Project Report

on

STOCK PRICE PREDICTION USING MACHINE LEARNING

(A dissertation submitted in partial fulfillment of the requirements of Bachelor of Technology in Computer Science and Engineering of the Maulana Abul Kalam Azad University of Technology, West Bengal)

Submitted By
Suchismita Das
Sharmistha Sett
Pallabi Maji
Debarati Paul
Bandana Bharti

Under the guidance of
Prof. Ruben Roy
Dept. of Computer Science and Engineering

Government College of Engineering and Leather Technology
(Affiliated to MAKAUT, West Bengal)
Kolakta - 700106, WB

2022-2023

DECLRATION

We hereby declare that the project work being presented in the project proposal entitle “StockPrice Prediction Using Machine Learning” in partial fulfilment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY at Government College of Engineering and Leather Technology is an authentic work carried out under the guidance of Prof. RUBEN ROY. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

SL NO.	Name Of The Student	University Roll No.	Signature Of The Students
1	Suchismita Das	11200220039	
2	Sharmistha Sett	11200220040	
3	Pallabi Maji	11200220043	
4	Debarati Paul	11200220045	
5	Bandana Bharti	11200220046	

CERTIFICATE OF APPROVAL

This is to certify that the project report on "**STOCK PRICE PREDICTION USING MACHINE LEARNING**" is a record of bonafide work, carried out by Suchismita Das, Sharmistha Sett, Pallabi Maji, Debarati Paul and Bandana Bharti under my guidance and supervision

In my opinion, the report in its present form is in conformity as specified by Government College of Engineering and Leather Technology and as per regulations of the Maulana Abul Kalam Azad University of Technology, West Bengal. To the best of my knowledge the results presented here are original in nature and worthy of incorporation in project report for the B.Tech. Program in Computer Science and Engineering.

Signature of
Supervisor/ Guide

Signature of
Head, Dept. of CSE

ACKNOWLEDGEMENT

With great pleasure, I would like to express my profound gratitude and indebtedness to **Prof. Ruben Roy**, Information Technology, Government College of Engineering and Leather Technology, W.B. for his continuous guidance, valuable advice and constant encouragement throughout the project work. His valuable and constructive suggestions at many difficult situations are immensely acknowledged. I am in short of words to express his contribution to this thesis through criticism, suggestions and discussions.

I would like to take this opportunity to thank Prof. Ruben Sir, Project Coordinator and Prof. Santanu Halder, HOD, Department of Computer Science & Engineering and Information Technology, Government College of Engineering and Leather Technology.

ABSTRACT

In this project we attempt to implement machine learning approach to predict stock prices. Machine learning is 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. There are two types of stocks. You may know of intraday trading by the commonly used term "day trading." Inter day traders hold securities positions from at least one day to the next and often for several days to weeks or months. 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. While predicting 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.

CONTENTS

Chapter 1: INTRODUCTION **8 - 10**

1.1 Motivation of the project : **8**
1.2 Background : **8**
1.3 Summary of the project: **8 -9**
1.4 System Development: **9**
1.4.1 System Analysis: 9
1.4.2 System Design: 9
1.5 Hardware/Software used: **9 - 10**
1.5.1 Software Requirement: 9
1.5.2 Hardware Requirements 9
1.5.3 Functional Requirements 9
1.5.4 Non-Functional Requirements: 10

Chapter 2: DATA AND WORK FLOW **11 - 12**

2.1 Data Flow [Basic]: **11**

2.2 Data Flow [Main]: **11**

2.3 Work Flow: **11 -12**

Chapter 3: DATA SOURCE AND DATASET LIST **13 - 17**

3.1 Data Source : **13 - 15**

3.1.1 Import Libraries 14
3.1.2 Visualization 14
3.1.3 Moving Average for 100 MA 14
3.1.4 Moving Average for 100 MA,200MA 15
3.1.5 Final Visualization 15

3.2 Dataset List: **15-17**

3.2.1 splitting data into training and testing 15
3.2.2 Normalizing the Dataset 16
3.2.3 Creating X_train and Y_train Data Structures and reshape 17

