Stock Price Prediction using Recurrent Neural Networks and LSTM

by

Fatin Ishrak

Abstract

The nature of the stock market has always been ambiguous as it always fluctuates for various factors. It has always been difficult for the investors to invest because of the regular fluctuations. Here, this project aims to significantly reduce the risk of trending in the stock market using previous dataset with machine learning. This research leads to finding the most effective prediction model that generates the most accurate result with the lowest error percentage. Also, this could help predict financial outcomes and generate significant economic impact all over the country. Our model outperformed discussed other models and able to predict next 60 days stock market data with very minimal loss.

Keywords: Machine Learning; Stock Market; Prediction; RNN; RMSE; MAPE; LSTM

Table of Contents

List of Figures

Chapter 1

Introduction

The basic goal of statistical methods, a well-known investment approach, is to create a result that is uncorrelated to the market by speculating with various directions and positive expected returns [9]. Distance [5] and cointegration procedures [8] are used in traditional statistical methods, and the relationship between the two is assessed using the distance metric and the test of cointegration, respectively. This statistical framework has been created in the past with a number of stock market prediction models [9], [14], [16] due to the significance of the models. The financial market is quite unpredictable, and predictions are frequently skewed due to a number of factors. Taking this into account, this article employs LSTM to create a multi-parametric model with time series data for stock market prediction.

In the world of finance, stock market forecasting is crucial because, if done correctly, it may limit losses and lower market risks while also providing significant benefits. Financial data are typically very volatile, melancholy, nonstationary, and nonlinear [6]. Because of this, the forecast is particularly troublesome and has recently received a lot of attention from academics and the financial industries [1], [13], [7]. The prediction approach is reliant on models for time series analysis. Similar to the autoregressive integrated moving average (ARIMA) [19], this model is exceptional. The generalized autoregressive conditional heteroskedasticity (GARCH) model [10] is another common model that is similar to ARIMA. These algorithms identify failure-prone linear series that capture nonlinear financial data trends. The biological neural networks that process data [25], [34], [33] are inspired by the computing paradigm known as the neural network, which can handle nonlinear problems and advance numerous fields of artificial intelligence, such as natural language processing [30] and computer vision [23]. Similar to this, neural networks have gained popularity for forecasting time series data used in finance. Because it provides access to historical time series values through recurrent connections, the recurrent neural network (RNN) is one of the most well-known neural network techniques that has been employed extensively in the financial world [18], [26]. Unfortunately, RNN suffers from the vanishing gradient issue when utilized as a standard recurrent unit since back propagation of time training is challenging. A recurrent unit called long short-term memory (LSTM) [3] was proposed as a solution to this issue. Long-term memory storage is made possible by the LSTM, which has memory cells with self-connections that can store temporal states of data and multiplicative gates that can govern the flow of information. Even on a wide variety of data, including machine

translation [22], speech recognition [31], remaining life prediction [21], virus spread prediction [29], etc., LSTM works well for sequential data modeling, such as stock market price.

1.1 Research Problem

The LSTM functions effectively when given a single variable input, such as when performing word embeddings for machine translation or digitizing audio for speech recognition. To build the model, the LSTM uses the multi-variable input in the same manner as a long vector. As a result, there is no unique benefit to the multi variable input [32] [35]. Contrarily, the stock market contains complex data that may be influenced by a number of factors, with prior values being omitted. In order to forecast the stock market, it is crucial that recurrent units manage the multi-variable inputs. Thankfully, Guo et al. introduced the multi-variable LSTM in [32]. In contrast to the vector utilized in the classic LSTM, the hidden state is taken into account in the multi-variable LSTM as a matrix. As a result, the neurons benefit since the hidden states exclusively encode the features from each varied input. Because it can comprehend the features of each variable, LSTM is required for better prediction.

1.2 Research Objective

In this paper, an LSTM-based stock market prediction is implemented because LSTMs perform well when applied to time series data with multidimensional input. Below is a summary of this paper's primary contribution.

- This study employs the LSTM, which is created to solve time series data with regression, to handle the categorization of the time series so that it can function properly and accurately forecast the model. For a better understanding of the model, this paper provides a rather thorough discussion of the LSTM model, its workings, and overall design.
- Based on the outcome of the model's forecast, this paper carries out the strategic stock market investment process. This effort uses BRAC Bank stock data that includes pandemic scenario for this purpose. The model's performance was superior to that of other statistical techniques like SMA and EMA, and it can offer better investing strategy.

The rest of this paper is structured as follows. The related literature on the use of stock prediction models in statistical arbitrage is reviewed in Chapter 2. The models utilized in this paper are introduced in Chapter 3. The experimental results are described in Chapter 4. The pre-processing method and model implementation with performance assessment are illustrated in Chapter 5. The experimental results are displayed and examined in Chapter 6. The paper is concluded in Chapter 7 with a discussion of limitations and further research.

Chapter 2

Literature Review

In this research, the stock market prediction survey results have been taken into account.

2.1 Background

Over the past 20 years, the popularity of stock market forecasting in the research world has increased significantly. Major cases are discovered to use macroeconomic factors like stock returns as the only input for the first phase of study with the utilization of non-linear slants as financial exchange record returns. For the reason that financial exchange returns are raucous, uncertain, confused, and nonlinear in nature, this strategy focuses on nonlinear expectations of the stock market. Numerous functions, such as the binary threshold, the linear threshold, the hyperbolic sigmoid, and the brown functions, have been employed to forecast different values.

Since the stock market is a target market for the financial industry, predictions have been made using machine learning techniques. One of them is specific assessment, but it cannot consistently produce specific results, so it is crucial to develop methods for progressively more precise measurement. The backslide approach has its own limitations in light of the challenges and steps involved. This model behaves in a manner similar to the least squares approach, however it was fitted in an arbitrary sequence. For instance, by lessening a handicapped variant of the least squares setback work or the "non-appearance of fit" in another standard. Again, fitting nonlinear models can be done using the least squares method.

The influence ratio of financial and technical analysis on stock market forecasting employing random forest, AI, and various man-made indicators is highly exceptional, as is well known. Researchers have spent time identifying the optimal technique and making future advancements in order to increase accuracy. As a result, there are several ways to apply the recent stock market predictions. It should come as no surprise that no model is ideal for financial analysis like stock market. Each methodology has its own usage restrictions. In the previously mentioned paper, the stock value estimation was completed using the self-assured Timberland estimating, which is being used to indicate the cost of the stock using fiscal extents structure the prospective quarter. This is one method for optically visualizing the incident by moving approaching it with the use of a clever model that uses erratic behaviour to

forecast the future cost of the stock from recorded data. On the other hand, there are a variety of additional factors that affect the stock market and can cause changes. The accuracy of the stock value forecast model can be increased by using the cash related size in close proximity to a model that can strongly separate assumptions, such as the money-related authority's suspicions, the association's overall opinion, news from other sources, and even circumstances that cause the complete trade protection to alter.

[9] explains how challenging it is to produce stock value using a multi-source sample without being aware of protective trades. However, because to the internet's ability to connect academics, the process has become less challenging with the utilization of many variables over time. Using several approaches and options based on precise, reliable data, such as using a feeling analyzer to suggest a striking connection between people's feelings and how their passion for express stocks affects them, are the only ways that budgetary trade information can be adequately predicted. The web news also made it simple to access the stock market, which informed everyone of its recent ups and downs. The utilization of historical data analysis and its effects on stock market price prediction were also discussed in the article. The cost of the stock or offer might be anticipated using historical data and its examples, but in practice it is necessary to use counts to foresee the expenses. The traditional frameworks only care about the diversity of an element used for forecasting. The latter is frequently accomplished with the aid of Genetic Algorithms (GA) or Artificial Neural Networks (ANN's) [16], but these tools fail to build a long-distance relationship between their stock costs transient dependencies.

RautSushrut et al. [2] estimate the stock market price movement using stock index data and a supervised learning classifier. A statistical AI methodology has been modeled after the computational analytical approach. Thus, the employment of support vector machines has demonstrated a strategic method of stock price prediction. In order to anticipate stock market prices, Manoj et al. [4] developed an LSTM network that can handle a linear problem. LSTM with hidden layers was employed by Roondiwala et al. [14]. The typical neuron was replaced with a memory cell from the LSTM, which may assist the memory cell interact with other nodes that remote access the input time effectively and construct a dynamic data capture model with high forecasting accuracy. The NIFTY50 dataset was used for the study. The acquisition of data is a crucial step in the model-training process because the algorithm differs greatly depending on the dataset. The topic that helps to identify the stock market prediction problem was studied with ANN by Kim et al. [16]. They illustrate the massive amounts of data that the system keeps forgetting, which causes the neuron to break and makes their predictions inaccurate. This is frequently caused by two factors: loads are established self-assuredly, and the loads become closer to the system's terminus. In that paper, LSTM usage was suggested.

