, with the hourly strategy achieving the best results. However, in case of Bitcoin only the 15-minute technique outperforms the benchmark, and it also performs the best for the SPX+BTC portfolio. Hourly and ensemble methods for the SPX+BTC portfolio, on the other hand, provide outcomes that are comparable to the benchmark.

Table 7: Performance metrics for strategies based on signals generated by Approach #4

	Cumulative Return	ARC%	ASD	IR*	MDD	MLD	IR**	Position changes
S&P 500 Index								
daily	115.81%	11.96%	17.96%	0.67	-28.55%	0.85	0.28	1.22%
hourly	**147.84%**	**14.25%**	**16.08%**	**0.89**	**-30.22%**	**1.25**	**0.42**	**1.92%**
15-minute	-20.77%	-3.36%	16.81%	-0.2	-37.63%	5.35	-0.02	0.97%
ensemble	128.86%	12.93%	16.81%	0.77	-34.15%	0.87	0.29	0.11%
buy-and-hold	114.57%	11.86%	17.96%	0.66	-33.97%	1.65	0.23	-
Bitcoin								
daily	3.47%	0.5%	75.36%	0.01	-91.44%	3.58	0.0	0.44%
hourly	1,111.38%	44.04%	80.07%	0.55	-96.09%	2.49	0.25	0.58%
15-minute	**3,489.22%**	**68.84%**	**85.5%**	**0.81**	**-97.7%**	**3.16**	**0.57**	**1.0%**
ensemble	1,468.35%	49.58%	85.48%	0.58	-87.44%	3.34	0.33	0.15%
buy-and-hold	2,340.49%	59.58%	75.27%	0.79	-83.43%	2.96	0.57	-
SPX+BTC								
daily	206.79%	17.89%	39.56%	0.45	-60.04%	2.96	0.13	-
hourly	1,129.09%	44.54%	42.19%	1.06	-67.11%	2.45	0.7	-
15-minute	**1,927.95%**	**55.56%**	**48.26%**	**1.15**	**-80.95%**	**3.03**	**0.79**	**-**
ensemble	952.57%	41.28%	45.57%	0.91	-52.5%	2.73	0.71	-
buy-and-hold	1,017.18%	42.52%	41.32%	1.03	-61.42%	2.86	0.71	-

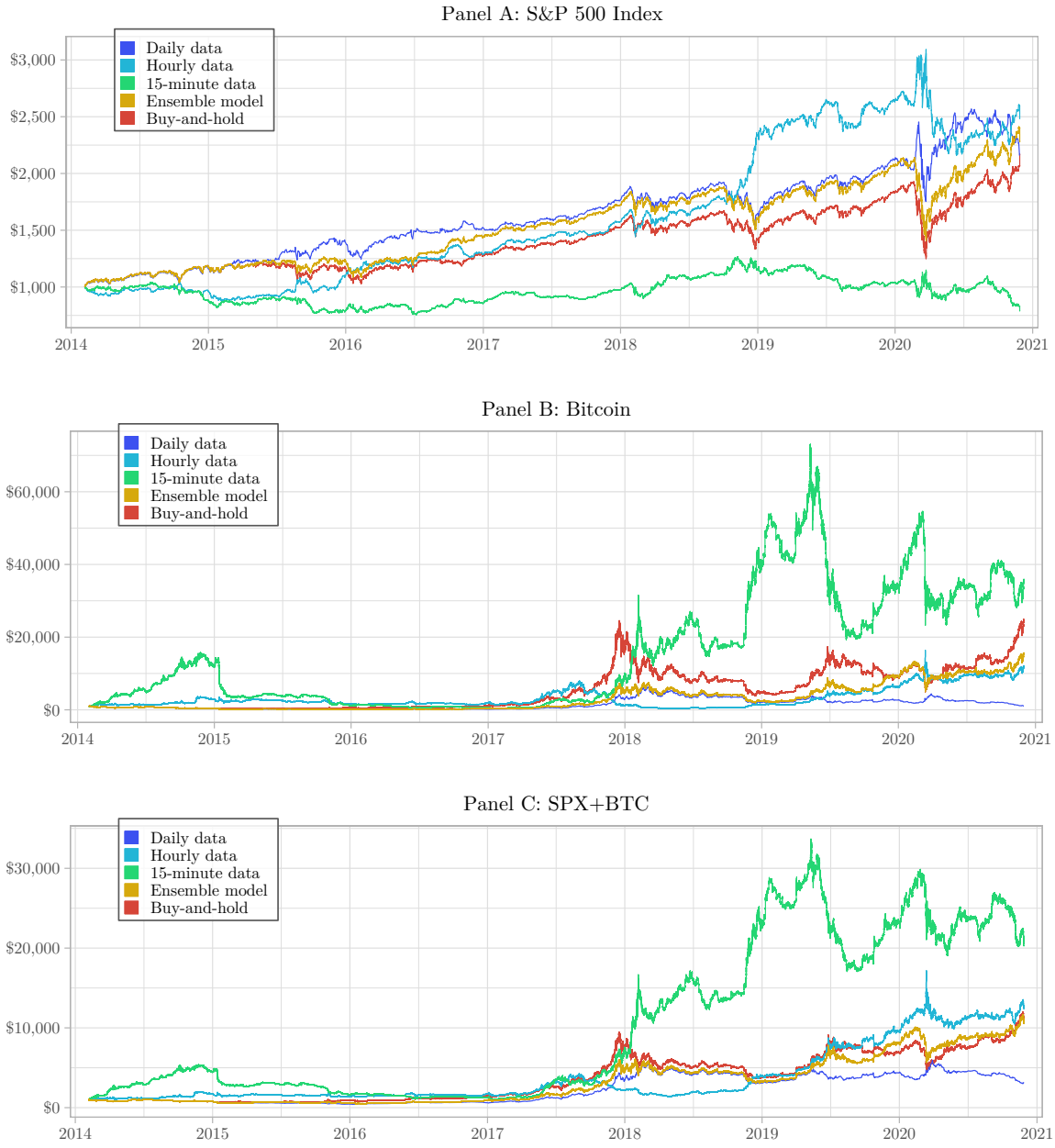
Note: Trading S&P 500 Index starts on 2014-02-06 and ends on 2020-11-27, while trading Bitcoin starts on 2014-02-01 and ends on 2020-12-01. The period for SPX+BTC portfolio was limited to S&P 500 Index trading period. Transaction costs are 0.005% for S&P 500 Index and 0.1% for Bitcoin. Ensemble model means combining daily, hourly and 15-minutes signals into one strategy. SPX+BTC assumes quarterly rebalancing, with weights for both assets equal to 50%. Both training and testing periods contain a year of observations for daily frequency, one month for hourly and two weeks for 15-minute. For each portfolio (S&P 500 Index, Bitcoin, or SPX+BTC), the strategy with the greatest Adjusted Information Ratio is shown in bold.

