STOCK VALUE PREDICTION USING DEEP LEARNING

A MINI PROJECT REPORT

Submitted by

KARTHIK.K (1902102)

NITHIN SOUNDAR. S. J (1902132)

RITHICK ROSHAN. R (190156)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

SRI RAMAKRISHNA ENGINEERING COLLEGE

[Educational Service: SNR Sons Charitable Trust]
[Autonomous Institution, Accredited by NAAC with 'A' Grade]
[Approved by AICTE and Permanently Affiliated to Anna University, Chennai]
[ISO 9001:2015 Certified and All Eligible Programmes Accredited by NBA]
Vattamalaipalayam, N.G.G.O. Colony Post,

COIMBATORE – 641 022

ANNA UNIVERSITY: CHENNAI-600 025

JULY 2021

SRI RAMAKRISHNA ENGINEERING COLLEGE

COIMBATORE – 641 022

BONAFIDE CERTIFICATE

16EC253- MINI PROJECT 1

Certified that this 16EC253 - Mini Project - I Report "**STOCK VALUE PREDICTION USING DEEP LEARNING"** is the bonafide work of **KARTHIK. K(1902132), NITHIN SOUNDAR. S. J (1902132), RITHICK ROSHAN.R(1902156)** who carried out the project under my supervision.

SIGNATURE
Mr. E. ESAKKI VIGNESWARAN, M.E.
Assistant Professor
Supervisor
Department of ECE
Sri Ramakrishna Engineering College
Coimbatore- 641 022

SIGNATURE
Dr. M. JAGADEESWARI, M.E., Ph.D.,
Professor
Head of the Department
Department of ECE
Sri Ramakrishna Engineering College
Coimbatore-641 022

Submitted for the Mini Project Viva-Voce Presentation held on 03.07.2021

INTERNAL EXAMINER

Mr.M.SELVAGANESH
Assistant Professor / ECE
Sri Ramakrishna Engineering College
Coimbatore- 641 022

EXTERNAL EXAMINER

Mrs.S.NAGARATHINAM,
Assistant Professor / ECE
Kumaraguru College of Technology
Coimbatore-641 049

Acknowledgment

We put forth our heart and soul to thank **ALMIGHTY** for being with us all through this technical adventure.

We express our heartfelt thanks to our esteemed Managing Trustee, **Sri. D. LAKSHMINARAYANASWAMY B.Tech., MBA,** and Joint Managing Trustee **Sri. R. SUNDAR, B.A, MBA.,** for their invaluable support.

We thank our Principal, **Dr. N. R. ALAMELU, B.E. (Hons.), M.E., Ph.D.,** for her benevolent patronage in carrying out this project.

We express our sincere respect and gratitude to our Professor & Head of the Department, Electronics and Communication Engineering **Dr. M. JAGADEESWARI M.E., Ph.D.,** for her continuous encouragement and motivation throughout the project.

We are obliged to our Mini Project coordinator, **Mr. M. Selvaganesh, M.E.,** for giving us essential technical expertise and support.

We are indebted to our Mini Project Guide, **Mr. E. Esakki Vigneswaran, M.E.,** for providing encouragement and guiding us throughout this project with enthusiasm and enabling us to complete it within the stipulated time.

We extend our thanks to all the teaching, non-teaching faculty members of our department and other departments, friends and family members for their support in carrying out the mini-project successfully.

ABSTRACT

Stock prices are driven by corporate earnings or profit expectations. If a trader thinks that the company's earnings are high or will rise further, they will raise the price of the stock. One way for shareholders to get a return on their investment is to buy low stocks and sell them at high prices. If the company performs poorly and the value of the stock declines, the shareholder will lose some or all of his investment at the time of sale. Therefore, accurate stock price information is important. Prediction of stock prices has been an important area of research for a long time. While supporters of the efficient market hypothesis believe that it is impossible to predict stock prices accurately, there are formal propositions demonstrating that accurate modeling and designing of appropriate variables may lead to models using which stock prices and stock price movement patterns can be very accurately predicted. In Stock Market Prediction, the aim is to predict the future value of the financial stocks of a company. This project has utilized the Long- Short Term Memory cell algorithm. LSTM are mini neural networks designed for larger neural networks.

These LSTMs at every feed forward iteration the cell can hold onto information from the previous step, as well as all previous steps. After pre-processing the collected data, it is split into training and testing datasets. LSTM algorithm is applied to the training dataset. The result is then analysed. By Using this algorithm, an accuracy of 0.968 has been achieved. The recent trend in stock market prediction technologies is the use of deep learning which makes predictions based on the values of current stock market indices by training on their previous values.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on a financial exchange. The successful prediction of a stock's future price will maximize investor's gains. This paper proposes a deep learning model to predict stock market price.

1.2 PROJECT OVERVIEW

When it comes to using deep learning in the stock market, there are multiple approaches a trader can do to utilize DL models. From determining future risk to predicting stock prices, deep learning can be used for virtually any kind of financial modeling.

Economic growth, politics, leadership and so many other factors influence the prediction of stock values.

We use algorithms to train with the data already present and then test the model for accuracy. The more training data, the more accurate model.