It is abundantly obvious from Selvin et althorough .'s analysis of financial data [32] that the prediction heavily depends on historical data and the reputation of the company. Additionally, the SVM technique has been used to lessen the issue. According to Loke et al. [6], the stock market's volatility should be taken into account when creating a model, with historical data analysis taking precedence and

time series data explaining how to anticipate the stock market using conventional methods based on statistical methodologies. The limitations of their work have been addressed using machine learning and AI approaches in order to get results that are more accurate. Since the stock market is a dynamic data set, they came to the conclusion that there is no possible universal solution. According to Zhang et al. [27], stock market forecasting can be crucial in today's society. They demonstrated that predictions can be made with a high degree of accuracy if the data is accurate, can be obtained from reliable sources, and the approach is consistent with the data. RNN and LSTM were used by Xing et al. [22] to update the model with historical equity of share price. They took a certain attribute from the data and used it to make predictions. Opening price, day High, day Low, previous day o price, close price, and date of trading are the characteristics of shares. To predict the stock price of a certain time frame the suggested approach makes time series analysis. A CNN model was suggested by Prosky et al. [18] to evaluate the sentiment of the stock market. It is a more generalized form of the gated recurrent system, according to Xao [1]. The evanescent gradient problem that RNN has is solved by the LSTM that supposed to be less defected other than rest of the deep learning approaches like traditional feed forward neural network. With the help of K-means algorithm a short term stock market prediction has implemented with LSTM.

2.2 Related Work

This study analyzes the frameworks and statistical arbitrage of stock market forecasting models. Huck created a straightforward structure that is nonetheless adaptable for use in stock market forecasting [11]. The predicting, ranking, and trading processes make up the framework's main technique. It requires n stocks, and the function shows the data for stocks i and j at time t. The predictor creates a return projection for the following scale for each stock i. The dataset's time dimension is understood by the author using an Elman network as a predictor. A ranking of all stocks is produced using the $ELECTREIII$ outranking method based on these projected returns, with a matrix of predicted spreads serving as the decision matrix. The last m steps stock was sold out, and the top m steps stock was taken out during trade. Applying the $S\&P100$'s weekly return, which covers stock prices from 1992 to 2006, This study displays a superb outcome in terms of return and directional forecasting. Then, Huck conducted additional research on the subject and employed multi-step ahead forecasting, which was more flexible and realistic [12].

Similar but less complex framework was developed by Krauss et al. [15]. A predictor has been trained using the lag in stock input returns. then makes a prediction about the likelihood that a stock will one day outperform the market. Each stock has gone through this process once, with the stocks being arranged in descending order according to the prediction's probability. The high rank represents the future stock market success that is most anticipated. If the first k stocks are long and the final k stocks are short, the ranking can go up to $2k$ stocks. With a mean daily return of 0.43 percent and no concern for transaction costs, random forests [20] produce the greatest results when applied empirically to the stocks that made up the $S\&P500$ index from 1992 to 2015. Additionally, Fischer et al. created the work stated above using an LSTM network to forecast the direction of movement

of data for the $S\&P500$ stock index that is out of sample [16]. LSTM achieves a daily return of 0.46 percent on the identical data set before transaction charges. For the same issue, Shen et al. [24] updated the gated recurrent unit (GRU) with the RNN. The margin-based loss has been minimized in the GRU support vector machine model, which was constructed with $softmax$ operation. They fared better than conventional GRU and SVM, according to that experiment. LSTM was also utilized to forecast stock market data by Lee et al. [17]. But they build diversified portfolios by establishing cutoffs for classifying companies as long or short. Based on investors' preferred degrees of risk, his system can create portfolios with those levels.

Only the lagged returns have been used as input in the above formula, which is based on closing price calculations. Since stock market data is generated in the financial sector, it is more complicated and subject to several influences. Because of this, if a prediction model is just built on previous pricing, it may not be accurate. In their prediction model, the authors [27], [28] employed a few technical indicators. They use PCA to minimize the input's dimension after adding the LSTM feature. However, there were no dense layers in the conventional LSTM to compute the inputs. As a result, in this study we not only add more variables as input features but also develop the LSTM-based prediction model.

Chapter 3

Model Architecture

The goal of this study is to use artificial neural networks to improve the precision of daily stock index price estimates. We obtain information from current market activity, where technical analysts utilize a type of safety analysis to predict price direction by examining past data. Because the gathered data comprises a range of values on various scales, the time series must first be fitted and normalized in order to enhance network training. There must first be a stage of preparation for the data. Technical analysis is used to locate the best technical indicators, whereas principal component analysis is used to identify features and reduce data. The NARX model is subsequently constructed using the PCA functional subset. After that, a data sample is trained using a serial-parallel architecture. After the training operations, the serial-parallel architecture is transformed into a parallelized network to carry out prediction tasks. The thorough work schedule for this study is shown in the accompanying graphic.

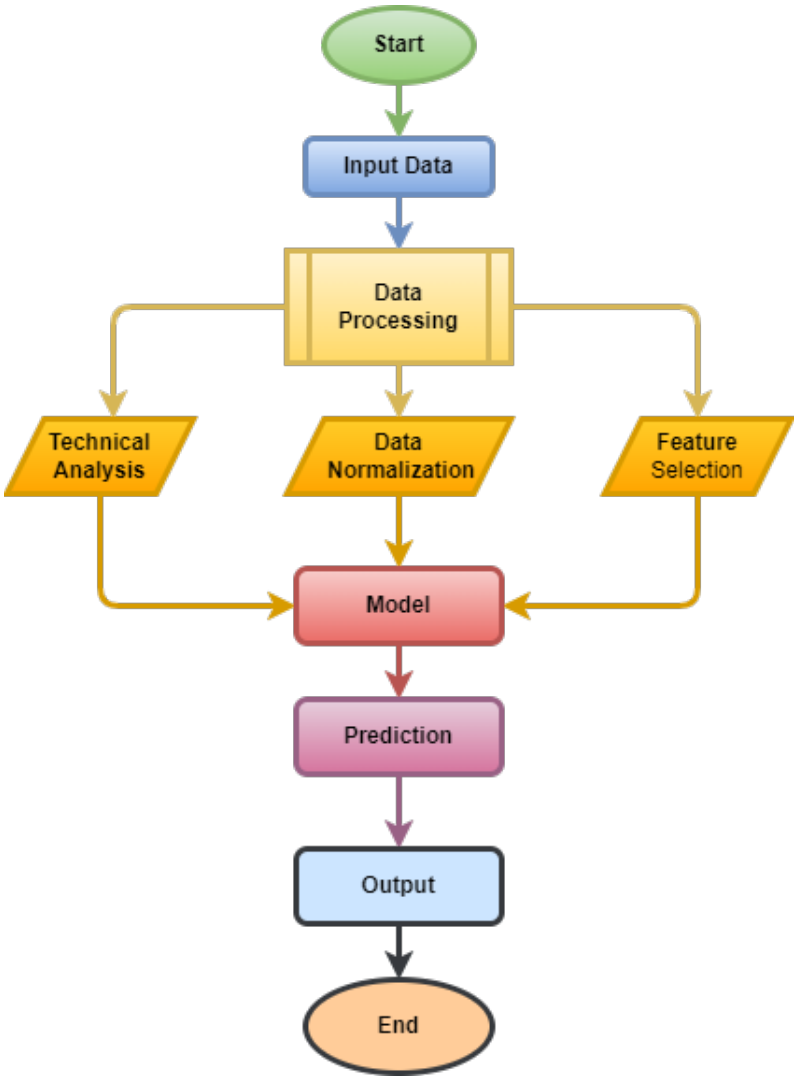

Figure 3.1: Work Plan

3.1 Principle Component Analysis

The principle component analysis (PCA) plays a vital role for the data that has too many fluctuations over the time scale such as stock market price. This paper implements two types of PCA and compares the result of the model for each. As the first PCA, this paper implements simple moving average (SMA). SMA provides the mean of N data points from the past and use the data to predict $N + 1$ value.

$$SMA = \frac{P_1 + P_2 + ... + P_n}{N} \tag{3.1}$$

P_1 to P_n are the data points that returns the past values where the SMA is the predicted value with size n. The size is the determiner of the precision value where the relation is the higher the better. However, the more size has been increased, it destabilize the model because of the granular fluctuations get smoothened off.