In addition, we can see that the strategies in Approach #4 change positions the least frequently when compared to other approaches. Same as in previous cases, the ensemble frequency position changes are the smallest.

Figure 9 displays equity lines for each method generated by Approach #4. The equity lines show that the 15-minute model takes a contrarian approach, earning the most when the price falls. This method works well for Bitcoin but poorly for S&P 500 Index.

There is no evidence in Approach #4 that, for S&P 500 Index, intra-day data models outperform inter-day tactics. On the other hand, for Bitcoin, intra-day and ensemble algorithms perform better than the daily approach. Because of this, the SPX+BTC portfolio performs the best when based on intraday signals while the ensemble model produces performance comparable to the buy-and-hold investment.

Figure 9: S&P Equity line for strategies generated by Approach #4.



Note: The charts show the strategies' net value in USD over time. Trading S&P 500 Index starts on 2014-02-06 and ends on 2020-11-27, while trading Bitcoin starts on 2014-02-01 and ends on 2020-12-01. For SPX+BTC portfolio trading period was restricted to S&P 500 Index trading period. Transactional costs are 0.005% for S&P 500 Index and 0.1% for Bitcoin. Ensemble model means combining daily, hourly and 15-minute signals into one strategy. SPX+BTC assumes quarterly rebalancing, with weights for both assets equal to 50%. Both training and testing periods contain a year of observations for daily frequency, one month for hourly, and two weeks for 15-minute.

4.5 Summary

Tables 8 and 9 summarize the outcomes of our analysis. Table 8 provides IR** metrics for all the LSTM architectures while Table 9 compares IR** statistics across tested approaches and assets. As we can see, there is no single strategy that produces profitable and consistent performance across all assets and frequencies. However, Approach #3 appears to be the best-performing model architecture, as it achieves the highest IR** in half of the cases. We can also observe that in general, classification tasks outperform regression tasks (in 8 out of 12 tested cases), particularly for intra-day data (in 5 out of 6 cases) and ensemble models (in 2 out of 3 cases).

We cannot say whether intra-day models performs better than inter-day models. For example, in Approach #1, intra-day models outperform daily models in case of S&P 500 Index but underperform in case of Bitcoin and SPX+BTC. In Approach #3 and $4, the dependence is less apparent in case of S&P 500 Index, but intra-day models outperform significantly inter-day ones for Bitcoin.

Ensembling frequencies generally improves performance when compared to pure 15-minute strategies (in 9 out of 12 cases), but it does not outperform other frequencies often (outperforms daily in 7 out of 12 cases and outperforms hourly in only 3 out of 12 cases). In general, the ensemble model averages the result of all three input (component) strategies. As a result, it can sometimes outperform all of the component strategies but it can also perform slightly worse than the average of the three components.

Combining assets (SPX+BTC portfolio) does not always improve strategy results. performing particularly poorly when one of the assets had performed incredibly badly. When a model performs fairly well for both assets, ensembling provides higher IR** (see ensemble frequency results for Approach #3 and Approach #4), resulting in significantly better risk-adjusted performance.

We can also see that tuning the hyperparameters (Approach #2) has no positive impact on the base case results (Approach #1). Approach #2 model actually performs worse, beating Approach #1 only in case of daily data for Bitcoin and SPX+BTC. This could be due to the use of an inappropriate objective function (MSE) when tuning the hyperparameters or overfitting.

Table 8: IR** statistics achieved by all the approaches

	Approach 1	Approach 2	Approach 3	Approach 4
SPX				
daily	0.11	0.03	**0.4**	0.28
hourly	0.46	0.08	**0.56**	0.42
15-minute	**0.28**	0	0.11	-0.02
ensemble	**1.29**	0.06	0.34	0.29
BTC				
daily	0.01	**1.47**	0	0
hourly	-0.4	-0.66	**5.28**	0.25
15-minute	-1.16	-1.21	0.39	**0.57**
ensemble	-1.12	-1.14	**0.4**	0.33
SPX+BTC				
daily	0.1	**2.27**	0.11	0.13
hourly	-0.1	-0.38	**3.95**	0.7
15-minute	-2.32	-2.61	**1.11**	0.79
ensemble	-1.42	-1.38	0.65	**0.71**

Note: Table presents IR** statistics for all the approaches, benchmark excluded. Strategy with the greatest IR** (for each frequency and asset) is highlighted.

Table 9: Summary of performance based on IR** for all approaches

	Best result	# strategies beating the benchmark	Best IR**	Average IR**	# intra-day strategies beating inter-day
Approach #1					
S&P500 Index	ensemble	3/4	1.29	0.535	2/2
Bitcoin	buy&hold	0/4	0.01	-0.6675	0/2
SPX+BTC	buy&hold	0/4	0.1	-0.935	0/2
Approach #2					
S&P500 Index	buy&hold	0/4	0.08	0.0425	1/2
Bitcoin	buy&hold	0/4	1.47	-0.385	0/2
SPX+BTC	daily	1/4	2.27	-0.525	0/2
Approach #3					
S&P500 Index	hourly	3/4	0.56	0.3525	1/2
Bitcoin	hourly	1/4	5.28	1.5175	2/2
SPX+BTC	hourly	2/4	3.95	1.455	2/2
Approach #4					
S&P500 Index	hourly	3/4	0.42	0.2425	1/2
Bitcoin	15-minute	0/4	0.57	0.2875	2/2
SPX+BTC	15-minute	1/4	0.79	0.5825	2/2

Note: Best result means a strategy or buy-and-hold investing with the greatest IR**. Columns Best IR** and Average IR** summarize all strategies, benchmark excluded. Intra-day strategies consist of hourly, 15-minute and ensemble models.

5 Sensitivity Analysis

Sensitivity analysis is used to assess how the study's results will alter under different assumptions. We decided to limit the assessment to Approach #3 for two reasons: (i) On average, it achieves the best IR** for both assets, which is worth investigating further, and (ii) it is not as computationally intensive as Approach #2 or Approach #4. We also limited the investigation to only one frequency - hourly data. Again, the model we chose performs the best, therefore it is worth investigating whether the results are robust. Additionally, it isn't as computationally intensive as 15-minute data. Finally, because there is less research on intra-day data, this will be more valuable to science.

