Master Thesis

Quantitative Finance

Predicting intraday stock returns using a hybrid ARIMA and long short-term memory neural network model

Student name:

Jesper Groenendijk

Student number:

417014

Supervised by:

Prof. Dr. M. van der Wel

Second assessor:

Drs. S.H.L.C.G. Vermeulen

ERASMUS UNIVERSITEIT ROTTERDAM

ERASMUS SCHOOL OF ECONOMICS

November 8, 2021

Abstract

Complex problems such as predicting stock returns often contain both linear and nonlinear structures. A hybrid methodology combining both linear and nonlinear models can take advantage of the unique strengths of the different models. The AutoRegressive Integrated Moving Average (ARIMA) is very popular for time series forecasting. More recently neural networks have proven to be very good in dealing with large amounts of data, making accurate time series predictions. Artificial neural networks (ANNs) and long short-term memory (LSTM) neural networks are among the best performing neural networks for time series prediction. This paper evaluates the performance of a hybrid methodology combining the ARIMA with the ANN, and an adjusted hybrid methodology combining the ARIMA with the LSTM in predicting daily and intraday stock returns. The hybrid model performance is evaluated relative to the models individually and other forecast combinations. The empirical results suggest that the hybrid methodology does improve in terms of forecasting accuracy upon the models individually and the different forecast combinations. The adjusted hybrid model outperforms the hybrid model in the majority of the cases.

Contents

1 Introduction

The stock market is extremely hard to model and predict. For investors it can be very beneficial to have the ability to predict price movements in order to find the right time buy or sell. Especially in quantitative finance stock price time series modeling and forecasting has become a major field of research, as accurately predicting prices is crucial in financial risk evaluation and asset allocation. Due to the development of stock exchanges it is now possible to trade and gather data at a frequency up to nanoseconds, and traded volumes are very high. Therefore significant profit can be achieved with slight improvement in predictions. The task of predicting these future stock prices can be done by using either a linear or nonlinear model. However, Zhang (2003) introduces a hybrid methodology that makes use of the strength of both the linear and nonlinear prediction models. Nonetheless, is this model accurate in predicting intraday stock returns, and is it possible to improve upon this hybrid model?

Stock returns can be decomposed into a discrete time series containing data. In time series forecasting we essentially try to predict a point in the future by analyzing the observed time series points in the past. Because time series often both contain linear and nonlinear patterns, it can be beneficial to combine linear models with nonlinear models to accommodate for both the linear and nonlinear structure. This paper uses the AutoRegressive Integrated Moving Average (ARIMA) model to account for the linear part. The ARIMA is a model class that explains a time series based on its past values, and uses these values to forecast future values. Stock and Watson (1998) find that the overall performance of these single methods can be improved when the models are combined. The ARIMA model alone is not adequate in modelling and predicting this time series data as it cannot deal with the nonlinear relationships. A nonlinear model such as Neural Networks alone is not capable of handling nonlinear and linear patterns equally well. By combining linear and nonlinear methods, complex structures in the data can be modeled more accurately.

Real problems often contain nonlinear patterns. To overcome this, several nonlinear models have been introduced in the literature. The traditional nonlinear approaches include bilinear models, autoregressive conditional heteroscedastic models and threshold autoregressive models. However, De Gooijer and Kumar (1992) show that the gain of using them is limited as these nonlinear models are developed for specific nonlinear patterns. More recently ANNs have been suggested as a nonlinear method for time series forecasting. Khashei and Bijari (2010) show that these neural networks are very flexible computing frameworks that can be applied to a wide range of time series forecasting problems with a high degree of accuracy. Current technology allows for huge amounts of data such as stock prices at the horizon of even a nanosecond, along with traded quantities for specific financial instruments. A neural network is able to work with a large amount of data along with many different input variables at the same time. This is very beneficial as it can deal with a large amount of indicators which possibly contain information in which direction a stock price could move. The main strength of the neural network is its ability to find patterns and irregularities and detect multi-dimensional non-linear connections in data. This gives it the potential to be very useful in modeling dynamic systems like the stock market.

There are several ways to combine the linear and nonlinear models. Zhang (2003) proposes a hybrid methodology to combine the ARIMA with artificial neural networks (ANNs). In the methodology the time series is first estimated using the linear ARIMA, then the error of the linear model remains containing the nonlinear relationship. This error is estimated by the ANN, finally the forecasted value and forecasted error are added together to form the combined forecast. The methodology is tested in forecasting the annual number of sunspots, the number of lynx trapped per year and the exchange rate between the British pound and US dollar. He finds that the combined model can be an effective way to improve forecasting accuracy achieved by either of the models used separately. This research evaluates the performance of the model of Zhang (2003) in predicting intraday stock returns, as the ANN can be a very effective machine learning technique for predicting stock returns. Vijh et al. (2020) use the ANN and random forest techniques to predict stock closing prices. They construct several technical indicators from the data such as the moving average to use as inputs for the model. The forecasts of the two machine learning techniques produce a low root mean-squared error (RMSE), this shows that the models are efficient in predicting stock closing price.

Although the hybrid methodology of Zhang (2003) is relatively old, the hybrid framework is still adopted in numerous papers for various predicting purposes. Ordóñez et al. (2019) use the hybrid methodology predict the remaining useful life of aircraft engines. The ARIMA is used first to estimate the predicted value, next the support vector machine is used to estimate the error of the ARIMA. The predictions are then combined to get the predicted value for the remaining useful life, the results are then compared to those obtained using a Vector AutoRegressive Moving Average (VARMA) model. The results show far greater prediction capability by the hybrid model over the VARMA model. Zhang et al. (2018) predict the short term electricity load using the hybrid methodology combining the ARIMA with a wavelet neural network. The simulation results show that the proposed hybrid methodology has the best performance over the comparison models.

This paper adjusts the hybrid model of Zhang (2003) by combining the ARIMA with a different method. Since the ARIMA is used to fit the data first, the model that fits the error term is of crucial importance for the forecasting accuracy. Zhang (2003) argues that in the traditional forecasting literature often linear models are combined. However, he finds that using dissimilar models or models that disagree strongly in the hybrid methodology will have lower generalization variance in the error. Therefore this paper will use dissimilar models to combine in an adjusted hybrid strategy. Since the year of 2003 in which the paper of Zhang is published. The literature regarding machine learning has developed rapidly, and the machine learning techniques have improved a lot. Rather et al. (2015) propose a robust novel hybrid model for prediction of stock returns. The proposed model is constituted of the ARIMA and exponential smoothing model to account for the linear part, and a recurrent neural network to account for the nonlinear part. The results show that the hybrid prediction model shows an outstanding prediction performance as it outperforms the recurrent neural network. Hence replacing the ANN in the hybrid model of Zhang (2003) with a recurrent neural network can be beneficial for predicting stock returns.

A recurrent neural network has a recurrent connection on the hidden state ensuring that the sequential information and dependencies in the input data are captured. This makes it very useful in time series prediction as the model now remembers previous inputs as well. The Long Short-Term Neural Network (LSTM) belongs to the class of Recurrent Neural Networks. This type of neural networks use their own internal state (memory) to process the input sequences. This is done by adding a hidden layer to provide feedback on itself, making it recurrent. Fischer and Krauss (2018) find that LSTM neural networks are inherently suitable for sequence learning, and that they outperform memory-free classification methods in predicting stock price movements. This paper adjusts the hybrid model of Zhang (2003) by combining the linear ARIMA with the LSTM instead of the ANN. Chang et al. (2018) use the LSTM model to predict electricity prices. However the performance of the LSTM alone was not very satisfactory. That is why this research decides to optimize the LSTM using a the stochastic gradient-based optimization of Adam (see Kingma and Ba (2015) for a detailed explanation). Adam has combined the advantages of popular optimization methods such as AdaGrad (see Lydia and Francis (2019)) and RMSProp (see Tieleman et al. (2012)). The results show that optimizing the LSTM model using Adam can significantly improve the prediction accuracy. Hence in this paper the LSTM will be optimized using the Adam optimization algorithm.

This paper evaluates the predicting performance of the different models in predicting both intraday returns on the horizon of one minute and daily returns. Intraday returns are especially of importance for day traders, these market participants use daytime price fluctuations to generate a profit and do not leave positions open overnight. If the financial numbers for a specific company are good, it is very likely that over next few days the stock price will go up and will yield a positive return. However, it is much less likely that the stock price will yield a positive return in the next few minutes. The intraday dynamics of stock prices are likely to be very different than daily stock prices. Therefore, if a model is accurate in predicting daily returns, the model need not to be accurate in predicting intraday returns and vice versa. That is why predicting daily and intraday returns are two different tasks. In this paper both tasks are evaluated.

The predictability of the returns is evaluated for both stocks and market indices. Stock prices are subject to the performance of one company, while the market index is subject to a lot of companies. Hence the return of the index is less volatile than the return of the stocks on the scale of both minutes and days. Therefore it is possible that a model able to accurate predict the return of the market index, but fails in predicting stock returns. These stocks and market indices are very liquid as they are traded very frequently. On trading days during exchange opening hours it is always possible to buy or sell the financial instrument for the actively traded price. Overall, it is clear that accurately predicting the return of either the stock or market index can lead to large profits.

In the hybrid methodology first the linear method is used to model the linear component. Next the residuals from the linear model containing the nonlinear relationship are modeled using the nonlinear method. However, Taskaya-Temizel and Casey (2005) find that one may not guarantee that the residuals of the linear component may comprise valid non-linear patterns. Hence, it might be sensible to try and avoid the hybrid methodology altogether and find an alternative method to combine the models. This could be done by producing weighted combinations of the predictions of the linear and nonlinear models. Therefore, this paper also analyzes different more straightforward model combinations to put the performance of the hybrid models in perspective. To put the practical relevance of these models into perspective, several trading strategies are constructed using the predictions obtained by the models. These trading strategies can help give an insight in the profitability of the different models. It can be possible that the adjusted hybrid model produces more accurate forecasts over the individual models, but still is not profitable in practice. Then to put the performance of the trading strategies into perspective, I construct several benchmark strategies based on the data alone. Do the model based trading strategies really improve upon the simple model-free strategies?

The main contribution of this paper is that to the best of my knowledge this is the first research to use a hybrid methodology to predict intraday stock prices. Hence, I extend the research of Zhang (2003) from an application perspective by evaluating their hybrid methodology in predicting intraday stock prices. But also from an methodological perspective by changing the methodology used within the hybrid model to increase predicting performance. This paper combines the more traditional literature of Zhang (2003), with the more recent literature like Fischer and Krauss (2018), Chang et al. (2018) and Vijh et al. (2020). The current literature focuses on the performance of the hybrid methodology compared to the models individually, this paper also focuses on the performance of the hybrid method compared to that of different model combinations. Finally, this paper provides insight in the practical relevance of the models by developing trading strategies with the model predictions.

The empirical results using the financial data are in line with the findings of Zhang (2003). The hybrid models do improve upon thee individual models in terms of forecasting accuracy. The hybrid models differ in the sense that the original hybrid model of Zhang (2003) makes use of the ANN, while the adjusted hybrid model makes use of the LSTM. The empirical results suggest that switching to an LSTM is a sensible thing to do, as the adjusted hybrid model outperforms the original hybrid model in the majority of the cases. For the trading strategies there is no strategy that consistently outperforms the others. This could be due to the fact that the models do not aim to maximize profit, but aim to minimize the mean squared error. In terms of profitability, the stock price movements have proven to be very hard to capture on the minute interval. Hence, the trading strategies are often not profitable on the scale of minutes. The daily observations in general contain less volatility and do produce profitable strategies in most cases for the individual and hybrid models.