The second PCA this paper implements is exponential moving average (EMA). EMA allows greater weight to the past samples. Thus allowing the more n data without

diminishing the recent trade in fluctuations.

$$EMA = \ (P_t * k) + EMA_t - 1 * (1 - k) \tag{3.2}$$

(3.2) illustrates any given price at time t as P_t whereas k is the weight of the P_t. $EMA_t - 1$ is the EMA value of $t - 1$ point computed with the equation (3.2). The wieght has been measured as $k = 2/(N + 1)$.

3.2 Long Short Term Memory

To implement the PCA parameters manually it is very difficult and almost impossible to optimize. This research mitigates the problem using more complex model so that the model can compute the each past data point's significance and provide optimized predictions. Thus, the implementation of Long Short Term Memory (LSTM) model provides the weight updation while training the ML model.

LSTM as a RNN provides the scope to work on data sequences and ease the learning with retaining only the relevant information from the time scope. The extracted information from the network that is learned by the model is added to a memory which gets update after each timestamp based on the significance of the new information to the sample.

The model implements the LSTM cell each with three gates illustrates as the input gate, the output gate and the forget gate. The three gates combinedly performs to learn the weights and determine the ration of current data sample to be remembered and past learned context should be forgotten. The cell state C_t represents the short term and the long term internal memory of a cell.

3.2.1 Input Gate

The input gates has been implemented to select the new information to be added and stored in the current C_t. A *sigmoid* function is implemented to reduce the input vector (i_t) values.

$$i_t = \sigma(W_i.[h_t - 1, x_t] + b_i) \tag{3.3}$$

After that, a *tanh* function modifies each value between $[-1, 1]$ $(C - t)$. An element by element matrix has been multiplied of i_t and C_t that represents the information that requires to be added in the current cell.

$$\simeq C_t = \ tanh(W_C.[h_t - 1, x_t] + b_C) \tag{3.4}$$

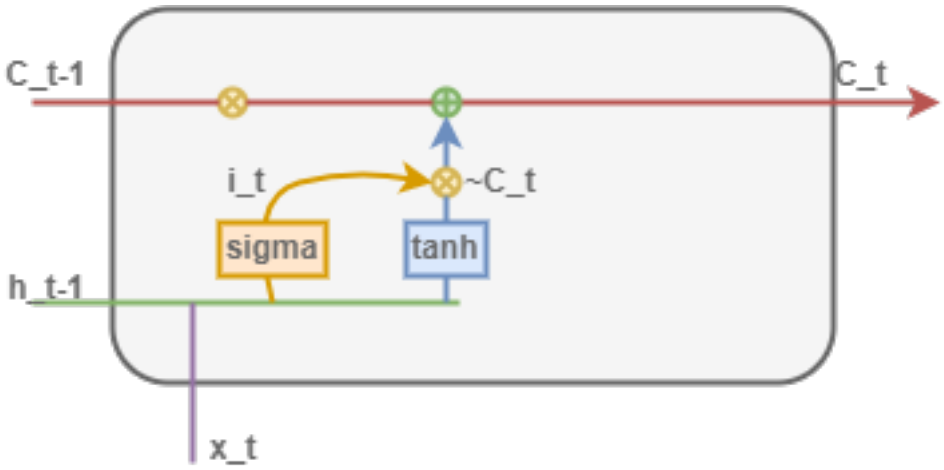

Figure 3.2: Input Gate

3.2.2 Output Gate

To control the output of the flowing to the next cell the output gate has been implemented. The output gate consists of a *sigmoid* function and then to filter the less important information a *tanh* function has been implemented. By this technique, the information needs to pass through is kept.The output (o_t) is calculated by the following equation.

$$o_t = \sigma(W_o.[h_t - 1, x_t] + b_o) \tag{3.5}$$

And the required (h_t) is as follows.

$$h_t = \ o_t * tanh(C_t) \tag{3.6}$$

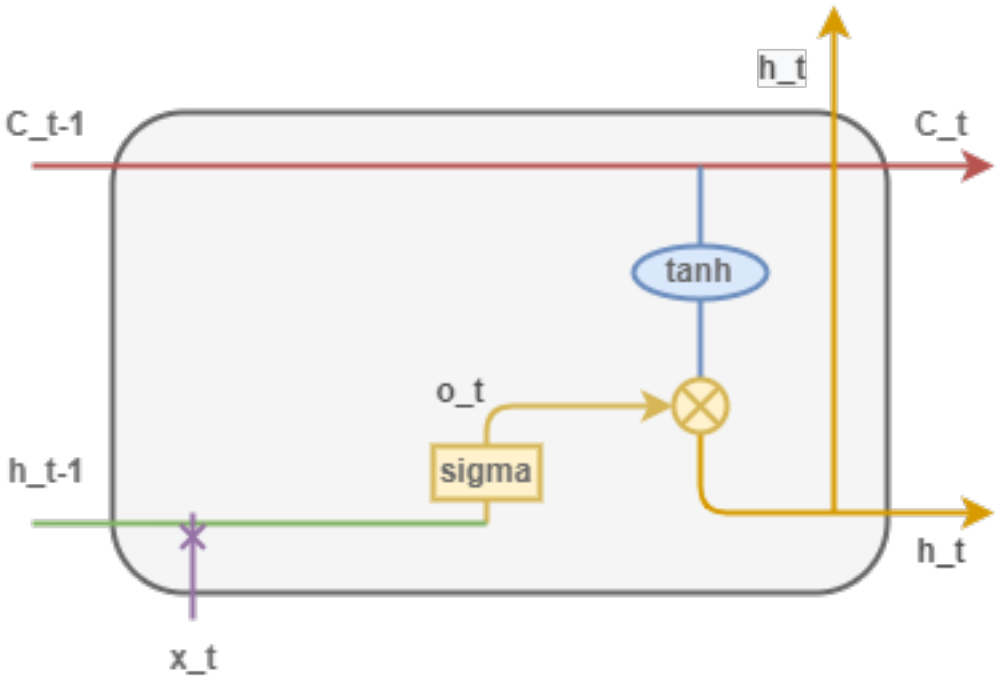

Figure 3.3: Output Gate

3.2.3 Forget Gate

The implementation of the forget gate confirms which information is to be forgotten by the model. The filtered information that the model can recognize as less important has been thrown away with this implementation of the forget gate. Mathematically, the forget gate has been implemented with the *sigmoid* function such as the output value range between $[0, 1]$ from the $C_t - 1$ state. 1 indicates the complete passing value and the 0 defines the completely filtered out values. The following equation has been implemented for the forget gate.

$$f_t = \sigma(W_f.[h_t - 1, x_t] + b_f) \tag{3.7}$$

The figure illustrates the functionality of forget gate in an LSTM cell.

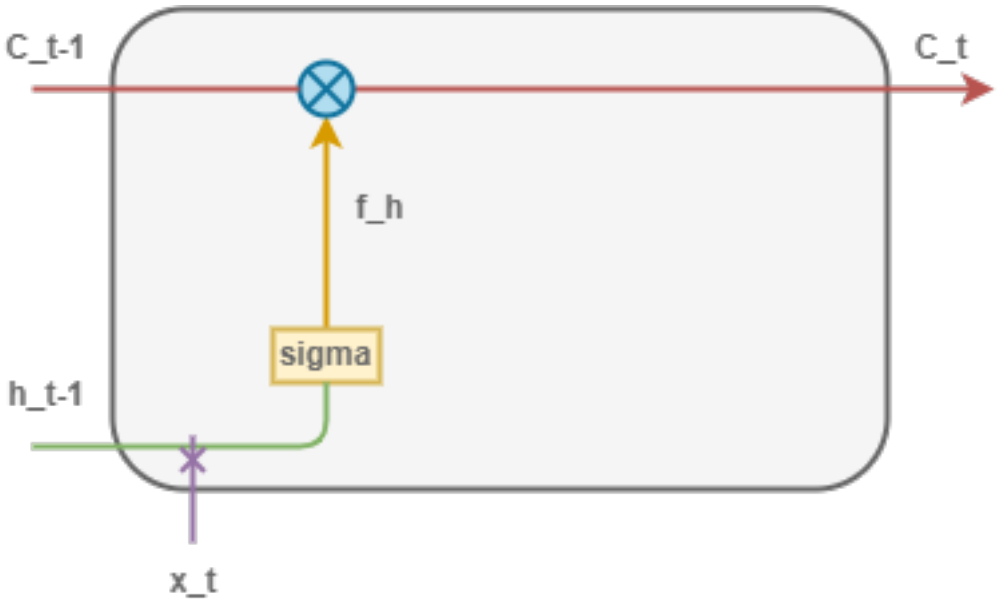

Figure 3.4: Forgate Gate

3.3 Performance Evaluation

In this section, the metric to measure the performance of the implemented model has been discussed. As stock price prediction is a regression problem, the two evaluation metric has been used. One of the used metric is Root Mean Squared Error (RMSE) and the other is Mean Absolute Percentage Error (MAPE).