1.3 CHAPTER ORGANIZATION

In this report chapters are organized as follows.

- Chapter 1 gives the brief introduction of the project.
- Chapter 2 provides the literature review of the project.
- Chapter 3 gives Software description of the project.
- Chapter 4 explains the methodology of the project.
- Chapter 5 summarizes the experimental results.

- Chapter 6 discusses the conclusion and future work.

1.4 SUMMARY

This chapter deals with the basic information, overview of the project and chapter wise organization is given in this chapter.

CHAPTER 2

LITERATURE SURVEY

2.1. INTRODUCTION

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on an exchange. The successful prediction of a stock's future price could yield significant profit. The efficient-market hypothesis suggests that stock prices reflect all currently available information and any price changes that are not based on newly revealed information thus are inherently unpredictable.

2.2 BACKGROUND WORK

Mojtaba Nabipour et al (2020) have proposed Predicting Stock Market Trends Using Machine Learning and Deep Learning Algorithms Via Continuous and Binary Data; a Comparative Analysis. This study aims to significantly reduce the risk of trend prediction with machine learning and deep learning algorithms. Four stock market groups, namely diversified financials, petroleum, non-metallic minerals and basic metals from Tehran stock exchange, are chosen for experimental evaluations. This study compares nine machine learning models (Decision Tree, Random Forest, Adaptive Boosting (Adaboost), eXtreme Gradient Boosting (XGBoost), Support Vector Classifier (SVC), Naïve Bayes, K-Nearest Neighbors (KNN), Logistic Regression and Artificial Neural Network (ANN)) and two powerful deep learning methods (Recurrent Neural Network (RNN) and Long short-term memory (LSTM). Ten technical indicators from ten years of historical data are input values, and two ways are supposed for employing them. Firstly, calculating the indicators by stock trading values as continuous data, and secondly converting indicators to binary data before using. Each prediction model is evaluated by three metrics based on the input ways. The evaluation results indicate that for the continuous data, RNN and LSTM outperform other prediction models with a considerable difference. Also, results show that in the binary data evaluation, those deep learning methods are the best; however, the difference becomes less because of the noticeable improvement of models' performance in the second way.

Ge Li and Ming Xiao (2019) have proposed Application of Deep Learning in Stock Market Valuation Index Forecasting. Long short-term memory (LSTM) model in deep learning can effectively describe the long memory of data and is suitable for predicting financial time series. Therefore, this paper uses LSTM model in deep learning to learn and forecast the stock market valuation indicator, price-earnings ratio (P/E ratio). Then the prediction bias is measured by forecast trend accuracy (FTA), average forecast deviation rate (AFDR), and root mean square error (RMSE). Empirical results show that LSTM model has a good predictive effect on P/E ratio sequence, indicating that there is practical research value for applying deep learning network algorithm to the field of stock market forecasting. At the same time, this paper also provides a reference for stock market investors.

Srinath Ravikumar and Prasad Saraf (2020) proposed Prediction of Stock Prices using Machine Learning (Regression, Classification) Algorithms. There have been significant developments in this field. Many researchers are looking at machine learning and deep learning as possible ways to predict stock prices. The proposed system works in two methods - Regression and Classification. In regression, the system predicts the closing price of stock of a company, and in classification, the system predicts whether the closing price of stock will increase or decrease the next day.

Mehak Usmani et al (2016) proposed Stock market prediction using machine learning techniques. The main objective of this research is to predict the market performance of Karachi Stock Exchange (KSE) on day closing using different machine learning techniques. The prediction model uses different attributes as an input and predicts market as Positive & Negative. The attributes used in the model includes Oil rates, Gold & Silver rates, Interest rate, Foreign Exchange (FEX) rate, NEWS and social media feed. The old statistical techniques including Simple Moving Average (SMA) and Autoregressive Integrated Moving Average (ARIMA) are also used as input. The machine learning techniques including Single Layer Perceptron (SLP), Multi-Layer Perceptron (MLP), Radial Basis Function (RBF) and Support Vector Machine (SVM) are compared. The results suggest that performance of KSE-100 index can be predicted with machine learning techniques.

Silva et al,(2020) proposed Automated Trading System for Stock Index Using LSTM Neural Networks and Risk Management. In this paper, an automated trading system is built to predict future trends of stock index prices. Using an LSTM-based agent to learn temporal patterns in the data, the algorithm triggers automatic trades according to the historical data, technical analysis indicators, and risk management. The results demonstrate that the proposed method, called LSTM-RMODV, shows better performance when compared with other methods, including the buy-and-hold technique. The proposed method also works in bear or bull market conditions, showing a rate over net income based on invested capital of 228.94%. That is, despite the low accuracy, the algorithm is capable of generating consistent profits when all the transaction costs are considered.

2.3 SUMMARY

From all the proposed method, experts have been studying and researching on the various trends that the stock market goes through. One of the major studies has been the attempt to predict the stock prices of various companies based on historical data.