CHAPTER 4: MACHINE LEARNING (LSTM) MODEL **18 -21**

4.1 Define LSTM model: **18**

4.2 LSTM Architecture : **18 -19**
4.2.1 Forget Gate 18
4.2.2 Input Gate 18
4.2.3 Output Gate 18

4.3 Working process Of LSTM: **19 -21**

4.3.1 Building the Model by Importing the Libraries and Adding Different Layers to LSTM 19
4.3.2 Summarize this model 20
4.3.3 Fitting the Model 20
4.3.4 Preparing the Input for the Model 21
4.3.5 Plotting the Actual and Predicted Prices of Stocks. 21

Chapter 5: CREATING A WEB APP THROUGH STREAMLIT **22 - 26**

5.1 Process of the Streamlit connection: **22**

5.2 Result after Connecting through API: **22 - 26**

AIM OF THE PROJECT **27**

FUTURE SCOPE **27**

CONCLUSION **27**

REFERENCES **28**

Chapter 1: INTRODUCTION

The financial market is a dynamic and composite system where people can buy and sell currencies, stocks, equities and derivatives over virtual platforms supported by brokers. The stock market allows investors to own shares of public companies through trading either by exchange or over the counter markets. This market has given investors the chance of gaining money and having a prosperous life through investing small initial amounts of money, low riskcompared to the risk of opening new business or the need of high salary career. Nodaway's, Time-series prediction is a common technique widely used in many real-world applications such as weather forecasting and financial market prediction. It uses the continuous data in a period of time to predict the result in the next time unit. Many timeseries prediction algorithmshave shown their effectiveness in practice. The most common algorithms now are based on Recurrent Neural Networks (RNN), as well as its special type- Long-short Term Memory (LSTM) and Gated Recurrent Unit (GRU). Stock market is a typical area that presents time- series data and many researchers' study on it and proposed various models. In this project, LSTM model is used to predict the stock price.

1.1 Motivation of the project :

Businesses primarily run over customer's satisfaction, customer reviews about their products. Shifts in sentiment on social media have been shown to correlate with shifts in stock markets. Identifying customer grievances thereby resolving them leads to customer satisfaction as wellas trustworthiness of an organization. Hence there is a necessity of an un biased automated system to classify customer reviews regarding any problem. In today's environment where we're justifiably suffering from data overload (although this does not mean better or deeper insights), companies might have mountains of customer feedback collected; but for mere humans, it's still impossible to analyse it manually without any sort of error or bias. Oftentimes, companies with the best intentions find themselves in an insights vacuum. You know you need insights to inform your decision making and you know that you're lacking them, but don't know how best to get them. Sentiment analysis provides some answers into what the most important issues are, from the perspective of customers, at least. Because sentiment analysis can be automated,decisions can be made based on a significant amount of data rather than plain intuition.

1.2 Background:

In the finance world stock trading is one of the most important activities. Stock market prediction is an act of trying to determine the future value of a stock other financial instrument traded on a financial exchange. This report explains the prediction of a stock using Machine Learning. The technical and fundamental or the time series analysis is used by the most of the stockbrokers while making the stock predictions The programming language is used to predict the stock market using machine learning is Python. In this report we propose a Machine Learning (ML) approach that will be trained from the available stocks data and gain intelligence and then uses the acquired knowledge for an accurate prediction. In this context this study uses a machine learning technique called Support Vector Machine (SVM) to predict stock prices for the large and small capitalizations and in the three different markets,employing prices with both daily and up-to-the-minute frequencies.

1.3 Summary of the project :

This project is mostly based on the approach of predicting the stock price using Long Short Term Memory (LSTM) and Recurrent Neural Networks (RNN) to forecast the stock value on NES data using various factors such as current market price, price-earning ratio, base value and other anonymous events. The efficiency of the model is analysed by comparing the true data and the predicted data using an RNN graph. Machine learning to predict stock price as see the model is able to predict the stock price very close to the actual price where this model captures the detailed feature and uses different strategies to make a prediction. The model train for all the NSE data from the internet and recognize the input and group them and provide input according to the user configuration this RNN based architecture proved very efficient in forecasting the stock price by changing the

configuration accordingly which also use backpropagation mechanism while gathering and grouping data to avoid mixing of data.

1.4 System Development :

Systems development is systematic process which includes phases such as planning,analysis, design, deployment, and maintenance. Here we will primarily focus on –

- Systems Analysis
- Systems Design

1.4.1 Systems Analysis

It is a process of collecting and interpreting facts, identifying the problems,and decomposition of a system into its components.
System analysis is conducted for the purpose of studying a system or its parts in orderto identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish theirpurpose.