To calculate the error RMSE uses the difference between the actual (A_t) and predicted ($F - t$) price values and returns an absolute error measure for N timescale. The following equation has been implemented to calculate RMSE.

$$RMSE = \sqrt{\frac{1}{N} * \sum_{t=1}^{N} (A_t - F_t)^2} \tag{3.8}$$

Next, MAPE considers the error looking at the true value thus measures relatively how off the predicted values are from the truth value rather than taking the actual difference. It helps dealing with data with more fluctuations and keep the error range in constant checking. The following formula has been implemented as the MAPE value.

$$MAPE = \frac{1}{N} * \sum_{t=1}^{N} \frac{A_t - F_t}{A_t} \tag{3.9}$$

Both of the metric of measuring the error is implemented to determine how close or far the stock price is after the prediction from the real world scenario.

Chapter 4

Dataset

For the implementation of the model, this paper uses the recent stock market data from Dhaka Stock Exchange (DSE). The recent stock price data has collected with the timescale of 15th September 2020 to 15th September 2022. For illustrating in this paper, the company BRAC Bank has been selected and all the data of the BRAC Bank and exploratory analysis has been shown in this paper. All the data of the DSE has been converted into the CSV file to prepossess and for the analysis.

4.1 Data Analysis

In this section, the exploratory data analysis has been discussed. To understand the data set, first it is required to visualize the data. For that purpose, the data frame has been visualized by the *pandas* library which is the most common data frame and manipulation library of python. The following figure illustrates the data.

DATE	#	TRADING CODE	LTP	HIGH	LOW	OPENP	CLOSEP	YCP	TRADE	VALUE	VOLUME
2022-09-15	1	BRACBANK	38.5	38.5	38.5	38.5	38.5	38.5	43	1.848	48012
2022-09-14	2	BRACBANK	38.5	38.5	38.5	38.5	38.5	38.5	39	0.793	20587
2022-09-13	3	BRACBANK	38.5	38.5	38.5	38.5	38.5	38.5	94	2.317	60191
2022-09-12	4	BRACBANK	38.5	38.5	38.5	38.5	38.5	38.5	81	4.452	115641
2022-09-11	5	BRACBANK	38.5	38.5	38.5	38.5	38.5	38.5	71	3.399	88276

Figure 4.1: Data Set

From the figure above we can see the tabular view of our model, here based on date the data is shown. The LTP stands for the Last Trading Price and the YCP stands for the Yesterday's Closing Price.

Because of using *Pandas* library to read the CSV file as dataframe and since the data is indexed by date, the data can also be indexed by the date column. For the analysis the data has taken from September 2020 to September 2022. It will also allow the model to visualize the trends for the model to work with the unpredictable occurrences such as COVID-19 situation. After loading data to the dataframe, it is important to see the main trends of the dataset to understand the fluctuations.

The following figure illustrates the HIGH and the LOW points of the BRAC Bank over the time scale.

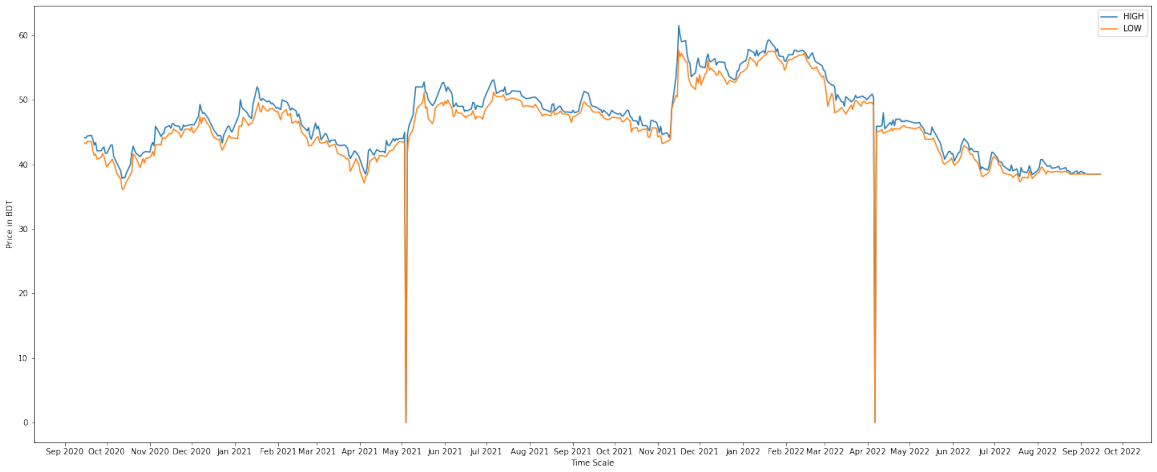

Figure 4.2: HIGH and LOW Points

As per the figure, it is visible that there are two sudden drop in the LOW price of the data, one is in May 2021 and another is in April 2022. Rest of the time it has the steady growth and downfall.

The challenging part for the ML model is to estimate the rapid changes of the data points correctly such as the values of the LOW in May 2021 and April 2022. This paper will focus on evaluating the model performance for predicting the recent values also another company share after training on the past data. Therefore, to mitigate the challenge it requires more parameters other than just the HIGH and the LOW values.

Like the same way applied previously to see the HIGH and LOW value, the OPEN and the CLOSE value is observed to analyse the data more vividly. Here, the open price is named as OPENP and similarly the close price of the dataset is named as CLOSEP. Plotting the Open and Close values for the time scale is shown below.

Figure 4.3: OPEN and CLOSE Price

Similar to LOW points, it is visible that OPEN price at May 2021 and April 2022 has gone down severely. As the high fluctuation can have impact on the model performance some other parameter has been taken under consideration.

Likewise the previous graphs, the Close price and Yesterday's Close price is been observed and illustrated in the following figure.

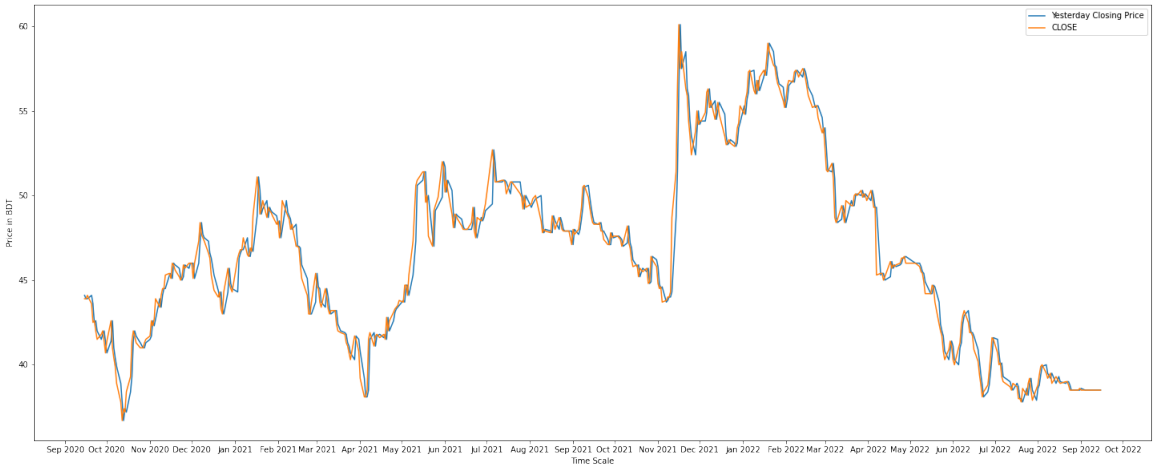

Figure 4.4: Close and Yesterday's Close Price

Here, it is clearly visible that there is no sudden fluctuations and it will help the model to train with good performance. The Volume of the data set should also need to visualize to understand the participation in the company by the traders that is another important aspect to determine the stock price for next day. The following figure illustrates the volume of the BRAC Bank stock market data.