CHAPTER 3

METHODOLOGY

3.1 BLOCK DIAGRAM

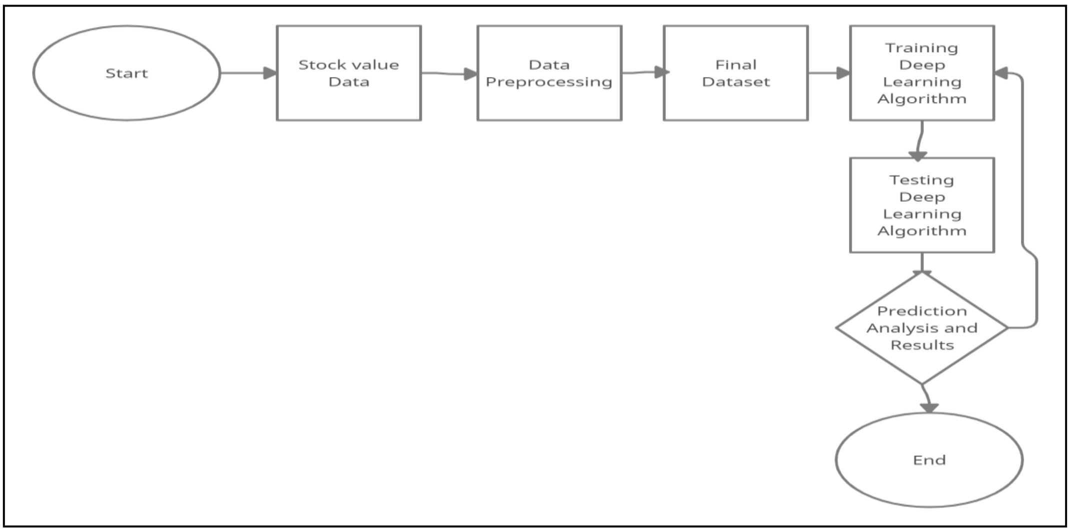


Fig. 3.1 Block Diagram

The figure 3.1 shows the process of Stock value prediction algorithm. Each process of the block will be explained detail below.

3.2 IMPLEMENTATION OF PREDICTION ALGORITHM

1. Import Libraries.
2. Import dataset.
3. Data summary and check for missing values. Drop the rows with any null values.
4. Plot a graph for Close price history of period vs close price graph.
5. Scale and transform the target and features column using MinMaxScaler.
6. Function to split dataset into Train and Test Dataset. Convert the x_train and y_train to numpy arrays. After the conversion, reshape the data.

7. Build the LSTM model. Compile the model. Train the model.

8. Prediction on Train and Test data.

9. Plot a graph for Prediction Model of period vs close price.

```
data.head()
```

Date	Open	High	Low	Close	Adj Close	Volume
2015-11-18	463.799988	465.649994	454.975006	456.000000	436.671021	5142766.0
2015-11-19	459.450012	469.350006	458.625000	467.375000	447.563873	5569752.0
2015-11-20	467.000000	476.399994	462.774994	473.424988	453.357422	5167930.0
2015-11-23	475.000000	478.950012	473.100006	476.875000	456.661224	4800026.0
2015-11-24	476.500000	485.799988	475.524994	483.850006	463.340515	6768886.0

Fig. 3.2 Input Data

Fig 3.2 shows the first five rows of the input data.

3.3 ACTIVATION FUNCTION

The choice of activation function in the hidden layer will control how well the network model learns the training dataset. The choice of activation function in the output layer will define the type of predictions the model can make. The activation function used in hidden layers is typically chosen based on the type of neural network architecture. The LSTM commonly uses the Sigmoid activation for recurrent connections and the Tanh activation for output.

The sigmoid activation function is also called the logistic function. It is the same function used in the logistic regression classification algorithm. The function takes any real value as input and outputs values in the range 0 to 1. The larger the input (more positive), the closer the output value will be to 1.0, whereas the smaller the input (more negative), the closer the output will be to 0.0.

The hyperbolic tangent activation function is also referred to simply as the Tanh (also “*tanh*” and “*TanH* “) function. It is very similar to the sigmoid activation function and even has the same S-shape. The function takes any real value as input and outputs values in the range -1 to 1. The larger the input (more positive), the closer the output value will be to 1.0, whereas the smaller the input (more negative), the closer the output will be to -1.0.