1.4.2 System Design:

It is a process of planning a new business system or replacing an existingsystem by defining its components or modules to satisfy the specific requirements. Before planning, you need to understand the old system thoroughly and determine how computers can best be used in order to operate efficiently.

1.6 Hardware/Software used:

1.5.1 Software Requirement

- Python 3.6
- Vs Code
- Jupiter Notebook
- Windows 7 Or Above
- Any Browser

1.5.2 Hardware Requirement

- RAM: 8 GB
- Storage: 500 GB
- CPU: 2 GHz or faster
- Architecture: 32-bit or 64-bit

1.5.3 Functional Requirements

Functional requirements describe what the software should do (the functions).
Think about the core operations.

Because the “functions” are established before development, functional requirements should be written in the future tense. In developing the software for Stock Price Prediction,some of the functional requirements could include:

- The software shall accept the tw_spydata_raw.csv dataset as input.
- The software should shall do pre-processing (like verifying for missing data values)on input for model training.
- The software shall use LSTM ARCHITECTURE as main component of thesoftware.
- It processes the given input data by producing the most possible outcomes of aCLOSING STOCK PRICE.

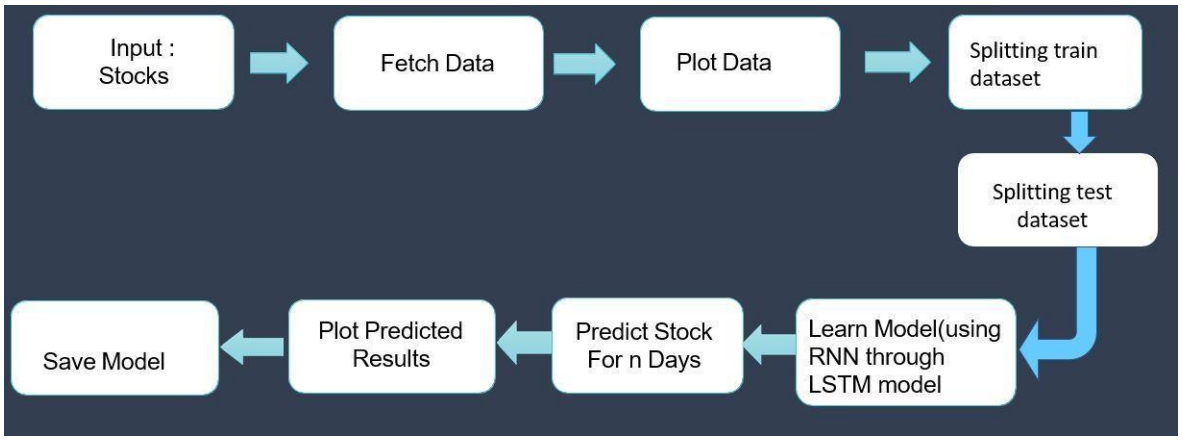
1.5.4 Non-Functional Requirements:

Product properties

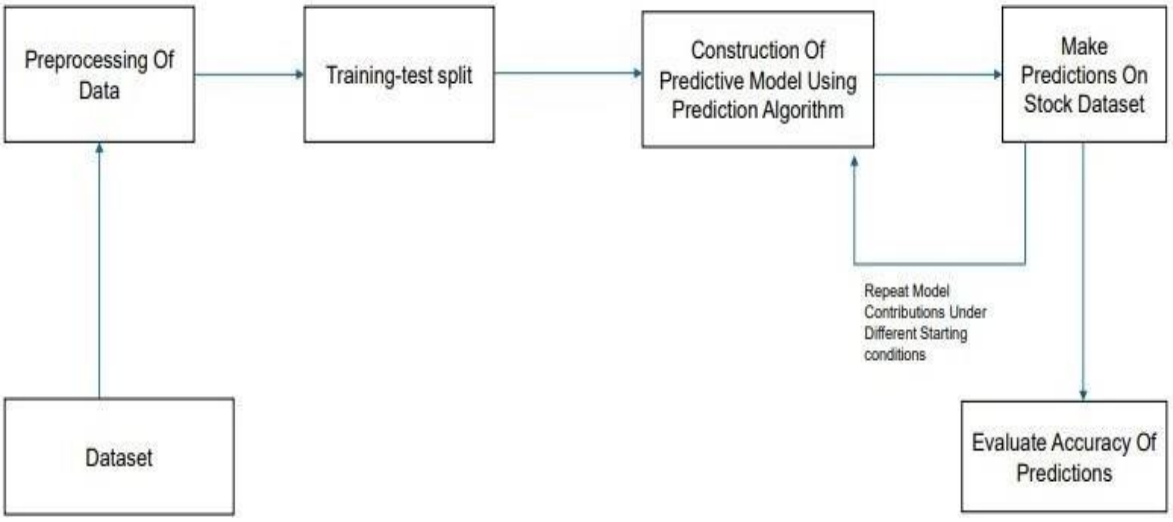
- Usability: It defines the user interface of the software in terms of simplicity of understanding the user interface of stock prediction software, for any kind of stocktrader and other stakeholders in stock market.
- Efficiency: maintaining the possible highest accuracy in the closing stock prices inshortest time with available data.