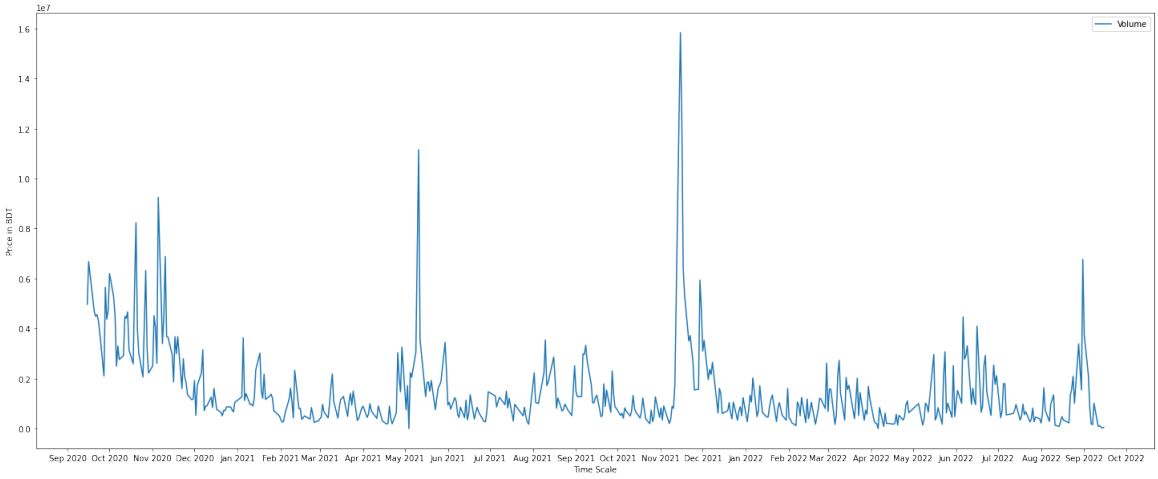

Figure 4.5: Volume

As seen in the figure, the volume is fluctuating more often. As the stock prices are unpredictable and contains high fluctuating values at time scale, it is important to get the right feature engineering processes to improve the model performance. The next section, the details analysis has been provided for the above analysis.

4.2 Stock Price as the Time Series Data

Despite the volatility, stock market price are not just a randomly generated numbers. Therefore, it is necessary to analyze the data as a sequence of discrete time data. Thus, the time series observation is granted as successive points in any given time scale over the time.

As the sequential nature of time series data, the sequence of information needs to aggregate. For the feature engineering the moving average is used that has the ability to smooth the short term fluctuations. This paper holds the 80% data to use for training purpose and 20% data for the prediction over trained values.

4.3 Fundamental Analysis

It is crucial to first look at the intrinsic values, such as tangible assets, investment statements, consumer behavior, and strategic objectives, for the fundamental analysis. For this to be an accurate predictor for the long-term investment, it depends on both historic data and current data. To calculate revenues, assets, liabilities, costs, and other financial metrics, historical and current data is crucial. However, in order to predict short-term market changes, it is necessary to analyze the stock market's technical elements, which are covered in the following part.

4.4 Technical Analysis

Technical analysis is the vital measure of the stock market price prediction with ML models. For this implementation, the measurable data from stock market activities has been thoroughly analyzed including stock prices, historical returns and the volume of historical trades. These quantitative information later used for identifying trading signals to identify the pattern of market movement of the stock market.

Technical analysis is been used to measure the short term trading purpose as the intention of the implementation is to predict the stock price for the short term instead of long term investment. For this paper, as a technical analysis, the sole focus is established on SMA and EMA and evaluated both to find the difference to see the model performance for each of them. LSTM as a deep learning RNN platform has utilized for the time series data and compared with the discussed technical analytical methods.

Chapter 5

Implementation

For data preprocessing, we used the scikit-learn library, Keras, and Tensorflow, with Keras acting as the front end of the machine learning. The data we used in this research includes a variety of parameters, including the most recent trading price, yesterday's closing price, the volume of BRAC BANK stock on the Dhaka Stock Exchange (DSE), as well as OPEN and CLOSE. We used RMSE and MAPE to optimize the model for training purposes.

5.1 Data Preprocessing

We used a fixed ranged normalizer for our model because the dataset was real-time data from DSE. By removing more significant numerical factors that can unfairly interact with the model and cause bias, this has improved our model and aided in achieving quick convergence.

We started by defining the features and the goal for this sector. After that, we rescaled the numbers using a *StandardScaler* function while maintaining a range of 0 to 1. Because stock prices fluctuate occasionally and can sometimes go from maximum to minimum, we restrict using $MinMaxScaler$. We choose the *ClosePrice* of the BRAC BANK stock data as the feature. After that, we divided the data in half and chose 80% of it for training. The coding portion of this method is shown in the following figure.

```python
data = stock_data_ft.filter(['CLOSEP'])
dataset = data.values
training_data_len = math.ceil(len(dataset) * .8)

training_data_len
```

Figure 5.1: Data Featuring

Then, using a *StandardScaler* with values between 0 and 1, we scale our data. The $MinMaxScaler$ function in the $Scikit-learn$ library is excellent in preprocessing data. However, it won't work well for stock price data because the price we receive on a given day could change significantly the following day. Therefore, combining different pieces of data won't produce improved results and will produce inaccurate predictions. As a result, we used the *StandardScaler* function as seen below.

Figure 5.2: Data Scaling

The dataset has since been split into training and testing sets. For each training step, the LSTM models require a timestep of data. Here, for 10 data samples, the LSTM will weigh the first 9 input samples in one step and then take 10 data samples to forecast the 10th one. As a result, this solution has adopted a more solid strategy than the one offered by $scikit-learn$, the $train\_test\_split$. The splitting function that this paper defines specifies a window size of n_steps that runs from the first location of the data to the nth step sample of the dataset. the latter section because the LSTM will use $n\_steps - 1$ samples for training and forecast the final sample for each time step. The major input for the loss calculation is the error of this forecast. Because the dataset is lacking the tenth data point for the current step, the remaining nine samples for the tenth step cannot be forecasted. The function starts with the dataset and builds the window size from there. The intended value for the n steps index has been retained by the y_train. The next 60 days has been selected to prediction. Below is an illustration of the function.

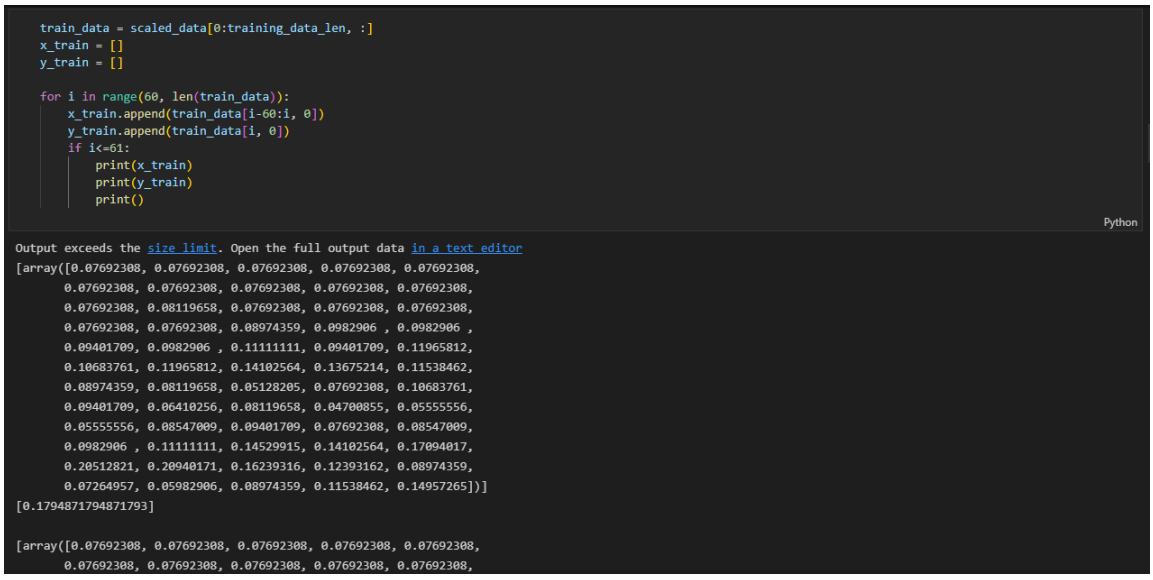

Figure 5.3: Train Test Splitting

After splitting the data, this paper used two steps' worth of historical data to forecast the outcome of the present price, despite the fact that time-series data does not allow for data shuffles. In order to forecast CLOSEP, the model will use YCP and OPENP. The dataset has been resized such that the array maintains a similar dimension and transformed into a numpy array for tensor calculation. The function is depicted in

the following figure.

```python
x_train, y_train = np.array(x_train), np.array(y_train)
```