This paper is structured as follows. Section 2 provides a detailed description of the data used in this research, as well as the technical indicators that are constructed from the data. Section 3 describes the different models, hybrid models, model combinations and trading strategies used in this paper. Section 4 presents the results obtained from the models and trading strategies, and provides a robustness check. Finally Section 5 concludes and discusses some shortcomings of this research as well as suggestions for future research.

2 Data

This section elaborates on the data used in this research. First section 2.1 provides a description of the financial data and its properties. Next section 2.2 gives a detailed explanation of the technical indicators used for predicting purposes.

2.1 Financial data

This paper uses intraday stock prices for two major indices of the world and two large-cap stocks. The two stock indices are the S&P 500 and the NASDAQ. The SP 500 stands for the Standard and Poor's 500, this index tracks 500 large U.S. companies across a span of industries and sectors. The NASDAQ is an acronym for National Association of Securities Dealers Automated Quotations, this index tracks roughly 3,000 companies traded on the NASDAQ exchange. These indices are very actively traded and a good indicator of the state that the market is in. The two large-cap stocks are Microsoft which is one of the leaders in the technology sector, and Amazon which is one of the leaders in the retail trading sector. These are two well established companies from different sectors which are less sensitive to specific news items, giving them more potential to be predictable with just financial data.

These stocks and market indices are very liquid, to give an example Amazon is traded over 6 million times daily over the span of the horizon used in this paper. As there are always a large amount of buyers and sellers for these financial instruments, the difference between the best bid and ask price is very small. The bid price implies the highest price a market participant is willing to pay for a security, and the ask price implies the lowest price a market participant is willing to accept for the security. On trading days during exchange opening hours it is always possible to buy or sell the financial instrument for the actively traded price. If one can accurately predict the future price or return of one of these stocks or indices, you have a very big advantage over other market participants. Hence incorporating accurate predictions into a trading strategy could potentially lead very large profits.

The data is collected at time horizons of one minute and daily stock prices, along with their traded quantities. The data is obtained from the TradingView network. The minute data for the stock prices of Amazon and Microsoft date from the 14^{th} of December 2020 to the 2^{nd} of March 2021. The minute data of the market indices date from the 1^{st} of March 2021 to the 14^{th} of may 2021. The dates of this data are according to the maximum availability of minute data on TradingView at time of gathering the data. The daily data dates from the first of January 2003 until the 19^{th} of may 2021 for all stocks and market indices. This period starts some time after the dot-com bubble, which largely affected stocks like Microsoft and Amazon, running up to the date the data is acquired. From this data several technical indicators are constructed which can be used as input for the neural networks. The minute data runs from market opening of the New York Stock Exchange (NYSE) at 9:30 am until market closing time at 4 pm containing 390 minute observations per day. Some of the minute data missed observations during the day. For these missing observations it is assumed there were no registered trades hence the price of the missing minute was the same as the previous price. For Amazon this occurred in 74 observations, S&P500 missed 4 observations and NASDAQ missed 8 observations.

The stock prices show different patterns across the horizon. Figure 1 shows the development of the prices for the different stocks and indices over time. It can be seen that the minute prices for the two different stocks are very volatile, especially minute prices of Amazon. The minute prices for the market indices are less volatile as they are subsidiary to much more factors. Hence I expect the stock prices of Amazon and Microsoft to be much harder to predict than the prices of the S&P 500 and the NASDAQ index. The daily prices show a similar pattern as the stock market has grown rapidly in the past 16 years. Here the prices have increased exponentially over the past few years. However the fraction of times the price went either up or down in still pretty evenly spread, this is shown in Table 1. Hence I expect the trading strategies to achieve more extreme values for the cumulative returns for the daily prices.

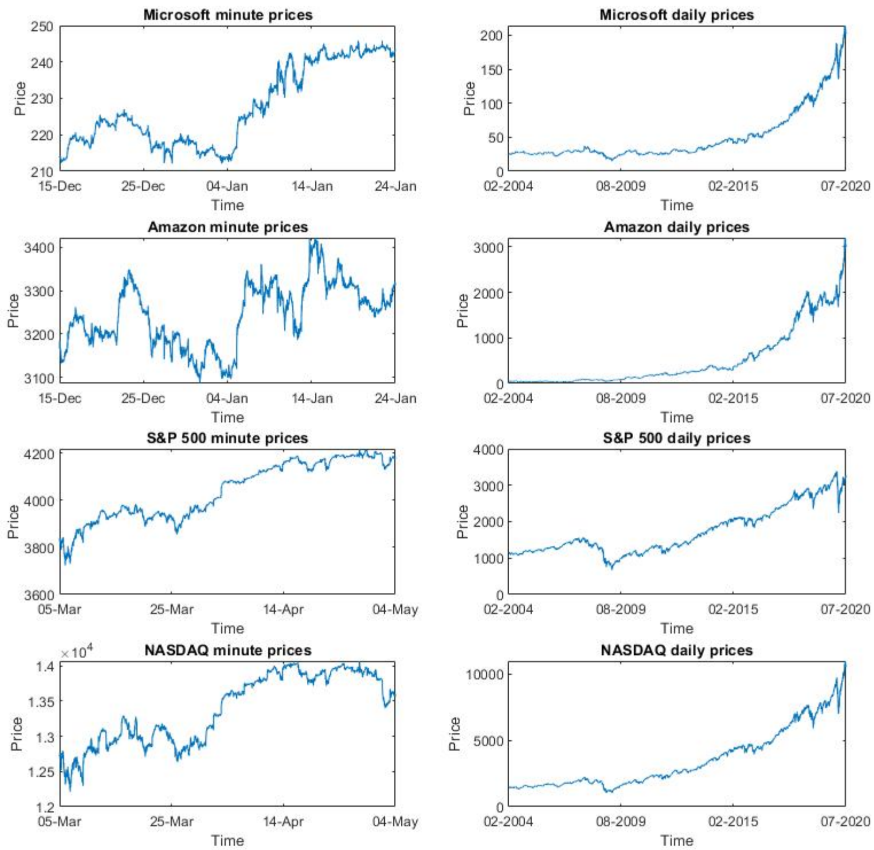

Figure 1: Minute prices and daily prices for the different stocks and market indices.

Because the data contains are real prices as traded on the exchange, the minute observations contain a gap between market closing and market opening every trading day. During this period between closing and opening a lot can change in the financial market, which can have a large impact on the price of the stock. Hence the price difference between market closing and market opening can be significantly large. This is visualized in the returns of the left graph in Figure 2. This overnight difference in price is extremely hard to predict using just financial data from the previous day. The ARIMA model as described in the methodology Section 3 makes use of at most seven lags. Therefore the first seven minutes of the day are being used as a burn-in period. This implies that these observations are not predicted by the model but are only used to help predict the price in the remainder of the trading day. Removing the first seven returns of the day results in the right graph in Figure 2. This time series contains a lot less outliers which improves the predictability.

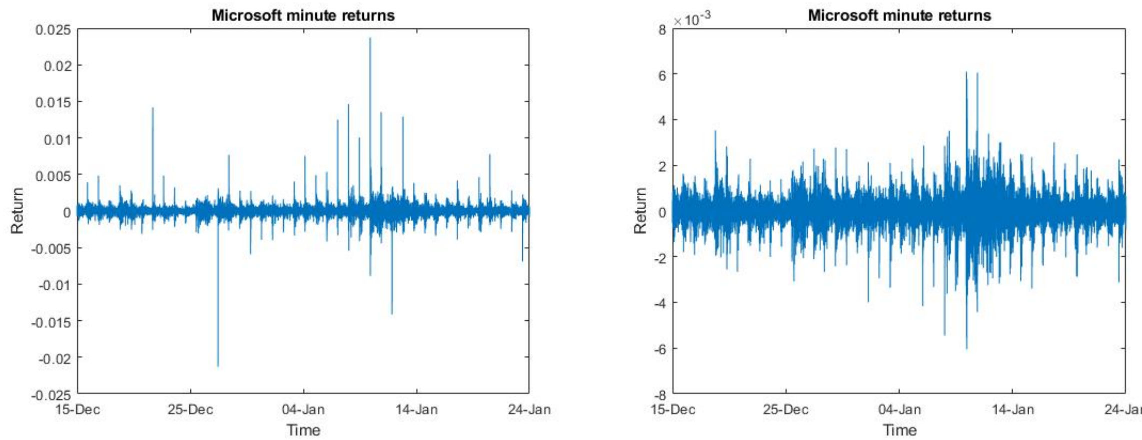

Figure 2: Microsoft minute returns with (left) and without (right) the first seven minutes of the day.

Table 1 shows the summary statistics to provide more insight into the data. The amount of observations differ for the minute data as Microsoft and Amazon trade for 390 minutes each day, and the S&P500 and NASDAQ trade for 400 and 391 minutes per day respectively. The standard deviation shows the financial instruments are very volatile, especially in the daily data it shows that the prices have changed a lot. The upticks and downticks indicate whether the price has moved up or down with respect to the previous observation. For the minute data the numbers do not sum to one as some times the price stays the same.

Table 1: Summary statistics

	Microsoft		Amazon		S&P 500		Nasdaq	
Statistic	Minute	Daily	Minute	Daily	Minute	Daily	Minute	Daily
Observations	20280	4250	20280	4250	21600	4250	21114	4250
Standard deviation	10.57	52.03	78.64	833.30	126.3160	759.03	460.30	2,875
Upticks	0.4963	0.5073	0.4972	0.5158	0.4971	0.4731	0.5030	0.5478
Downticks	0.4829	0.4927	0.5024	0.4842	0.4839	0.5269	0.4969	0.4522

2.2 Technical indicators

To improve the forecasting accuracy, in this paper several technical indicators are constructed of the data. These technical indicators contain certain characteristics of the data, giving it statistically and economically significant predictive power (see Neely et al. (2014)). The most commonly used indicators in the literature that have proved to be effective are selected. Among others, the indicators are based on research done by Tanaka-Yamawaki and Tokuoka (2007) and Kara et al. (2011). The indicators will be used for all neural networks such that the model performance can be compared using the same available information. The technical indicators can be divided into three categories: trend indicators, momentum indicators and volatility indicators.

In order to get a measure of the direction and strength of that direction, this paper uses trend indicators. This is done by averaging over the recent past. If the price of a stock is above the value of this indicator it can be interpreted as a rising trend. If the price of a stock is below the value of this indicator it can be interpreted as a downwards trend. Among the trend indicators used in this research is the moving average. The moving average is calculated by taking the average price over the last 10 observations. Next is the exponential moving average, this is also calculated as the average price over the last 10 observations but then assigning a larger weight to the most recent observations. Finally is the Moving Average Convergence Divergence (MACD), this calculates the difference between the different moving averages using different sample sizes. These moving averages are all commonly used in the literature. Among others they are used in both Tanaka-Yamawaki and Tokuoka (2007) and Kara et al. (2011).