Chapter 2: DATA AND WORK FLOW

2.1 Data Flow[Basic]:

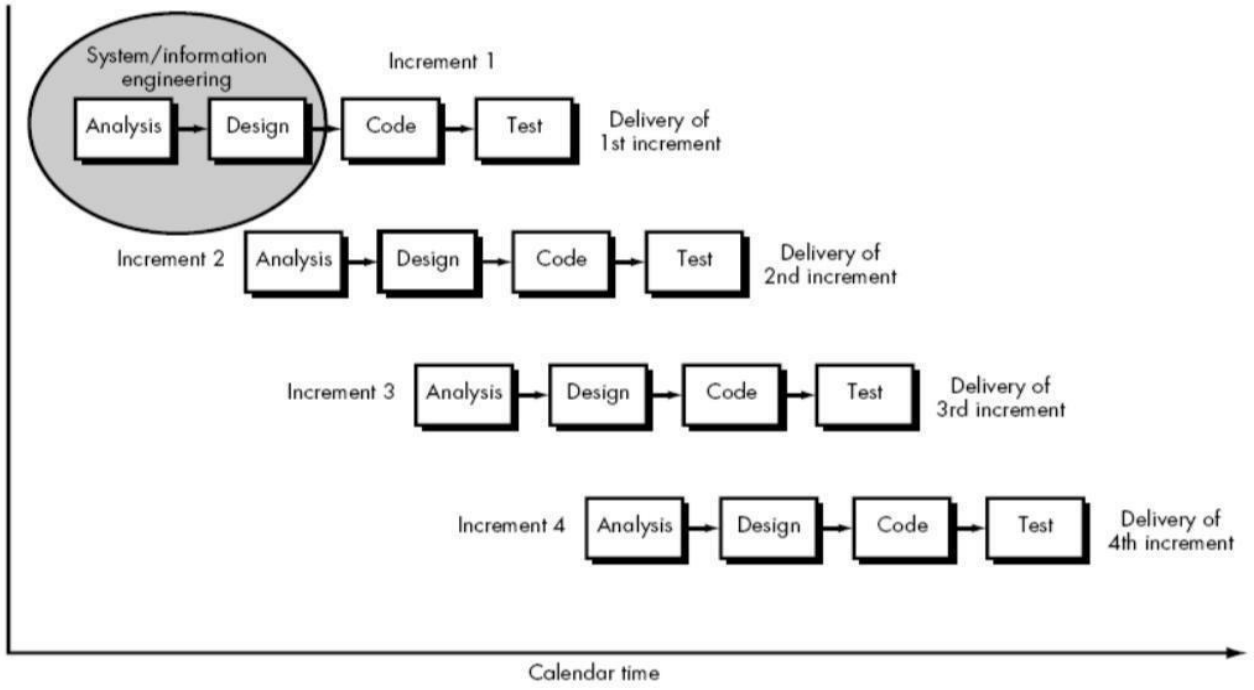


2.2 Data Flow[Main]:



2.3 Work Flow:

This Document plays a vital role in the development life cycle (SDLC) as it describes the complete requirement of the system. It is meant for use by the developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process. INCREMENTAL MODEL was being chosen.



The developer is responsible for:

- Developing the system, which meets the SRS and solving all the requirements ofthe system.
- Demonstrating the system and installing the system at client's location afterthe acceptance testing is successful.
- Submitting the required user manual describing the system interfaces to work onit and also the documents of the system.
- Conducting any user training that might be needed for using the system.
- Maintaining the system for a period of one year after installation.