Figure 5.4: Numpy Array Conversion

```python
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
```

Figure 5.5: Reshaping the Train Data

5.2 LSTM Model

Since LSTM is an application of RNN and RNN is based on sequential data, this implementation has used the ”Sequential” module for model definition and the ”LSTM” module for single unit LSTM models. To obtain the output, the dense layer has been introduced. A third three layer dense network has been added to the three layer LSTM model. The LSTM model is shown in the subsequent figure.

```python
model = Sequential()

model.add(LSTM(50, return_sequences=True, input_shape=(x_train.shape[1], 1)))
model.add(LSTM(50, return_sequences=True))
model.add(LSTM(50, return_sequences=False))

model.add(Dense(25))
model.add(Dense(10))
model.add(Dense(1))
```

Figure 5.6: LSTM Model

The model has then been fitted to the practice data. The following figure shows the output that the system generated.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 10, 100)           42400

 dense (Dense)               (None, 10, 1)             101

=================================================================
Total params: 42,501
Trainable params: 42,501
Non-trainable params: 0
_________________________________________________________________
```

Figure 5.7: Model Output

5.3 Performance Evaluation

The test set has been transformed into a *numpy* array for tensor calculation in order to evaluate the model. The array has then been modified to enable easy array multiplication. The model has then undergone prediction. Finally, the model is inverted so that the date might go from HIGH to LOW. The code is demonstrated in the following figure.

```python
x_test = np.array(x_test)
```

Python

```python
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1], 1))
```

Python

```python
predictions = model.predict(x_test)
predictions = scaler.inverse_transform(predictions)
```

Python

Figure 5.8: Model Prediction

Additionally, the RMSE and MAPE values have been verified to assess the model's performance. This section is covered in the chapter titled Results.

Chapter 6

Result

This article conducted great model loss where it is evident that loss reduction stagnates after as little as 15 epochs due to thorough preprocessing and clean data. After 20 epochs, the learning rate is reduced by the model. The graph of loss against the number of epochs is shown in the figure.

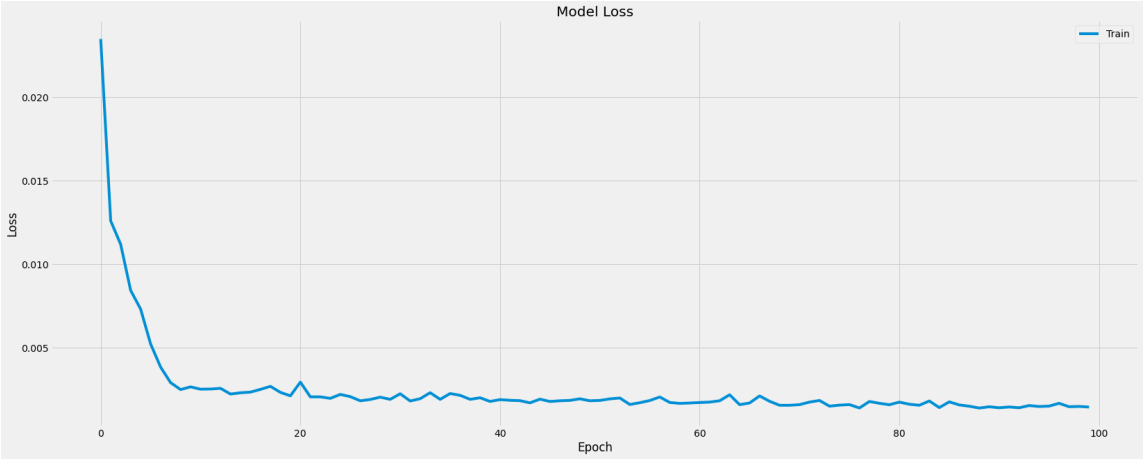

Figure 6.1: Model Loss

The forecast was then plotted to see how the model performed. On CLOSEP, the forecast was made. It is clear that the forecast and the valid set are so similar that the valid set can hardly be seen. As a result, this paper's research revealed that this model can, to a certain extent, mimic stock price movements. It can be seen that the model also fit the falling curve over recent prices based on the decrease of CLOSEP. The model's forecast is depicted in the next figure.

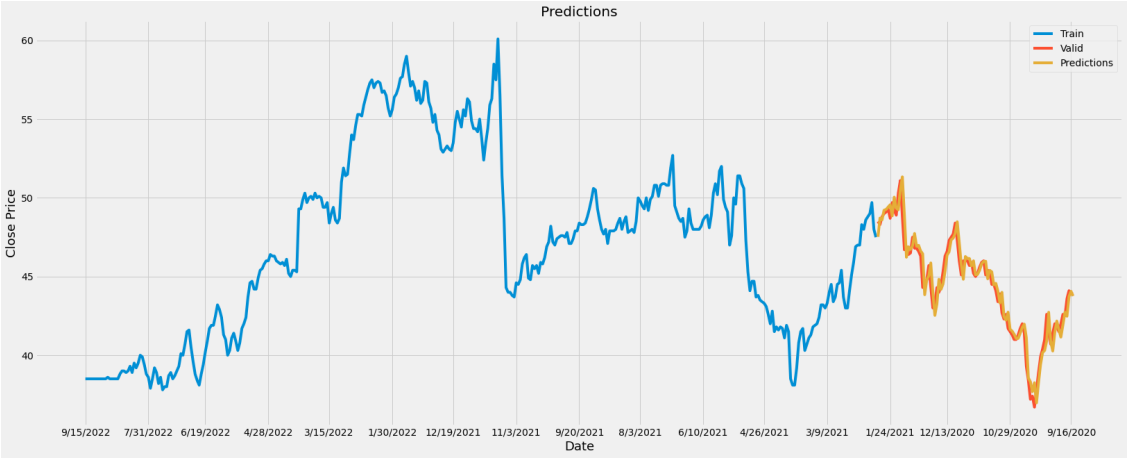

Figure 6.2: Model Prediction

6.1 Performance Evaluation on Test Set

With the use of a *numpy* array, the RMSE and MAPE have been calculated for the prediction and test set as shown below.

```
# RMSE
rmse = np.sqrt(np.mean(((predictions - y_test)**2)))
rmse
```

Python

```
0.8029217765691206
```

```
# MAPE
mape = np.mean(np.abs((y_test - predictions) / y_test)) * 100
mape
```

Python

```
1.406721782741874
```

Figure 6.3: RMSE and MAPE

The observable RMSE and MAPE values have been used to plot the forecast and the valid set. The forecast clearly followed the validation set in a good way.

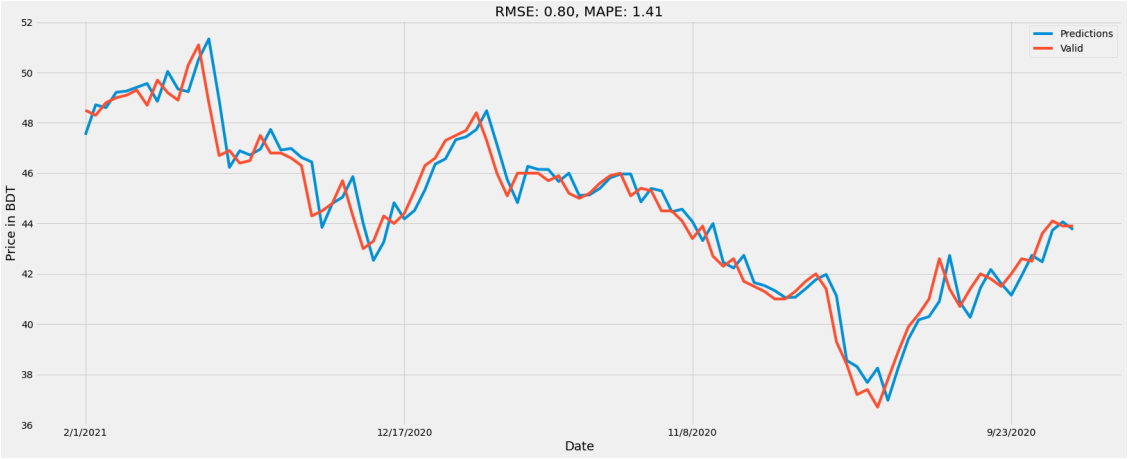

Figure 6.4: Prediction and Valid Set

Finally, the validation of the model is shown in the following figure where CLOSEP and Predictions are side by side compared. Here, it is clear that the model's fluctuation ranges from almost 1 in the beginning to as low as 0.14 after the completion of 100 epochs.