The momentum indicators can be used to identify the momentum and speed of the price change. The momentum indicators used in this research are the Relative Strength Index (RSI) and the simple momentum. The RSI calculates the ratio between the upticks and downticks indicating whether the stock is overbought or oversold. If the price often increased relative to the previous timestep the value of the RSI will be higher. The simple momentum is calculated by taking the difference of the current price and the previous price indicating whether the price is rising or falling. To provide a better insight of the RSI indicator Figure 3 on the next page illustrates a part of the graph of the RSI of Microsoft using minute intervals. Here the upper bound and lower bound are set at 70% and 30% respectively can be interpreted as signals. If the RSI crosses the upper bound the stock can be considered as overbought, meaning that he current price is considered to be overvalued based on its past performance. Therefore, it could be expected that the price is about to drop. Hence, it might be a good time to sell. The opposite holds for when the RSI crosses the lower bound. The RSI and simple momentum indicator are again very commonly used in stock price prediction, they are both used in the research of Tanaka-Yamawaki and Tokuoka (2007) and Kara et al. (2011).

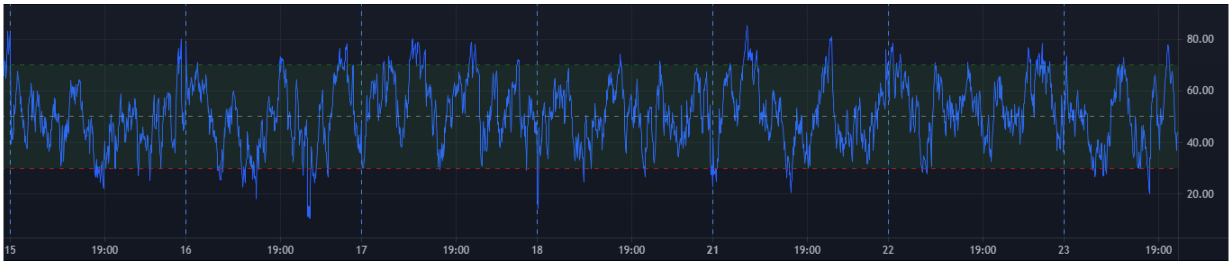

Figure 3: The RSI for Microsoft using minute intervals.

Finally the volatility indicators provide insight in the volatility of the price of the financial instrument. The volatility indicator used in this research is the historical volatility. This indicator calculates how much the price deviated from its mean in the past 10 observations. If the value of this indicator is high, this can indicate that price changes in the near future will be higher as well. The historical volatility is commonly used in stock price prediction, Wang and Kim (2018) use the historical volatility together with the MACD to predict the stock price trend. The mathematical formulas for the different indicators are given in Table 2. Here the price at timestep i is denoted by p_i and the weight at timestep i is denoted by w_i.

Table 2: Mathematical formulas for the technical indicators

Indicators	Formulas	Default Values
Moving Average	$\frac{1}{n}\sum_{i=1}^{n} p_i$	$n = 10$
Exponential moving average	$\frac{1}{n}\sum_{i=1}^{n} w_i p_i$	$n = 10$
MACD	$\frac{1}{m}\sum_{j=1}^{m} w_j p_j - \frac{1}{k}\sum_{i=1}^{k} w_i p_i$	$m = 26,\ k = 12$
RSI	$100 - \frac{100}{1+\frac{AverageGain}{AverageLoss}}$	$r = 14$
Historical Volatility	$\frac{1}{n}\sum_{i=1}^{n}(p_i - \bar{p})$	$n = 10$
Momentum	$p_i - p_{i-1}$	

The moving average and exponential moving average are based on a window of 10 timesteps. The MACD is calculated using a window of 26 and 12 timesteps. The RSI is calculated using the average gain and average loss over a window of 14 timesteps. The historical volatility is calculated over the last 10 timesteps.

3 Methodology

In this section the different models and trading strategies are discussed. First section 3.1 gives a brief description of the Autoregressive Integrated Moving Average (ARIMA). Next section 3.2 elaborates on the artificial neural network (ANNs). Followed by Zhang's hybrid model in section 3.3 and the adjusted hybrid model in section 3.4. The adjusted hybrid model makes use of the Long Short-Term Memory (LSTM) neural networks, this is explained in section 3.5. Then to put the hybrid model performance into perspective, several other model combinations are introduced in section 3.6. In order to evaluate the models section 3.7 introduces the evaluation criteria. Finally to test the models for potential profitability, section 3.8 elaborates on the trading implementation.

3.1 The ARIMA Model

The Autoregressive Integrated Moving Average (ARIMA) model is a linear model. The ARIMA methodology is based on the research of Zhang (2003). Here it is assumed that the future value of a stock is a linear function of past observations and random errors. The process generates time series of the form:

$$y_t = \theta_0 + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t - \theta_1 \epsilon_{t-1} - \theta_2 \epsilon_{t-2} - \cdots - \theta_q \epsilon_{t-q}, \tag{1}$$

where y_t and ϵ_t represent the actual values and error terms at time t respectively. The model parameters are ϕ_i and θ_j for $i = 1, 2, \ldots, p$ and $j = 0, 1, \ldots, q$, where p and q are integers and represent the model order. The error terms ϵ_t are assumed to be random independent and identically distributed with mean zero and variance σ^2.

The ARIMA model in (1) can take on several important special cases. If $p = 0$ the model becomes an Moving Average model of order q, and when $q = 0$ the model becomes an Autoregressive model of order p. The ARIMA model determines the appropriate model orders of (p, q). Box and Jenkins (1970) have developed a practical approach of building ARIMA models consisting of three iterative steps. These three steps are model identification, parameter estimation and diagnostic checking. In the model identification step it is checked whether the model has some theoretical autocorrelation properties. If the time series is generated from an ARIMA process, it should have theoretical autocorrelation properties. These theoretical properties can then be matched with the empirical properties to identify one or more potential models for the time series. The autocorrelation and partial autocorrelation functions of the sample data can then be used as tools to determine the order of the ARIMA model.

In building an ARIMA model, stationarity is a necessary condition. Hence, in the identification step data transformation is often needed. To transform the financial data to stationary data the stock prices are converted to returns. Then the stock returns are checked for stationarity using the augmented Dickey-Fuller test. The statistical characteristics of a stationary time series such as the mean and the autocorrelation structure are constant over time. If the time series contains trend and heteroscedasticity, differencing and power transformation can be applied to stabilize the variance and remove the trend before the ARIMA model can be fitted. The parameter estimation step is straightforward. The parameters are estimated using a nonlinear optimization procedure minimizing the overall measure of errors. In contrast to the neural networks, the ARIMA model does not receive the different technical indicators as input. This can be a disadvantage, however more input does not necessarily lead to more accurate forecasts.

In the last step diagnostic checking the model is checked whether it is adequate. This basically implies that it is checked if the model assumptions about the errors ϵ_t are satisfied. This can be done by examining the goodness of fit for several diagnostic statistics and plots of the residuals of the model to the historical data. If the model is not adequate, a new model should be build which is then followed by the steps of parameter estimation and diagnostic checking. Finally this three-step procedure is generally repeated multiple times until a satisfactory model is selected. The final model can then be used for predicting.

3.2 Artificial Neural Networks

Artificial Neural Networks (ANNs) are flexible computing frameworks which are suitable for modelling nonlinear time series. Because ANNs are able to parallel process a lot of information from data, it can approximate a large class of functions with a high accuracy. The network contains a set of artificial neurons, these are called the nodes. between the nodes there is a set of directional edges. The goal of the network is to process the input and assign specific weights to the input to obtain an output. These weights are designed in such a way that the output is as close as possible to the true value. In order to achieve this, the model is trained to determine optimal weights in a pre-specified training sample. This hold-out sample has the sole purpose to train the neural network and is not used for testing. The models do not require prior assumptions and the model form is largely determined by the characteristics of the data. Single hidden layer feedforward network is a widely used model for forecasting and time series modeling. This model is characterized by a network of three layers of processing units which are connected by acyclic links, this is visualized in Figure 4. Following the notation of Zhang (2003) the output y_t is of the following mathematical form:

$$y_t = \alpha_0 + \sum_{j=1}^{q} \alpha_j g(\beta_{0j} + \sum_{i=1}^{p} \beta_{ij} y_{t-i}) + \epsilon_t, \tag{2}$$

where α_j and β_{ij} are the model parameters often referred to as the connection weights for $i = 0, 1, \ldots, p$ and $j = 1, 2, \ldots, q$ with p the number of input nodes and q the number of hidden nodes. The ANN receives the different technical indicators as input. The logistic function $g(\cdot)$ is often used as the layer transfer function and is of the form:

$$g(x) = \frac{1}{1 + \exp(-x)}.$$

Such that the ANN model in (2) is a nonlinear mapping function of the past observations to the future value y_t which takes the form:

$$y_t = f(y_{t-1}, \ldots, y_{t-p}, w) + \epsilon_t,$$

where w is the vector of all parameters and f is the function that is determined by the network structure and connection weights. Hence the neural network essentially is a nonlinear autoregressive model.

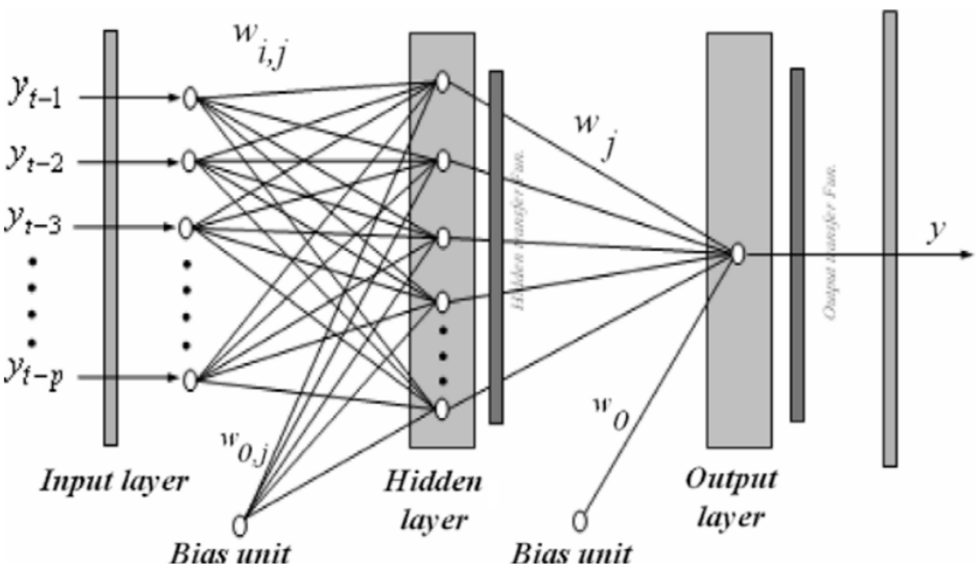

Figure 4: The neural network structure from Khashei and Bijari (2010) on page 481.

Hornik et al. (1990) find that the simple neural network as described in (2) is surprisingly powerful as it can approximate the arbitrary function if the number of hidden nodes q is sufficiently large. In practice it often turns out that simple network structures with a small number of hidden nodes work well out of sample. This could be due to an overfitting effect in the neural network modelling process. This overfitting effect implies that the model has a good fit for the sample used for building and training the model, but has a poor performance for predicting data out of the sample.

