

A  
Group Project Report  
on

**“MARKET TREND ANALYSIS USING MACHINE  
LEARNING”**

Submitted in Partial Fulfillment of the Requirements

For the award of the Degree of

**Bachelor of Technology**  
in

**Electronics & Computer Engineering (ECM)**

Submitted

by

**OM JAJU (20311A1901)**

**Y VISHAL (20311A1904)**

**G PHANEENDRA (20311A1958)**

Under the Guidance of

**MRS.K. ARUNA KUMARI**

(Assistant Professor)

**Dept. of ECM**



**Department of Electronics & Computer Engineering**

**Sreenidhi Institute of Science & Technology**

**2022-23**

**DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING**  
**SREENIDHI INSTITUTE OF SCIENCE & TECHNOLOGY**  
**(AUTONOMOUS)**



**CERTIFICATE**

This is to certify that the Project work entitled “**MARKET TREND ANALYSIS USING MACHINE LEARNING**”, submitted by **Om Jaju (20311A1901), Y Vishal (20311A1904), G Phaneendra (20311A1958)**, towards partial fulfilment for the award of Bachelor Degree in Electronics & Computer Engineering from Sreenidhi Institute of Science & Technology, Ghatkesar, Hyderabad, is a record of bonafide work done by him/ her. The results embodied in the work are not submitted to any other University or Institute for award of any degree or diploma.

**Internal Guide**

K.Aruna Kumari  
Assistant Professor  
Department of ECM

**Project Coordinator**

K.Naga Sailaja  
Assistant Professor  
Department of ECM

**Head of Department**

Dr. D. Mohan  
Professor & HOD  
Department of ECM

**Date:**

**External Examiner**

## **ACKNOWLEDGEMENT**

We would like to extend our sense of gratitude to our guide **K.ARUNA KUMARI ,Assistant professor, ECM** and project coordinator **K.NAGA SHAILAJA,Assistant professor, ECM** for giving us their constant guidance, support and motivation throughout the period this course work was carried out. Her readiness for consultation at all times, his educative comments and assistance even with practical things have been invaluable. We are thankful that she gave us the freedom to do the work with my ideas.

We convey our sincere thanks to all the faculties of ECM department, Sreenidhi Institute of Science and Technology, for their continuous help, co-operation, and support to complete this project.

We are very thankful to **Dr. D. Mohan, Head of ECM Department**, Sreenidhi Institute of Science and Technology, Ghatkesar for providing an initiative to this project and giving valuable timely suggestions over our project and for their kind cooperation in the completion of the project.

We convey our sincere thanks to **Dr.T.Ch. Siva Reddy**, Principal and **Prof. CV Tomy**, Executive Director, Sreenidhi Institute of Science and Technology, Ghatkesar for providing resources to complete this project.

Finally, we extend our sense of gratitude to almighty, our parents, all our friends. teaching and non-teaching staff, who directly or indirectly helped us in this endeavour.

**OMJAJU – 20311A1901**

**Y VISHAL– 20311A1904**

**G PHANEENDRA - 20311A1958**

## **DECLARATION**

This is to certify that the work reported in the present Project Work titled “**Market Trend Analysis Using Machine Learning**” is a record work done by us in the **Department of Electronics and Computer Engineering, Sreenidhi Institute of Science and Technology, Yamnampet, Ghatkesar.**

The report is based on the project work done entirely by us and not copied from any other source.

**OMJAJU – 20311A1901**

**Y VISHAL– 20311A1904**

**G PHANEENDRA - 20311A1958**

## ABSTRACT

Stock market is one of the major fields that investors are dedicated to, thus stock market price trend prediction is always a hot topic for researchers from both financial and technical domains. In this research, our objective is to build a state-of-the-art prediction model for price trend prediction, which focuses on short-term price trend prediction. In today's world where data is the new currency, computer machines are capable enough to find and predict the future by using the previous data and the algorithms which are becoming more precise day by day. The proposed solution by us includes the pre-processing of the previous data. Our solution implemented the engineering techniques of the LSTM model to predict the market trend.

While predicting [7] the actual price of a stock is an uphill climb, we can build a model that will predict whether the price will go up or down. In this project we attempt to implement machine learning approach to predict stock prices. Machine learning effectively implemented in forecasting stock prices. The objective is to predict the stock prices in order to make more informed and accurate investment decisions. We propose a stock price prediction system that integrates mathematical functions, machine learning, and other external factors for the purpose of achieving better stock prediction accuracy and issuing profitable trades. The recent trend in stock market prediction technologies is the use of machine learning which makes predictions based on the values of current stock market indices by training on their previous values. Machine learning itself employs different models to make prediction easier and authentic. This project focuses on the use of Regression and LSTM based Machine learning to predict stock values. LSTMs are very powerful in sequence prediction problems because they're able to store past information. This is important in our case because the previous price of a stock is crucial in predicting its future price.

The stock price [3] of a company is changed every day based on their exchange of products and raw goods. A company provides the shares to increase the production of their company production. To predict the value of the share is not an easy task because it will change based on many factors. Many investors try to invest their amount in a company to get profit. Here we have a problem with finding the company share value is going to increase or decrease for that day end. In this case, we are trying to use technology for attaining perfect value. Technology is evolving swiftly. There are many features to be acknowledged while we try to build a machine learning model. In a day stock market has three major values high, low, open and close. Scrutinize the historical data to predict value is going to increase or decrease. Many machine learning algorithms are used to predict the value but as these types of time forecasting problems, we can apply SVM and Recurrent Neural Network.

## **TABLE OF CONTENTS**

Abstract	i
List of Figures	iv
Chapter – 1 Introduction	1-2
1.1 Stock market	
1.2 LSTM Model	
Chapter – 2 Block Diagram	3-4
2.1 Gates	
Chapter – 3 Literature Survey	5-6
3.1 Research paper	
3.2 Literature Survey papers published	
Chapter – 4 Flowchart	7-8
4.1 Diagram	
4.2 Explanation	
Chapter – 5 Code	9-13
Chapter – 6 Result	14-18
6.1 Execution	

6.2 Final Output	
Chapter – 7 Future Scope	19-20
7.1 Scope in specific field	
Chapter – 8 Conclusion	21-22
Chapter – 9 References	23

## **TABLE OF FIGURES**

FIGURE S.NO	FIGURE NAME	PAGE NO.
2.1.1	LSTM Model	3
4.1.1	Flowchart	7
6.1.1	Importing Libraries	14
6.1.2	Dataset information before processing	14
6.1.3	Dataset information after processing	15
6.1.4	Training Dataset	15
6.1.5	Implementing LSTM model	16
6.1.6	Graph of training dataset	16
6.1.7	Testing dataset	17
6.1.8	Final Graph	17
6.2	Final output	18



# CHAPTER – 1

## INTRODUCTION

### 1.1 STOCK MARKET

The stock price [7] fluctuations are uncertain, and there are many interconnected reasons behind the scene for such behavior. The possible cause could be the global economic data, changes in the unemployment rate, monetary policies of influencing countries, immigration policies, natural disasters, public health conditions, and several others. All the stock market stakeholders aim to make higher profits and reduce the risks from the thorough market evaluation. The major challenge is gathering the multifaceted information, putting them together into one basket, and constructing a reliable model for accurate predictions. Stock price prediction is a complex and challenging task for companies, investors, and equity traders to predict future returns. Stock markets are naturally noisy, non-parametric, non-linear, and deterministic chaotic system.

