DIFFERENT APPROACHE TO STOCK PRICE PREDICTION: THE USE OF RNN-LSTM AS A MACHINE LEARNING TECHNIQUE FOR STOCK PRICE PREDICTION

Iman Khamis, MLIS*

Northwestern University

*Address to which the correspondence should be:
Iman.khamis@northwestern.edu

Abstract

Machine learning has been used successfully over the years to predict stock prices, however, used methods (like Artificial Neural Networks and Convolution Neural Networks) has an accuracy error of average 20%. This error percentage makes the predicted results not functional to finance practitioners. The research problem is to devising a method using Recurrent Neural Network that can improve the accuracy of the results and reduce the error average to 6% . The research argues use of Recurrent Neural Network to predict stock prices with accuracy error of less than 6%.

The motivation behind the stock price prediction using machine learning is to forecast the next day's stock price for a specific firm to help traders make informed decisions. Some challenges emerge during the prediction process, among these challenges is the nature of the stock market as it is very volatile on the individual stock prices. This research aims to investigate the effectiveness of using different Recurrent Neural Network models as a method for the purpose of stock price forecasting. A set of parameters and hyperparameters are chosen that will aid in the selection of the best performance of the various available models that will take in consideration the insensitivity of the stock market.

The data for the analysis collected from Google finance using the python. The data files contain historical stock prices (last 12 years) for 29 of 30 of Dow Jones Industrial Average (DJIA) companies (excluding 'V'). Successfully, two models were devised using RNN-LSTM, parameter and hyperparameter were tuned, finally best performing model error percentage was less than 6%.

Declaration

I declare that this dissertation is my own original work and that all sources have been acknowledged. It is 6197 words in length.

Table of Contents

List of Figures

List of Tables

List of Abbreviation

ADAM	Adaptive Moment Estimation
AMH	Adaptive Market Hypothesis
ANN	Artificial Neural Network
ATR	Average True Range
AUC	Area Under the Curve
DJIA	Dow Jones Industrial Average
EMA	Exponential Moving Average
EMH	Efficient Market Hypothesis
FFNN	Feedforward Neural Network
GA	Genetic Algorithms
LR	Linear Regression
LSTM	Long Short-Term Memory
MACD	Moving Average Convergence/Divergence
MAPE	Mean Absolute Percentage Error
MSE	Mean Square Error
ReLU	Rectified Linear Unit
ROC	Rate of Change
RMSE	Root Mean Square Error
RMSprop	Root Mean Squared Propagation
RNN	Recurrent Neural Network
RSI	Relative Strength Index
SMA	Simple Moving Average
SO	Stochastic Oscillator
SVM	Support Vector Machines
SVR	Support Vector Regression

Acknowledgement

The completion of research would not have been achieved without the contribution of the following people:

- Special thanks go to Dr Alianna Maren, for valuable feedback and for encouragements.
- I would like to thank Dr Syamala Srinivasan for her support and guidance. Her Guidance helped me to become a confident researcher. She helped in many ways, both professionally and personally.
- Thanks also go to members of Kaggle for providing me with all information needed to fulfill this research.
- I would like to thank also the staff and colleagues at the Northwestern University.
- Last but not least, I thank my family, who has been very supportive, loving, caring and understanding, during my master, without their support I would have never succeeded.

1. Introduction

1.1 Background

Stock price prediction helps make the stock market be more efficient as investors would be able to envision the future of the stock market (Campanella et al., 2016; Urquhart and Hudson, 2013; Shonkwiler, 2013; Malkiel, 2003). However, other researchers believe that the stock market price depends on investors who analyze all information they could get to understand the stock price, hence prediction of the future stock price would not be of high value to the investors (Manahov and Hudson, 2014; Malkiel, 2003).

The efficient market hypothesis (EMH) states that future prices will only be predicted based on events that will happen in the future, therefore, the hypothesis suggests that it is not possible to predict stock prices because stocks follow a Random Walk (RW) model (Shonkwiler, 2013; Hull, 2009; Malkiel, 2003; Urquhart and Hudson, 2013; Malkiel, 2003). The random walk (RW) model suggests that prediction of the stock market prices is not possible, and any predicted profit made will not be in a sustained way. The random walk model is popular among financial experts, nevertheless, alternatives have been proposed like the Adaptive Market Hypothesis (Lo, 2017). The Adaptive Market Hypothesis (AMH) states that investors are willing to adapt to change, hence, these adaptations will make the market unstable. Therefore, AMH encourages using different methods to predict stock prices.

Stock prices are determined by supply and demand between traders, consequently, finance practitioners need to perform a technical analysis (analyze past stock price) and/or a fundamental analysis (study the company growth) before making trading decisions (Thomsett, 2015; Rockefeller 2011). “Technical analysts and fundamental analysts are diametrically

opposed to one another" Krantz (2016). Technical analysis suggests that only the momentum of the market is main drive behind the price of the stock. However, fundamental analysis suggests that technical analysis is not enough to determine stock price, additionally, related economic and financial factors must be taken in consideration. As well, financial practitioners argue that technical indicators are not enough for accurate prediction, fundamental indicators must be used together with technical indicators (Cavalcante et al., 2016; Cavalcante and Oliveria, 2015).

Finance and economic researchers have adapted the technical and fundamental analysis together in stock price generation (Hong and Wu, 2016; Chen et al., 2015; Amini et al., 2015; Wafi et al., 2015b), however, researchers prefer to do technical analysis and technical indicators. The reason behind using technical indicators by researchers, is that these indicators are available and ready for use anytime.

Time series data is unique as the relationship between input data and target variable changes, which is why it is important to choose a learning method that can manage these changes otherwise trained models would be less effective (Cavalcante and Oliveria, 2015). Any changes whether economic, political, environmental, or others, have a great impact on stock prices. According to Rockefeller (2011), the dynamic nature of the stock market could have an impact on the machine-learning-based prediction models.

Finance practitioners tend to measure the state of the stock by market indicators like Volatility Index (VIX) (Achelis, 2000; Rockefeller, 2011). Volatility Index (VIX) is widely used in the finance world, the index represents the expectations of stock market. Hight VIX means high volatility of the stock price, low VIX means the opposite (Rockefeller 2011). Clustering techniques have been widely used in financial time series data prediction, as well, clustering

approaches have been used to develop successful forecasting models over the years (Tsinaslanidis and Kugiumtzis, 2014; Cherif et al., 2011; Wu and Lee, 2015). This research investigates a time series analysis for the dataset DJIA 30 Stock Time Series. The data set contains historical stock data for DIJA 30 companies. Finally, a set of relevant parameters will be identified, which will be modified and developed for accurate prediction. Mean Square Error (MSE) was used to evaluate the accuracy of the various models.

1.2 Research Problem

The stock exchange market is one of the pillars of the national economy. The stock exchange allows the transfer of different goods between different brokers. Additionally, the stock exchange market is worth more than thousands of billions of dollars, which makes it an especially important investment tool among individuals. The importance of the stock market rises from the fact that most trades happen in the stock market, not only monetary exchange but as well many types of goods. The value of the stock is given by the volume of transactions of the stock. Predicting the trend of a certain stock in the future is considered quite valuable among finance practitioners.

It is critical to determine the decrease/increase of the stock market to help investors buy/sell the goods to avoid economic loss. This problem has been attracting researchers to build models to be able to forecast the stock market using different methods. Among the popular machine learning algorithms are used ANN (Artificial Neural Networks) and SVM (Support Vector Machines). This research will use Neural Networks to build different models to predict stock market prices in the future.