Chapter 3: DATA SOURCE AND DATASET LIST

3.1 Data Source :

```
import yfinance as yf
```

For using *YFinance* we first need to install it, we can install it by pip install finance. We will start by importing the required library i.e. *yfinance*.

Selecting the Stock by its Symbol

```
symbol = 'SBIN.NS'
ticker = yf.Ticker(symbol)
```

By using ticker function we pass the stock symbol for which we need to download the data.

Downloading the Stock data

```
stock = ticker.history(start ='2019-01-01', end ='2020-01-01')
```

```
stock
```

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
2019-01-01 00:00:00+05:30	290.248787	293.370802	286.687757	292.297607	11837127	0.0	0.0
2019-01-02 00:00:00+05:30	291.809779	295.126902	285.956022	286.736511	25559853	0.0	0.0
2019-01-03 00:00:00+05:30	287.809727	288.346310	283.029165	284.004791	17548347	0.0	0.0
2019-01-04 00:00:00+05:30	284.980398	291.712212	284.395016	290.395111	19514041	0.0	0.0
2019-01-07 00:00:00+05:30	293.712258	294.151302	288.004868	289.078033	14579399	0.0	0.0

Now let us download the stock data using the 'history' function. We can pass the argument like start and end to set a specific time period else we can set period to max which will download all the stock data since it's listing in the market.

Data listed through the Following

```
# get the list of all stocks listed on NSE
nse = pd.read_csv('ind_nifty50list.csv')
pd.set_option('display.max_colwidth', 100)

# get the name of all the stocks
stock_list = nse['Symbol'].to_list()
stock_company = nse['Company Name'].to_list()

# user input
user_input = st.selectbox('Select Stock', stock_company)
user_input_start_date = st.date_input('Start Date', value = pd.to_datetime('2019-01-01'))
user_input_end_date = st.date_input('End Date', value = pd.to_datetime('2020-01-01'))
```

3.1.1 Import Libraries

```
import numpy as np
import pandas as pd
import yfinance as yf
import streamlit as st
import matplotlib.pyplot as plt
import pandas_datareader as data
from keras.models import load_model
from sklearn.preprocessing import MinMaxScaler
```

These are some specific libraries which we have used for the project development

3.1.2 Visualization

```
plt.plot(stock['Date'], stock['Close'])
plt.show()
```

Now we can see the closing price vs time chart graph where X axis denotes time and Y axis denotes closing price.

3.1.3 Moving Average for 100 MA

MA is a popular method to smooth out random movements in the stock market. Similar to a sliding window, an MA is an average that moves along the time scale/periods; older data points get dropped as newer data points are added.

```
plt.plot(stock.Close)
plt.plot(ma100,'r')
plt.xlabel("Days")
plt.ylabel("Price")
plt.title("Stock Price with 100MA")
plt.show()
```

In this section we see the 100MA moving average. For the 100 days it not to able create any moving average. But for the 101 value we get the moving average .That is 101 index which will be just find the min of the previous 100 closing values and it will plot on the graph by using the rolling function.

3.1.4 Moving Average for 100 MA,200MA

```
plt.figure(figsize = (12,6))
plt.plot(stock.Close)
plt.plot(ma100,'r')
plt.plot(ma200,'b')
plt.xlabel("Days")
plt.ylabel("Price")
plt.title("Stock Price with 100MA and 200MA")
plt.show()
```

As the same of 100days moving average we find the 200days moving average. Here the first 200 day is null but for the 201 index the moving average value is there. So in this graph we see the both 100MA and 200MA together.

3.1.5 Final Visualization

```
plt.figure(figsize=(12, 6))
plt.plot(y_test, 'b', label = 'Original Price')
plt.plot(y_predicted, 'r', label = 'Predicted Price ')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```

Here we get the prediction vs original graph .The original price can be shown In Blue and the predicted price will be shown in red line and the x_axis for the “time” and the y_axis for the “price”.

3.2 Dataset List:

3.2.1 splitting data into training and testing

So we should split data for train and test. We use open stock price column to train ana test our data. Here is the code and output.