A proper model developed with an optimal set of attributes can predict stock price reasonably well and better inform the market situation. A plethora of research has been published to study how certain variables correlate with stock price behavior. [4] A varying degree of success is seen concerning the accuracy and robustness of the models. One possible reason for not achieving the expected outcome could be in the variable selection process. There is a greater chance that the developed model performs reasonably better if a good combination of features is considered. One of the contributions of this study is selecting the variables by looking meticulously at multiple aspects of the economy and their potential impact in the broader markets.

The field of quantitative analysis in finance has a long history. Several models ranging from naive to complex have been developed so far to find the solution to financial problems. However, not all quantitative analyses or models are fully accepted or widely used. One of the first attempts was made in the seventies by two British statisticians, Box and Jenkins, using mainframe computers. They developed the Auto- Regressive Integrated Moving Average (ARIMA) model utilizing only the historical data of price and volume. A varying degree of success is seen concerning the accuracy and robustness of the models. One possible reason for not achieving the expected outcome could be in the variable selection process. There is a greater chance that the developed model performs reasonably better if a good combination of features is considered.

One of the contributions of this study is selecting the variables by looking meticulously at multiple aspects of the economy and their potential impact in the broader markets. The field of quantitative analysis in finance has a long history. Several models ranging from naive to complex have been developed so far to find the solution to financial problems. However, not all quantitative analyses or models are fully accepted or widely used. One of the first attempts was made in the seventies by two British statisticians, Box and Jenkins, using mainframe computers.

The stock market is a complex and dynamic system, influenced by numerous factors such as economic indicators, company performance, news events, and investor sentiment. Predicting stock market movements accurately is a challenging task due to the inherent volatility and non-linear nature of financial markets. However, advancements in machine learning techniques, particularly the use of recurrent neural networks (RNNs) with Long Short-Term Memory (LSTM) cells, have shown promise in improving stock market predictions. LSTM [5] is a type of RNN that is specifically designed to handle long-term dependencies and capture temporal patterns in sequential data. It has gained popularity in various fields, including natural language processing, speech recognition, and time series analysis. LSTM networks are well-suited for predicting stock market prices because they can effectively model the complex relationships and dependencies that exist within historical price data.

## 1.2 LTSM MODEL

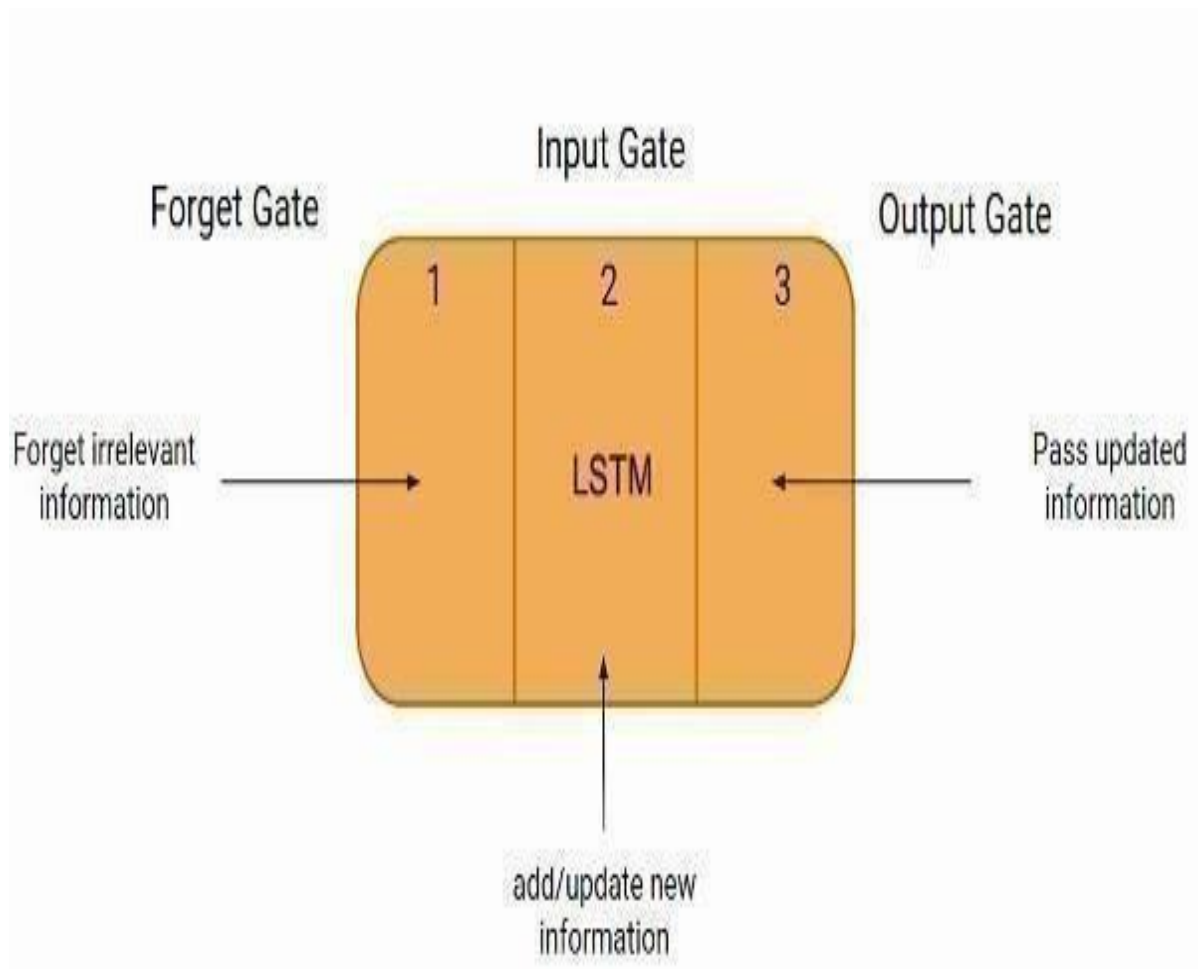
The LSTM model consists of interconnected memory cells that store information over extended periods. Each memory cell is equipped with gating mechanisms that control the flow of information and selectively remember or forget certain inputs. To train an LSTM model for stock market prediction, historical price data and relevant technical indicators are typically used as input features. The model learns from patterns and trends in the data to make predictions about future price movements. The training process involves adjusting the weights and biases of the LSTM network using optimization algorithms, such as stochastic gradient descent, to minimize the prediction errors. Once trained, the LSTM model can be used to generate predictions for future stock market prices. It takes in the current and past price data as input and produces an output that represents the predicted price at the next time step.