1.3 Research Question

- Which set of parameters, and architecture, would result in better stock price prediction performance? Answering this question should provide insight into the nature of the machine-learning technique, and manners to improve it. To be able to investigate this research question a dataset of financial time series of 30 Stock will be used to build different models aiming to predict stock market prices. The dataset used includes historical stock prices (last 12 years) for 29 of 30 DJIA companies.

1.4 Aims and Objectives

This research aims to explore the following:

1- The impact of using various models on the accuracy of the model in predicting stock price.

2- Evaluate the performance of each machine learning-based model created for stock price prediction.

1.5 Research Methods

To build a successful predicting model, machine learning algorithms will be used. Research suggests that a Recurrent Neural Network (RNN) is one of the recommended machine learning algorithms to deal with financial time prediction problems (Cavalcante et al., 2016; Nassirtoussi et al., 2014; Atsalakis and Valavanis, 2009). In this research, Long Short-Term Memory (LSTM) was used.

The reason LSTM is used is that LSTM model has the ability to adapt to the concept of sequential correlation. Sequential correlation refers to the importance of the sequences in the data presented in the Model. Financial time-series prediction problems require models that can

manage sequential correlation appropriately. In other words, the model should remember the past and present, hence memory is especially important in the LSTM model.

The key difference between RNN and other neural networks is the relationship between input and output. In traditional neural networks input and output are independent of each other, however, in RNN especially LSTM input and output are not independent. In prediction problems, the built network must remember the previous data, for this reason, the usage of LSTM will be more appropriate. Finally, LSTM has a unique architecture that give it the ability to effectively deal with historical data. LSTM model has 3 gates: input gate, output gate, and forget gate, these gates allow LSTM to identify which important information to keep, while forgetting the rest.

1.6 Overview of the Structure of the paper

The research is divided into five chapters. Chapter 1 is the introduction providing a background of machine learning-based stock price prediction. Chapter 2 provides an overview of the literature published on the application of machine learning methods in stock price prediction. It introduces neural networks, the Efficient Market Hypothesis, and Adaptive Market Hypothesis. As well, as covering the fundamental and technical analysis. Chapter 3 discusses the results of the experiments conducted with respect to technical analysis using input sets. Chapter 4 focuses on analyzing the results from the models built in chapter 3. Finally, chapter 5 discusses conclusions and suggestions for future research.

2. Literature Review

2.1 Background

The goal of this chapter is to provide a holistic overview to the main task addressed in this research which is forecasting stock prices using machine learning techniques.

Machine Learning is a rising field of Artificial Intelligence (AI), in ML researchers build models (mathematical and statistical models) from the data collected by the researchers. Therefore, the models based on machine learning algorithms can predict and give decisions based on the input data. Researchers divide machine learning tasks into these main types: (Bishop, 2006)

1- Classification: in classification tasks, the researcher's task is to identify a set of categories to which the input data belongs to.

2- Regression tasks are based on prediction based on historical data as input.

Goodfellow and others (2016) classified machines into the following types:

- Supervised learning is based on learning a function from training sets, training sets are composed of an input feature vector x and the desired out for this specific input y. Scientists train our model to a function that maps the input on the output hence predicting y from x (normally by calculating the probability).
- Unsupervised learning is based on the learning representation of the transformation of the input without any targeted output. The model here is learning the properties of the structure of the dataset for example learning the probability distribution p(x).

2.1.1 Deep learning and neural networks

Deep learning is a field of machine learning based on learning from the data directly by learning the data representations. The models are normally built by stacking layers of networks to make depth to our model which will give us more advanced models. Researchers consider a neural network as function approximator f (x: w) where the parameters (w also known as weights) learn to match the function f(x) (Goodfellow et al., 2016).

2.1.2 Artificial neural networks

Artificial neural networks are systems that are built by several neural layers and based on the neurons in human brains (Goodfellow et al., 2016). Each layer is represented by a vector where each value represents a single node. The layers in the network are classified into the input layer, output layers, and hidden layers in-between. The number of units in the hidden layers determines the width of the network. The layers are connected where each unit in a layer is connected to the unit in the previous later via the weights. Therefore, the sum of weights in the input layer is calculated and the value is passed in a vector representing the input layer.

A bias unit is added to the input and hidden layers, these units are not connected to the previous layer and represent an added weight learned for each layer. Additionally, an activation function (a non-linear function) is applied to the weighted sum of each unit to give the final output. It is important in any neural network to control the hyperparameter, such as the type of the layers in the network, the loss function, and the optimizer (Chollet, 2018).

Here more details about each hyperparameter:

1- Layers

There are different types of layers each type serves a specific purpose, for example, dense layers are used to process simple vector data. Convoluted layers are used when the spatial structure of the input is significant. Recurrent layers are used when there are sequence input data with the temporal structure like in time series prediction problems. The number of hidden layers and activation function are the hyperparameters chosen with every layer type chosen.

2- Loss function

It is a function that represents the performance of the model built during the training process. For example, the loss function for the Weight parameter will measure the difference between predicted y and targeted y. The numerical value is considered the feedback that will be used to evaluate our model. However, in regression tasks, the loss functions are the L1 and L2 distances, mean absolute error (MAE) and mean squared error (MSE) (Bishop, 2006; Goodfellow et al., 2016). (Figure 2.1)

Finally, to improve the performance of the neural network, researchers regularize the weights by adding a penalty term to the cost function, which is usually the norm of the L1 or L2 of the respective weights or outputs of neurons (Goodfellow et al., 2016).

3- Optimizer

Optimizer is the gradient-based algorithm that keeps updating the weights in the network based on the value of the loss function. There are plenty of methods that can be used as optimizers all based on the Gradient Descent (GD) (Amari, 1993). GD is based on the chain rule and gets back the directional derivative of a composite function like neural networks. GD

functions work on minimizing the loss function by updating the value of the weights toward the opposite direction of the gradient (Goodfellow et al., 2016). The learning rate is reduced as the loss function approaches a local minimum, different commonly used optimizers are based on the same principle as Adam, RMSprop, or Adagrad.

2.1.3 Training and evaluating the network

To train and evaluate the data, divide the data into three subsets training, validation, and test. When the network is learning the model iterated over the training set in batches of several samples. Iterations are called epochs, after each epoch, the validation set is presented to test the model's performance with new data. That is why weights are not updated during the validation process, and use the validation set as hyperparameter tuning. Finally, after a fixed number of epochs, the model with the lowest error for the validation set will be selected. If there is no improvement in the validation after a certain number of epochs the model comes to an early stop. Finally, the selected model goes to evaluation by processing the test subset one at a time. Dropout is a commonly used procedure where the model randomly ignores some layers' output (Srivastava et al., 2014). The main reason to use dropout layers is to prevent overfitting the training set as these will result in poor performance when it comes to validation and test subsets.