There is no systematic rule in determining the number of hidden nodes q as it is data dependent. Next to the choice of q, an important task of the ANN is to determine the number of lagged observations p. This parameter is very crucial as it plays a big role in determining the nonlinear autocorrelation structure of the time series. There are many different approaches in finding the optimal settings for a neural network, however these methods are often very complex in nature and hard to implement. Currently there is no clear-cut method that can provide an optimal solution for the determination of these parameters. That is why Khashei and Bijari (2010) test numerous networks with varying numbers of input and hidden nodes, and select the network based on the estimated generalization error for each.

Once a network is selected it is ready for training. Similar to the ARIMA model building, the parameters are estimated such that the error is minimized. This is done with efficient nonlinear optimization algorithms. To make the ANN a strong competitor for the other methods several optimization algorithms are used. These algorithms include the Levenberg-Marquardt backpropagation as specified by Ranganathan (2004) on page 101-110. Also the gradient descent optimization algorithm based on the steepest gradient, and finally BFGS quasi-Newton backpropagation as described by Nawi et al. (2006) on page 152-157 is used in this study. The model is then trained using a specific sample of the data, and later evaluated using a separate hold-out sample using the best performing optimization algorithm.

3.3 Zhang's hybrid model

The ARIMA and ANN models have both achieved success in their linear and nonlinear domain. However, complex problems such as predicting stock price returns often contain both linear and nonlinear structures. There is no universal model that is suitable for all situations. To overcome the linear and nonlinear patterns in time series Zhang (2003) propose a hybrid model that considers a time series to be composed of a linear autocorrelation structure and a nonlinear component. First the linear component is modeled by the ARIMA model, such that residuals will only contain the nonlinear relationship. Let ϵ_t denote the residual at time t for the linear model, then

$$\hat{L}_t = Y_t + \epsilon_t,$$

where $\hat{L}_t$ denotes the forecasted return and Y_t represents the actual return at time t. Now that the stock returns are estimated with the linear model, this error term remain containing the nonlinear relationship. In order to discover the nonlinear relationships, the residuals are modeled using ANNs. With n input nodes the ANN model for the residuals is:

$$\hat{\epsilon}_t = f(e_{t-1}, \ldots, e_{t-n}) + \eta_t,$$

where f is a nonlinear function determined by the neural network, e_t denotes the residuals of the linear model at time t and η_t is the random error. This hybrid model takes advantage of the unique strengths of the linear ARIMA modeling and nonlinear ANN modeling. Now the forecast of the ARIMA model at time t is denoted at $\hat{L}_t$ and the forecast for the error term of the ARIMA model estimated by the ANN is denoted by $\hat{\epsilon}_t$. Finally the forecast of the ARIMA is added together with the forecasted value for the error to get the forecasted return. This leads to the combined forecast for the estimated return $\hat{Y}_t$ of the form:

$$\hat{Y}_t = \hat{L}_t + \hat{\epsilon}_t.$$

This specific combination method can be an effective way to improve forecasting accuracy for complex problems that have both linear and nonlinear structures. The empirical results of Zhang (2003) clearly suggests that this is indeed the case. Following the literature, this paper regards this hybrid model as the best performing model thus far. Now this paper investigates if this also holds in predicting intraday stock prices, and whether it is possible to improve upon this hybrid model.

3.4 Adjusted hybrid model

The hybrid model proposed by Zhang (2003) has proven to be effective in making accurate forecasts using different types of data. Since then the framework has been adopted in many different academic papers as described in the introduction. However, the question remains whether the methodology is suitable to predict intraday stock prices or not. Moreover, is it possible to improve upon the current hybrid methodology in order to gain forecasting accuracy in predicting intraday stock prices? This paper aims to achieve this by adjusting the methods used in the hybrid methodology and evaluate the performance of the adjusted hybrid model in terms of forecasting accuracy.

The adjusted hybrid model takes a similar approach to Zhang (2003). Again, the time series is assumed to be composed of a linear autocorrelation structure and nonlinear component. First the linear component is modeled by the ARIMA model leaving the residuals containing the nonlinear relationship. Then to capture the nonlinear patterns in the residuals, the residuals are modeled by long short-term memory (LSTM) neural networks, which is an recurrent neural network (RNN) technique. A RNN has a recurrent connection on the hidden state ensuring that the sequential information and dependencies in the input data are captured. This makes it very useful in time series prediction as the model now remembers previous inputs as well. Fischer and Krauss (2018) find that LSTM neural networks are inherently suitable for sequence learning, and that they outperform memory-free classification methods in predicting stock price movements.

However, RNNs face the problem of vanishing gradients. The gradients vanish to zero during backpropagation because the derivative of the activation functions get too low. The LSTM model avoids this problem such that the error can flow backwards through several layers giving it the possibility to learn from memories of events that occurred several time steps earlier. In this way the model is more capable in handling long sequences of data compared to ANNs.

3.5 Long Short-Term Memory neural networks

The Long Short-Term Memory (LSTM) neural networks is a recurrent neural network (RNN). The RNN is skilled in dealing with sequential data, this is done by processing input sequences through its own internal state (memory). The LSTM adds a hidden layer to provide feedback on itself over time. All observations are assumed to be independent. Recurrent neural networks are trained by propagating the error backwards over the different time steps. However, when this is done over a long horizon the network faces the problem of vanishing gradients. The network is trained in such a way that the weights of the input are constantly updated to minimize the error. In the case of the vanishing gradient, the gradient gets too small such that the effect vanishes and the weights do not change in the future. To fix this problem the LSTM neural network was introduced by Hochreiter and Schmidhuber (1997). The structure of the LSTM neural network is slightly more complicated as there are memory cells in the hidden layer instead of nodes. These memory cells can selectively add or remove information from the cell state, and then communicate this information to the next cell. This makes the network very capable of dealing with long term market dependencies in data.

The LSTM neural network is built in several layers, the input layer, output layer and multiple hidden layers. In this network the input layer receives a several explanatory variables, this is equal to the amount of technical indicators used plus one for the historical price which is always used. The memory of the LSTM model is contained in the memory cells in the hidden layers. The memory cell contains three different gates which process the input. In order to describe this procedure this paper follows the notation used in Fischer and Krauss (2018), and also followed by Groenendijk, Smits and van Wijngaarden (2020). At every timestep t the gates receive an input x_t and a previous cell state h_{t-1}. First the input gate i_t processes the data and determines what information from the data is added to the cell state s_t. Next the forget gate f_t determines what information to remove from the cell state. Then finally the output gate o_t determines what information is used as output. This process is visualized in Figure 5.

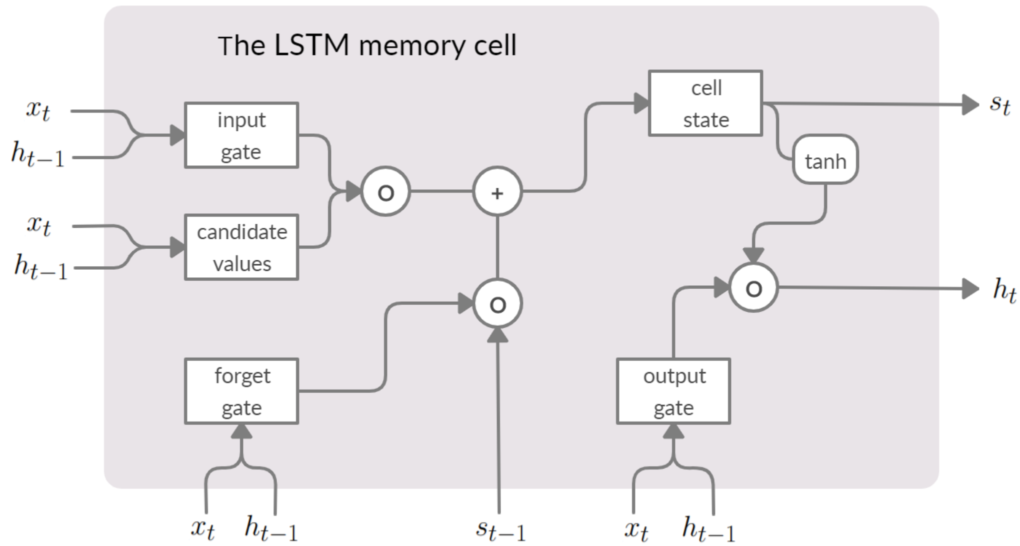

Figure 5: The LSTM memory cell from Groenendijk, Smits and van Wijngaarden (2020) page 12.

At every timestep t the memory cell starts the updating procedure. In order to describe this procedure this paper follows the notation of Fischer and Krauss (2018), which is also used in Groenendijk, Smits and van Wijngaarden (2020). The procedure starts with the input vector x_t. The activation values for the input gate, forget gate and output gate are denoted by i_t, f_t and o_t respectively. The cell state and candidate values that could potentially be added to the cell state are denoted by s_t and $\tilde{s}_t$. To adjust the cell state, weight matrices $W_{j,k}$ are produced for all the gates and candidate values such that $j = \{i, f, o, \tilde{s}\}$. The weight matrices are produced to adapt the input x_t and previous cell state h_{t-1} such that $k = \{x, h\}$. Each gate and candidate value also produces a bias vector denoted by $b_f, b_{\tilde{s}}, b_i$ and b_o. Finally the output for the LSTM layer is denoted by h_t.

Next up is the updating procedure of the LSTM layer at time t, at this stage the cell state s_t and output h_t are determined. First in this updating procedure it is determined what information is removed from the previous cell state h_{t-1}. This is done by the forget gate by taking the corresponding weight matrices $W_{f,x}$ and $W_{f,h}$ together with the current input x_t, previous cell state h_{t-1} and bias term b_f. These are then used to be scaled by the sigmoid function. This function is denoted by $\sigma(\cdot)$, and scales the values per element between zero and one. If the scaled value is close to zero this corresponds to completely forget, and if the value is close to one this corresponds to completely remember. Finally, you get the activation value f_t at time t:

$$f_t = \sigma(W_{f,x}x_t + W_{f,h}h_{t-1} + b_f).$$

Now it is the question of what information from the data should be added to the cell state s_t. The LSTM layer determines this by means of two operations. First the candidate values $\tilde{s}_t$ are computed, these values could be added to the cell state. This is done using the hyperbolic tangent function denoted as $\tanh(\cdot)$. Next the activation values i_t for the input gates are calculated using the sigmoid function by:

$$\tilde{s}_t = \tanh(W_{\tilde{s},x}x_t + W_{\tilde{s},h}h_{t-1} + b_{\tilde{s}}),$$
$$i_t = \sigma(W_{i,x}x_t + W_{i,h}h_{t-1} + b_i).$$

In the third step the LSTM layer determines the new cell states s_t using the calculated candidate values and activation values. The calculation of these cell states is done with the Hadamard product denoted by $\circ$, this product multiplies vectors per element.

$$s_t = f_t \circ s_{t-1} + i_t \circ \tilde{s}_t.$$

Finally, the output h_t of the memory cell is determined. This is done using the output gate o_t as:

$$o_t = \sigma(W_{o,x}x_t + W_{o,h}h_{t-1} + b_o),$$
$$h_t = o_t \circ \tanh(s_t).$$

For every timestep t the input sequence containing past returns and several technical indicators is sent to the LSTM neural network and adapted by these steps. For this research the network is programmed using the programming and numeric computing platform MATLAB. During the process of training the LSTM neural network the weights and bias terms are adapted in such a way the the mean squared error is minimized. For the minute predictions 80% of the data rounded to the nearest full day is used for training, and 20% for testing. For example the data for the minute prices of Microsoft contains 52 days with 390 minute observations per day. Here the first 16380 minutes (80,8%) are used for training, and the remaining 3900 minutes (19,2%) are used for testing. For the daily predictions the amount of observations is considerably less (4250 days). Here 90% of the data (3825 days) is used for training, and 10% (125 days) is used for testing. The models are then trained using the training data and different combinations of the technical indicators with a learning rate of 0.005 and an epoch of 500. Then select the technical indicators with the best performance in the training sample are selected based on a grid-search between the different combinations, to use for the testing sample.