By [2] iteratively feeding the predicted prices back into the model as input, it can generate a sequence of predicted prices, allowing for the forecasting of future market trends. It's important to note that while LSTM models have shown promise in stock market prediction, they are not infallible, and accurate predictions are not guaranteed. This capability allows LSTM networks to capture both short-term and long-term dependencies in the input data, making them particularly useful for predicting stock market trends. Stock price predictions are very important among many business people and the public. People can make a lot of money or lose their financial income from a stock market job. Algorithm predictions and models can be used to make future predictions applied to historical data. Predicting the future has been a daunting task, one that many have found difficult to understand.

This type of prediction is even more appealing when it involves money and risks such as Stock Market speculation. Researchers are conducting research on stock market forecasts from a variety of fields, including computer science and business.

## CHAPTER – 2

### BLOCK DIAGRAM



2.1.1 –LSTM MODEL

LSTM is a special network structure with three “gate” structures.

[5] Three gates are placed in an LSTM unit, called input gate, forgetting gate and output gate. While information enters the LSTM’s network, it can be selected by rules. Only the information conforms to the algorithm will be left, and the information that does not conform will be forgotten through the forgetting gate.

## 2.1 GATES

1) **Input gate** - Decides how much information from current input flows to the cell state. The input gate determines how much of the new input ( $x(t)$ ) should be stored in the cell state ( $c(t)$ ). It takes the current input ( $x(t)$ ) and the previous hidden state ( $h(t-1)$ ) as inputs and outputs a value between 0 and 1 for each element of the cell state.

2) **Forget gate** - Decides how much information from the current input and the previous cell state flows into the current cell state. The forget gate decides how much of the previous cell state ( $c(t-1)$ ) should be forgotten. It takes the current input ( $x(t)$ ) and the previous hidden state ( $h(t-1)$ ) as inputs and outputs a value between 0 and 1 for each element of the cell state.

3) **Output gate** - Decides how much information from the current cell state flows into the hidden state, so that if needed LSTM can only pick the long-term memories or short-term memories and long-term memories. The output gate determines how much of the cell state ( $c(t)$ ) should be exposed as the output. It takes the current input ( $x(t)$ ) and the previous hidden state ( $h(t-1)$ ) as inputs and outputs a value between 0 and 1 for each element of the cell state.

In summary, the LSTM model uses these equations to update and control the flow of information through the memory cell and hidden state. The input gate determines new information to be stored, the forget gate controls what information is forgotten, the update gate calculates the new memory cell values, and the output gate determines the output at the current time step.

## CHAPTER – 3

### LITERATURE SURVEY

#### 3.1 RESEARCH PAPERS

[1] **Stock Market Prediction Using Machine Learning Techniques, IEEE 2020-** Naadun Sirimevan et al., [4] The Stock Market Prices play a crucial role in today' economy. Researchers have discovered that social media platforms such as twitter and web news tend to influence the decision-making process of any individual. In this research behavioural reflex towards web news is taken into count to reduce the gap and make the prediction much more accurate. Precise predictions were made for a day, a week and two weeks here after.

[2] **Share Price Prediction using Machine Learning Technique, IEEE 2019-** Jeevan B et al., Lately stock market has been the talk of the town with more and more people from academics and business showing interest in it. This paper mostly deals with the approach towards predicting stock prices using RNN (Recurrent Neural Network) and LSTM (Long Short Term Memory) on National Stock Exchange using numerous elements such as the present-day market price as well as anonymous events. A recommendation system along with models constructed on RNN and LSTM methods are used in selecting the company is also mentioned in this paper.

[3] **An LSTM-Method for Bit- coin Price Prediction: A Case Study Yahoo Finance Stock Market, IEEE 2019-** Ferdiansyah et al., Bit-coin is a type of Cryptocurrency and currently is one of a kind of investment on the stock market. Stock markets are inclined by several risks. And bit-coin is one kind of crypto currency that keeps rising in recent years, and sometimes suddenly falls without knowing influence on the stock market. There's a need for automation tools to predict bit-coin on the stock market because of its fluctuations. This research study studies how to create mode prediction bit-coin stock market prediction using LSTM. Before confirming the results the paper tries to measure the results using RMSE (the Root Mean Square Error). The RMSE will at all times be larger or equal to the MAE. The RMSE metric assesses how well a model can calculate a continuous value. The method that is applied on this research to predict Bit- coin on the stock market Yahoo finance can forecast the result above \$12600 USD for the next couple of days after prediction.

[4] **Research on Stock Price Prediction Method Based on Convolutional Neural Network, IEEE 2019-** Sayavong Lounnapha et al. This paper intends for a prediction model for stock price which is centered at the convolutional neural networks, that has exceptional capability of learning on its own. The data set is taught and tested relating the behaviours of both Convolutional Neural Networks and Thai stock market. The result shows that the model on grounds of Convolutional Neural Networks can effectually recognize the altering trend in stock market price and envisage it which provides significant allusion for stock price forecast. The accuracy of the prediction is found to be elevated, and it could also be promoted in the field of finance.