2.1.4 Activation functions

The activation function is the nonlinear function that is applied to the weighted sum of inputs then the numerical value is calculated in the final value (Goodfellow et al. 2016). This non-linear function that is applied between layers increases the ability to learn in a complex function. Here are some of the most common types of activation functions:

1-Rectifier Linear Unit (ReLU) (Figure 2.2)

2- Sigmoid (Figure 2.3)

3- Hyperbolic tangent (Figure 2.4)

4- Softplus (Figure 2.5)

f(x) = ln (1+exp x)

2.1.5 Recurrent layers

Recurrent neural networks RNN are feedforward networks with backpropagation connections. The main difference between RNN, DNN, and CNN is learning and memory as both DNN and CNN do not have memory. When incorporate the backpropagation with layers it allows the RNN to learn from the temporal structure of the input like the data selected for this research (Chollet, 2018). There are different types of RNN

1- Vanilla RNN

It the simple type of RNN, in this network the recurrent connections are in-between hidden layers that enable the network to generate output at each time step.

$$\mathbf{y}^t = f_{out}(\mathbf{W}_{out} \cdot \mathbf{h}^t), \quad \text{were,} \quad \mathbf{h}^t = f_h(\mathbf{W}_{in} \cdot \mathbf{x}^t + \mathbf{W}_h \cdot \mathbf{h}^{t-1}),$$

Where x^t (input), h^t (hidden), and y^t (output). F(out) the output activation functions, and F(h) the hidden layers activation functions. W(in) is the input-to-hidden weight matrix. W(h) is the hidden-to-hidden weight matrix, and W (out) is hidden-to-output weight matrix (Chollet, 2018). Conclusion from the equation above is that RNN learns to map the input sequence to a hidden sequence that contains relevant features to the current time step t. The network as well updates the h(t) based on the hidden state h(t-1), to finally obtain the output feature map.

2- Bidirectional RNN

The bidirectional RNN was created to improve the long-term memory capabilities of RNN. The bidirectional RNN is two RNNs one moving from the start of the sequence while the other RNN moves from its end. This way the bidirectional recurrent layers can access the long-term data by the forward and backward propagations (Graves and Schmidhuber, 2005; Graves et al., 2013).

$$\mathbf{y}^t = f_{out}(\mathbf{W}_{out} \cdot \mathbf{h}^t + \mathbf{V}_{out} \cdot \mathbf{g}^t), \qquad \mathbf{g}^t = f_g(\mathbf{V}_{in} \cdot \mathbf{x}^t + \mathbf{V}_g \cdot \mathbf{g}^{t+1}),$$

Where V(in) is input-to-hidden weights, V(g) hidden-to-output weight V(out) moving forward in the network. G(t) hidden state, f(g) activation function.

2.1.6 Long Short-Term Memory (LSTM)

One of the problems facing RNNs is vanishing gradient and exploding gradient. A vanishing gradient happens when the gradient decreases exponentially across the layers preventing the network from learning as the weights of the earlier layers could not be updated. The exploding gradient is the opposite when the gradient increases exponentially making the network unstable (Pascanu et al., 2013). LSTM networks refer to gated RNNs where neural networks learn when to reset the flow of information or control it to overcome the problems of vanishing/exploding gradients. In LSTM some gates are implemented with sigmoid function, which is known as input gate, output gate, and forget gate (Gers et al., 1999). The three gates of the LSTM are the following:

$$g_{in}^{t} = \sigma(W_{in} \cdot x^{t} + W_{h} \cdot h^{t-1}),$$

$$g_{out}^{t} = \sigma(U_{in} \cdot x^{t} + U_{h} \cdot h^{t-1}),$$

$$g_{forget}^{t} = \sigma(V_{in} \cdot x^{t} + V_{h} \cdot h^{t-1}),$$

Where w(in), u(in), and v(in) are the weight for input matrices for input. While w(h), u(h), and v(in) are the recurrent weight matrices, and h(t-1) is the output feature from the previous step (t-1). The layers in LSTM us updated frequently

$$c^{t} = g_{forget}^{t} \times c^{t-1} + g_{in}^{t} \times \tanh(C_{in} \cdot x^{t} + C_{h} \cdot h^{t-1}),$$ (Chollet, 2018).

2.2 Different Market Hypothesis

There are different market hypotheses, in this chapter we will discuss Efficient Market Hypothesis and Adaptive Market Hypothesis. The Efficient Market Hypothesis assumes that the market is made up of rational investors, meaning that the current stock price is based on current information that is based on data from the past and the expectations of the future (Shonkwiler, 2013). In this hypothesis, the speed of the information

and the spread of the information is called efficiency. Levels of market efficiency come in three forms, the *weak form* which means that past prices cannot be used to predict the future hence technical analysis cannot be used, and the *semi-strong form* which means no publicly available information can be used to predict the future price hence no fundamental analysis can be used, and the *strong form* of efficiency means even insider information cannot be used to predict the future price (Shonkwiler, 2013). Therefore, EMH assumes that stock prices cannot be predicted overall.

On the other hand, Adaptive Market Hypothesis (AMH) assumes that stock prices are based on an interaction of various factors and that “market efficiency is dynamic and context-dependent” (Manahov and Hudson, 2014). Therefore, according to AMH stock prices can be predicted as the market evolves (Lo, 2017). As well, AMH remarks that investors have both rational and irrational behaviors that change according to the political and technical environment change. Investors adapt to these changes hence there is an opportunity to predict the future price.

2.2.1 Technical analysis

Technical analysis is based on the assumption that history will repeat itself (Tsinaslanidis and Zapranis, 2016). In other words, technical analysis “try to classify these repetitive investment/trading behaviors and their corresponding impacts on the market prices” (Tsinaslanidis and Zapranis, 2016). Technical Indicator is used in technical analysis, and it is the application of math formulas to historical prices (Open, Close, High, and Low) as well as the number of traded shares for the specific stock.

2.2.2 Fundamental analysis

Fundamental analysis refers the use of studies regarding the business and the company's products to determine the expected value of the company's stock in the future (Krantz, 2016). In this type of analysis, researchers depend on the operation side of the company and the economic environment around it. The importance of the fundamental analysis lies in the accuracy of the valuation of the stock provided by fundamental analysts.

2.2.3 Technical and fundamental comparison

Both technical and fundamental analyses are vastly different, technical analysis believes in stock price is driven by the momentum of the market and its past value. However,

fundamental analysis believes that it is mistaken to consider the momentum of the market in estimating stock price because it might lead to "groupthink" that will mislead the stock price (Krantz 2016). Schwalger and Turner (1995) think it is better to combine these two analyses type, fundamental analysis will help traders decide which stock to trade, while technical analysis will help traders to determine which time is best to trade the stock.

Krantz (2016) believes that fundamental analysis can give long-term investors insights that are not based on short-term volatility. Also, Krantz suggests that fundamental analysts should pay attention to technical indicators to overcome the lack of timing and to be able to spot changes in the market. Also, Chen et al. (2016) discusses the benefits of using technical and fundamental indicators together, like using F-Score (fundamental indicator) with a technical indicator (like information ratio) when investigating a stock will yield more accurate assumptions.

Another study compared the predictive performance of both indicators in one day. Wafi et al. (2015b) argue that technical indicators outperformed fundamental indicators when predicting the price of a stock for one day. However, when one day ahead returns was being forecasted, the technical indicators performed poorly.

2.3 Use of Machine Learning in stock price prediction

Statistical Techniques and Machine Learning Techniques are the two techniques that have been used to predict stock prices (Cavalcante et al. 2016). Statistical techniques are built on the idea that factors that drive the stock prices are linear. However, time-series stock price data has a non-linear and noisy nature, hence machine learning methods would be more appropriate to be used (Cavalcante et al., 2016; Hsu et al., 2016).

Cavalcante et al. (2016) set up a trading framework to be used in predicting financial time-series stock prices (Figure 2.6). In the framework, there are steps involved in the forecasting process using machine learning techniques. The following section reviews the application of machine learning techniques to predict stock prices.

2.3.1 Input

Historically researchers used technical indicators as inputs (Cavalcante et al., 2016). Indicators that normally are used in forecasting stock prices are "simple moving average (SMA), exponential moving average, relative strength index (RSI), rate of change (ROC), moving average convergence/divergence (MACD), and Stochastic oscillator and average true range (ATR)" (Krollner et al., 2010). The reason behind the popularity of technical indicators is their availability, and accessibility, unlike fundamental indicators. However, some researchers used fundamental analysis, while others used both like Chandwani and Saluja (2014) in forecasting stock prices in India.