In order to train the neural network, the networks uses the stochastic gradient-based optimization algorithm

Adam (Kingma and Ba (2015)). This optimization algorithm adapts in such a way that the mean squared error is minimized. Hence, the difference between the predicted return and the actual return is as small as possible. The difference between this stochastic gradient descent and the ordinary gradient descent is that in the stochastic case the gradient is replaced by an estimate of the gradient. The first and second moment of the gradient are initialised with zeros. Then iteratively the first and second moments are estimated by taking linear combinations of the data. Because of the initialisation in zero, there is a slight bias in the estimate. Therefore a bias correction term is introduced. To gain more insight in the algorithm, the pseudo-code developed by Kingma and Ba (2015) for this algorithm is included in Appendix A.

3.6 Forecast combinations

The hybrid methodology supposes that the residuals from the linear model only contains the nonlinear relationship. However, Taskaya-Temizel and Casey (2005) find that residuals of the linear component do not necessarily contain the non-linear relationship. Therefore different forecast combinations are evaluated putting weights on the individual forecasts of the models separately. If these simple forecast combinations outperform the hybrid methodology in predicting accuracy then there is no need for the sophisticated hybrid methods. Two different types of combinations are evaluated, time-varying minimum variance weights and fixed equal weights.

3.6.1 Time-varying minimum variance weights

In order to combine different forecasts without using the hybrid methodology, one could implement time-varying minimum variance weights on the forecasts. Timmermann (2006) describes an optimal weighting scheme minimizing the variance of the forecast error. This weighting scheme and notation has been adopted from my earlier work in Groenendijk (2019). However previously I used the weighting scheme in an expanding window, now I use the weighting scheme using a rolling window approach. In this way the weights assign a larger weight to models which produce forecasts with a low variance in the recent error. For this paper there are two models that produce two different forecasts $\hat{Y}_{1,t}$ and $\hat{Y}_{2,t}$. It is assumed that the individual forecasts are unbiased. Because the forecasts are unbiased the expected value is $E[Y_t] = \hat{Y}_{i,t}$ such that $E[e_{i,t}] = 0$. The forecast errors corresponding to the two models are denoted as $e_{1,t}$ and $e_{2,t}$, with variance σ_i^2 for $i = 1, 2$ and covariance $\sigma_{12} = \rho_{12}\sigma_1\sigma_2$. The combined forecast is then the weighted sum of the two forecast, this is mathematically denoted as:

$$\hat{Y}_t^{cb} = \omega\hat{Y}_{1,t} + (1-\omega)\hat{Y}_{2,t},$$

with forecast weight ω. This combined forecast is different from the hybrid models in section 3.3 and section 3.4 in the sense that here both the ARIMA, ANN and LSTM directly forecast the return. These forecasts are then added together with a specific weight for each forecast. While in the hybrid methodology the ARIMA directly forecasts the return, and the ANN and LSTM both forecast the ARIMA error. These forecasts are then added together without weights aiming for a more accurate forecast of the return. The time-varying minimum variance combinations aims to achieve a combined forecast with a minimal error of the combined forecast $e_t^{cb} = Y_t - \hat{Y}_t^{cb}$. The variance of this forecast error is calculated as:

$$\begin{aligned} e_t^{cb} &= Y_t - \omega \hat{Y}_{1,t} - (1-\omega)\hat{Y}_{2,t} \\ &= \omega e_{1,t} + (1-\omega) e_{2,t} \\ \sigma_{cb}^2(\omega) &= \omega^2 \sigma_1^2 + (1-\omega)^2 \sigma_2^2 + 2\omega(1-\omega)\sigma_{12}. \end{aligned}$$

Next to get a lower MSE for the combined forecast, the variance of the combined forecast σ_{cb}^2 is minimized with respect to ω. In every timestep t the weights are determined using a rolling window approach. Finally after minimizing the variance this leads to the weights:

$$\omega^* = \frac{\sigma_2^2 - \sigma_{12}}{\sigma_1^2 + \sigma_2^2 - 2\sigma_{12}} \text{ and } 1 - \omega^* = \frac{\sigma_1^2 - \sigma_{12}}{\sigma_1^2 + \sigma_2^2 - 2\sigma_{12}}.$$

This weighting scheme ensures that the model with least variance in the recent forecasting error is assigned a greater weight. If a forecast gets a negative weight this can also be valuable information by going short in a forecast in order to reduce the error. In the research of Engle et al. (1984) they combine bivariate ARCH models using time-varying weights to minimize the forecast error variance. They find that this combination is improves in performance relative to the models individually. However, the minimum variance combination does not necessarily outperform the simple equally weighted combination.

Finally, the question remains of what value v for the length of the rolling window is optimal. Studies including the work of Newbold and Granger (1974) considered values of the length v between one and twelve periods. They find that the rolling window approach generally performs best for the longest windows $v = 9$ and $v = 12$. However, more recent research of Clark and McCracken (2009) find that a rolling window of $v = 40$ works best. As this value is heavily dependent on the data that is being used. I perform a grid search over different rolling window sizes ranging between one and sixty using in-sample predictions. The rolling window of size $v = 40$ has the best predicting accuracy, hence this value is used for predicting out of sample.

3.6.2 Fixed equal weights

Just like Groenendijk (2019), the second weighting scheme this paper considers is the simple combination of fixed equal weights on the predictions of the linear and nonlinear models. This is also known as the $\frac{1}{N}$ combination, and seems like a naive allocation. However, DeMiguel et al. (2009) among others have proven that this allocation can perform really well empirically. This paper considers the combination of the linear ARIMA model with either the nonlinear ANN or LSTM. This leads to the combination of the different forecasts with equal weights of $\omega = 0.5$. Hence, the combined forecasts is:

$$\hat{Y}_t^{cb} = \frac{1}{2}\hat{Y}_{1,t} + (1 - \frac{1}{2})\hat{Y}_{2,t}.$$

This combined forecast has a error variance of:

$$\sigma_{ew}^2 = \frac{1}{4}\sigma_1^2 + \frac{1}{4}\sigma_2^2 + \frac{1}{2}\rho_{12}\sigma_1\sigma_2.$$

When you analyze this variance relative to that of the minimum variance combined forecast you get:

$$\frac{\sigma_{ew}^2}{\sigma_{cb}^2(\omega^*)} = \left(\frac{(\sigma_1^2 + \sigma_2^2)^2 - 4\sigma_{12}^2}{4\sigma_1^2\sigma_2^2(1 - \rho_{12}^2)}\right).$$

Timmermann (2006) show that the variance of the error of the equal weighted combination is smaller than the minimum variance combination if and only if $\sigma_1 = \sigma_2$. Armstrong (2001) use this specific combination to compute equally weighted forecasts. He finds that in 30 empirical comparisons the error reduces significantly.

3.7 Forecast Evaluation

The performance of the different models, hybrid models and combined models can be evaluated by means of the root mean-squared error (RMSE) and the hit rate. Chai and Draxler (2014) provide a discussion between two widely used metrics in the literature, the RMSE and Mean Absolute Error (MAE). They find that the MAE can be more beneficial for estimation procedures. However, when comparing different model performances it is beneficiary to punish the large errors more. This allows for a better distinction between the models. Hence, the RMSE is adequate for evaluating the relative forecasting performance between the models.

Next to the RMSE this paper evaluates the models by means of the hit rate. This is calculated as the fraction of times the model correctly predicted a price increase or decrease relative to the previous price. This performance measure gives an indication if the model is capturing price movement accurately. If a model always predicts close to the price in the previous timestep, it will most likely be very close to the future price. Hence the prediction error will be reasonably low. However, the predictions will not capture the movement of the price very well unless there is a high amount of momentum in the stock price. Therefore the hit rate will provide a good insight whether the model captures the price movement, and whether it can be profitable in practice. Finally the trading strategies are evaluated by the mean and the cumulative return. The mean is defined as the average generated return per trade. The cumulative return represents the generated return over the entire horizon.

3.8 Trading implementation

Now to get a better insight in the practical relevance of the different models, several trading strategies are constructed using the forecasts of the different models. In order to make the trading strategies feasible I need to make several assumptions explained in subsection 3.8.1. Next subsection 3.8.2 elaborates on the trading strategies based on the models. These trading strategies are kept simple as they do not aim to maximize profit, but are there provide insight in the practical relevance of the models relative to each other. Does a low root mean squared error actually result in a profitable strategy?

3.8.1 Assumptions and Setup

The models and trading algorithms are tested using simulation. In order to achieve this some assumptions are required. These are the most basic assumptions to make the trading strategies work in a simulation setting, and are similar to the ones used in Groenendijk, Smits and van Wijngaarden (2020). This paper makes three basic assumptions to restrict the simulation environment as little as possible.

1. **Closing price execution**
   It is assumed that you can buy and sell at the closing price of the current minute or day, at that point in time.
2. **No impact on the market**
   The order of buying or selling a stock does not have an impact on the market in a way that it affects the future price.
3. **No transaction costs**
   There are no transaction costs directly related to buying or selling.

The first assumption ensures the trade can be made at the closing price of the current interval. This allows the model to trade using simulation and compare the performance of the different models and trading strategies. As this price is actually traded at the closing of the interval, this is very close to practice. Next I assume the trade made no direct impact on the current market price. This means that the trade did not push the price up or down. Since the trading strategies are buying and selling very liquid financial instruments, it is highly unlikely that the trade would impact the current price. Finally it is assumed that the trade does not incur transaction costs. Transaction costs are very subjective to the type of trader. Individual investors might incur very high transaction costs, while high frequency trading firms might incur no transaction costs at all. This assumption can always be relaxed later on.

3.8.2 Trading strategies

In order to gain more insight in the practical relevance of these models I impose several trading strategies. The purpose of these trading strategies is to demonstrate whether the models could be used to develop profitable trading strategies in practice. The first strategy is simple, buy one unit of the stock when the predicted stock price is higher than the current stock price with a holding time of one interval period, and sell one unit if the predicted price is lower. Then to put the performance of this trading strategy in perspective, several benchmark strategies can be deployed. These benchmark strategies include the always long, always short, and momentum strategy. The always long and short strategy buys and sells at every timestep respectively. The momentum strategy buys if the price went up in the current time step with respect to the previous timestep, and sells if the price went down.