3.4 IMPLEMENTATION RESULTS

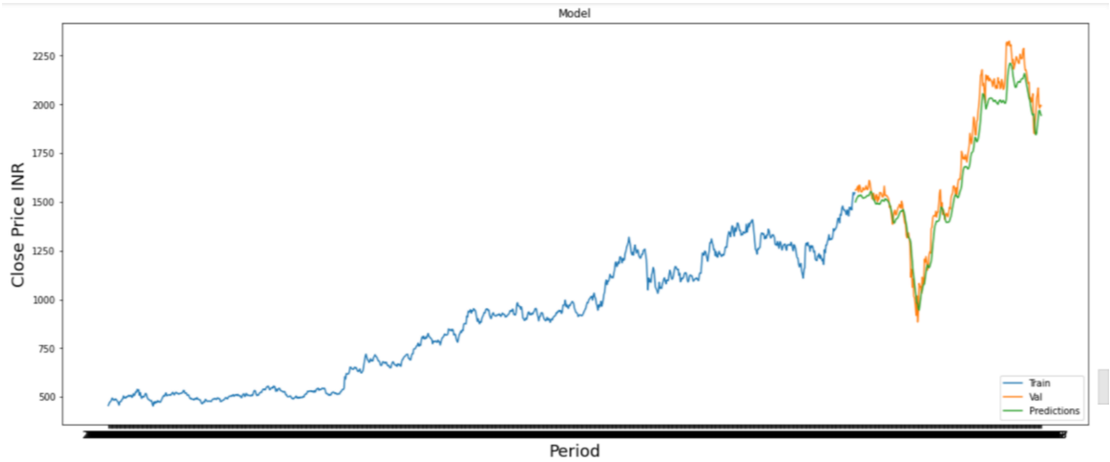


Fig. 3.3 Result Graph

Fig. 3.3 shows the graph between time and closing stock value of predicted and actual Reliance stock values using LSTM.

3.5 SUMMARY

In this chapter discusses the methodology of this project. It explains the method which is used in the prediction algorithm.

CHAPTER 4

SOFTWARE DESCRIPTION

4.1 INTRODUCTION

Stock prices are driven by corporate earnings or profit expectations. If a trader thinks that the company's earnings are high or will rise further, they will raise the price of the stock. If the company performs poorly and the value of the stock declines, the shareholder will lose some or all of his investment at the time of sale. Therefore, accurate stock price information is important. Prediction of stock prices has been an important area of research for a long time. In Stock Market Prediction, the aim is to predict the future value of the financial stocks of a company.

The recent trend in stock market prediction technologies is the use of deep learning which makes predictions based on the values of current stock market indices by training on their previous values.

4.2 DEEP LEARNING (LSTM)

Deep learning is an artificial intelligence (AI) function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Deep learning is a subset of machine learning in artificial intelligence that has networks capable of learning unsupervised from data that is unstructured or unlabelled. Also known as deep neural learning or deep neural network.

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series.

4.3 GOOGLE COLAB

Colaboratory or "Colab" for short is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

4.4 PSEUDOCODE FOR PREDICTION ALGORITHM

STEP 1 Import Libraries.

```
[ ] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    from sklearn.metrics import mean_squared_error
    from keras.models import Sequential
    from keras.layers import LSTM,Dropout,Dense
    from sklearn.preprocessing import MinMaxScaler
    from sklearn.metrics import accuracy_score
```

STEP 2 Import dataset.

```
data = pd.read_csv('/content/drive/MyDrive/Reliance.csv', index_col = 'Date')
data.head()
```

STEP 3 Data summary and check for missing values. Drop the rows with any null values.

```
[ ] data.dropna(axis = 0, inplace = True)
```

STEP 4 Plot a graph for Close price history of period vs close price graph.

```
plt.figure(figsize=(16,8))
plt.title('Close Price History')
plt.plot(data['Close'])
plt.xlabel('Period', fontsize=18)
plt.ylabel('Close Price INR', fontsize=18)
plt.show()
```

STEP 5 Scale and transform the target and features column using MinMaxScaler.

```
scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(close)

scaled_data
```

STEP 6 Function to split dataset into Train and Test Dataset. Convert the x_train and y_train to numpy arrays. After the conversion, reshape the data.

```
train_data = scaled_data[0:int(train_data_len), :]
#Split the data into x_train and y_train data sets
x_train = []
y_train = []

for i in range(60, len(train_data)):
    x_train.append(train_data[i-60:i, 0])
    y_train.append(train_data[i, 0])
    if i<= 61:
        print(x_train)
        print(y_train)
        print()

# Convert the x_train and y_train to numpy arrays
x_train, y_train = np.array(x_train), np.array(y_train)

#Reshape the data
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
# x_train.shape
```

STEP 7 Build the LSTM model. Compile the model. Train the model.

```
#from keras.models import Sequential
#from keras.layers import Dense, LSTM

#Build the LSTM model
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape= (x_train.shape[1], 1)))
model.add(LSTM(50, return_sequences= False))
model.add(Dense(25))
model.add(Dense(1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

#Train the model
model.fit(x_train, y_train, batch_size=1, validation_split = 0.1, epochs=3)
```

STEP 8 Prediction on Train and Test data.

```
test_data = scaled_data[train_data_len - 60: , :]

x_test = []
y_test = close[train_data_len:, :]
for i in range(60, len(test_data)):
    x_test.append(test_data[i-60:i, 0])

x_test = np.array(x_test)

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

preds = model.predict(x_test)
preds = scaler.inverse_transform(preds)

rmse = mean_squared_error(preds , y_test )
np.sqrt(rmse)
```

STEP 9 Plot a graph for Prediction Model of period vs close price.

```
train = target[:train_data_len]
valid = target[train_data_len:]
valid['Predictions'] = preds

plt.figure(figsize=(20,8))

plt.title('Model')
plt.xlabel('Period', fontsize=18)
plt.ylabel('Close Price INR', fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
plt.legend(['Train', 'Val', 'Predictions'], loc='lower right')

plt.show()
```

Fig. 4.1 Pseudocode for Prediction Algorithm

In fig 4.1 explains each step of the pseudocode for Prediction algorithm.