Some of the machine learning techniques that have been used by researchers to predict stock prices with high accuracy are Artificial Neural Network and SVR as both ANN and SVR optimized by Genetic Algorithms (GA). Weng et al. (2017) used financial time series data from various sources and P/E ratio to predict stock market price, the performance of the models created (ANN, SVR, DT) were compared by using Area Under the Curve (AUC) indicator.

2.3.2 Pre-process the data

Pre-processing data before analysis include the following: cleaning the data by removing missing values, etc., and standardizing the data for the machine learning methods to learn

effectively. Researchers recommend using ratios rather than the actual values as input, especially when using a neural network as a machine learning technique (Vanstone and Finnie 2009). Using ratios will enhance the ability to generalize the findings. Another important step is addressing the nulls or missing values, analysts might choose to delete missing rows, however, Romero and Balch (2014) argue that the common approach is to fill forward "to treat missing values as the same level as the last known value".

Feature selection is an important step in pre-processing the data. Feature selection means "to discard attributes that appear to be irrelevant" (Russell and Norvig, 2010). Feature selection helps in reducing the inputs to only relevant ones and decreases the "curse of dimensionality" (Bellman, 1961). Feature selection also is known as the filter or wrapper method (Torgo 2017). The difference between the filter and wrapper method is that the filter method does not take into consideration predicting approach, while the wrapper method does.

2.3.3 Popular methods

Cavalcante et al. (2016) categorize the machine learning approaches into three main categories:

1- Models use a single machine learning technique

2-Models use a combination of machine learning techniques with optimization

3-Models use a combination of different single models

To predict time-series data researchers have been using a single machine learning technique (e.g., ANN, SVR, etc.). Cavalcante et al. (2016) argue that ANN and SVR are "the more common soft computing techniques applied in forecasting financial time series". Cavalcante et al. (2016) added "the majority of work which proposed the use of ANNs for

solving the financial forecasting problem have used a multi-layer feed-forward neural network (MLP) trained with backpropagation algorithm with remarkable success." The reason is that ANN and SVR are enormously powerful in feature extraction. Hadavandi et al. (2010) used a GA (Genetic Algorithm) to optimize the parameters of the neural network to predict stock prices (Figure 2.7).

2.3.4 Evaluations of the models

Performance evaluation is an integral part of building up a successful machine learning model. The performance measures used by scientists are statistical (RMSE, MAE, MSPE, etc.) (Atsalakis and Valavanis, 2009). Twenty-six studies out of seventy-two used statistical measures. Researchers suggest using "the MAE or RMSE if all your forecasts are on the same scale" and use "the MAPE if you need to compare forecast accuracy on several series with different scales, unless the data contain zeros or small values or are not measuring a quantity" (Hyndman and Athanasopoulos, 2014).

2.4 Conclusion

This section provided definitions of fundamental machine learning concepts and trading-related concepts like the Efficient Market Hypothesis, Adaptive Market Hypothesis. As well, it discussed the fundamental analysis and technical analysis as the two methods used by analysts to decide on the trading stock in the market.

Machine learning has been a popular and successful method to predict stock prices. The discussion of the application of the machine learning methods in forecasting studies showed that machine learning approaches depend on technical indicators. It is highlighted that as models developed the technical indicators and fundamental as well should be taken in consideration.

3. Results and Discussion

This chapter will provide analysis of the results in a timeline manner. The reason behind devising the models is to inquire a model that is better performing than existing models. As literature suggests, models created using other machine learning techniques have 20% margin error, the model created should have margin error less than 20%. The following paragraphs will give an overview of each model separately before discussing and interpreting the results.

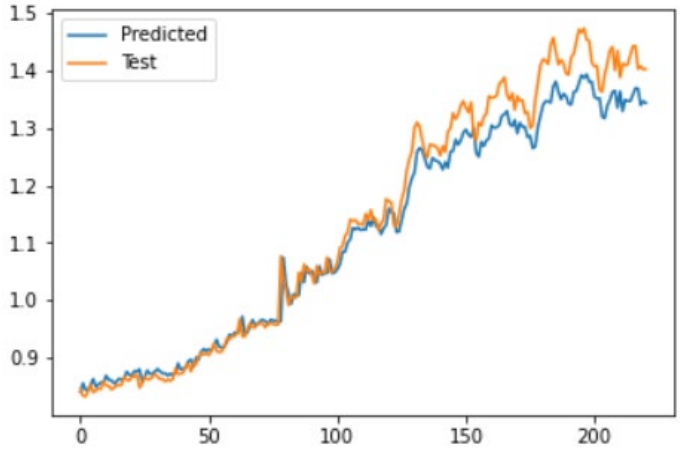
Model One

The Model is built as following: The first layer is an embedding layer, then an LSTM which has three hidden later, and a final Dense Layer. The model is compiled using *RMSprop* optimizer and the loss function is *MSE*. In this experiment the model will be fit using 60 epochs (Fig 4.1).

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_3 (LSTM)                (None, 30, 120)           58560
_________________________________________________________________
lstm_4 (LSTM)                (None, 30, 80)            64320
_________________________________________________________________
lstm_5 (LSTM)                (None, 60)                33840
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 61
=================================================================
Total params: 156,781
Trainable params: 156,781
Non-trainable params: 0
_________________________________________________________________
```

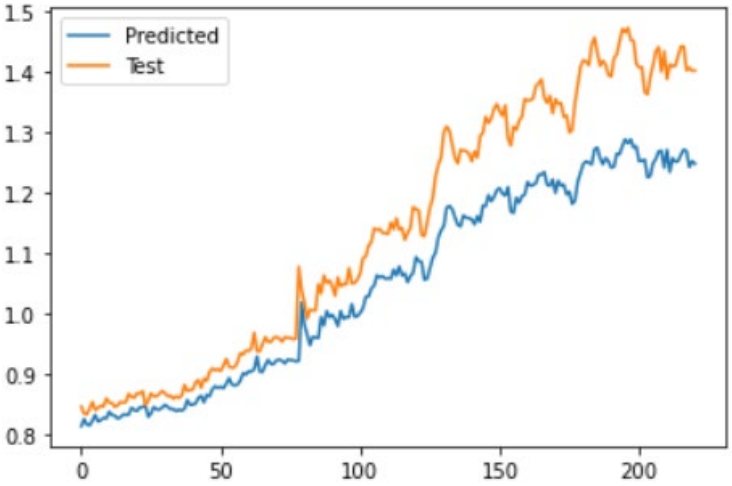
Fig 4.1: Model One (RNN-LSTM) architecture

The graph (Fig 4.2) is showing that the model is performing very well, as the curve of predicted and test data are quite close to each other. However, the graph starts to split at the end indicating inaccurate results. Hence, this model needed to be modified to get more accurate prediction.



(Fig 4.2: Model One, experiment one: Results showing the predicted and test results compared in one graph)

The experiment (Model one, Experiment two) is repeated with editing the architecture to achieve better accurate results. A Dropout layer will be added to prevent overfitting. The graph (Fig 4.3) is showing that the model is not performing well, as the curve of predicted and test lines are widely split. The reason could be the dropout layer, as dropout randomly deactivate some neurons while the model is still probably learning which resulted is under-fitting model.



(Fig 4.3: Model One, experiment two: Results showing the predicted and test results compared in one graph)

Model Two

The LSTM architecture for the second model will be slightly different from model one to achieve better accuracy. The model contains three LSTM layers (10, 20, 30 neurons in the hidden layers) (Fig 4.4).