4 Results

In this section the results obtained from the models described in Section 3 are discussed. This section is split into three parts. The first part in section 4.1 contains the prediction results of the individual models, combined models and the hybrid models and their relative performance. The second part in section 4.2 consists of the trading performance of the strategies based on the models and the benchmark strategies. Finally the last part in section 4.3 includes a robustness check to verify if the trading and prediction results are robust to changes in the data or not.

4.1 Model performance

In this study, all neural networks are implemented using the same set of technical indicators described in Section 2. All predictions are one-step-ahead forecasts. The root mean squared error (RMSE) and hit rate are selected as forecasting accuracy measures.

Table 3: Model performance

		Minute prices		Daily prices	
Stock	Model	Hit rate	RMSE	Hit rate	RMSE
MSFT	ARIMA	0.492	7.522	0.575	0.0163
	ANN	0.492	6.988	0.538	0.0157
	LSTM	0.487	9.364	0.550	0.0154
	Hybrid	**0.509**	7.008	0.519	**0.0151**
	Adjusted Hybrid	0.505	**6.732**	**0.568**	0.0155
	Equal weighted[1]	0.503	7.070	0.535	0.0155
	Minimum variance[1]	0.507	9.181	0.564	0.0160
	Equal weighted[2]	0.478	7.606	0.547	0.0154
	Minimum variance[2]	0.498	8.293	0.538	0.0164
AMZN	ARIMA	0.501	8.284	0.535	0.0196
	ANN	**0.527**	9.474	**0.552**	0.0204
	LSTM	0.497	9.138	0.538	0.0205
	Hybrid	0.501	**7.951**	0.512	0.0198
	Adjusted Hybrid	0.496	8.092	0.521	**0.0195**
	Equal weighted[1]	0.509	8.364	0.533	0.0197
	Minimum variance[1]	0.503	9.925	0.517	0.0200
	Equal weighted[2]	0.498	8.303	0.524	0.0197
	Minimum variance[2]	0.497	9.989	0.526	0.0207
S&P500	ARIMA	0.514	3.542	**0.542**	0.0093
	ANN	0.509	3.336	0.509	0.0097
	LSTM	0.497	4.921	0.502	0.0094
	Hybrid	0.505	3.285	0.495	**0.0093**
	Adjusted Hybrid	**0.516**	**3.038**	0.521	0.0094
	Equal weighted[1]	0.514	3.386	0.519	0.0095
	Minimum variance[1]	0.497	4.833	0.521	0.0101
	Equal weighted[2]	0.507	3.801	0.524	0.0094
	Minimum variance[2]	0.493	6.489	0.528	0.0097
NASDAQ	ARIMA	0.524	4.854	0.479	0.0186
	ANN	**0.525**	4.525	0.507	0.0148
	LSTM	0.513	5.239	0.515	0.0155
	Hybrid	0.523	4.249	**0.526**	0.0146
	Adjusted Hybrid	0.515	**4.016**	0.505	**0.0133**
	Equal weighted[1]	0.523	4.604	0.460	0.0149
	Minimum variance[1]	0.505	7.166	0.486	0.0148
	Equal weighted[2]	0.521	4.759	0.481	0.0144
	Minimum variance[2]	0.497	6.529	0.502	0.0136

Note: all values for the RMSE on the minute interval should be multiplied by 10^{-4}. The combined model with superscript [1] is a combination of the ARIMA and ANN, the combined model with superscript [2] is a combination of the ARIMA and LSTM.

Table 3 gives the forecasting results for the different stocks on the interval of one minute and daily prices using the different models. The hybrid model refers to the original hybrid model of Zhang (2003) that combines the ARIMA and ANN. Following the literature this paper considers this model as the best performing hybrid model yet in terms of predicting accuracy. The adjusted hybrid model is the proposed hybrid methodology combining the ARIMA and LSTM. The equal weighted and minimum variance combinations are simply combining the forecasts of the ARIMA with either the ANN or LSTM as described in Section 3.6. For reading convenience, the best performing results of each category are in bold.

The results for the Microsoft stock using minute observations show that model which achieves the best hit rate only produces a hit rate of 50.9% which is just slightly higher than the expected 50%. In the data section it is observed that the directional movements of the minute prices are very bouncy, upward and downward movements alternate very frequently. This movement has proven to be very hard to capture. Also, the models aim to minimize the mean squared error and not optimize the hit rate as it is not a classifier. The results for the model errors show that the adjusted hybrid model is the most accurate as it has the lowest RMSE. It achieves a RMSE reduction of 3.7% over the ANN which is the second best performing model in this category. The results for daily prices display much higher values for the hit rate. This is expected as in the sample the daily prices contain much more momentum as the price rose a lot over the past years. Here the model with the best forecasting accuracy is the original hybrid model with a RMSE reduction of 2.0% over the LSTM model. The results values for the RMSE are much closer relative to the results for the minute prices as the test sample size is also much smaller (3900 minute observations vs. 425 daily observations).

The outcome for the Amazon stock show a similar pattern. Again the hit rate for the minute observations is around 50%. Only the ANN achieves a slightly higher hit rate of 52.7%. In terms of RMSE the original hybrid model produces the most accurate forecasts. Although the hybrid model is closely followed by the adjusted hybrid model, the hybrid models do improve upon the models individually. The hit rate on the interval of daily observations is again generally higher than 50%, with the ANN achieving a hit rate of 55.2%. In terms of predicting accuracy the adjusted hybrid model produces the most accurate forecasts by a very thin margin. The results are again very close, hence you can not pick a clear 'best' model.

For the market indices the results are again comparable. The obtained values for the hit rate using minute observations are around 50% and somewhat above that value for the NASDAQ. This could be related to the fact that the minute prices of the market indices are less volatile relative to the stock prices, as they are dependent on many more factors. For the S&P500 and the NASDAQ the adjusted hybrid and ANN produced the highest hit rate with values of 51.6% and 52.5% respectively. Regarding the prediction accuracy the adjusted hybrid model produced the most accurate forecasts for both market indices. For the S%P 500 the adjusted hybrid model reached a error reduction of 7.5% over the original hybrid model and for the NASDAQ it reduced the error by 5.5% over the original hybrid model. The daily results again produce somewhat higher hit rates of 54.2% and 52.6%. The most accurate models in terms of prediction error are the hybrid and adjusted hybrid models for the S&P 500 and the NASDAQ respectively. However, the differences between the best and second best performing models are again very small.

Overall the results for the minute intervals in Table 3 show that the adjusted hybrid produces the most accurate forecasts in three out of four cases using minute intervals, with the adjusted hybrid being a very close second in the case it is not the best. The daily results show that the hybrid and adjusted hybrid both produce the most accurate forecasts in two out of four cases.

Regarding the individual model performances, the LSTM generally produces a relatively higher RMSE over the ARIMA and ANN. Figure 6 illustrates the prediction error over time of the individual models for the Microsoft stock using minute intervals. The trading days are separated in the graph by the vertical dashed lines. Each trading day consists of 390 minutes, as mentioned in the data section the first seven minutes are used as a burn-in period to account for the overnight price changes. Hence, in the graph 383 minutes remain each trading day. The ARIMA and ANN model rarely show outliers in the error. However, it can be seen that at the start of each trading day the LSTM is considerably less accurate than the ARIMA and ANN models. This could be due to the memory of the LSTM model. As the model predictions are very inaccurate during the first seven minutes of the trading day, the LSTM remembers this as the error is backpropagated across many timesteps. Hence, this error is carried forward for a longer period of time relative to the ARIMA and ANN. This pattern of errors across time repeats over all the different stocks.

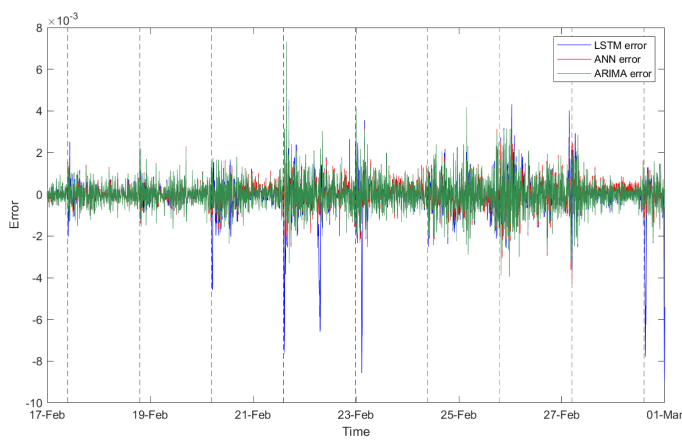

Figure 6: Prediction errors of the individual models for Microsoft using minute intervals.

Figure 7 visualizes the actual return and the predicted return of the NASDAQ using minute observations for the individual models (left) and the hybrid models (right). The figure is split into two graphs to improve visibility. The figure shows that the LSTM produces pretty noisy forecasts, this is in line with the findings above. However, this effect fades as the adjusted hybrid is observed which consists of the ARIMA and LSTM combined. Here the returns are predicted using ARIMA in the first step, the ARIMA makes use of up to seven lags. In the second step the ARIMA error terms are estimated using the LSTM. Hence, the hybrid model does not encounter a lot of outliers in the error terms. This leads to more conservative predictions valuing the long term memory. The ARIMA produces more noisy forecasts relative to the ANN. This leads to the effect that the model is either very right or very wrong. In the RMSE large errors are punished more severely, hence the RMSE of the ARIMA is also larger. Looking at the hybrid models the adjusted hybrid model produces slightly more conservative forecasts over the original hybrid model. This also contributes to a lower RMSE.

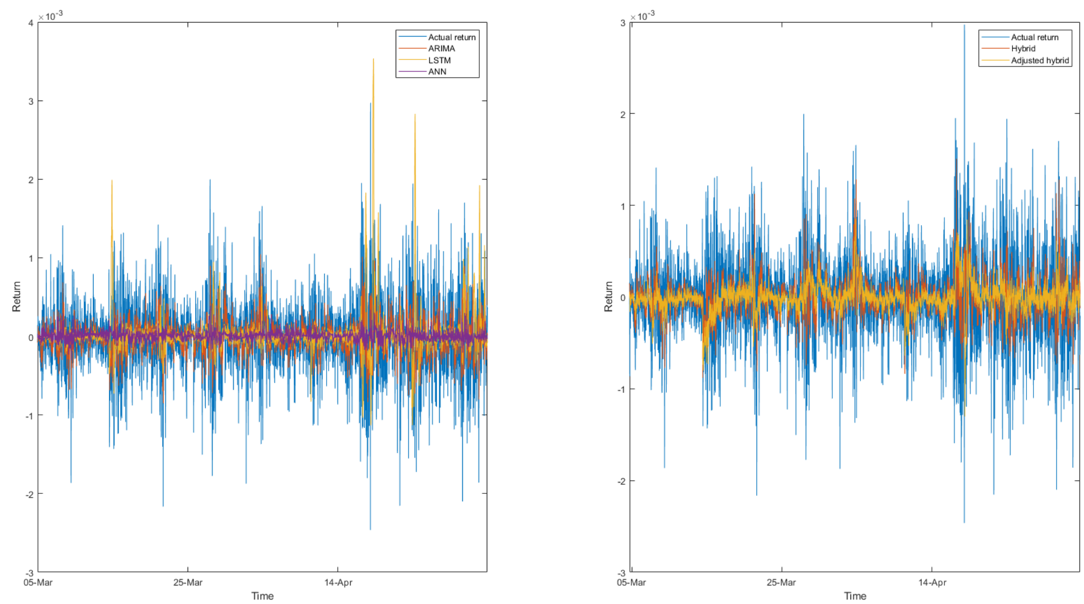

Figure 7: Predicted returns for the individual models and hybrid models.