### 3.2 LITERATURE SURVEY PAPERS PUBLISHED:

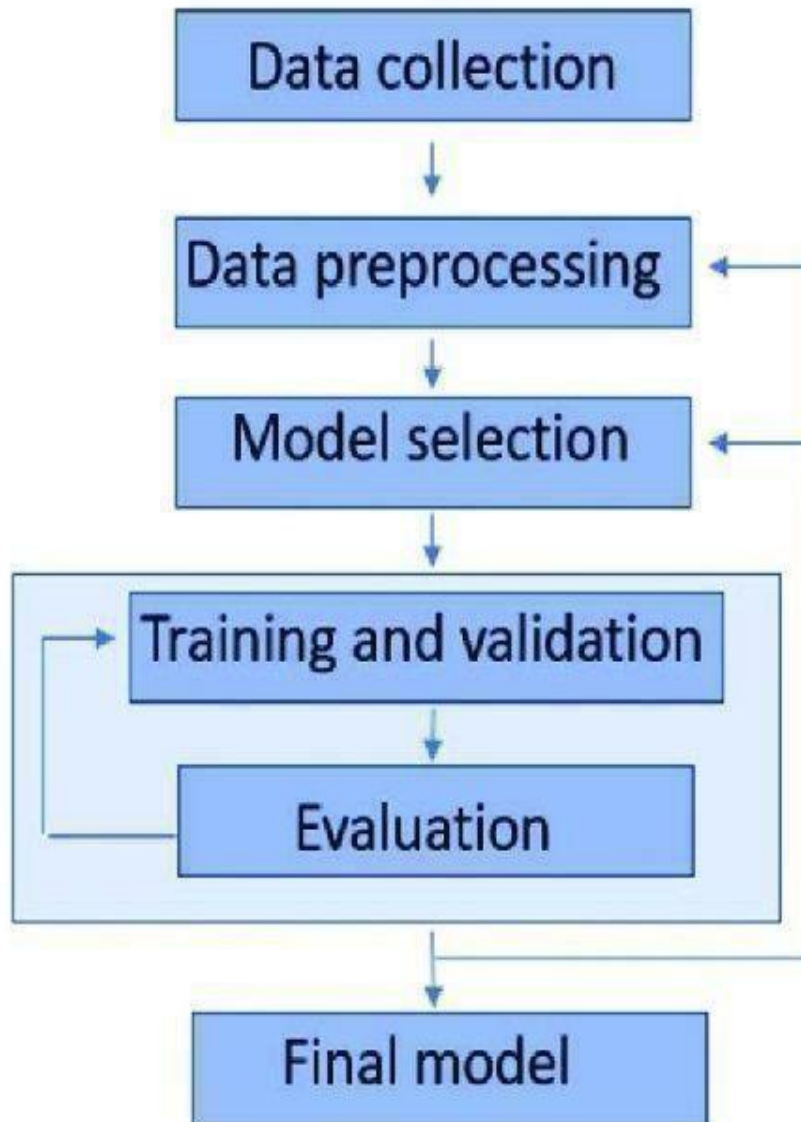
S.NO	PAPERS	YEAR	PUBLISHER
1	"Stock price prediction using LSTM and CNN-LSTM models"	2020	Elsevier
2	"Deep learning with long short-term memory for stock market forecasting"	2017	IEEE
3	"Stock price prediction using LSTM and evolutionary-based feature selection"	2019	Springer
4	"Enhancing stock price prediction using LSTM with attention mechanism"	2015	Elsevier
5	"Long short-term memory networks for anomaly detection in time series data"	2022	Springer
6	"Stock price prediction using LSTM-RNN with indicator fusion"	2016	ACM
7	"Combining LSTM with news topics for stock trend prediction"	2019	ACM

## CHAPTER – 4

### FLOWCHART

#### 4.1 - DIAGRAM

---



4.1.1 –FLOWCHART

## 4.2 - -EXPLANTION

**Data Collection :** Collect previous data using database and stored information.

**Data Preprocessing :** The carrying out of operations on data, especially by a computer, to retrieve, transform, or classify information.

**Model Selection :** Selecting the LTSM model to predict the future trend of stock market.

**Training and Validation :** During training, validation data infuses new data into the model that it hasn't evaluated before. Validation data provides the first test against unseen data, allowing data scientists to evaluate how well the model makes predictions based on the new data.

**Final Model :** The final result is presented in the form of a graph to convey the change in stock market price.

LSTMs [5] use a series of 'gates' which control how the information in a sequence of data comes into, is stored in and leaves the network. There are three gates in a typical LSTM; forget gate, input gate and output gate. These gates can be thought of as filters and are each their own neural network. Data is collected using database files and then preprocessed by importing libraries. Importing libraries is an essential step in building an LSTM model (or any machine learning model) as it allows you to leverage pre-existing functionality and tools that facilitate the development and training of the model. Here are some common libraries that are often imported when working with LSTM models.

These libraries provide a wide range of functionalities that streamline the development process, simplify complex tasks, and improve the performance of LSTM models. By importing these libraries, you gain access to their functions and classes, saving you time and effort in implementing common operations and ensuring efficient execution of your code.



## CHAPTER – 5

### CODE

```
import numpy as np # Library for numerical computing.
```

```
import pandas as pd # Library for data manipulation and analysis.
```

```
import matplotlib.pyplot as plt #Library for creating static, animated, and interactive  
visualizations.
```

```
from sklearn.preprocessing import MinMaxScaler #Module for data preprocessing
```

```
data.head() #Displaying the first few rows of the Data Frame
```

```
data.info()
```

```
data["Close"] = pd.to_numeric(data.Close, errors='coerce') #Convert the "Close" column of the Data Frame  
tonumeric data type, replacing any non- numeric values with NaN .
```

```
from keras.models import Sequential #Class for creating sequential models in Keras.
```

```
from keras.layers import Dense, LSTM, Dropout #Classes for defining layers in a neural network.
```

```
data = pd.read_csv('Google_train_data.csv') #Reading the CSV file "Google_train_data.csv" andstoring  
it in the variable "data".
```

```
data = data.dropna() #Drop any rows in the DataFrame that contain NaN values, removing missing or  
invaliddata
```

```
trainData = data.iloc[:, 4:5].values #Extract a subset of the DataFrame containing only the values from the  
column at index 4 (5th column) and store it in "trainData".
```

`trainData.shape` **#Display the shape of the "trainData" array.**

`for i in range(60, 1149)` **# Iterate over the range from 60 to 1149 (length of the data) #60 : timestep // 1149 :**

`X_train.append(trainData[i-60:i, 0])` **#Appends a sequence of 60 previous values (from index i- 60 to i-1) from the first column (0th column) of the trainData array to the X\_train list. These sequences will be used as input features for the model.**

`sc = MinMaxScaler(feature_range=(0,1))` **# Create an instance of the MinMaxScaler class with the feature range set to (0, 1).**

`trainData = sc.fit_transform(trainData)` **#Use the fit\_transform() method of the MinMaxScaler object to scale the "trainData" array.**

`X_train.append(trainData[i-60:i, 0])`

`y_train.append(trainData[i, 0])`

`X_train, y_train = np.array(X_train), np.array(y_train)` **#In the first two lines, empty lists X\_train and y\_train are created to store the input and output sequences for the model.**

`X_train=np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))` **#Reshape the X\_train array to add an additional dimension for the batch size**

`X_train.shape` **# Display the shape of the**

`X_train` **array.** `(X_train.shape[0], X_train.shape[1], 1)`

`model = Sequential()` **#Create an instance of the Sequential class to build the model.**

`model.add(LSTM(units=100, return_sequences=True, input_shape=(X_train.shape[1], 1)))` **#Add an LSTM layer with 100 units, returning sequences, and specify the input shape.**

`model.add(Dropout(0.2))` **#Add a dropout layer with a dropout rate of 0.2.**

`model.add(LSTM(units=100, return_sequences=True))` **#Add another LSTM layer with 100 units, returning sequences.**

`model.add(Dropout(0.2))` **#Add another dropout layer with a dropout rate of 0.2.**

`model.add(LSTM(units=100, return_sequences=True))` **#Add another LSTM layer with 100 units, returning sequences.**

`model.add(Dropout(0.2))` **#Add another dropout layer with a dropout rate of 0.2. In this code, a sequential model is created using the Sequential class from Keras.**

`model.add(LSTM(units=100, return_sequences=False))` **#Add a final LSTM layer with 100 units, not returning sequences. LSTM layers are a type of recurrent neural network layer that can handle sequential data.**

`model.add(Dropout(0.2))` **#Add another dropout layer with a dropout rate of 0.2. Overfitting refers to a situation where a machine learning model performs.**

`hist = model.fit(X_train, y_train, epochs=20, batch_size=32, verbose=2):`

**#In this code, the fit() function of the model object is called to train the model. The function takes several parameters.**