```
Model: "sequential_31"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_23 (LSTM)               (None, 60, 10)            600
_________________________________________________________________
dropout_9 (Dropout)          (None, 60, 10)            0
_________________________________________________________________
lstm_24 (LSTM)               (None, 60, 20)            2480
_________________________________________________________________
dropout_10 (Dropout)         (None, 60, 20)            0
_________________________________________________________________
lstm_25 (LSTM)               (None, 30)                6120
_________________________________________________________________
dropout_11 (Dropout)         (None, 30)                0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 31
=================================================================
Total params: 9,231
Trainable params: 9,231
Non-trainable params: 0
```

Fig 4.4: Model Two (RNN-LSTM) architecture

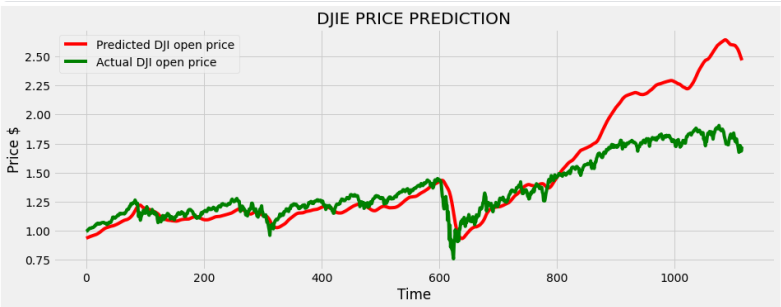
The activation function for the input layer is ReLU. The loss function is MSE, and optimizer is ADAM. In this experiment, the number of epochs will be set to be 10, this number will determine the number of times the training happens over the dataset. The batch size will be set to 32 samples, this will set the number of samples the LSTM model train on before updating the next gradient. Finally, by calculating the accuracy (mean of the test data / mean of the predicted data), the model can be evaluated accurately. The accuracy percentage is 93.74% which is considered good accuracy percentage, however, some hyperparameters can be manipulated or change some features to achieve better results (Fig 4.5). However, any modification must be executed with cautious not to overfit.