Generally the equal weighted and minimum variance weighted model combination do not outperform the models individually. The equally weighted combination often gets close to the individual models in terms of RMSE. However, the minimum variance combination often does not. This could be due to the minimum variance combination putting too much weight on forecasts that recently performed well, but need not to be the best in the near future. Due to the volatile nature of the data, the volatility in the model errors is also very high. This is reflected in the weights that are attached to the different models. The weights combining the ARIMA forecasts with the LSTM forecasts for the S&P 500 using minute observations are visualized in Figure 8. Both the weights are visualized to show that they indeed sum to one. Here it can be seen that the weights take on large positive and negative values, leading to unstable model combinations. This could potentially lead to less accurate forecast combinations.

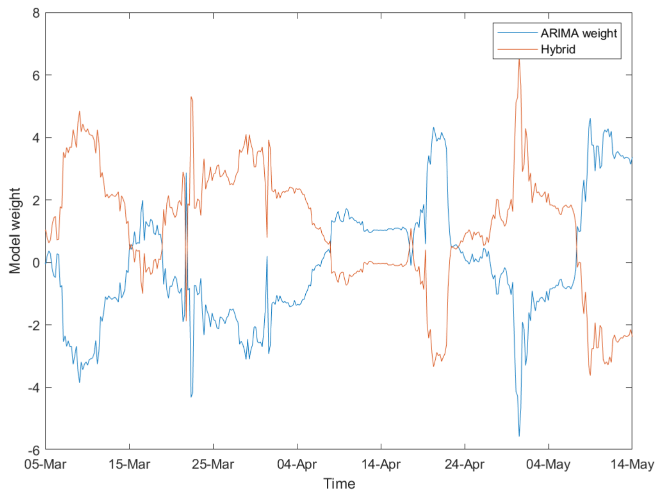

Figure 8: Model weights combining the ARIMA and LSTM forecasts

4.2 Trading performance

For the trading performance I evaluate several trading strategies based on the models and benchmark strategies. The strategies based on the models buys one unit of stock if the predicted price is higher than the current price (the return is positive), and sells the stock in the next period. If the predicted price is lower than the current price (the return is negative), it sells one unit of stock now and buys it back in the next period. The benchmark strategies include always long, always short and momentum. The always long and short buy and sell at every timestep respectively. The momentum strategy buys if the price went up in the current timestep relative to the previous, and sells if it went down. The results are shown in Table 4. Again for reading convenience, the best performing results of each category are in bold.

Table 4: Trading performance

		Minute prices		Daily prices			Minute prices		Daily prices	
Stock	Model	Mean	Cum	Mean	Cum	Index	Mean	Cum	Mean	Cum
MSFT	ARIMA	-1.03	-4.0	0.15	62.7	S&P500	1.18	5.1	0.04	16.9
	ANN	0.09	0.3	0.16	69.0		0.80	3.5	-0.01	-4.6
	LSTM	-0.34	-1.3	0.14	59.7		-7.10	-3.1	0.02	8.0
	Hybrid	-0.56	-2.1	0.16	67.6		0.55	2.3	0.02	10.6
	Adjusted Hybrid	0.74	2.8	**0.17**	**74.0**		0.50	2.1	**0.08**	**32.6**
	Equal weighted[1]	-0.03	-0.1	0.14	59.5		**1.27**	**5.5**	0.00	-1.3
	Minimum variance[1]	**1.26**	**4.8**	0.11	47.1		-0.09	-0.4	0.00	1.4
	Equal weighted[2]	-1.62	-6.2	0.14	57.7		0.49	2.1	0.01	4.7
	Minimum variance[2]	-0.66	-2.5	0.07	29.5		-1.16	-5.0	0.00	0.7
	Always long	0.64	2.4	0.12	52.3		-0.10	-0.4	0.03	14.6
	Always short	-0.64	-2.4	-0.12	-52.3		0.10	0.4	-0.03	-14.6
	Momentum	-1.91	-7.3	-0.13	-53.7		0.44	1.9	0.01	3.2
AMZN	ARIMA	-0.60	-2.3	0.04	15.6	NASDAQ	**3.24**	**13.7**	0.00	-1.9
	ANN	**3.46**	**13.3**	0.08	35.1		2.72	11.5	0.01	5.8
	LSTM	1.79	6.9	**0.14**	**58.2**		2.07	8.7	**0.17**	**74.0**
	Hybrid	-0.66	-2.5	0.01	3.8		2.38	9.9	0.04	18.3
	Adjusted Hybrid	-1.66	-6.2	0.06	26.1		2.23	9.2	0.05	20.8
	Equal weighted[1]	0.67	2.6	0.07	30.7		3.17	13.4	-0.07	-27.7
	Minimum variance[1]	0.16	0.6	0.03	14.3		0.56	2.4	-0.05	-20.6
	Equal weighted[2]	0.22	0.8	0.13	56.1		2.67	11.3	-0.02	-7.3
	Minimum variance[2]	0.07	0.3	0.03	11.6		-0.23	-1.0	0.03	14.2
	Always long	-0.46	-1.8	0.04	17.8		-0.56	-2.3	0.05	20.2
	Always short	0.46	1.8	-0.04	-17.8		0.56	2.3	-0.05	-20.2
	Momentum	-2.88	-11.0	0.08	34.0		2.99	12.6	-0.05	-19.4

Note: all values for the mean on the minute interval should be multiplied by 10^{-3}. The combined model with superscript 1 is a combination of the ARIMA and ANN, the combined model with superscript 2 is a combination of the ARIMA and LSTM.

The results for the stocks using minute observations show that majority of the trading strategies produce negative cumulative returns. In the data section it is observed that the directional movements of the minute prices are very bouncy. Upward and downward movements alternate very frequently. This movement has proven to be very hard to capture. Therefore the hit rate is close to 50% for most models, this is again reflected here. A noticeable result is that despite the hybrid model being the performing model in terms of hit rate for Microsoft using minute observations, it still produces a negative return. While the adjusted hybrid model produces a slightly positive return. This could be due to the hybrid model predicting the majority of the small movements correctly, but predicting some relatively larger movements in the wrong direction resulting in a negative cumulative return. On the contrary, the results using daily observations show only positive returns for all the trading strategies based on the models. The daily observations are far less volatile and the prices have experienced experienced major price increases. This gives it the potential to be very profitable, resulting in cumulative gains up to 74% and 58.2% for Microsoft and Amazon respectively.

The empirical results for the market indices using minute observations show that there is profitability. All the models produce profitable trading strategies with the exception of the LSTM and the minimum variance combinations. This is in line with the findings above as the LSTM and the minimum variance combinations are less accurate relative to the other models. As the prices rely on many more factors relative to the stocks, the price movements are less volatile. This is reflected in the results as the models are better able to capture the movements, and achieve cumulative gains of 5.7% and 13.7% for the S&P 500 and the NASDAQ respectively. The results using daily observations show again mainly positive returns for the individual and hybrid models. The combined models and the benchmark strategies often produce negative returns.

Overall is there no trading strategy that consistently outperforms the other. This could be due to the fact that the models aim to minimize the mean squared error. This results in more accurate forecasts, but does not necessarily lead capturing the directional movement of the price more accurately. The trading strategies based on the model predictions do show favourable returns over the benchmark strategies. The results using the minute observations show that for the stocks it is very hard to be consistently profitable, however for the market indices the results are much more promising. For the daily observations the results show more potential, leading to consistently profitable strategies based on the individual and hybrid models.

4.3 Robustness

Finally I include a robustness check to verify whether the same results hold using a different set of data. For the stock price predictions data of the stock price of Apple indicated with ticker AAPL is used. Apple is again a very large-cap stock and a leader in the technology sector just like Microsoft. For the market index data of the Dow Jones Industrial Average with ticker DJI is used. The Dow Jones is a stock market index just like the NASDAQ and the S&P 500, it is comprised of 30 prominent companies listed on stock exchanges in the United States. The minute data for Apple dates from the 10th of May 2021 until the 21st of July 2021. The minute data for the Dow Jones dates from the 6th of July 2021 until the 21st of July 2021. The daily data for both the stock and market index date from the 1st of January 2003 until the 21st of July 2021. The models and trading strategies follow the exact same procedure as mentioned before. The results are show in Table 5. Again for reading convenience, the best performing results of each category are in bold.

Table 5: Results for the robustness check

		Minute prices		Daily prices		Minute prices		Daily prices	
Stock	Model	Hit rate	RMSE	Hit rate	RMSE	Mean	Cum	Mean	Cum
AAPL	ARIMA	0.504	6.262	0.537	0.0243	0.49	2.1	0.20	92.7
	ANN	**0.528**	5.934	0.529	0.0256	0.08	0.3	0.18	83.5
	LSTM	0.508	10.128	0.522	0.0239	1.73	7.3	0.11	53.6
	Hybrid	0.523	5.904	0.501	0.0240	**3.23**	**13.3**	0.17	78.9
	Adjusted Hybrid	0.514	**5.748**	0.531	**0.0237**	1.61	6.7	0.17	79.2
	Equal weighted[1]	0.507	6.031	0.526	0.0245	1.39	5.8	0.18	82.8
	Minimum variance[1]	0.494	11.241	0.531	0.0256	-0.21	-0.9	0.14	62.3
	Equal weighted[2]	0.512	7.107	**0.573**	0.0240	2.12	8.9	**0.43**	**197.4**
	Minimum variance[2]	0.509	10.898	0.564	0.0250	0.77	3.3	0.32	145.2
	Always long	-	-	-	-	0.31	1.3	0.23	106.6
	Always short	-	-	-	-	-0.31	-1.3	-0.23	-106.6
	Momentum	-	-	-	-	0.92	3.9	-0.23	-106.9
DOW	ARIMA	0.502	2.931	0.563	0.0169	-0.10	-0.3	0.07	32.2
	ANN	0.494	2.716	0.527	0.0189	-0.37	-1.0	**0.12**	**58.3**
	LSTM	**0.508**	3.401	0.576	0.0169	-0.32	-0.9	0.12	56.9
	Hybrid	0.503	2.691	0.507	**0.0163**	0.12	0.3	0.05	21.2
	Adjusted Hybrid	0.475	**2.512**	0.527	0.0167	-0.75	-2.0	0.03	13.2
	Equal weighted[1]	0.480	2.962	0.561	0.0167	-0.85	-2.3	0.12	55.4
	Minimum variance[1]	0.501	3.217	0.524	0.0203	-0.59	-1.6	0.12	53.3
	Equal weighted[2]	0.485	2.934	**0.579**	0.0168	-0.68	-1.8	0.12	54.2
	Minimum variance[2]	0.502	3.388	0.524	0.0190	-0.44	-1.2	0.06	28.9
	Always long	-	-	-	-	-0.81	-2.2	0.07	32.2
	Always short	-	-	-	-	**0.81**	**2.2**	-0.07	-32.2
	Momentum	-	-	-	-	-0.12	-0.3	-0.04	-19.8

Note: all values for the RMSE on the minute interval should be multiplied by 10^{-4}. All values for the mean on the minute interval should be multiplied by 10^{-3}. The combined model with superscript [1] is a combination of the ARIMA and ANN, the combined model with superscript [2] is a combination of the ARIMA and LSTM.