4.5 SUMMARY

In this chapter shows does not need any external microcontroller or circuit to find the book. A fully functional system is required to achieve the result. It also does not need any Image Processing or multispectral camera to capture book, a surveillance camera is enough to made easy and cost efficient.

CHAPTER 5

EXPERIMENTAL RESULTS AND DISCUSSION

5.1 INTRODUCTION

Reliance stock value has been used in this project. Stock Value data of about 1200 values is split into training and testing data in the ratio of 80:20. These data are fed into the LSTM Model and results were obtained. This data is obtained from Kaggle.com.

5.2 RESULTS

- The resulting graph when epochs at 3.

Date	Close	Predictions
2019-11-25	1561.550049	1603.417114
2019-11-26	1560.250000	1614.795532
2019-11-27	1569.849976	1621.120605
2019-11-28	1580.300049	1626.831543
2019-11-29	1551.150024	1633.610962

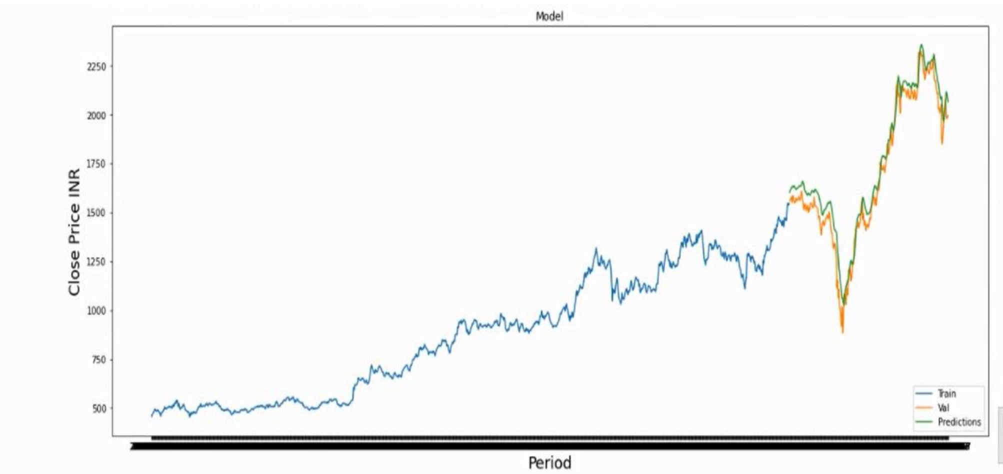


Fig. 5.1 Output-1

Fig.5.1 shows the values and graph of actual and predicted Reliance stock values using LSTM when epochs at 3.

- The resulting graph when epochs at 7

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

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

#Train the model
model.fit(x_train, y_train, batch_size=1, validation_split = 0.1, epochs=7)
```