```
accuracy = (mean_y_test / mean_y_pred)*100
accuracy
```

```
93.74532108826469
```

Fig 4.5: Model Two, experiment one accuracy result

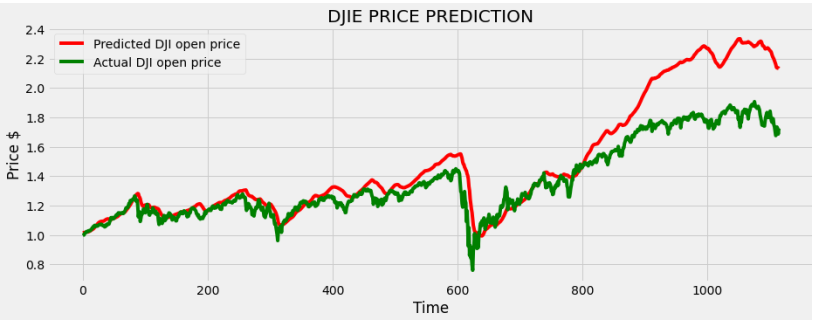
The graph (Fig 4.6) is showing a good performing model as well. However, the split ending in the graph indicates enhancement is needed.



(Fig 4.6: Model Two, experiment one: Results showing the predicted and test results compared in one graph)

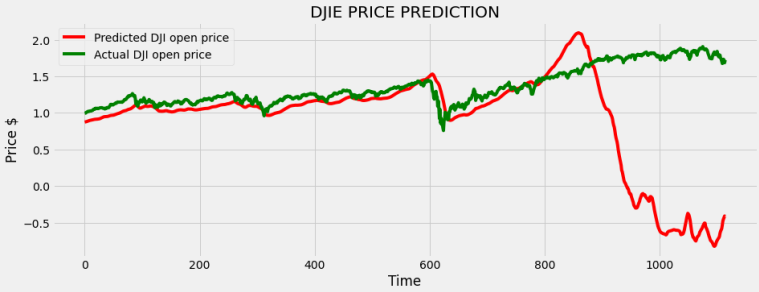
Because the model has not reach less than 6% error, the second is repeated four times with editing some hyperparameter to achieve the best performing model.

1. In this experiment the proposed alteration is adding the Dropout layers. The accuracy resulted dropped, this indicated that the model is indeed of the neurons to learn (Fig 4.4).



(Fig 4.7: Model Two, experiment two: Results showing the predicted and test results compared in one graph)

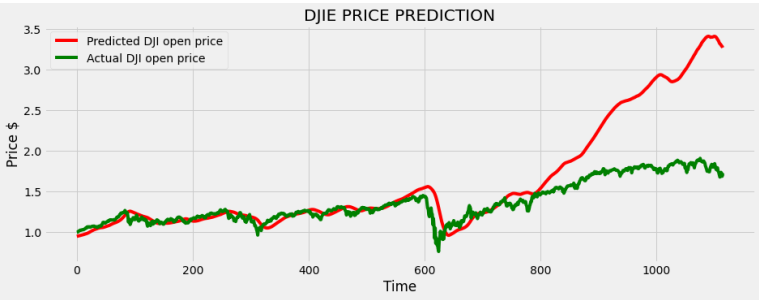
2. In next experiment the alteration will be increasing the epochs number from 10 to 30. As the number of epochs increases, the weights will be changed in the LSTM model. This change will cause the boundary change from underfitting to best fitted model.



(Fig 4.8: Model Two, experiment three: Results showing the predicted and test results compared in one graph)

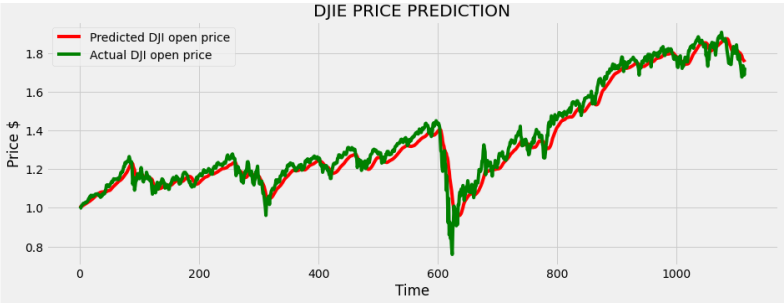
The graph (Fig 4.8) shows that the model is highly overfitting and the accuracy dropped. The epoch in LSTM model refers to the number of times the entire training dataset iterate. The number of times the training dataset passed through the model caused it to overfit on the training set.

3. The following experiment (number 4) the alteration will be increase the epochs from 32 to 60 and decrease the patch size to 20. The batch size is an important hyperparameter that can be adjusted to get a better-performing model. Increasing the batch size will train the model and increases the accuracy of the learning process. However, the smaller the batch size the faster the training dynamic, and less noise, hence less tendency to overfitting. By editing the batch size to sixty the model is prone to overfit. As well, neural network systems generally, are prone to overfitting small batch size normally is better. Figure 4.9 shows the accuracy has improved and the model is slightly overfitting.



(Fig 4.9: Model Two, experiment four: Results showing the predicted and test results compared in one graph)

4. The last experiment (number 5) the editing will increase the neuron in the hidden layers as follow 32, 64, 128. The LSTM models perform well for prediction the reason is the mechanism of working neurons in the model. For the LSTM model to work, neurons must be involved in the decision making. However, not all neurons are of the same importance. Some neurons are more important effects than others. Hence increasing the number of neurons in the model will surely increase the ability of the model to understand the training dataset and pass the important weights to the following neurons. The results are quite good (Fig 4.10); however, the model is also slightly overfitting. It is clear in this experiment the effect of increasing the number of neurons in the hidden layers on the performance of our model.



(Fig 4.10: Model Two, experiment four: Results showing the predicted and test results compared in one graph)

4. Conclusion and Future Research

4.1 Conclusion

The conclusion section will include a discussion of what was achieved and an interpretation of the results. What was achieved in this research is that two models were built and successfully predict the stock prices. Each model has been edited multiple times to achieve better accuracy of the prediction results. The two models and different experiments conducted in this research aimed to understand the impact of changing the model architecture and hyperparameters on accuracy. The main goal was to achieve accuracy higher than what was achieved in the published literature to help traders gain insights into the stock market.

To achieve the main goal and answer the research question, the architecture of the RNN-LSTM model was changed twice (for example, the number of hidden layers). As well, some hyperparameters (for example, learning rate number of epochs) was altered for each model to edit the model performance.

According to the results, each model reacted differently to the change proposed. For example, removing dropout layers ended up with highly overfitting in some models. Finally, increasing the number of epochs doesn't always guarantee to increase in the learning rate of our model.

Results in detail:

In experiment one (Model one) the LSTM with three hidden later is performing quite well. However, the accuracy of the result is low could be due to overfitting issue. So, the architecture of the model was edited to get more accurate prediction results. In the second experiment (Model one) a dropout layer was added to prevent overfitting. The results show that

the model is not performing better, in fact, the prediction is less accurate than in the previous experiment. The reason could be because the dropout method turns off some neurons randomly, and that process could be happening before the model is well-developed which leads to less accurate results (Table 4.1).

Second model is proposed to achieve accuracy higher than 94%. The architecture has changed to have a smaller number of neurons and reduced the number of epochs. 93.7% accuracy of the result was achieved; however, the model failed to achieve the aimed accuracy, which means the model need alteration to achieve research goal.

In the following experiment (Model 2, experiment 2) the dropout layer was removed, however, the model did not perform better. In fact, the accuracy of the model dropped to 91.9%. The reason could be learning did not happen over enough epochs.

In the next experiment (Model 2, experiment 3) the number of epochs will be increased to 30 instead of 10. The results as shown in the (Figure 4.9), the model is highly overfitting on the training data that it failed in predicting accurately.

Next experiment (Model 2, experiment 4) the training epochs dropped to ten in order not to overfit, as well the patch size was decreased. The results below in the table shows the model is performing better than previous experiments, however it is overfitting (Table 4.1).

Apparently, the model needs more neurons to learn accurately. So, last edit to the model architecture would be adding more neurons, decrease the number of epochs while increase the batch size. The model result is quite remarkable, as shown in the table below, however, it is slightly overfitting.

Exp.	Descriptions	Results
(Model 1), Ex. 1	LSTM three layers (12, 80, 60) NO dropout with 60 epochs	Figure 4.1
(Model 2), Ex. 1	LSTM three layers (10, 20, 30) WITH dropout with num_epochs = 10, batch_size = 32	93.7% accuracy on prediction
(Model 2), Ex. 2	LSTM three layers (10, 20, 30) WITHOUT dropout with num_epochs = 10, batch_size = 32	91.9% accuracy on prediction
(Model 2), Ex. 3	LSTM three layers (10, 20, 30) WITH dropout with num_epochs = 30, batch_size = 32	Figure 4.9
(Model 2), Ex. 4	LSTM three layers (10, 20, 30) WITH dropout with num_epochs = 10, batch_size = 20	109%
(Model 2), Ex. 5	LSTM three layers (32, 64, 128) WITH dropout with num_epochs = 11, batch_size = 32	101%

(Table 4.1: Results and architecture of the six experiments conducted in this research)