`model.add(Dense(units=1))` **#Add a dense layer with 1 unit.**

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

**#Compile the model with the Adam optimizer and the mean squared error loss function.**

`Hist = X_train  
y_train  
epochs=20  
batch_size=32`

`plt.plot(hist.history['loss'])` **# Plot the training loss values from the training history.**

`plt.title('Training model loss')` **# Set the title of the plot to 'Training model loss'.**

`plt.ylabel('loss')` **#Set the label for the y-axis to 'loss'.**

`plt.xlabel('epoch')` **#Set the label for the x-axis to 'epoch'.**

`plt.legend(['train'], loc='upper left')` **#Add a legend to the plot with the label 'train' and position it in the upper left corner.**

`plt.show()` **#Display the plot.**

`matplotlib.pyplot` **#In this code, the plot() function from the module is used to create a line plot of the training loss values.**

`title()`, `xlabel()`, and `ylabel()` **#functions are used to set the title of the plot and labels for the x-axis and y-axis, respectively.**

`legend()` **#function is used to add a legend to the plot, with the label 'train'.**

`show()` **# function is called to display plot.**

## **#Testing Data**

```
testData = pd.read_csv('Google_test_data.csv')
```

```
testData["Close"]=pd.to_numeric(testData.Close  
, errors='coerce')
```

```
testData = testData.iloc[:, 4:5]
```

```
y_test = testData.iloc[60:, 0:].values
```

```
inputClosing = testData.iloc[:, 0:].values
```

```
inputClosing_scaled = sc.transform(inputClosing)
```

```
inputClosing_scaled.shape
```

```
X_test.append(inputClosing_scaled[i- timestep:i, 0])
```

```
X_test = np.array(X_test)
```

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

```
X_test.shape
```

```
testData = testData.dropna()
```

```
plt.plot(y_test, color='red', label='Actual Stock Price')
```

```
y_pred = model.predict(X_test) y_pred #In this code, the predict() function of the trained model is called to make predictions on the X_test data.
```

```
predicted_price=sc.inverse_transform(y_pred)
```

**#plotting the graph**

```
plt.plot(predicted_price,color='green', label='Predicted Stock Price')
```

```
plt.title('Google Stock Price Prediction')
```

```
plt.xlabel('Time')
```

```
plt.ylabel('Stock Price')
```

```
plt.legend()
```

```
plt.show()
```

plot() function from the matplotlib.pyplot module

title(), xlabel(), and ylabel()

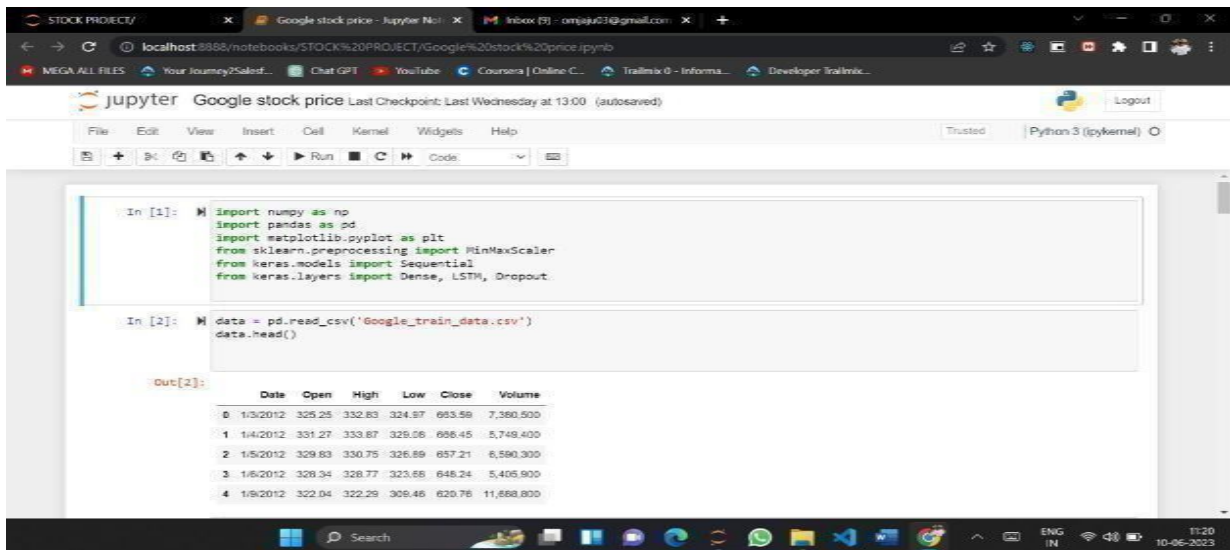
legend()

show()

## CHAPTER – 6

### RESULT

#### 6.1 – EXECUTION



The screenshot shows a Jupyter Notebook interface with the following code and output:

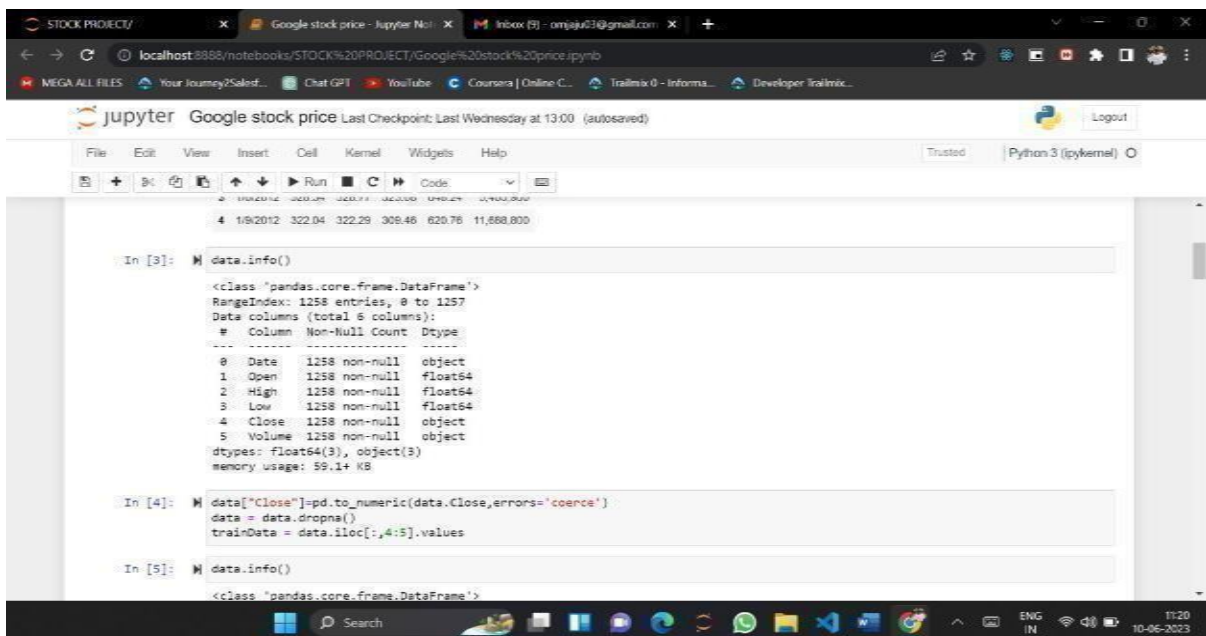
```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout
```

```
In [2]: data = pd.read_csv('Google_train_data.csv')
data.head()
```