```
#splitting data into training and testing
data_train = pd.DataFrame(stock['Close'][0:int(len(stock)*0.70)])
data_test = pd.DataFrame(stock['Close'][int(len(stock)*0.70):int(len(stock))])



print(data_train.shape)
print(data_test.shape)
```

```
(170, 1)
(73, 1)
```

```
data_train.head()
```

	Close
0	292.297607
1	286.736511
2	284.004791
3	290.395111
4	289.078033

```
data_test.head()
```

	Close
170	284.590179
171	277.760803
172	267.272827
173	273.565582
174	267.370331

3.2.2 Normalizing the Dataset

Now we need to define a scaler to normalize the data. MinMaxScalar scales all the data to be in the region of 0 and 1. You can also reshape the training and test data to be in the shape .

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range = (0,1))
```

```
data_train_array = scaler.fit_transform(data_train)
data_train_array
```

```
array([[0.35260111],
       [0.30191165],
       [0.277012  ],
       [0.33525983],
       [0.32325466],
       [0.40329   ],
       [0.40329   ],
       [0.4055134 ],
       [0.38061347],
       [0.35971553],
       [0.37083141],
       [0.38417068],
       [0.33303671],
       [0.31124948],
       [0.28901745],
       [0.26945306],
       [0.2374387 ],
       [0.24988866],
       [0.22632254],
```

```
data_train_array.shape
```

```
(170, 1)
```

3.2.3 Creating X_train and Y_train Data Structures and reshape

Now we are dividing into two parts X_train and Y_train . The X_train is the first 100 days which we are going to take for prediction and the Y_train is going to be the predicted value. After getting first predicted value,the first predicted value will apend to the X_train for 100 days prediction and then will predict Y_train . This is how the whole prediction will continue after that reshape the data.

```
x_train =[]
y_train =[]

for i in range(100, data_train_array.shape[0]):
    x_train.append(data_train_array[i-100: i])
    y_train.append(data_train_array[i, 0])

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

```
x_train.shape
```

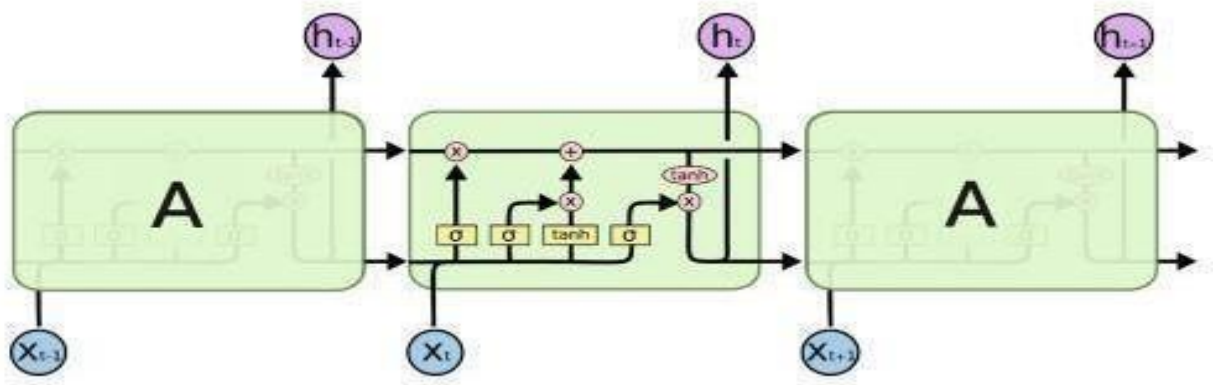
```
(70, 100, 1)
```

CHAPTER 4: MACHINE LEARNING (LSTM) MODEL

4.1 Define LSTM model:

LSTMs are predominantly used to learn, process, and classify sequential data because these networks can learn long-term dependencies between time steps of data. Common LSTM applications include sentiment analysis, language modeling, speech recognition, and video analysis.

4.2 LSTM Architecture:



4.2.1 Forget Gate

A forget gate is responsible for removing information from the cell state.

• The information that is no longer required for the LSTM to understand things or the information that is of less importance is removed via multiplication of a filter.
• This is required for optimizing the performance of the LSTM network.
• This gate takes in two inputs; h_t-1 and x_t. h_t-1 is the hidden state from the previous cell or the output of the previous cell and x_t is the input at that particular time step.

4.2.2 Input Gate

• Regulating what values need to be added to the cell state by involving a sigmoidfunction. This is basically very similar to the forget gate and acts as a filter for allthe information from hi-1 and x_t.
• Creating a vector containing all possible values that can be added (as perceived fromh_t-1 and x_t) to the cell state. This is done using the tanh function, which outputs values from -1 to +1.
• Multiplying the value of the regulatory filter (the sigmoid gate) to the created vector(the tanh function) and then adding this useful information to the cell state via addition operation.

4.2.3 Output Gate

The functioning of an output gate can again be broken down to three steps:
• Creating a vector after applying tanh function to the cell state, thereby scaling thevalues to the range -1 to +1.
• Making a filter using the values of h_t-1 and x_t, such that it can regulate the valuesthat need to be output from the vector created above. This filter again employs a sigmoid function.

• Multiplying the value of this regulatory filter to the vector created in step 1, andsending it out as a output and also to the hidden state of the next cell.

4.3 Working process of LSTM:

LSTMs work in a three-step process.

- The first step in LSTM is to decide which information to be omitted from the cell in that particular time step. It is decided with the help of a sigmoid function. It looks at the previous state (ht-1) and the current input xt and computes the function.
- There are two functions in the second layer. The first is the sigmoid function, and the second is the tanh function. The sigmoid function decides which values to let through (0 or 1). The tanh function gives the weightage to the values passed, deciding their level of importance from -1 to 1.
- The third step is to decide what will be the final output. First, you need to run a sigmoid layer which determines what parts of the cell state make it to the output. Then, you must put the cell state through the tanh function to push the values between -1 and 1 and multiply it by the output of the sigmoid gate.

With this basic understanding of LSTM, we can dive into the hands-on demonstration part of this project regarding stock price prediction using machine learning.

4.3.1 Building the Model by Importing the Libraries and Adding Different Layers to LSTM

We will use the Sequential and LSTM modules provided by Tensorflow Keras to build a simple

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