The tested assumptions that will be changed are the model's structure or major hyperparameters. First, we modify the in-sample and out-of-sample periods by making them twice as long or twice as short. We also put several optimizers (Adam, Nadam, and RMSprop) and loss functions (log loss, hinge, and squared hinge) to the test. Then we look at whether input variable normalization (MinMax in the range (-1,1) or (0,1) or Robust Scaler normalization) produces the best results. Then, during training, we adjust the sequence length (7, 15, 30) and the number of epochs (25, 50, 100). Finally, we examine how sensitive the outcome is to transaction costs. With the ceteris paribus supposition applied to all other parameters, each parameter is adjusted one at a time. After these tests are performed, the outcomes are compared with the "Buy&Hold" benchmark and the base case scenario.

Because we adjust the size of the input and output data, we also change the out-of-sample period, which will start on 2014-03-10 for S&P 500 Index and 2014-03-03 for Bitcoin across all frequencies and scenarios to guarantee that the results are comparable.

Table 10: Performance metrics for Sensitivity Analysis for S&P500 Index

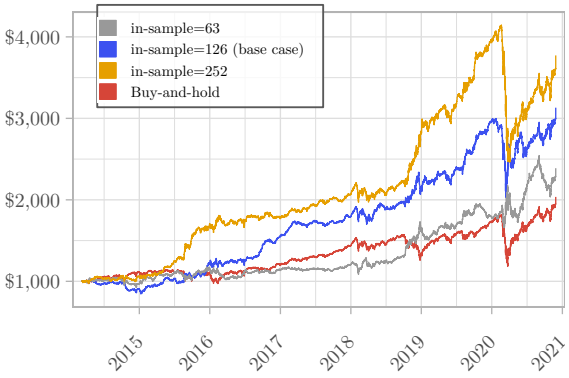
	Cumulative Return	ARC%	ASD	IR*	MDD	MLD	IR**	Position changes
Base case								
Approach #3	213.1%	18.5%	16.14%	1.15	-32.25%	1.48	**0.66**	3.35%
Buy-and-hold	103.6%	11.15%	16.14%	0.69	-34.85%	1.65	0.22	-
In-sample period								
63	138.82%	13.82%	16.14%	0.86	-20.69%	2.84	0.57	3.31%
252	277.25%	21.83%	16.14%	1.35	-40.53%	1.12	**0.73**	2.76%
Out-of-sample period								
63	258.73%	20.92%	16.14%	1.3	-21.43%	1.08	**1.27**	4.33%
252	197.69%	17.62%	16.14%	1.09	-34.85%	1.27	0.55	2.23%
Optimizer								
Nadam	187.3%	17.0%	16.14%	1.05	-35.43%	1.09	0.51	2.61%
RMSprop	269.88%	21.48%	16.14%	1.33	-25.09%	0.95	**1.14**	2.84%
Loss function								
hinge	58.48%	7.09%	16.15%	0.44	-25.44%	2.04	0.12	0.59%
squared hinge	-41.26%	-7.61%	16.15%	-0.47	-49.08%	9.6	-0.07	0.63%
Input variable normalization								
MinMax(-1,1)	181.28%	16.63%	16.14%	1.03	-18.3%	1.65	**0.94**	6.02%
RobustScaler	69.88%	8.2%	16.14%	0.51	-37.65%	1.13	0.11	7.19%
Sequence lengths								
7	369.13%	25.85%	16.13%	1.6	-22.32%	0.49	**1.86**	3.12%
30	188.47%	17.07%	16.14%	1.06	-41.22%	1.25	0.44	2.59%
Number of epochs								
25	235.99%	19.75%	16.14%	1.22	-26.97%	1.48	**0.9**	2.14%
100	110.35%	11.69%	16.14%	0.72	-21.07%	1.11	0.4	4.97%
Transaction costs								
0.01%	221.7%	18.98%	16.14%	1.18	-32.22%	1.48	**0.69**	3.35%
0.025%	173.44%	16.14%	16.14%	1.0	-32.38%	1.51	0.5	3.35%

Note: Trading begins on 2014-03-10 and ends on 2020-11-27. Base case (Approach #3) assumes in-sample period=126, out-of-sample perdiod=126, Optimizer=Adam, Loss function=log-loss, normalization=MinMax(0,1), sequence length=15, epochs=50 and transaction costs 0.005% for S&P 500 Index and 0.1% for Bitcoin.

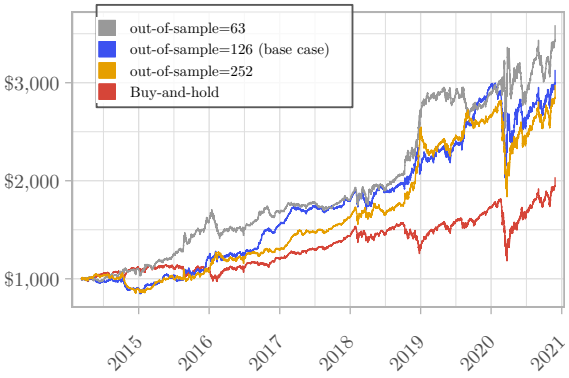
The findings of the sensitivity analysis for the S&P 500 Index are shown in Table 10 and Figure 10. We can see that the base model does not always produce the best outcomes for the S&P Index. A longer in-sample time to train the model, and predictions for a shorter out-of-sample period produce better results. RMSprop is a better optimizer in this case as well. The highest performance is obtained when using the log loss (base case) function. MinMax scaler in the range (-1,1) enables a steadier result and greater IR** than our base case scaler. Intriguingly, better outcomes are obtained with shorter sequence lengths and with fewer training epochs. Last but not least, as anticipated, decreased transaction costs enable higher performance.