Out[2]:

	Date	Open	High	Low	Close	Volume
0	1/3/2012	325.25	332.83	324.87	653.58	7,368,500
1	1/4/2012	331.27	333.87	328.06	666.45	5,748,400
2	1/5/2012	329.83	330.75	326.89	657.21	6,580,300
3	1/6/2012	328.34	328.77	323.68	648.24	5,405,900
4	1/9/2012	322.04	322.29	309.46	620.76	11,668,800

##### 6.1.1 –IMPORTING LIBRARIES



The screenshot shows a Jupyter Notebook interface with the following code and output:

```
In [3]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1258 entries, 0 to 1257
Data columns (total 6 columns):
#   Column  Non-Null Count  Dtype  
---  -
0   Date    1258 non-null      object  
1   Open    1258 non-null      float64  
2   High    1258 non-null      float64  
3   Low     1258 non-null      float64  
4   Close   1258 non-null      object  
5   Volume  1258 non-null      object  
dtypes: float64(3), object(3)
memory usage: 59.1+ KB
```

```
In [4]: data["Close"] = pd.to_numeric(data.Close, errors='coerce')
data = data.dropna()
trainData = data.iloc[:,4:5].values
```

```
In [5]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

##### 6.1.2 –DATASET INFORMATION BEFORE PROCESSING

```

In [4]: data["Close"] = pd.to_numeric(data.Close, errors='coerce')
        data = data.dropna()
        trainData = data.iloc[:,4:5].values

In [5]: data.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 1149 entries, 0 to 1257
Data columns (total 6 columns):
#   Column  Non-Null Count  Dtype  
---  --
0    Date      1149 non-null     object  
1    Open      1149 non-null     float64  
2    High      1149 non-null     float64  
3    Low       1149 non-null     float64  
4    Close     1149 non-null     float64  
5    Volume    1149 non-null     object  
dtypes: float64(4), object(2)
memory usage: 62.8+ KB

In [6]: sc = MinMaxScaler(feature_range=(0,1))
        trainData = sc.fit_transform(trainData)
        trainData.shape

```

### 6.1.3 – DATASET INFORMATION AFTER PROCESSING

```

In [6]: sc = MinMaxScaler(feature_range=(0,1))
        trainData = sc.fit_transform(trainData)
        trainData.shape

Out[6]: (1149, 1)

In [7]: X_train = []
        y_train = []

        for i in range(60,1149): #60 : timestep // 1149 : Length of the data
            X_train.append(trainData[i-60:i,0])
            y_train.append(trainData[i,0])

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

In [8]: X_train = np.reshape(X_train,(X_train.shape[0],X_train.shape[1],1)) #adding the batch_size axis
        X_train.shape

Out[8]: (1089, 60, 1)

In [9]: model = Sequential()

        model.add(LSTM(units=100, return_sequences = True, input_shape =(X_train.shape[1],1)))
        model.add(Dropout(0.2))

        model.add(LSTM(units=100, return_sequences = True))
        model.add(Dropout(0.2))

```

### 6.1.4 –TRAINING DATASET

```
STOCK PROJECT/ Google stock price - Jupyter Notebook - omjaju03@gmail.com
localhost:8888/notebooks/STOCK%20PROJECT/Google%20stock%20price.ipynb
jupyter Google stock price Last Checkpoint: Last Wednesday at 13:00 (autosaved)
Python 3 (ipykernel)

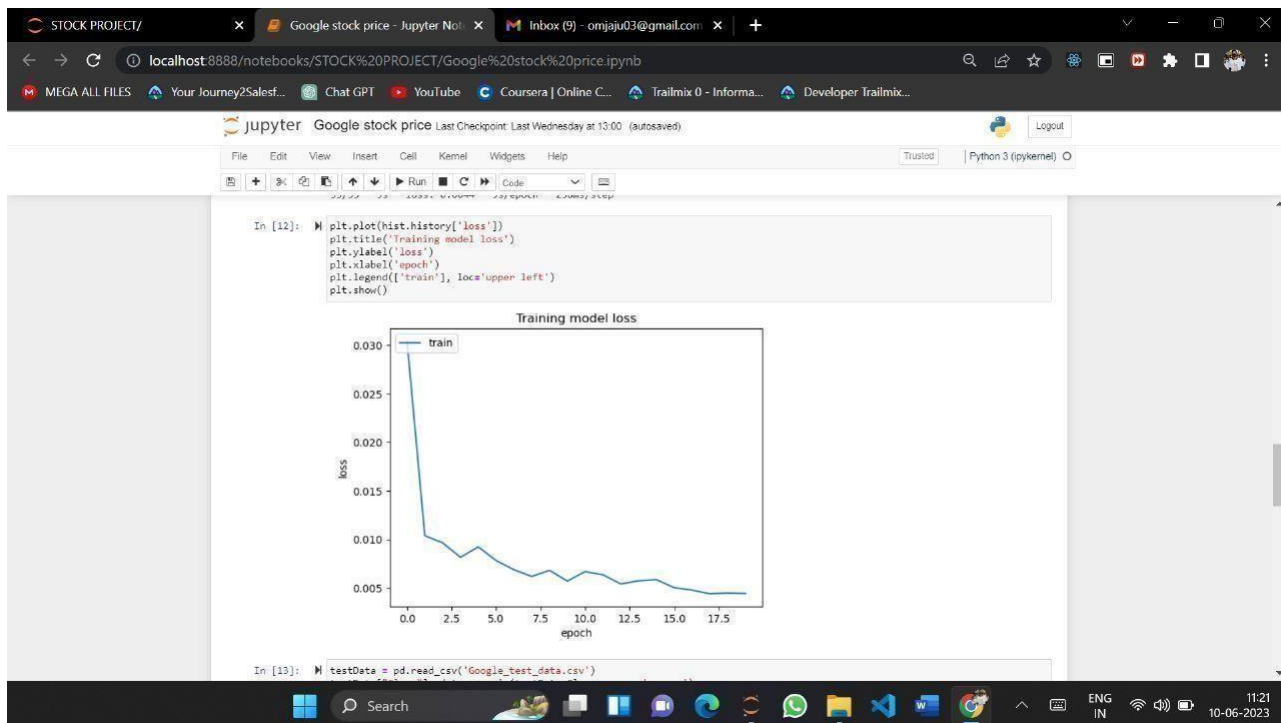
Out[8]: (1089, 60, 1)