```
model = Sequential()
model.add(LSTM(units = 50, activation = 'relu', return_sequences = True, input_shape = (x_train.shape[1], 1)))
model.add(Dropout(0.2))

model.add(LSTM(units = 60, activation = 'relu', return_sequences = True))
model.add(Dropout(0.3))

model.add(LSTM(units = 80, activation = 'relu', return_sequences = True))
model.add(Dropout(0.4))

model.add(LSTM(units = 120, activation = 'relu'))
model.add(Dropout(0.5))


model.add(Dense(units = 1))
```

4.3.2 Summarize this model

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 100, 50)           10400

 dropout (Dropout)           (None, 100, 50)           0

 lstm_1 (LSTM)               (None, 100, 60)           26640

 dropout_1 (Dropout)         (None, 100, 60)           0

 lstm_2 (LSTM)               (None, 100, 80)           45120

 dropout_2 (Dropout)         (None, 100, 80)           0

 lstm_3 (LSTM)               (None, 120)               96480

 dropout_3 (Dropout)         (None, 120)               0

 dense (Dense)               (None, 1)                 121

=================================================================
Total params: 178,761
Trainable params: 178,761
Non-trainable params: 0
_________________________________________________________________
```

4.3.3 Fitting the Model

```
model.compile(optimizer = 'adam', loss = 'mean_squared_error')
model.fit(x_train, y_train, epochs = 50)
```

```
Epoch 1/50
3/3 [==============================] - 5s 192ms/step - loss: 0.4221
Epoch 2/50
3/3 [==============================] - 1s 196ms/step - loss: 0.3223
Epoch 3/50
3/3 [==============================] - 1s 180ms/step - loss: 0.1691
Epoch 4/50
3/3 [==============================] - 1s 186ms/step - loss: 0.1141
Epoch 5/50
3/3 [==============================] - 1s 185ms/step - loss: 0.1138
Epoch 6/50
3/3 [==============================] - 1s 195ms/step - loss: 0.1131
Epoch 7/50
3/3 [==============================] - 1s 185ms/step - loss: 0.0923
Epoch 8/50
3/3 [==============================] - 1s 184ms/step - loss: 0.0971
Epoch 9/50
3/3 [==============================] - 1s 182ms/step - loss: 0.0851
Epoch 10/50
3/3 [==============================] - 1s 207ms/step - loss: 0.0816
Epoch 11/50
3/3 [==============================] - 1s 191ms/step - loss: 0.0919
Epoch 12/50
3/3 [==============================] - 1s 197ms/step - loss: 0.0833
Epoch 13/50
3/3 [==============================] - 1s 196ms/step - loss: 0.0842
Epoch 14/50
3/3 [==============================] - 1s 190ms/step - loss: 0.0830
Epoch 15/50
3/3 [==============================] - 1s 191ms/step - loss: 0.0947
Epoch 16/50
3/3 [==============================] - 1s 181ms/step - loss: 0.0741
Epoch 17/50
```