```
Epoch 1/7
833/833 [==============================] - 40s 25ms/step - loss: 0.0032 - val_loss: 6.2493e-04
Epoch 2/7
833/833 [==============================] - 20s 24ms/step - loss: 5.0554e-04 - val_loss: 4.8935e-04
Epoch 3/7
833/833 [==============================] - 20s 24ms/step - loss: 5.5249e-04 - val_loss: 3.7316e-04
Epoch 4/7
833/833 [==============================] - 20s 24ms/step - loss: 2.4783e-04 - val_loss: 0.0015
Epoch 5/7
833/833 [==============================] - 20s 24ms/step - loss: 2.0970e-04 - val_loss: 0.0011
Epoch 6/7
833/833 [==============================] - 20s 24ms/step - loss: 1.7941e-04 - val_loss: 3.2008e-04
Epoch 7/7
833/833 [==============================] - 20s 24ms/step - loss: 1.5237e-04 - val_loss: 0.0011
<keras.callbacks.History at 0x7f56c0be8290>
```

Date	Close	Predictions
2019-11-25	1561.550049	1458.528687
2019-11-26	1560.250000	1469.604736
2019-11-27	1569.849976	1469.328735
2019-11-28	1580.300049	1476.974487
2019-11-29	1551.150024	1488.103027

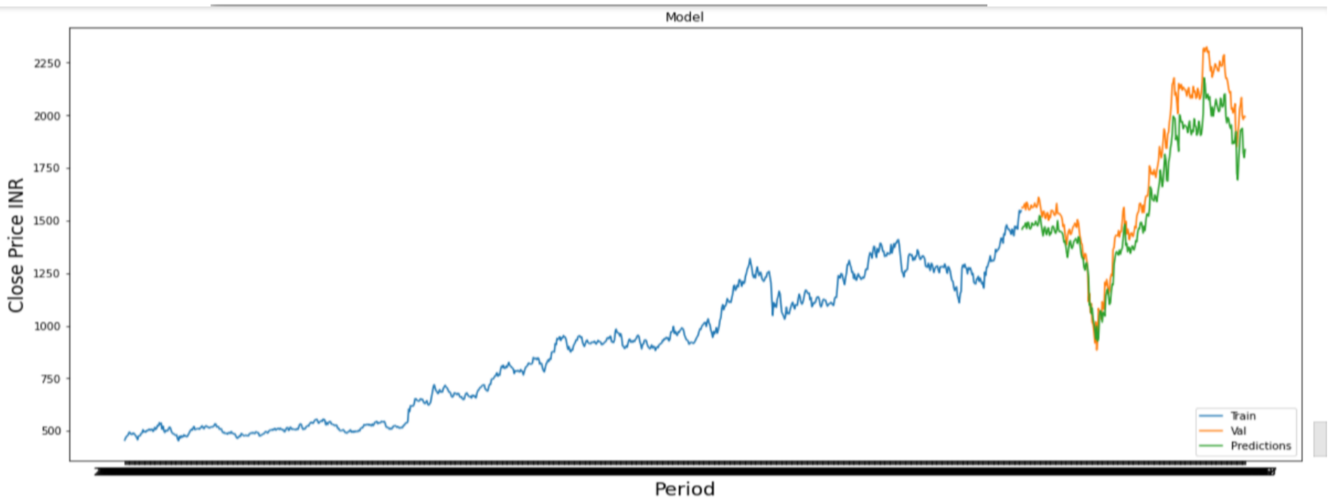


Fig. 5.2 Output-2

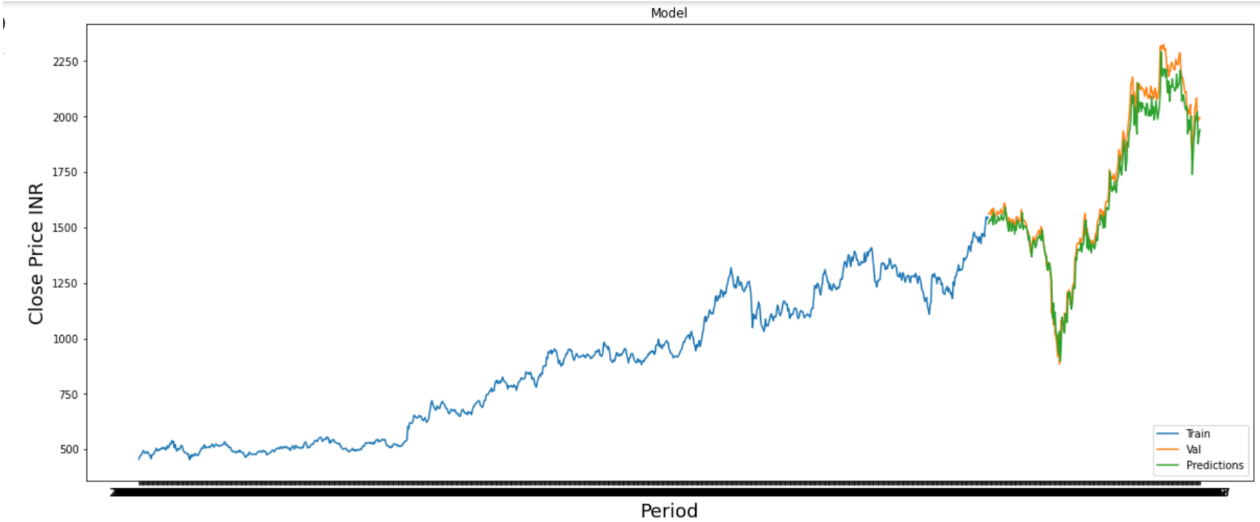
Fig.5.2 shows the values and graph of actual and predicted Reliance stock values using LSTM when epochs at 7.

- The resulting graph when epochs at 9

```
# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

#Train the model
model.fit(x_train, y_train, batch_size=1, validation_split = 0.1, epochs=9)
```

```
Epoch 1/9
833/833 [==============================] - 25s 26ms/step - loss: 0.0023 - val_loss: 0.0063
Epoch 2/9
833/833 [==============================] - 21s 25ms/step - loss: 5.4436e-04 - val_loss: 4.9223e-04
Epoch 3/9
833/833 [==============================] - 21s 25ms/step - loss: 4.3765e-04 - val_loss: 5.6322e-04
Epoch 4/9
833/833 [==============================] - 21s 25ms/step - loss: 2.7113e-04 - val_loss: 3.4284e-04
Epoch 5/9
833/833 [==============================] - 21s 25ms/step - loss: 2.9460e-04 - val_loss: 4.8435e-04
Epoch 6/9
833/833 [==============================] - 20s 24ms/step - loss: 2.4032e-04 - val_loss: 2.0747e-04
Epoch 7/9
833/833 [==============================] - 23s 27ms/step - loss: 1.5282e-04 - val_loss: 1.7543e-04
Epoch 8/9
833/833 [==============================] - 21s 25ms/step - loss: 1.2693e-04 - val_loss: 2.0685e-04
Epoch 9/9
833/833 [==============================] - 21s 25ms/step - loss: 1.7629e-04 - val_loss: 2.5092e-04
<keras.callbacks.History at 0x7f56b753a210>
```



Date	Close	Predictions
2019-11-25	1561.550049	1517.553589
2019-11-26	1560.250000	1533.612427
2019-11-27	1569.849976	1528.428589
2019-11-28	1580.300049	1541.221313
2019-11-29	1551.150024	1551.611084

Fig. 5.3 Output-3

Fig.5.3 shows the values and graph of actual and predicted Reliance stock values using LSTM when epochs at 9.

- Output of Stock values of Reliance Using Time Series Model