In [9]: model = Sequential()
        model.add(LSTM(units=100, return_sequences = True, input_shape =(X_train.shape[1],1)))
        model.add(Dropout(0.2))
        model.add(LSTM(units=100, return_sequences = True))
        model.add(Dropout(0.2))
        model.add(LSTM(units=100, return_sequences = True))
        model.add(Dropout(0.2))
        model.add(LSTM(units=100, return_sequences = False))
        model.add(Dropout(0.2))
        model.add(Dense(units =1))
        model.compile(optimizer='adam',loss="mean_squared_error")

In [11]: hist = model.fit(X_train, y_train, epochs = 20, batch_size = 32, verbose=2)

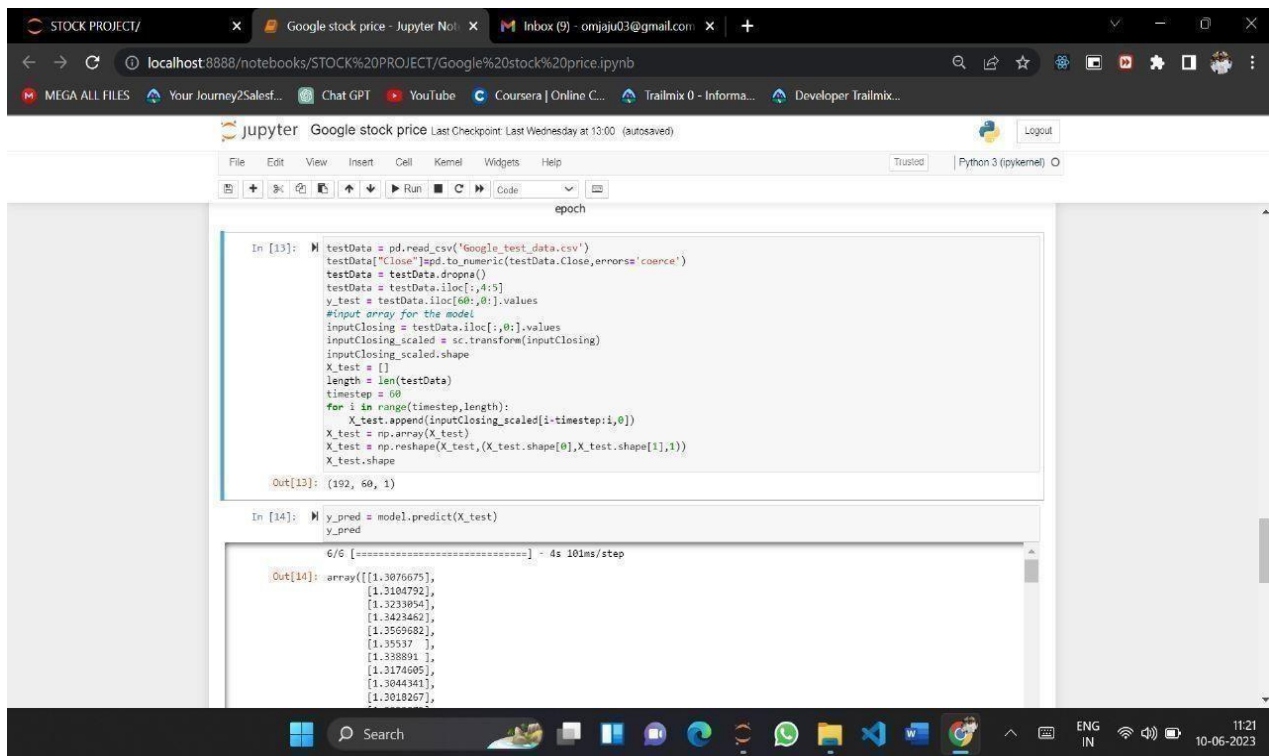
Epoch 1/20
35/35 - 24s - loss: 0.0304 - 24s/epoch - 691ms/step
Epoch 2/20
35/35 - 10s - loss: 0.0104 - 10s/epoch - 281ms/step
Epoch 3/20
```

### 6.1.5 –IMPLEMENTING LSTM MODEL

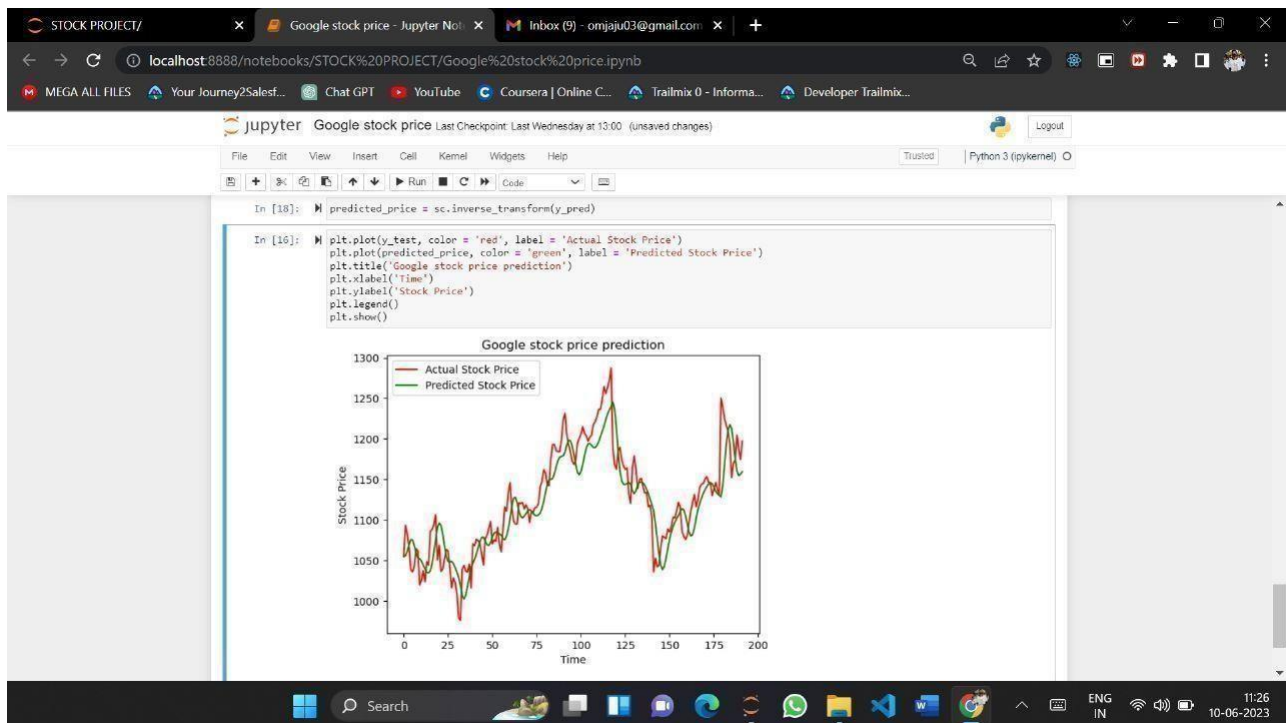


### 6.1.6 –GRAPH OF TRAINING DATASET





### 6.1.7 –TESTING DATASET



### 6.1.8 –FINAL GRAPH

## 6.2 – FINAL OUTPUT



6.2.1 –FINAL OUTPUT

## **CHAPTER – 7**

### **FUTURE SCOPE**

Stock price prediction [3] is one of the not topics in that field of machine learning. The prediction of stock price not only helps developers also help the investors to invest in a profitable company and gain some profit. By using LSTM we get more accuracy than other algorithms in machine learning. Here we only considered the closing price of each day and created the model and get the closest predicted value. We also take 60 days of data for particular y days that also helped the model to recognize the pattern in the sequence data and predict the next one. We applied the same model on 7 top companies for 10 days that lead to getting a conclusion that for some company it gets the closest value and for some, it gets very large difference there we get 60% of closest prediction.

The further enhancement we try to add new feathers to the existing one that feather is news and sentiments of the country and company. These feathers increase the model to find the accurate value at add times. The popularity of stock market trading is growing rapidly, which is encouraging researchers to find out new methods for the prediction using new techniques. The forecasting technique is not only helping the researchers but it also helps investors and any person dealing with the stock market. In order to help predict the stock indices, a forecasting model with good accuracy is required. In this work, we have used one of the most precise forecasting technology using Long Short-Term Memory unit which helps investors, analysts or any person interested in investing in the stock market by providing them a good knowledge of the future situation of the stock market.

When compared with ARIMA (Auto Regressive Integrated Moving Average) algorithm, it is shown that ARIMA algorithm understands the past data and does not focus on the seasonal part. Therefore accuracy is less. LSTM provides more accurate results than ARIMA algorithm. The future enhancement includes comparing the accuracy of LSTM with other prediction algorithms. LSTM is more accurate than any other prediction algorithms.

## 7.1 Scope in specific fields :

**Algorithmic Trading:** LSTM models can be utilized to develop algorithmic trading systems that make predictions about future stock prices and automate the buying and selling of stocks based on those predictions. Such systems can take into account historical price data, market indicators, and other relevant factors to generate trading signals.

**Risk Management:** Accurate stock price predictions can aid in risk management for investment firms and individual traders. LSTM models can help identify potential market risks by forecasting price movements, enabling proactive risk mitigation strategies and better-informed decision-making.

**Portfolio Optimization:** LSTM models can be employed in optimizing investment portfolios by predicting future stock prices. By integrating LSTM predictions into portfolio optimization algorithms, investors can allocate their assets optimally based on the expected performance of various stocks.

**Market Research and Analysis:** [6] LSTM models can assist market researchers and analysts in generating insights and identifying trends in stock markets. By analyzing historical data and predicting future price movements, these models can aid in understanding market behavior, identifying patterns, and uncovering market inefficiencies.

**Financial Planning and Wealth Management:** LSTM-based stock price prediction models can be leveraged in financial planning and wealth management services. By providing reliable forecasts of stock prices, these models can help individuals and financial advisors make informed decisions about investment strategies, retirement planning, and wealth accumulation.