Finally, to answer the research question "Which set of parameters, and architecture, would result in better stock price prediction performance?" different models were built. Results showed that the model is built using 32, 64, and 128 hidden layers, 32 batches, 10 epochs, the activation function is ReLU, the loss function is MSE, and the optimizer is ADAM, is the best performing machine learning model based on the datasets. The ability of the prediction models relies on the machine learning method used.

4.2 Future Research

Cavalcante et al. (2016) highlighted the importance of using deep learning-based methods for financial forecasting. The recommendation for future work would be enhancements/changes

to the proposed models through changing the parameters (inputs, and other machine learning methods) and also updating taken methods.

Another future work suggestion would be applying the proposed models in health-related time-series forecasts. Zhang et al. (2014) highlighted the importance of forecasting epidemic diseases (e.g., brucellosis, hepatitis A, hepatitis B, etc.). Zhang et al. (2014) stated that these diseases have a relationship with seasonal change and collecting data over time will eventually help in epidemic prediction.

Finally, the developed models can be used in electricity load prediction. It is an area where there are non-linear relationships and has "dependency on various exogenous factors including time, day, weather, seasonal economic aspects, and social activities," which "make the load forecasting a difficult task" (Mohan et al., 2018). There are similarities between the characteristics of electricity load forecasting and financial time series forecasting that will make using LSTM model prediction possible in electricity load.

References

Amari, S. Backpropagation, and stochastic gradient descent method. *Neurocomputing*, 5(*4-5*):185–196, 1993.

Amini, Amir, Rahnama, G., Alinezhad, Alireza. Ranking and Managing Stock in the Stock Market Using Fundamental and Technical Analyses. J*ournal of Modern Processes in Manufacturing and Production*, 4(*3*), 45-57, 2015.

Atsalakis, George S., and Valavanis, Kimon. Surveying stock market forecasting techniques - Part II: Machine learning methods. *Expert Systems with Applications*. 36, 5932-5941, 2009.

Bellman, Richard E. *Adaptive Control Processes: A Guided Tour*. Princeton: Princeton University Press, 1961.

Bishop, Christopher. *Pattern recognition and machine learning*. springer, 2006.

Campanella, Francesco, Mustilli, Mario, and D'Angelo, Eugenio. Efficient Market Hypothesis and Fundamental Analysis: An Empirical Test in the European Securities Market. *Review of Economics & Finance*. 27-42, 2016.

Cavalcante, Rodolfo, Brasileiro, Rodrigo C., Souza, Victor L.F., Nobrega, Jarley P., Oliveira, Adriano L.I. *Computational Intelligence and Financial Markets: A Survey and Future Directions. Expert Systems with Applications*. 55, 194-211, 2016.

Cavalcante, Rodolfo C., and Oliveira, Adriano L. I. An approach to handle concept drift in financial time series based on Extreme Learning Machines and explicit Drift Detection. *International joint conference on neural networks* (IJCNN). 1–8, 2015.

Cavalcante, Rodolfo C., and Oliveira, Adriano Lorena Inácio. FEED: Feature Extraction for Explicit Concept Drift Detection in Time Series. *International joint conference on neural networks (IJCNN)*, 2016.

Chandwani Deepika, Saluja, Manminder Singh. Stock Direction Forecasting Techniques: An Empirical Study Combining Machine Learning System with Market Indicators in the Indian Context. *International Journal of Computer Applications*. 92(*11*), 8-17, 2014.

Chen, Kai, Zhou, Yi, Dai, Fangyan. A LSTM-based method for stock returns prediction: A case study of China stock market. *IEEE International Conference on Big Data (Big Data), Proceedings*, 2823-2824, 2015.

Chen, Hong-Yi, Cheng-Few, Lee, Wei-Kang, Shih. Technical, Fundamental, and Combined Information for Separating Winners from Losers. *Pacific-Basin Finance Journal*. 39, 224-242, 2016.

Cherif, A., Cardot, H., Bone R. SOM time series clustering and prediction with recurrent neural networks. *Neurocomputing*. 74, 1936 – 1944, 2011.

Chollet, François. *Deep Learning with Python*. Manning Publications Co., 2018.

Gers, F.A., Schmidhuber, J., and Cummins, F. *Learning to forget: Continual prediction with LSTM*. IET, 1999.

Goodfellow, Ian, et al. *Deep learning.* MIT press, 2016.

Hadavandi, Esmaeil, Ghanbari, Arash, and Abbasian-Naghneh, Salman. Developing an Evolutionary Neural Network Model for Stock Index Forecasting.Advanced Intelligent Computing. *Theories and Applications - Communications in Computer and Information Science*. 93, 407-415, 2010.

Hong, KiHoon, Wu, Eliza. The roles of past returns and firm fundamentals in driving US stock price movements. *International Review of Financial Analysis*. 43, 62-75, 2016.

Hull, John. *Options, Futures, and Other Derivatives*. New Jersey: Pearson Education International, 2009.

Hyndman, Rob, Athanasopoulos, George. *Forecasting: principles and practice*. OTexts, 2014.

Ibidapo, Isaac, Adebiyi, Ayodele, Okesola, Olatunji. Soft Computing Techniques for Stock Market Prediction: A Literature Survey. *Covenant Journal of Informatics and Communication Technology*. 5-2, 2017.

Krantz, Matt. *Fundamental Analysis for Dummies*. Indianapolis, Indiana, USA: Wiley Publishing, 2016.

Krollner, Bjoern, Vanstone, Bruce, and Finnie, Gavin. Financial time series forecasting with machine learning techniques: A survey. *The European symposium on artificial neural networks: Computational and machine learning*. Bruges, Belgium, 2010.

Lo, Andrew. *Adaptive Markets Financial Evolution at the Speed of Thought*. Press Princeton, 2017.

Malkiel, Burton G. The Efficient Market Hypothesis and Its Critics. *Journal of Economic Perspectives*. 17, 59 – 82, 2003.

Manahov, Viktor, Hudson, Robert. A note on the relationship between market efficiency and adaptability. *New evidence from artificial stock markets. Expert Systems with Applications*. 41, 7436-7454, 2014.

Mohan N., Soman K. P., Kumar S. S. A data-driven strategy for short-term electric load forecasting using dynamic mode decomposition model. *Applied Energy* 232, 229–244, 2018.

Nassirtoussi, Arman Khadjeh, Aghabozorgi, Saeed, Wah, The Ying, Ngo, David Chek Ling. Text mining for market prediction: A systematic review. *Expert Systems with Applications*. 41, 7653-7670, 2014.

Pascanu, Razvan, Mikolov, Tomas, and Bengio, Yoshua. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning,* 2013.

Rockefeller, Barbara. *Technical Analysis for Dummies. Indianapolis*, Indiana: Wiley Publishing, 2011.

Romero, Philip J., Balch, Tucker R. *What Hedge Funds Really Do: An Introduction to Portfolio Management*. Business Expert Press, 2014.

Russell, Stuart J., and Norvig, Peter. *Artificial Intelligence: A modern Approach*. New Jersey: Pearson, 2010.

Schwager, Jack D., and Turner, Steven C. *Futures: Fundamental Analysis*. Wiley, 1995.

Shonkwiler, Ronald W. *Finance with Monte Carlo*. New York: Springer, 2013.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(*1*):1929–1958, 2014.

Thomsett, Michael C. *Getting Started Getting Started in Stock Analysis*. Hoboken, Singapore: John Wiley Publishing, 2015.

Torgo, Luis. *Data mining with R: learning with case studies*. Chapman & Hall/CRC, Florida, 2017.

Tsinaslanidis, Prodromos E., Kugiumtzis, Dimitris. A prediction scheme using perceptually important points and dynamic time warping. *Expert Systems with Applications*. 41, 6848–6860, 2014.

Tsinaslanidis, Prodromos E., Zapranis, Achilleas D. *Technical Analysis for Algorithmic Pattern Recognition*. Springer, 2016.