```
valid
```

Python

DATE	CLOSEP	Predictions
2/1/2021	48.5	47.523907
1/31/2021	48.3	48.715164
1/28/2021	48.8	48.603325
1/27/2021	49.0	49.215321
1/26/2021	49.1	49.263077
...	...	...
9/21/2020	42.5	42.734486
9/20/2020	43.6	42.476765
9/17/2020	44.1	43.734726
9/16/2020	43.9	44.064240
9/15/2020	43.9	43.762161

97 rows × 2 columns

Figure 6.5: Validation Results

6.2 Model Comparison

To compare the model, this paper discuss the statistical approaches like Simple Moving Average (SMA) and Exponential Moving Average (EMA).

6.2.1 Simple Moving Average

To SMA a *numpy* array with a split of 0.8 has been created from the train and test data. The mean has been computed using CLOSEP, and the SMA has been plotted with the prediction value. Figure comparing SMA and prediction as follows.

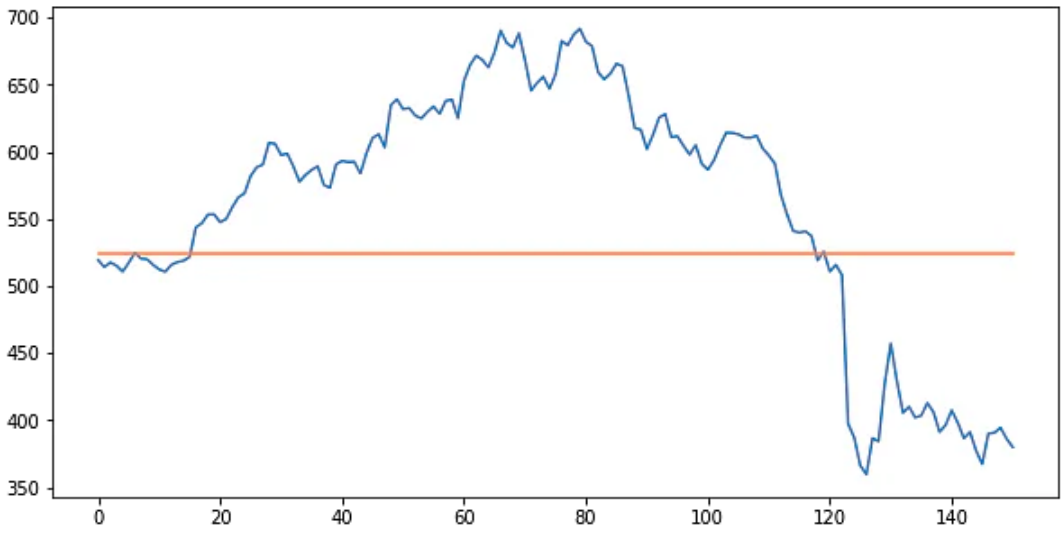

Figure 6.6: SMA vs LSTM

6.2.2 Exponential Moving Average

This work uses the Python *statsmodels* package, which contains the *SimpleExpSmoothing* routines, to implement EMA. The smoothing level was adjusted with a parameter during model fitting to achieve the best results. We discovered the same graph, but the RMSE and MAPE are significantly lower than SMA, indicating greater performance. The LSTM model, however, is well ahead of the SMA and EMA. By tweaking the hyperparameters, such as the amount of cells, batch size, or loss function, we can make our LSTM model better. However, using data up to 10 years old up to 5 years worth of data would be quite beneficial for the model. The EMA vs. LSTM model is shown in the figure.

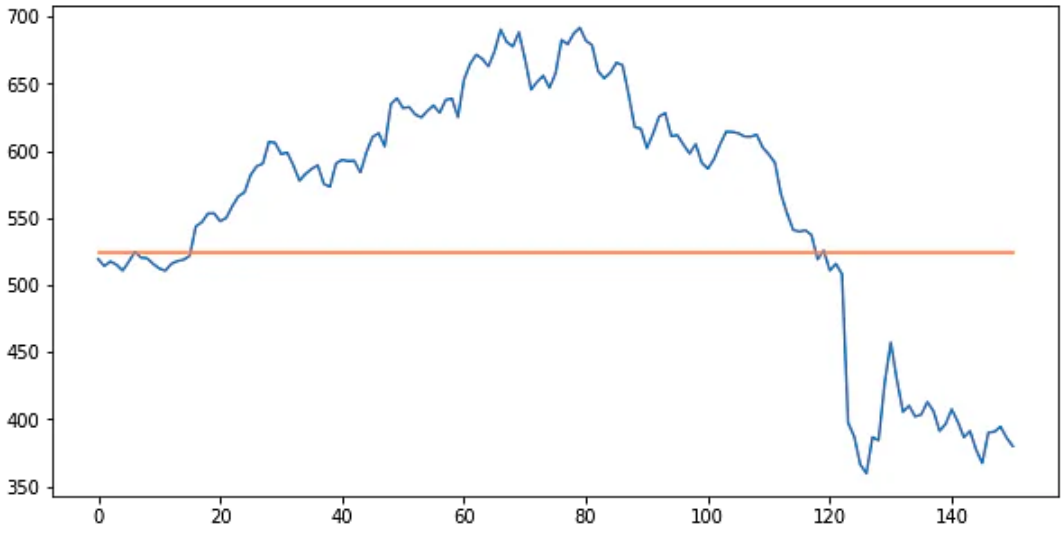

Figure 6.7: EMA vs LSTM

This study explored the use of stock market data as a time series to solve the

challenge of stock price prediction. The outcome also clarifies the logic behind three popular time series forecasting techniques: the simple moving average (SMA), exponential moving average (EMA), and LSTMs. Additionally, real-world stock data were used in this implementation to predict the values of the BRAC Bank stock using these three techniques. A performance comparison utilizing the RMSE and MAPE error metrics was also done.

Chapter 7

Conclusion

This work presents an application of the LSTM model for stock price prediction that can solve the single input variable problem on market time sequence. RNN and dense networks are used by the model for features that are similar to variables. Recurrent layer units use the LSTM to encode the lengthy sequential input with a dense matrix that stores the encoded variables. The attention mechanism that takes into account each feature has been engaged by the introduction of latent variables. Applications may be built based on this model and the RNN and dense layer that can handle the features. The experimental data consists of BRAC Bank data for the pandemic situation from 2018 to 2022. Given the significance of market risk and price prediction, this experiment outperformed the model outlined in chapter 2 and was able to anticipate future data with little loss.

Analyzing stock market changes is an important task for many reaserchers and study. Moreover, this study provides information based on PCA and ANN. The primary technique is often PCA, which chooses the appropriate input variables taking technical calculation indicators and the NARX model, which uses feature extraction from stock exchange data, into account. This PCA-NARX model outperforms ANN for time series data, such as financial market data. To sum up, the paper will focus on reducing the errors of the financial probability of the stock market through a model combining Principal Component Analysis (PCA) and Artificial Neural Networks (ANN). For the unpredictable fluctuation often the investors are leading towards counting risks over profits. Our model can minimize the probability of errors using the huge database and machine learning capabilities. By increasing the accuracy rate of previous data analysis simply the model will present a report with less errors.

The LSTM can handle the data as a time series, allowing the model to predict from a variety of features because the hidden layer or dense layer is tensorized. This study uses VOLUME and CLOSEP to forecast the stock market. To make things more complex and obtain more precise findings, however, several additional variables can be added. The resources for computing and the clean, readily available data are the main constraints. Our next project will involve expanding the model's functionality and using it in real time through a web application. This research can be done for a variety of market shares in the future. With the support of trustworthy data, a rising number of LSTM layers, and more processing capacity, predictions will be increasingly accurate. Last but not least, we can use sentiment analysis as a

parameter and refine it to get better results.

Bibliography

[1] C. Granger, "Forecasting stock market prices," *International Journal of Forecasting*, vol. 8, pp. 3–13, Jun. 1991. DOI: 10.1016/0169-2070(92)90003-R.

[2] C. Cortes and V. Vapnik, "Support vector network," *Machine Learning*, vol. 20, pp. 273–297, Sep. 1995. DOI: 10.1007/BF00994018.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735.

[4] ——, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735.

[5] W. Goetzmann, E. Gatev, and K. Rouwenhorst, "Pairs trading: Performance of a relative value arbitrage rule," *SSRN Electronic Journal*, vol. 19, Jan. 1999. DOI: 10.2139/ssrn.141615.

[6] R. Cont, "Empirical properties of asset returns: Stylized facts and statistical issues," *Quantitative Finance*, vol. 1, pp. 223–236, Jan. 2001. DOI: 10.1080/713665670.