**Cryptocurrency Trading:** LSTM models can be applied to predict the prices of cryptocurrencies, which are known for their high volatility. By forecasting cryptocurrency prices, traders and investors can make informed decisions in this rapidly evolving market.

## CHAPTER – 8

### CONCLUSION

From the research done so far it could be concluded that the RNN and LSTM [2]libraries are very effective in determining the stock price trends effectively relative to the actual market trend. At the same time what we could find out is that the python libraries that were used as a part of the training process were not very optimal. As far as the training speed is considered the functions that we use from the mathematics principle have a lot faster speed comparatively and they consist of more detailed designs and significant improvements when tested under various situations. However, the python library functions are considered to be more adaptable. From our work done so far we can easily tell that certain stock trends can be predicted easily on the basis of certain general rules and regulations of the stock. This is the main reason behind the existence of the private placement institutes.

Few things such as optimization of the neural network parameters as well as the training process however always has much room for improvement. All these points would be considered as further steps in the research. Stock investing has attracted the interest of many investors around the world. However, making a decision is a difficult task as many things are involved. By investing successfully, investors are eager to predict the future of the stock market. Even the slightest improvement in performance can be enormous. A good forecasting system will help investors make investments more accurate and more profitable by providing supporting information such as future stock price guidance.

In addition to historical prices, other related information [6] could affect prices such as politics, economic growth, financial matters and the atmosphere on social media. Numerous studies have proven that emotional analysis has a significant impact on future prices. Therefore, the combination of technical and basic analysis can produce very good predictions. However, they should be considered as one component of a comprehensive investment strategy, working in conjunction with other fundamental and technical analysis techniques. Stock price prediction using LSTM models holds promise and potential in the field of finance and investment. By leveraging the power of deep learning and recurrent neural networks, LSTM models have shown capabilities in capturing complex patterns and dependencies in sequential data like historical stock prices. However, it is important to draw certain conclusions regarding their application in stock price prediction.

It is crucial to consider that stock markets are highly efficient and adaptive systems, incorporating vast amounts of information and the actions of numerous participants. As such, even the most advanced models, including LSTM, may face limitations in capturing and incorporating all relevant factors that drive stock price movements.

Furthermore, it is essential to approach stock price prediction using LSTM models with caution and employ rigorous validation and evaluation techniques. Models should be thoroughly tested on historical data and assessed based on various performance metrics, such as accuracy, precision, recall, and profitability measures. Robust backtesting and out-of-sample testing should be conducted to assess the model's effectiveness in real-world scenarios.

Lastly, while LSTM models [5] can aid in decision-making and risk management, they should not be viewed as standalone tools for making investment decisions. Expertise in finance, domain knowledge, and a holistic understanding of the stock market dynamics remain crucial for interpreting and contextualizing the predictions provided by LSTM models. In conclusion, LSTM models offer exciting possibilities for stock price prediction, and their application can contribute to informed decision-making, risk management, and financial planning.

## **CHAPTER – 9**

### **REFERENCES**

- [1] [www.simplilearn.com](http://www.simplilearn.com)
- [2] [www.google.com](http://www.google.com)
- [3] [www.researchgate.com](http://www.researchgate.com)
- [4] [https://www.researchgate.net/publication/341482418\\_A\\_Survey\\_on\\_Stock\\_Market\\_Prediction\\_Using\\_Machine\\_Learning\\_Techniques](https://www.researchgate.net/publication/341482418_A_Survey_on_Stock_Market_Prediction_Using_Machine_Learning_Techniques)
- [5] [https://en.wikipedia.org/wiki/Long\\_short-term\\_memory](https://en.wikipedia.org/wiki/Long_short-term_memory)
- [6] <https://www.kaggle.com/code/faressayah/stock-market-analysis-prediction-using-lstm>
- [7] [https://en.wikipedia.org/wiki/Stock\\_market](https://en.wikipedia.org/wiki/Stock_market)