Figure 10: Sensitivity Analysis #1: S&P 500 Index

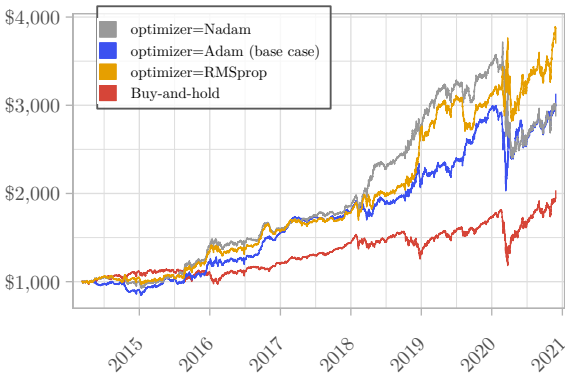
A: in-sample period



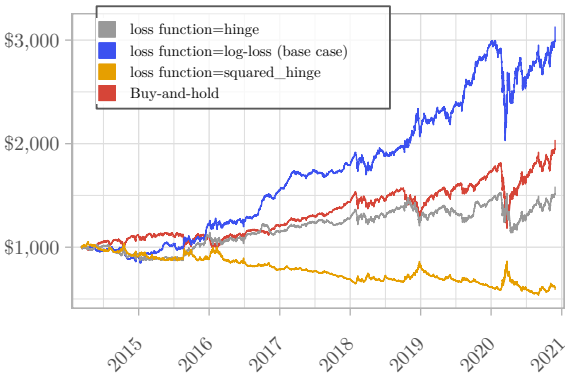
B: out-of-sample period



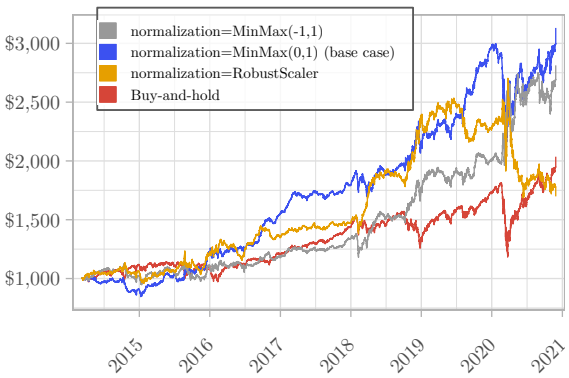
C: optimizer



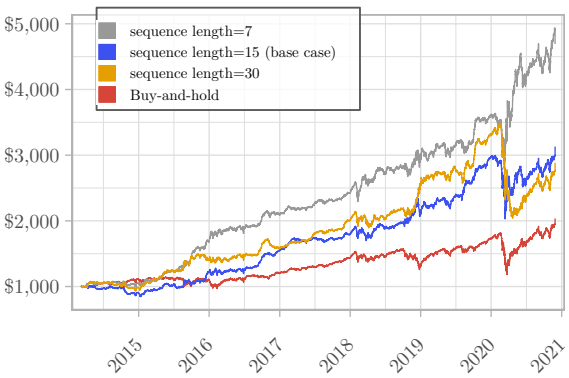
D: loss function



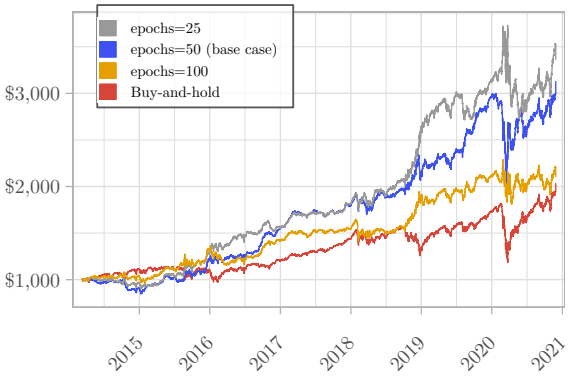
E: input variable normalization



F: sequence length



G: number of epochs



H: transactional costs

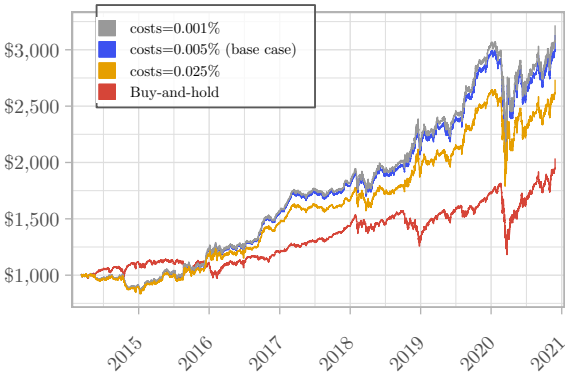


Table 11: Performance metrics for Sensitivity Analysis for Bitcoin

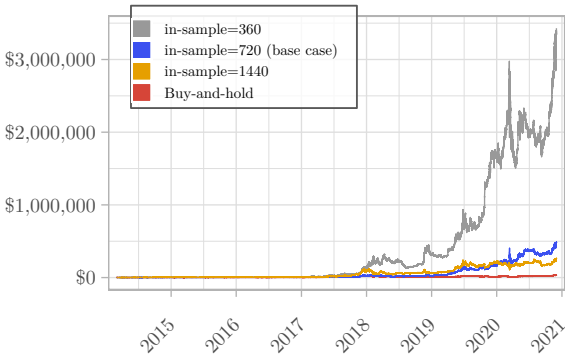
	Cumulative Return	ARC%	ASD	IR*	MDD	MLD	IR**	Position changes
Base case								
Approach #3	49,010.67%	150.32%	78.99%	1.9	-65.71%	1.3	**4.35**	0.94%
Buy-and-hold	3,373.06%	69.1%	78.99%	0.87	-83.94%	2.96	0.72	-
In-sample period								
360	342,954.24%	233.8%	78.99%	2.96	-61.7%	1.01	**11.22**	1.03%
720	26,575.60%	128.69%	79.01%	1.63	-66.16%	1.4	3.17	1.41%
Out-of-sample period								
360	480,853.69%	250.93%	79.02%	3.18	-66.68%	1.13	**11.95**	1.82%
720	282,084.00%	224.29%	78.99%	2.84	-76.56%	1.78	8.32	0.95%
Optimizer								
Nadam	78,244.90%	168.24%	78.99%	2.13	-73.94%	1.18	**4.85**	0.9%
RMSprop	28,675.45%	131.27%	79.0%	1.66	-81.93%	2.29	2.66	1.79%
Loss function								
hinge	7,307.44%	89.17%	78.99%	1.13	-87.56%	2.41	1.15	0.05%
squared hinge	-99.96%	-68.36%	78.99%	-0.87	-99.98%	5.88	-0.59	0.0%
Input variable normalization								
MinMax(-1,1)	3,416.80%	69.41%	79.04%	0.88	-68.95%	2.3	0.88	3.75%
RobustScaler	24.51%	3.3%	79.11%	0.04	-91.81%	5.88	0.0	5.58%
Sequence lengths								
7	86,783.85%	172.38%	79.0%	2.18	-70.76%	2.1	5.32	1.0%
30	172,872.33%	201.62%	79.0%	2.55	-71.76%	1.22	**7.17**	0.88%
Number of epochs								
25	256,580.46%	219.77%	78.98%	2.78	-74.31%	1.17	**8.23**	0.44%
100	3,400.54%	69.3%	79.06%	0.88	-76.07%	1.52	0.8	3.59%
Transaction costs								
0.02%	119,092.53%	185.44%	78.98%	2.35	-64.93%	1.15	**6.71**	0.94%
0.5%	-97.93%	-43.68%	80.93%	-0.54	-98.95%	6.65	-0.24	0.94%

Note: Trading begins on 2014-03-03 and ends on 2020-12-01. Base case (Approach #3) assumes in-sample period=126, out-of-sample perdiod=126, Optimizer=Adam, Loss function=log-loss, normalization=MinMax(0,1), sequence length=15, epochs=50 and transaction costs 0.005% for S&P 500 Index and 0.1% for Bitcoin.