```
In [40]: df_valid[["VWAP", "Forecast_ARIMAX"]].plot(figsize=(14, 7))
```

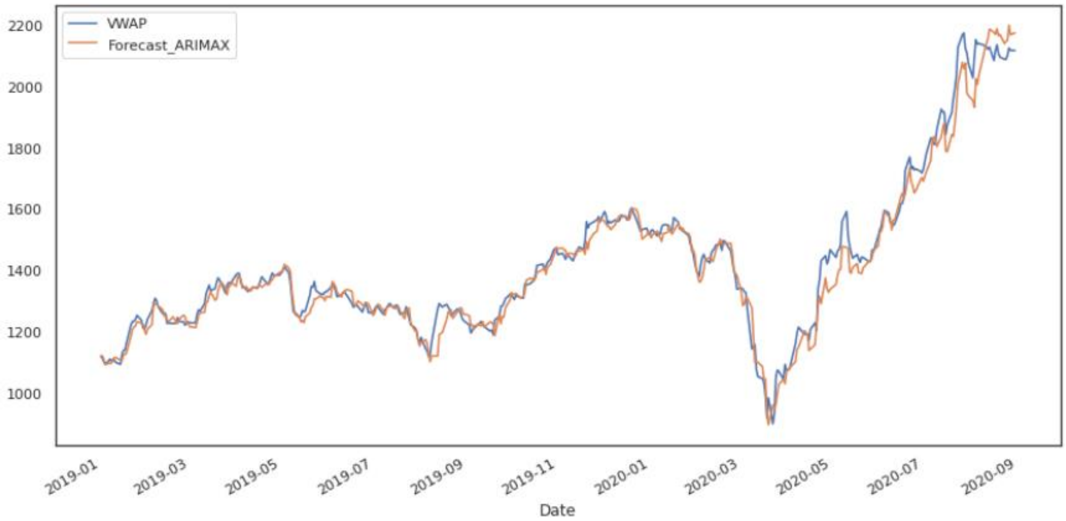


Fig. 5.4 Output-4

Fig. 5.4 shows the Output of Stock values of Reliance Using Time Series Model

Date	Close	Predictions
2019-11-25	1561.550049	1517.553589
2019-11-26	1560.250000	1533.612427
2019-11-27	1569.849976	1528.428589
2019-11-28	1580.300049	1541.221313
2019-11-29	1551.150024	1551.611084
2019-12-02	1586.500000	1512.656494
2019-12-03	1578.900024	1566.289307
2019-12-04	1552.699951	1546.039795
2019-12-05	1550.849976	1516.487549
2019-12-06	1554.900024	1522.532227
2019-12-09	1572.599976	1529.239868
2019-12-10	1561.949951	1550.612549
2019-12-11	1562.400024	1531.333496
2019-12-12	1568.199951	1534.704102
2019-12-13	1582.900024	1541.481201

Fig. 5.5. Output-5

Fig. 5.5 shows 15 rows of actual and predicted results of Reliance stock values using LSTM.

- Comparison between output of stock values of Reliance using LSTM and output of stock values of Reliance using Time Series Analysis.

S.No	MODEL	COMPANY	RMSE	ACCURACY SCORE
1	LSTM	Reliance	84.53635253496131	0.968
3	Time Series Analysis	Reliance	100.77775126216939	0.926

Fig. 5.6 Output-6

Fig. 5.6 shows comparison between output of stock values of Reliance using LSTM and output of stock values of Reliance using Time Series Analysis.