[7] L. Feng, T. Dillon, and J. Liu, "Inter-transactional association rules for multi-dimensional contexts for prediction and their application to studying meteorological data," *Data Knowledge Engineering*, vol. 37, pp. 85–115, Apr. 2001. DOI: 10.1016/S0169-023X(01)00003-9.

[8] G. Vidyamurthy, "Pairs trading : Quantitative methods and analysis / g. vidyamurthy.," Jan. 2004.

[9] M. Avellaneda and J.-H. Lee, "Statistical arbitrage in the u.s. equities market," *Quantitative Finance*, vol. 10, pp. 761–782, Jul. 2008. DOI: 10.2139/ssrn.1153505.

[10] H. Shalit, D. Alberg, and R. Yosef, "Estimating stock market volatility using asymmetric garch models," *Applied Financial Economics*, vol. 18, pp. 1201–1208, Aug. 2008. DOI: 10.1080/09603100701604225.

[11] N. Huck, "Pairs selection and outranking: An application to the sp 100 index," *European Journal of Operational Research*, vol. 196, pp. 819–825, Jul. 2009. DOI: 10.1016/j.ejor.2008.03.025.

[12] ——, "Pairs trading and outranking: The multi-step-ahead forecasting case," *European Journal of Operational Research*, vol. 207, pp. 1702–1716, Dec. 2010. DOI: 10.1016/j.ejor.2010.06.043.

[13] J. Agrawal, D. V. Chourasia, and A. Mittra, "State-of-the-art in stock prediction techniques," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 2, pp. 1360–1366, Jan. 2013.

[14] C.-F. Huang, C.-J. Hsu, C.-C. Chen, B. Chang, and C.-A. Li, "An intelligent model for pairs trading using genetic algorithms," *Computational intelligence and neuroscience*, vol. 2015, p. 939 606, Aug. 2015. DOI: 10.1155/2015/939606.

[15] C. Krauss, X. Do, and N. Huck, "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the sp 500," *European Journal of Operational Research*, vol. 259, Oct. 2016. DOI: 10.1016/j.ejor.2016.10.031.

[16] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, Dec. 2017. DOI: 10.1016/j.ejor.2017.11.054.

[17] S. I. Lee and S. J. Yoo, "A new method for portfolio construction using a deep predictive model," in *Proceedings of the 7th International Conference on Emerging Databases*, Springer, 2018, pp. 260–266.

[18] K. Al-Thelaya, E.-S. El-Alfy, and S. Mohammed, "Evaluation of bidirectional lstm for short-and long-term stock market prediction," Apr. 2018, pp. 151–156. DOI: 10.1109/IACS.2018.8355458.

[19] X. Zhan, Y. Li, R. Li, X. gu, O. Habimana, and W. Haozhao, "Stock price prediction using time convolution long short-term memory network: 11th international conference, ksem 2018, changchun, china, august 17–19, 2018, proceedings, part i," in. Aug. 2018, pp. 461–468, ISBN: 978-3-319-99364-5. DOI: 10.1007/978-3-319-99365-2_41.

[20] S. Athey, J. Tibshirani, and S. Wager, "Generalized random forests," *Annals of Statistics*, vol. 47, pp. 1179–1203, Apr. 2019. DOI: 10.1214/18-AOS1709.

[21] F. Wang, X. Liu, G. Deng, X. Yu, H. Li, and Q. Han, "Remaining life prediction method for rolling bearing based on the long short-term memory network," *Neural Processing Letters*, vol. 50, Dec. 2019. DOI: 10.1007/s11063-019-10016-w.

[22] J. Zhang and C. Zong, "Deep learning for natural language processing," in. Feb. 2019, pp. 111–138, ISBN: 978-3-030-06072-5. DOI: 10.1007/978-3-030-06073-2_5.

[23] S. Liang, Y. Khoo, and H. Yang, "Drop-activation: Implicit parameter reduction and harmonious regularization," *Communications on Applied Mathematics and Computation*, vol. 3, Oct. 2020. DOI: 10.1007/s42967-020-00085-3.

[24] Y. Liu, J. Trajkovic, H.-G. Yeh, and W. Zhang, "Machine learning for predicting stock market movement using news headlines," Nov. 2020, pp. 1–6. DOI: 10.1109/IGESSC50231.2020.9285163.

[25] F. Meng, X. Zeng, Z. Wang, and X. Wang, "Anti-synchronization of fractional-order chaotic circuit with memristor via periodic intermittent control," *Advances in Mathematical Physics*, vol. 2020, pp. 1–8, Jan. 2020. DOI: 10.1155/2020/5158489.

[26] V. Chang, X. Man, Q. Xu, and C.-H. Hsu, "Pairs trading on different portfolios based on machine learning," *Expert Systems*, vol. 38, May 2021. DOI: 10.1111/exsy.12649.

[27] Y. Gao, R. Wang, and E. Zhou, "Stock prediction based on optimized lstm and gru models," *Scientific Programming*, vol. 2021, 2021.

[28] Y. Hu, "Stock forecast based on optimized lssvm model," *Computer science*, vol. 48, no. S1, pp. 151–157, 2021.

[29] S. Kumar, R. Sharma, T. Tsunoda, K. Thirumananseri, and A. Sharma, "Forecasting the spread of covid-19 using lstm network," *BMC Bioinformatics*, vol. 22, Jun. 2021. DOI: 10.1186/s12859-021-04224-2.

[30] R. Chiong, Z. Fan, Z. Hu, and S. Dhakal, "A novel ensemble learning approach for stock market prediction based on sentiment analysis and the sliding window method," *IEEE Transactions on Computational Social Systems*, vol. PP, pp. 1–11, Jan. 2022. DOI: 10.1109/TCSS.2022.3182375.

[31] F. Gao, J. Zhang, C. Zhang, X. Shuang, and C. Ma, "Long short-term memory networks with multiple variables for stock market prediction," *Neural Processing Letters*, pp. 1–19, Oct. 2022. DOI: 10.1007/s11063-022-11037-8.

[32] T. Lees, S. Reece, F. Kratzert, D. Klotz, M. Gauch, J. de Bruijn, R. Sahu, P. Greve, L. Slater, and S. Dadson, "Hydrological concept formation inside long short-term memory (lstm) networks," *Hydrology and Earth System Sciences*, vol. 26, pp. 3079–3101, Jun. 2022. DOI: 10.5194/hess-26-3079-2022.

[33] W. Wang, "Further results on mean-square exponential input-to-state stability of stochastic delayed cohen-grossberg neural networks," *Neural Processing Letters*, Aug. 2022. DOI: 10.1007/s11063-022-10974-8.

[34] X. Gu, A. Metcalfe, N. Cook, C. Aldrich, and L. George, "Exploratory analysis of multivariate drill core time series measurements," *ANZIAM Journal*, vol. 63, pp. C208–C230, Jan. 2023. DOI: 10.21914/anziamj.v63.17192.

[35] ——, "Exploratory analysis of multivariate drill core time series measurements," *ANZIAM Journal*, vol. 63, pp. C208–C230, Jan. 2023. DOI: 10.21914/anziamj.v63.17192.