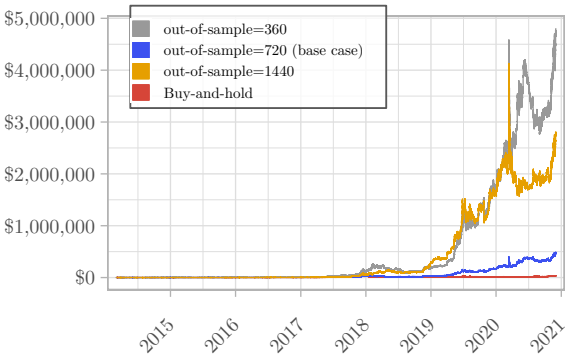
Table 11 and Figure 11 present the results of the sensitivity analysis performed on Bitcoin. It is evident that the base model does not always result in the best performance. Better outcomes are obtained with a shorter in-sample training period for the model. Performance improved whether the out-of-sample time was increased or decreased. In this instance, Nadam is a superior optimizer. The log loss (base case) function yields the best results, while the base scaler MinMax(0,1) also contributes to the best performance. It's interesting to note that both shorter and longer sequence lengths provide better results. Fewer epochs improve performance of the strategy. Not to mention, as expected, lower transaction costs enable better performance.

Figure 11: Sensitivity Analysis #2: Bitcoin

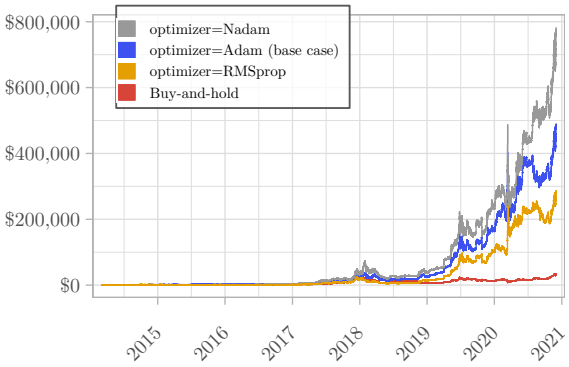
A: in-sample period



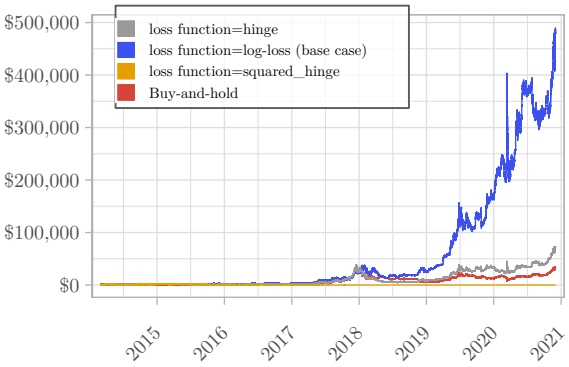
B: out-of-sample period



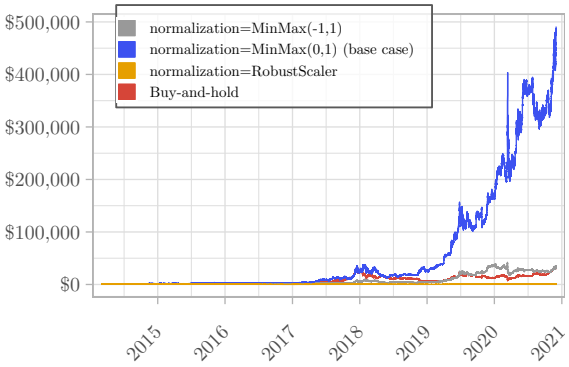
C: optimizer



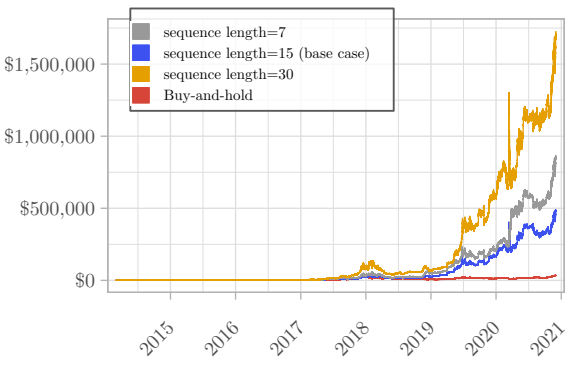
D: loss function



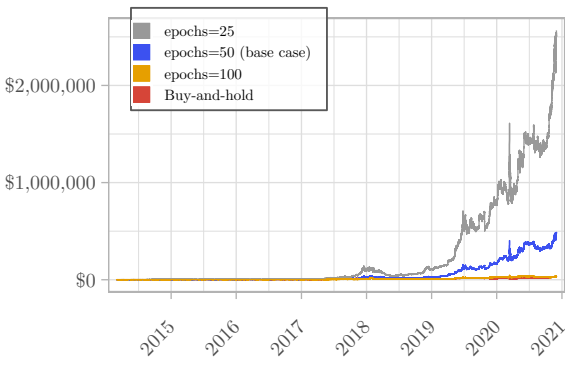
E: input variable normalization



F: sequence length



G: number of epochs



H: transactional costs

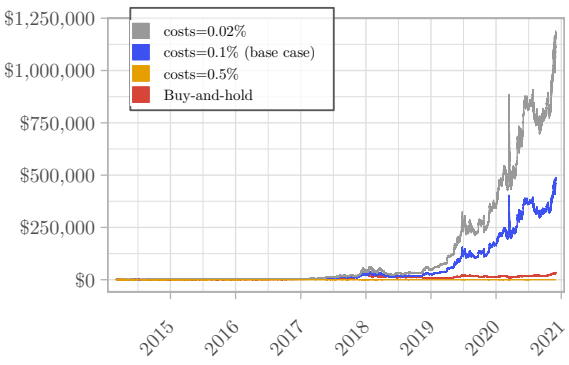


Table 12: Performance metrics for Sensitivity Analysis for SPX+BTC

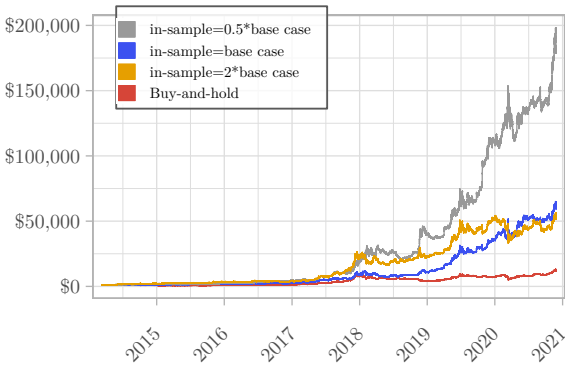
	Cumulative Return	ARC%	ASD	IR*	MDD	MLD	IR**
Base case							
Approach #3	6,026.46%	84.43%	43.3%	1.95	-38.15%	0.9	**4.32**
Buy-and-hold	1,119.35%	45.06%	40.57%	1.11	-58.3%	1.99	0.86
In-sample period							
0.5*base case	18,261.51%	117.13%	44.18%	2.65	-36.51%	0.9	**8.51**
2*base case	5,215.02%	80.57%	42.26%	1.91	-39.28%	0.92	3.91
Out-of-sample period							
0.5*base case	23,953.51%	126.03%	44.63%	2.82	-39.77%	0.82	**8.95**
2*base case	17,612.86%	115.97%	43.63%	2.66	-51.34%	1.58	6.01
Optimizer							
Nadam	8,004.19%	92.26%	43.86%	2.1	-40.93%	0.87	**4.74**