Urquhart, Andrew, Hudson, Robert. Efficient or adaptive markets? Evidence from major stock markets using very long run historic data. *International Review of Financial Analysis*. 28, 130-142, 2013.

Vanstone, Bruce, Finnie, Gavin. An empirical methodology for developing stockmarket trading systems using artificial neural networks. *Expert Systems with Applications*. 36(*3*), 6668-6680, 2009.

Wafi, Ahmed, Hassan, and Adel Mabrouk. "Fundamental Analysis Vs Technical Analysis in the Egyptian Stock Exchange – Empirical Study*." International Journal of Business and Management Study – IJBMS* 2 (10/19 2015): 212-18.

Weng, Bin, Ahmed, Mohamed, Megahed, Fadel. Stock market one-day ahead movement prediction using disparate data sources. *Expert Systems with Applications*. 79, 153-163, 2017.

Wu, Shin Fu, Lee, Shei-Jue. Employing local modeling in machine learning based methods for time-series prediction, *Expert Systems with Applications*. 42, 341-354, 2015.

Zhang X., Zhang T., Young A. A., Li X. Applications and Comparisons of Four Time Series Models in Epidemiological Surveillance Data. *PLoS ONE* 9(*2*): e88075, 2014.

Appendix A

Figure 2.1

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

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

Figure 2.2

$$f(x) = x^+ = \max(0, x)$$

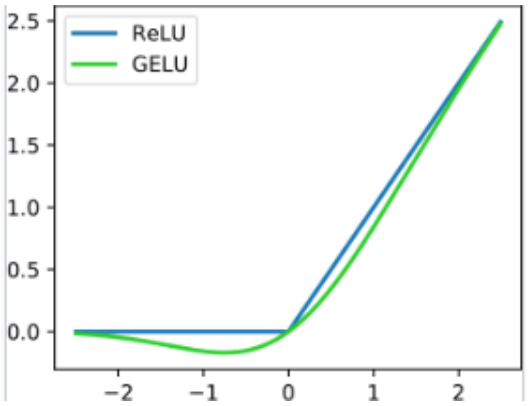


Figure 2.3

$$S(x) = \frac{1}{1 + e^{-x}}$$

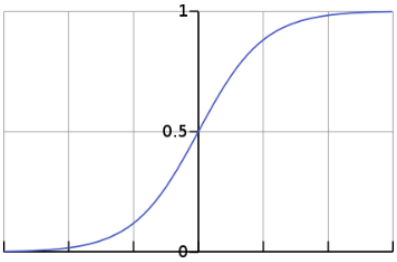


Figure 2.4

$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1}$$

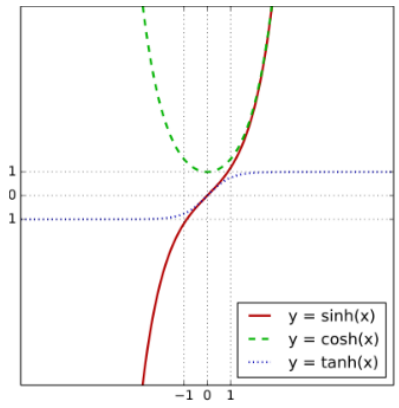


Figure 2.5

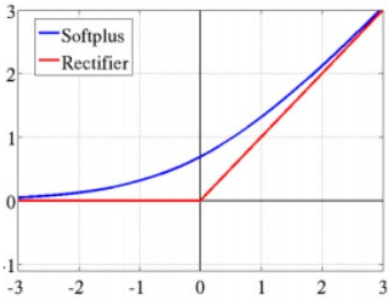


Figure 2.6

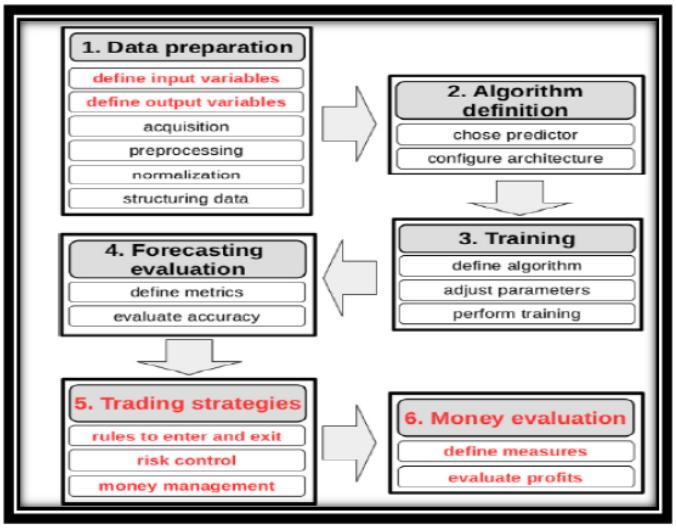
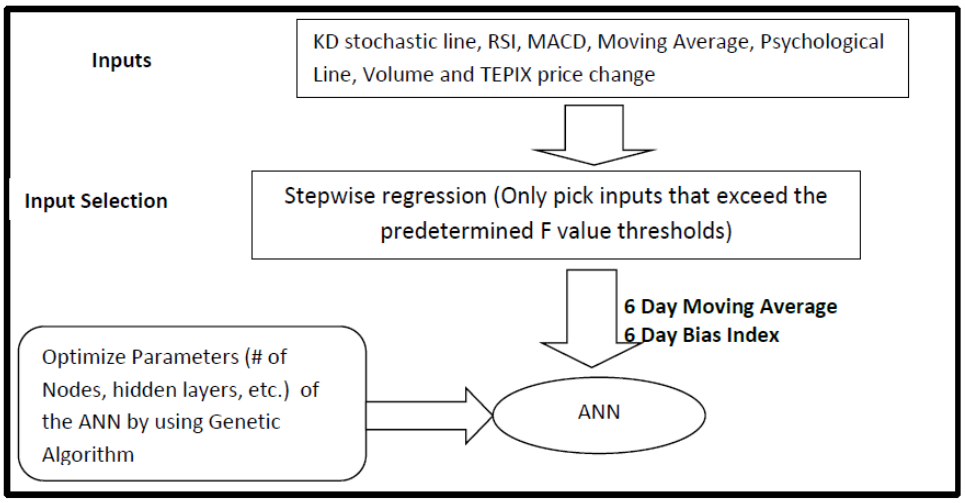


Fig 2.7


Inputs
KD stochastic line, RSI, MACD, Moving Average, Psychological Line, Volume and TEPIX price change
Input Selection
Stepwise regression (Only pick inputs that exceed the predetermined F value thresholds)
6 Day Moving Average
6 Day Bias Index
Optimize Parameters (# of Nodes, hidden layers, etc.) of the ANN by using Genetic Algorithm
ANN

Appendix B

Development of LSTM Model

This chapter introduce deep learning architectures to different models to predict stock prices for financial time series data. As stated earlier, Machine learning techniques have been popularly used in predicting stock prices because the algorithm allows the network to learn the data history and be able to predict the future data (Cavalcante et al., 2016; Nassirtoussi et al., 2014; Atsalakis and Valavanis, 2009).

3.1 Data

The Data used in training the models is adapted from the dataset named 'S&P 500 Stock data'. This data is gathered from Google finance using the python library 'pandas_datareader' and available on Kaggle Website. The period of the data was from January 2006 to January 2018. The data is containing 13 years of stock data, the data is presented in two formats:

1- Folders, each folder contains files of data for individual stocks, labelled by their stock ticker name.

2- CSV files contain this same data, presented in merged .csv files.

3.2. Model Design

Research suggests that Artificial Neural Network (ANN) and Support Vector Regression (SVR) are the most popular machine-learning methods used in stock price prediction. Additionally, surveys show that researchers use machine learning techniques in financial analysis using technical indicators rather than fundamental indicators (Ibidapo et al., 2017; Cavalcante et al., 2016; Atsalakis and Valavanis, 2009).

In this research Recurrent Neural Network is chosen as the financial stock price is time-series data, and the recurrent characteristics of RNN are needed for accurate prediction. The train-test data is identified, then the data is scaled using *MinMaxScaler* to train the data. Data normalization is an important step where the independent variables of data are standardized.

The parameters of the network need to be tuned for the model to perform well. Placeholders must be defined to hold the expected values, finally, the loss function is defined to calculate the Mean Squared Error (MSE). After running the model, the MSE value will indicate whether the parameters of the RNN need to be tuned or not.

3.3 Building the LSTM Model

The main goal behind building the models is to forecast the close price of the index based on the historical stock price Open, Close, High, Low and Volume data. Generally, a basic LSTM cell has three gates which are responsible for keeping the values over time intervals. In other words, the LSTM cell has its memory which enables it to act like a brain cell when making decisions. Defining the cost function is an important step in building up the model. In the RNN model, the cost function will be the ReLU (Rectified Linear Unit). As LSTM learns the weights of each layer node by node, the activation function will define which nodes to be activated for the following layer. ReLU activation function was selected as it is a non-linear activation function, and its computational requirements are the least among the rest of the non-leaner activation functions. The LSTM used in building up the model is basic where each cell has the number of units equivalent to the number of nodes in a feed-forward neural network.

The aim of this research is to build a model with error less than 6%. In other to achieve this, two models will be built for performance comparison and evaluation. The first model starts with an embedding layer that is responsible for creating dense vectors for incoming input. The

embedding layer will help reduce the dimensionality. In the embedding layer, similar input will have similar embeddings which will enable the model to understand the context of the input data.

The second layer in the LSTM model will have several memory units (smart neurons). The last Dense Layer with several neurons will convert the encoded vectors to one dimension. Finally, compiling the model will happen by using the Root Mean Squared Propagation "RMSprop" optimizer and MSE as a loss function. The RMSprop optimizer is very close to the concept of the gradient descent algorithm. The RMSprop will "restricts the oscillations in the vertical direction" meaning the increase of the learning rate can take place horizontally. Mean squared error (MSE) is used as a loss function that will calculate the squared difference between true and predictive values. The number of epochs is another important hyperparameter, which is the number of completer iterations through the training dataset. Also, in neural networks the training of the model happens in rounds (epochs), in each epoch, the network learns and changes the weights accordingly.

To prevent the model from overfitting, a Dropout layer can be added. These hyperparameters and others (like the cost function, the type of initialization for the starting weights, the number of epochs, the batch size, and the learning rate during the training process in the following experiments) can be edited accordingly to provide a different performance.

The second model will be built to compare and select the better performing model. In the second model, 60 days only will be chosen as a historical period for this model. The difference between the first model and the second model will be in the chosen the optimizer.

In this model the chosen optimizer is adaptive moment estimation (ADAM). ADAM optimizer will incorporate adaptive learning rates based on momentum. The optimizer will help the network learn weights in the best possible manner by minimizing the loss function.