```
# Plot the data
train = data[:training_data_len]
valid = data[training_data_len:]
valid['Predictions'] = predictions
# Visualize the data
plt.figure(figsize=(16,6))
plt.title('Model')
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
plt.legend(['Train', 'Val', 'Predictions'], loc='lower right')
plt.show()
```

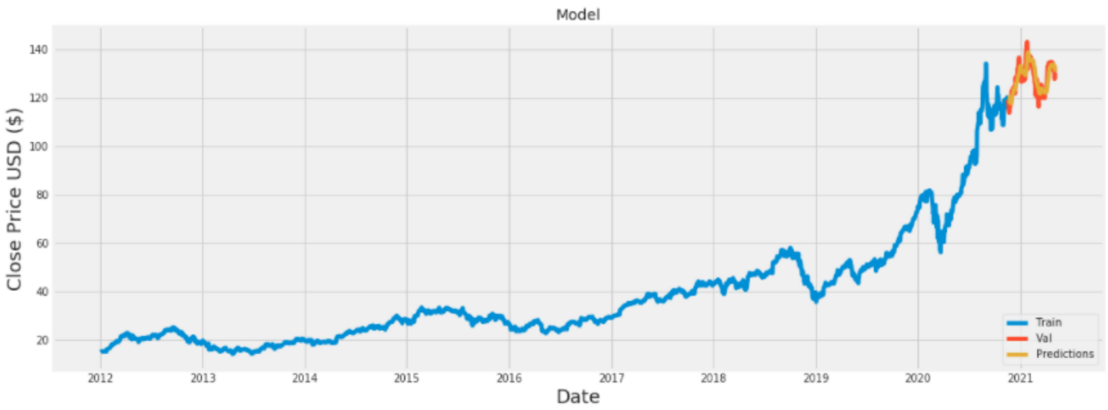


Fig. 5.7 Output-7

Fig. 5.7 shows the graph of actual and predicted stock values of Apple using LSTM.

5.3 SUMMARY

This chapter deals with the experimental results of the working algorithm. This shows the experimental results after implementation.

CHAPTER 6
CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

Predicting stock market returns is a challenging task due to consistently changing stock values which are dependent on multiple parameters which form complex patterns. The historical dataset available on company's website consists of only few features like high, low, open, close, adjacent close value of stock prices, volume of shares traded etc., which are not sufficient. This work consists of two parts: data extraction and pre-processing of the Reliance stock market dataset, and stock price trend prediction model based on the long short-term memory (LSTM). The stock values of Reliance were pre-processed, and the data was split into training and testing data. This data was fed into the training algorithm and results were obtained.

6.2 FUTURE SCOPE

For future work, deep learning models could be developed which consider financial news articles along with financial parameters such as a closing price, traded volume, profit and loss statements etc., for possibly better results.

REFERENCES

1. Nabipour, Mojtaba & Nayyeri, Pooyan & Jabani, Hamed & Band, Shahab & Mosavi, Amir. (2020). Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis on the Tehran stock exchange. IEEE Access. PP. 1-1. 10.1109/ACCESS.2020.3015966.
2. G. Li, M. Xiao and Y. Guo, "Application of Deep Learning in Stock Market Valuation Index Forecasting," 2019 IEEE 10th International Conference on Software Engineering and Service Science *(ICSESS)*, 2019, pp. 551-554, doi: 10.1109/ICSESS47205.2019.9040833.
3. S. Ravikumar and P. Saraf, "Prediction of Stock Prices using Machine Learning (Regression, Classification) Algorithms,*"* 2020 International Conference for Emerging Technology (INCET), *2020, pp. 1-5, d*oi: 10.1109/INCET49848.2020.9154061.
4. M. Usmani, S. H. Adil, K. Raza and S. S. A. Ali, "Stock market prediction using machine learning techniques," *2016 3rd International Conference on Computer and Information Sciences (ICCOINS)*, 2016, pp. 322-327, doi: 10.1109/ICCOINS.2016.7783235.,
5. T. R. Silva, A. W. Li and E. O. Pamplona, "Automated Trading System for Stock Index Using LSTM Neural Networks and Risk Management," *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1-8, doi: 10.1109/IJCNN48605.2020.9207278.

APPENDIX

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import LSTM,Dropout,Dense
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import accuracy_score
data = pd.read_csv('/content/drive/MyDrive/Reliance.csv', index_col = 'Date')
data.head()
data.dropna(axis = 0, inplace = True)
data.head()
data.shape
plt.figure(figsize=(16,8))
plt.title('Close Price History')
plt.plot(data['Close'])
plt.xlabel('Period', fontsize=18)
plt.ylabel('Close Price INR', fontsize=18)
plt.show()
target = data.filter(['Close'])
close = target.values
train_data_len = int(np.ceil(len(close) * .8 ))
train_data_len
scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(close)

scaled_data
train_data = scaled_data[0:int(train_data_len), :]
```

```python
x_train = []
y_train = []

for i in range(60, len(train_data)):
    x_train.append(train_data[i-60:i, 0])
    y_train.append(train_data[i, 0])
    if i<= 61:
        print(x_train)
        print(y_train)
        print()


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



x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape= (x_train.shape[1], 1)))
model.add(LSTM(50, return_sequences= False))
model.add(Dense(25))
model.add(Dense(1))

model.compile(optimizer='adam', loss='mean_squared_error')

model.fit(x_train, y_train, batch_size=1, validation_split = 0.1, epochs=5)
test_data = scaled_data[train_data_len - 60: , :]

x_test = []
y_test = close[train_data_len:, :]
for i in range(60, len(test_data)):
    x_test.append(test_data[i-60:i, 0])
```

```
x_test = np.array(x_test)

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

preds = model.predict(x_test)
preds = scaler.inverse_transform(preds)

rmse = mean_squared_error(preds , y_test )
np.sqrt(rmse)
train = target[:train_data_len]
valid = target[train_data_len:]
valid['Predictions'] = preds

plt.figure(figsize=(20,8))

plt.title('Model')
plt.xlabel('Period', fontsize=18)
plt.ylabel('Close Price INR', fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
plt.legend(['Train', 'Val', 'Predictions'], loc='lower right')

plt.show()
valid.head()
```