RMSprop	6,388.10%	86.01%	43.21%	1.99	-48.66%	1.64	3.52
Loss function							
hinge	1,941.96%	56.62%	42.73%	1.33	-57.92%	1.83	1.3
squared hinge	-96.20%	-38.5%	38.93%	-0.99	-97.55%	5.87	-0.39
Input variable normalization							
MinMax(-1,1)	1,561.04%	51.89%	41.81%	1.24	-38.75%	1.3	1.66
RobustScaler	136.31%	13.65%	40.46%	0.34	-62.7%	2.88	0.07
Sequence lengths							
7	12,111.98%	104.35%	43.06%	2.42	-41.17%	0.81	6.14
30	14,280.77%	109.38%	44.16%	2.48	-44.0%	1.17	**6.16**
Number of epochs							
25	20,617.74%	121.07%	43.9%	2.76	-41.65%	0.92	**8.02**
100	1,211.41%	46.64%	43.33%	1.08	-52.7%	1.49	0.95
Transaction costs							
0.2*base cost	9,992.09%	98.64%	43.49%	2.27	-37.68%	0.82	**5.94**
5*base cost	478.21%	29.82%	42.53%	0.7	-47.33%	1.52	0.44

Note: Trading begins on 2014-03-10 and ends on 2020-11-27. Base case (Approach #3) assumes in-sample period=126, out-of-sample perdiod=126, Optimizer=Adam, Loss function=log-loss, normalization=MinMax(0,1), sequence length=15, epochs=50 and transaction costs 0.005% for S&P 500 Index and 0.1% for Bitcoin. SPX+BTC portfolio assumes quarterly rebalancing, with weights for both assets equal to 50%.

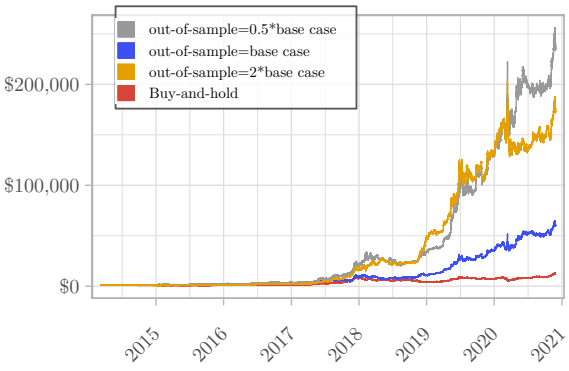
The findings of the sensitivity analysis conducted on the SPX+BTC portfolio are shown in Table 12 and Figure 12. We may use this to determine which criteria would be most effective for the performance of combined two assets. It is clear that the performance of the base model is not always the optimal. Shorter in-sample training times lead to better results. Whether the out-of-sample period was extended or shortened, performance improved. Furthermore, Nadam is a better optimizer. The best performance is produced by the log loss (base case) function, and also the base scaler MinMax(0,1). It's noteworthy to remark that better outcomes may be obtained with both shorter and longer sequence lengths, but with fewer epochs. As anticipated, higher performance is made possible by decreased transaction costs.