The outcome shows similar results for the model accuracy in terms of RMSE. Again the adjusted hybrid model is the best performing model using the minute observations in two out of two cases. Regarding the individual models the LSTM again underperforms relative to the ARIMA and ANN. For the daily observations the hybrid and adjusted hybrid produces the most accurate forecasts in one out of two cases. However, the values for the RMSE using daily observations are very close. The obtained hit rates are relatively higher using the daily observations, with a favourable outcome for the equal weighted model combination of the ARIMA and LSTM. There is no model that consistently outperforms the rest in terms of hit rate so this is not very surprising.

Now for the trading performance the results for the minute price of the Apple stock actually shows a favourable outcome over the market index of the Dow Jones. Looking at the data the stock price of Apple increased very steadily over the horizon, while the Dow Jones suffered some of the biggest drops of the year as the fear for the COVID delta variant hit hard. The results for the daily results are again more promising with all positive returns for the strategies based on the individual, hybrid and combined models.

5 Conclusion

This paper considers three different models to make intraday and daily stock return predictions. These models include the ARIMA, ANN and LSTM. Next I include the hybrid model of Zhang (2003) combining the ARIMA and ANN making use of the unique strength of the different models. Then, in an attempt to improve upon the hybrid model of Zhang (2003) the adjusted hybrid model is introduced combining the ARIMA and LSTM. In order to put the performance of the hybrid models in perspective several methods combining the forecasts of the individual models are evaluated. To gain an insight in the practical relevance of the models several trading strategies based on the model predictions are constructed. Finally a robustness check evaluating the performance of the models for the stock data of Apple and market index data of the Dow Jones is included.

Regarding the model performance results of Table 3 there are several conclusions. First off it is very hard to predict in which way the stock is going to move on a scale of minutes. Therefore the obtained values for the hit rate are mostly around 50%. This task becomes less difficult when less noisy data is observed. This generally is the case when the daily data is observed, here the hit rate is mostly above 50%. Concerning the prediction accuracy, the LSTM is not very accurate in predicting the return using minute observations. This is mainly due to the nature of the data as the market leaves a 'gap' between market closing and opening with very large price changes overnight. The LSTM often produces inaccurate forecast for the first minutes of the day leading to large errors. Due to the memory property of the model these errors are carried forward and cause large fluctuations in the upcoming predictions. This issue is eased when the hybrid models are observed.

The hybrid models take advantage of the unique strength of the linear and nonlinear models. Complex problems such as predicting stock price returns can contain both linear and nonlinear structures. Combining the methods using the hybrid methodology can be an effective way to improve upon the forecasting performance. The empirical results for the intraday return predictions suggest that this is indeed the case, as the hybrid models outperform the individual models. The two hybrid models both fit the ARIMA model first to the data. Followed by the fitting of the neural networks to the error terms of the ARIMA model. This helps ease the overfitting problem that is often related to the neural networks. The hybrid models differ in the sense that the original hybrid model of Zhang (2003) makes use of the ANN, while the adjusted hybrid model makes use of the LSTM. The empirical results suggest that switching to an LSTM is a sensible thing to do, as the adjusted hybrid model outperforms the original hybrid model in the majority of the cases.

For the daily observations there is no clear 'best' model using this specific data set. The daily data set is divided in a training sample (90%) and a test sample (10%), due to the few observations in the test sample the results for the different models are very close. However, there is exponential growth in the global market in the most recent years. This can be seen in Figure 1 in the data section 2. Due to these price movements it is of interest for the neural networks to contain as much as possible of these type of movements in the training sample in order to learn from these patterns. Price movements of ten years ago are most likely not very representative for the price movements of today. Especially after the start of the COVID-19 pandemic in March 2020, the price changes have become very volatile.

Regarding the trading performance, there is no trading strategy that consistently outperforms the others. This could be due to the fact that the models aim to minimize the mean squared error. This results in more accurate forecasts, but does not necessarily lead capturing the directional movement of the price more accurately. In terms of profitability, the stock price movements have proven to be very hard to capture on the minute interval. Hence, the trading strategies are often not profitable on the scale of minutes. The daily observations in general contain less volatility and do produce profitable strategies in most cases for the individual and hybrid models.

For future research there are many options. First off, to improve the LSTM model more hyperparameter combinations and features could be considered to improve the prediction accuracy. Next, in order to better capture the volatility in the data, an ensemble method of LSTM networks could be considered. Furthermore, different nonlinear methods could be considered for the hybrid methodology other than the LSTM or ANN to capture the nonlinear part in the data.

In order to get a better insight in the profitability of the models, for future research I suggest converting the models to classification models. In this way the models can be trained to capture the movement of the price whether it will go up or down in the next period. Also to get closer to practice, the data should be contain more order book data including the best bid and ask prices. This data can ensure that you can actually buy or sell the stock for that price at the given time. This could relax the trading assumptions, creating a more realistic simulation setting. This data can also be used to construct more insightful technical indicators such as the price imbalance. Furthermore, more advanced trading strategies could be considered to test the full potential of the models and maximize profitability.

References

Armstrong, J. S. (2001), Combining forecasts, *in* 'Principles of forecasting', Springer, pp. 417–439.

Box, G. and Jenkins, G. (1970), Control, *Halden-Day, San Francisco* .

Chai, T. and Draxler, R. R. (2014), Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature, *Geoscientific model development* 7(3), 1247–1250.

Chang, Z., Zhang, Y. and Chen, W. (2018), Effective adam-optimized lstm neural network for electricity price forecasting, *in* '2018 IEEE 9th international conference on software engineering and service science (ICSESS)', IEEE, pp. 245–248.

Clark, T. E. and McCracken, M. W. (2009), Improving forecast accuracy by combining recursive and rolling forecasts, *International Economic Review* 50(2), 363–395.

De Gooijer, J. G. and Kumar, K. (1992), Some recent developments in non-linear time series modelling, testing, and forecasting, *International Journal of Forecasting* 8(2), 135–156.

DeMiguel, V., Garlappi, L. and Uppal, R. (2009), Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy?, *The review of Financial studies* 22(5), 1915–1953.

Engle, R. F., Granger, C. W. and Kraft, D. (1984), Combining competing forecasts of inflation using a bivariate arch model, *Journal of economic dynamics and control* 8(2), 151–165.

Fischer, T. and Krauss, C. (2018), Deep learning with long short-term memory networks for financial market predictions, *European Journal of Operational Research* 270(2), 654–669.

Groenendijk, J. (2019), A comparison of direct, iterated and combined multistep conditional forecasts.

Groenendijk, J., Smits, S. and van Wijngaarden, D. (2020), The prediction performance of long short-term memory neural networks in a high-frequency setting.

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

Hornik, K., Stinchcombe, M. and White, H. (1990), Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks, *Neural networks* 3(5), 551–560.

Kara, Y., Boyacioglu, M. A. and Baykan, Ö. K. (2011), Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange, *Expert systems with Applications* 38(5), 5311–5319.

Khashei, M. and Bijari, M. (2010), An artificial neural network (p, d, q) model for timeseries forecasting, *Expert Systems with applications* 37(1), 479–489.

Kingma, D. P. and Ba, J. (2015), Adam: A method for stochastic optimization, *International Conference on Learning Representations* .

Lydia, A. and Francis, S. (2019), Adagrad—an optimizer for stochastic gradient descent, *Int. J. Inf. Comput. Sci* 6(5).

Nawi, N. M., Ransing, M. R. and Ransing, R. S. (2006), An improved learning algorithm based on the broyden-fletcher-goldfarb-shanno (bfgs) method for back propagation neural networks, *in* 'Sixth International Conference on Intelligent Systems Design and Applications', Vol. 1, IEEE, pp. 152–157.

Neely, C. J., Rapach, D. E., Tu, J. and Zhou, G. (2014), Forecasting the equity risk premium: the role of technical indicators, *Management science* 60(7), 1772–1791.

Newbold, P. and Granger, C. W. (1974), Experience with forecasting univariate time series and the combination of forecasts, *Journal of the Royal Statistical Society: Series A (General)* 137(2), 131–146.

Ordóñez, C., Lasheras, F. S., Roca-Pardinas, J. and de Cos Juez, F. J. (2019), A hybrid arima–svm model for the study of the remaining useful life of aircraft engines, *Journal of Computational and Applied Mathematics* 346, 184–191.

Ranganathan, A. (2004), The levenberg-marquardt algorithm, *Tutoral on LM algorithm* 11(1), 101–110.

Rather, A. M., Agarwal, A. and Sastry, V. (2015), Recurrent neural network and a hybrid model for prediction of stock returns, *Expert Systems with Applications* 42(6), 3234–3241.

Stock, J. H. and Watson, M. W. (1998), A comparison of linear and nonlinear univariate models for forecasting macroeconomic time series, Technical report, National Bureau of Economic Research.

Tanaka-Yamawaki, M. and Tokuoka, S. (2007), Adaptive use of technical indicators for the prediction of intra-day stock prices, *Physica A: Statistical Mechanics and its Applications* 383(1), 125–133.

Taskaya-Temizel, T. and Casey, M. C. (2005), A comparative study of autoregressive neural network hybrids, *Neural Networks* 18(5-6), 781–789.

Tieleman, T., Hinton, G. et al. (2012), Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, *COURSERA: Neural networks for machine learning* 4(2), 26–31.

Timmermann, A. (2006), Forecast combinations, *Handbook of economic forecasting* 1, 135–196.

Vijh, M., Chandola, D., Tikkiwal, V. A. and Kumar, A. (2020), Stock closing price prediction using machine learning techniques, *Procedia Computer Science* 167, 599–606.

Wang, J. and Kim, J. (2018), Predicting stock price trend using macd optimized by historical volatility, *Mathematical Problems in Engineering* 2018.

Zhang, G. P. (2003), Time series forecasting using a hybrid arima and neural network model, *Neurocomputing* 50, 159–175.

Zhang, J., Wei, Y.-M., Li, D., Tan, Z. and Zhou, J. (2018), Short term electricity load forecasting using a hybrid model, *Energy* 158, 774–781.

6 Appendix

Appendix A

This is the pseudo-code for the Adam algorithm developed by Kingma and Ba (2015). This code is literally taken from Kingma and Ba (2015), and I take no credit for the algorithm itself.

"

Algorithm 1 The Adam algorithm for stochastic optimization as described in Kingma and Ba (2015). g_t^2 indicates the elemntwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0 Initial parameter vector
$m_0 \leftarrow 0$ (Initialize 1^{st} moment vector)
$v_0 \leftarrow 0$ (Initialize 2^{nd} moment vector)
$t \leftarrow 0$ (Initialize timestep)
while *While* θ_t *not converged* **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1}$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\hat{m}_t \leftarrow m_t/(1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\hat{v}_t \leftarrow v_t/(1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon)$ (Update parameters)
end
return: θ_t (Resulting parameters)

"