Figure 12: Sensitivity Analysis #3: SPX+BTC

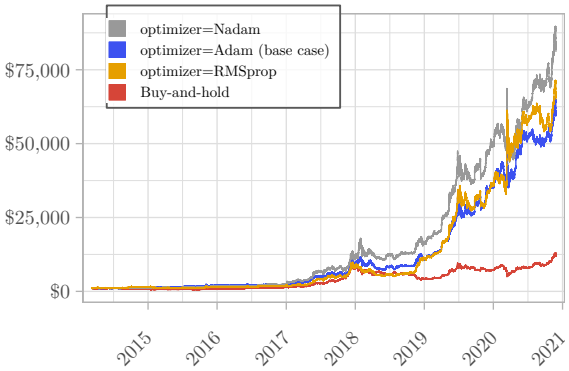
A: in-sample period



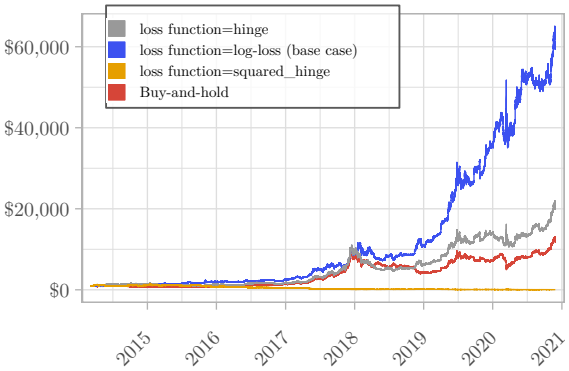
B: out-of-sample period



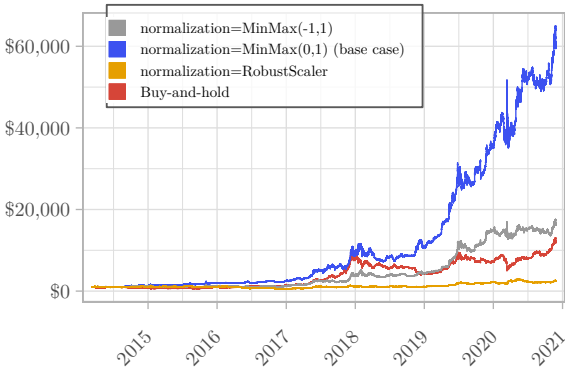
C: optimizer



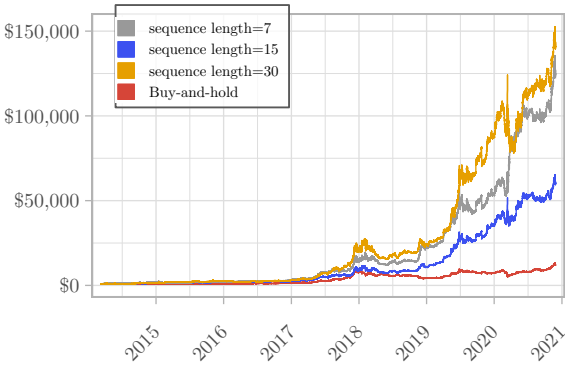
D: loss function



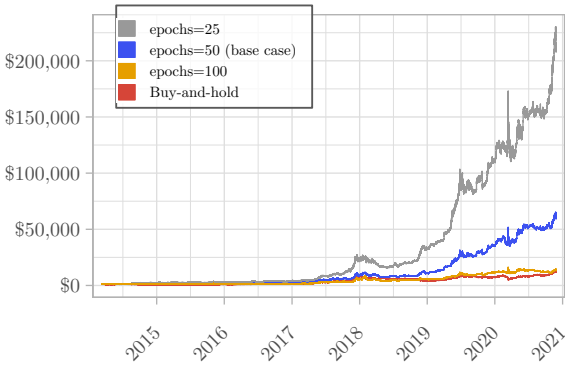
E:input variable normalization



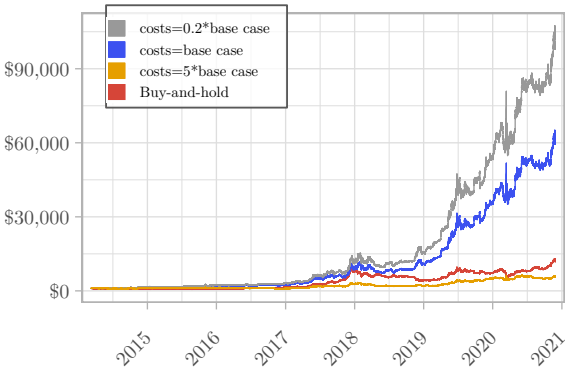
F: sequence length



G: number of epochs



H: transactional costs



The sensitivity analysis results show that our strategy is not resistant to changes in model structure or parameters; however, it is worth noting that the strategy for portfolio SPX+BTC almost always outperforms the benchmark, with only two exceptions - when we change the loss function or input variable normalization. The strategy appears to be quite resistant to changes in optimizer and produces comparable results for all three optimizers evaluated. Table 13 summarizes our sensitivity study, demonstrating how many portfolios (pure S&P 500 Index, pure Bitcoin, or a mix of the two) achieve the highest IR** measure for each parameter value.

We can see a few dependencies that are common to all assets. Firstly, the shortest length of out-of-sample period improves the results, secondly, log-loss loss function consistently produces the best outcomes. Surprisingly, the strategy performs better the fewer epochs are used during the training. Finally, lower transaction costs enhance the performance of the strategy.

Table 13: Summary of sensitivity analysis for all scenarios

A: In-sample duration		**B: Out-of-sample duration**	
in-sample=63	2	out-of-sample=63	3
126 (base case)	0	126 (base case)	0
in-sample=252	1	out-of-sample=252	0

C: Optimizer		**D: Loss function**	
optimizer=Nadam	2	loss function=hinge	0
Adam (base case)	0	log-loss (base case)	3
optimizer=RMSprop	1	loss function=squared_hinge	0

E: Input variable normalization		**F: Sequence length**	
normalization=MinMax(-1,1)	1	sequence length=7	1
MinMax(0,1) (base case)	2	15 (base case)	0
normalization=RobustScaler	0	sequence length=30	2

G: Number of epochs		**H: Transactional costs**	
epochs=25	3	costs=0.2*base case	3
50 (base case)	0	base case	0
epochs=100	0	costs=5*base case	0

Note: Table illustrates how many financial portfolios obtained the greatest IR** statistics for each parameter value during the sensitivity study

Conclusions and further research

Our study's objectives were to develop a successful trading strategy and to clarify which type of LSTM model architecture predicts better the values of particular financial

instruments, namely S&P 500 Index and Bitcoin. In order to accomplish this, we developed four LSTM approaches: the first one was an LSTM architecture based on literature with a regression task, the second was hyperparameter tuning of the first approach, the third was applying the same model from the first approach to a classification problem, and the fourth was an ensemble model of ten models from literature used for classification. All of the approaches were created in a walk-forward manner, with rolling windows of training, testing (and validation for Approach #2) periods.

The algorithms were tested on two assets: the S&P 500 Index and Bitcoin, as well as a portfolio made up of equal weights of each. For each portfolio, three data frequencies were used: daily, hourly, and 15 minute. We also developed an ensemble strategy using signals of all frequencies. The accuracy of forecasts was assessed by drawing an equity line for each model and comparing the Adjusted Information Ratio value (IR**). Only historical asset prices were used by our system to make decisions.

Five research questions were raised in this work:

RQ1. Which type of LSTM model architecture generates the best buy/sell signals for Bitcoin and/or S&P 500 Index algorithmic trading? The findings from all of our testing for the financial instruments were provided in Section 4. In Tables 8 and 9, summing performance of all the strategies, we see that on average classification approaches (Approaches #3 and #4) outperformed regression methods (Approaches #1 and 2), with Approach #3 producing the best results.

RQ2. Does intra-day data improve performance of transactional systems compared to systems using daily data? In Tables 8 and 9 there is no clear evidence that intra-day strategies outperform inter-day strategies. For regression problems, we find that daily data models outperform intra-day data models. However, for classification tasks, intra-day models dominate inter-day models in most cases.

RQ3. Does ensembling assets or signals frequencies improve the outcomes of investing strategies when compared to individual strategies? When compared to pure 15-minute strategies, ensembling frequencies generally improved performance, but it did not consistently outperform other frequencies. Ensembling assets did not always improve strategy results. When a strategy performed fairly well for both assets, combining them provided better risk-adjusted performance.

RQ4. Does hyperparameter tuning help achieve better performance of the investment strategy? We show results from hyperparameter tuning in Section 4.2 (Approach #2). It is evident that hyperparameter tuning had no beneficial effect. In fact, Approach #2 outperformed Approach #1 only in the case of daily data for Bitcoin and SPX+BTC.

RQ5. Do the results of the study change under different assumptions? Section 5 reported the findings of Sensitivity Analysis for Approach #3 and hourly frequency. We observed that the results vary greatly depending on the settings chosen. However, in the majority of circumstances, our model outperformed the benchmark, resulting in a robust profitable strategy.

Our research may be broadened in a variety of ways. The findings of our study differ greatly depending on the asset. Therefore, it would be advantageous to test additional asset classes to determine if they behave similarly. In our analysis, we only utilized the closing price as input. More inputs, such as OHLC (open-high-low-close), transaction volume, or technical indicators, may be useful. In our analysis, we create buy-sell signals for each asset separately and then combine the equity lines, assuming a constant weight of each asset in the portfolio. It would be good to build an ensemble LSTM model that predicts buy-sell signals for both assets or generates the weights of the assets in our

portfolio.

References

Abu-Mostafa, Y.S., Atiya, A.F. (1996). Introduction to financial forecasting. *Appl Intell 6, 205–213.*

Ariyo, A. & Adewumi, A. & Ayo, C. (2014). Stock price prediction using the ARIMA model. *UKSim 2014. 10.1109/UKSim.2014.67.*

Azari, A (2018). Bitcoin Price Prediction: An ARIMA Approach. *KTH Royal Institute of Technology*

Ballings, M. & Van den Poel, D. & Hespeels, N. & Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications, 42(20), 7046-7056.*

Baranochnikov I., Ślepaczuk R. (2022) A comparison of LSTM and GRU architectures with novel walk-forward approach to algorithmic investment strategy. *Working Papers of Faculty of Economic Sciences, University of Warsaw, WP 21/2022 (397).*

Barberis, N., Thaler, R. (2002). A Survey of Behavioral Finance. *Handbook of the Economics of Finance, 1(1), chapter 18, 1053-1128*

Billah, B. & King, M.L. & Snyder, R.D., Koehler, A.B. (2006). Exponential smoothing model selection for forecasting. *International Journal of Forecasting, 22(2), 239-247*

Box, G. E. P., & Jenkins, G. M. (1976). Time series analysis: Forecasting and control. *San Francisco: Holden-Day.*

Chen W-H., Shih J-Y., Wu S. (2006) Comparison of support-vector machines and back propagation neural networks in forecasting the six major Asian stock markets. *International Journal of Electronic Finance, (I), 49-67.*

Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance, 25(2), 383–417.*

de Faria, E.L. & Albuquerque, M.P. & Gonzalez, J.L. , Cavalcante, J.T.P. & Albuquerque, M.P. (2009). Predicting the Brazilian stock market through neural networks and adaptive exponential smoothing methods. *Expert Systems with Applications 36(10), 12506-12509.*

Gers, F. & Schmidhuber, J. & Cummins,F. (2000) Learning to Forget: Continual Prediction with LSTM. *Neural Computation, 12(10):2451–2471.*

Grudniewicz J., Ślepaczuk R. (2021) Application of machine learning in quantitative investment strategies on global stock markets. *Working Papers of Faculty of Economic Sciences, University of Warsaw, WP 23/2021 (371).*

Hochreiter, S., Schmidhuber, J. (1997). Long Short-term Memory. *Neural computation, 9, 1735-80.*

Hossain, M.A. & Rezaul, K. & Thulasiram, R.K. & Bruce, N. & Wang, Y. (2018). Hybrid Deep Learning Model for Stock Price Prediction. Paper at the 2018 IEEE Symposium Series on Computational Intelligence, Bangalore, India.

A.K. Jain; Jianchang Mao; K.M. Mohiuddin (1996) Artificial neural networks: a tutorial. *Computer. 29(3), 31-44*

Ji, S. & Kim, J. & Im, H. (2019) A Comparative Study of Bitcoin Price Prediction Using Deep Learning. *Mathematics. 7(10):898.*

Kamalov, F., Gurrib, I. & Rajab, K. (2021). Financial Forecasting with Machine Learning: Price Vs Return. *Journal of Computer Science, 17(3), 251-264.*

Kijewski, M. & Ślepaczuk, R. (2020) Predicting prices of S&P500 index using classical methods and recurrent neural networks. *Working Papers of Faculty of Economic Sciences, University of Warsaw, WP 27/2020 (333)*

Lahmiri, S. & Bekiros, S. (2020) Intelligent forecasting with machine learning trading systems in chaotic intraday Bitcoin market. *Chaos, Solitons & Fractals, 133(C).*

Lo, A.W., Hasanhodzic, J. (2010). The Heretics of Finance: Conversations with Leading Practitioners of Technical Analysis. *John Wiley and Sons.*

Malkiel, B.G. (1973). A Random Walk Down Wall Street. *(W.W. Norton & Co., New York).*

Malkiel, B.G. (2005). Reflections on the Efficient Market Hypothesis: 30 Years Later. *Financial Review 40(1):1-9.*

Michańków J., Sakowski P., Ślepaczuk R. (2022) The comparison of LSTM in algorithmic investment strategies on BTC and SP500 index. *Sensors 22, 917*

Olah, C. (2015). Understanding LSTM Networks. Available at: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (Accessed: 11 July 2022).

Ryś, P. & Ślepaczuk, R. (2018) Machine learning in algorithmic trading strategy optimization - implementation and efficiency. *Working Papers of Faculty of Economic Sciences, University of Warsaw, WP 25/2018 (284)*

Sepp, H. & Schmidhuber, J. (1997) Long Short-Term Memory. *Neural computation, 9(8):1735–1780.*

Shah, D. & Campbell, W. & Zulkernine, F. (2018). A Comparative Study of LSTM and DNN for Stock Market Forecasting. Presented at IEEE International Conference on Big Data (Seattle, Washington, USA).

Schulmeister, S. (2009) Profitability of technical stock trading: Has it moved from daily to intraday data? *Review of Financial Economics, 18(4).*

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