We use two LSTM layers in our model and implement drop out in between for regularization. The number of units assigned in the LSTM parameter is fifty. with a dropout of 10 %. Mean squared error is the loss function for optimizing the problem with adam optimizer. Mean absolute error is the metric used in our LSTM network as it is associated with time-series data

4.3.4 Preparing the Input for the Model

```
x_test = []
y_test = []

for i in range(100, input_data.shape[0]):
    x_test.append(input_data[i-100: i])
    y_test.append(input_data[i, 0])
```

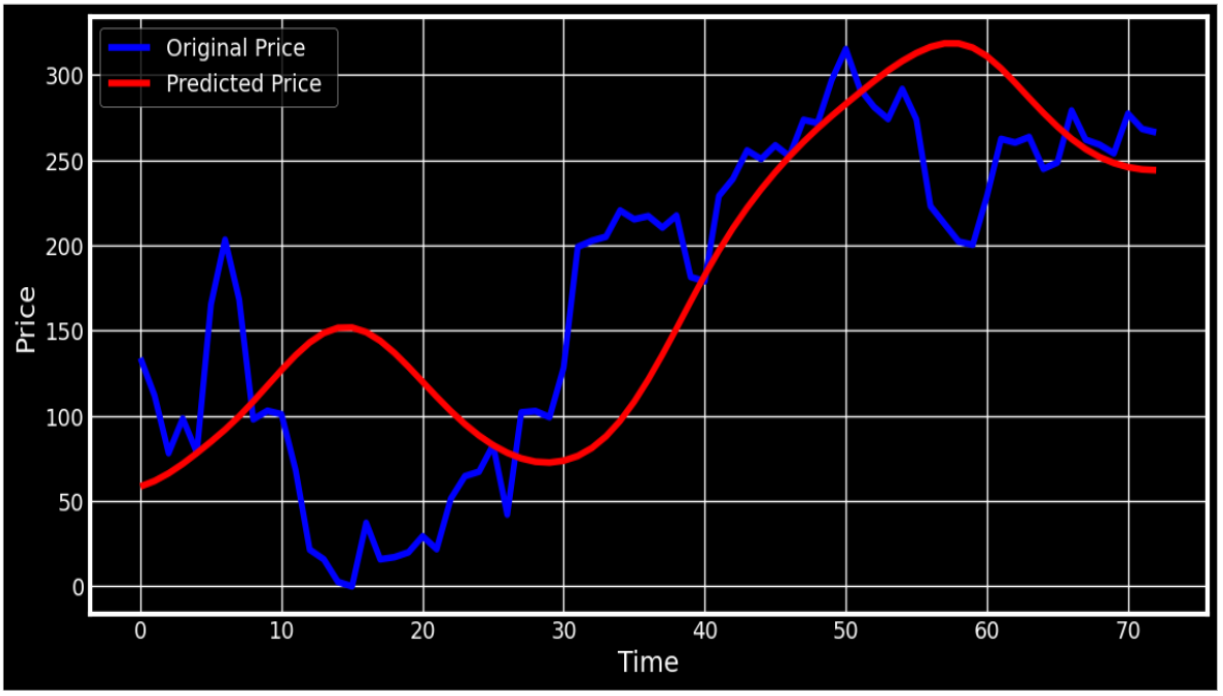
```
x_test, y_test = np.array(x_test), np.array(y_test)
print(x_test.shape)
print(y_test.shape)
```

```
(73, 100, 1)
(73,)
```

4.3.5 Plotting the Actual and Predicted Prices of Stocks.

As we can see below, the model can predict the trend of the actual stock prices very closely. The accuracy of the model can be enhanced by training with more data and increasing the LSTM layers.

```
plt.figure(figsize=(12, 6))
plt.plot(y_test, 'b', label = 'Original Price')
plt.plot(y_predicted, 'r', label = 'Predicted Price ')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```



Chapter 5: CREATING A WEB APP THROUGH STREAMLIT

5.1 Process of the Streamlit connection:

```
streamlit run app.py
```

We have created a file named app.py in vs code where we have used streamlit as open source to build user interface

5.2 Result after Connecting through API:

After connecting through the API, we are getting the y- finance's available company's past history data in our webpage . And here we are selecting any company and the date for the Stock Data Prediction .After selecting the company and date,month,and year we will get the predicted values.

Code :

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import pandas_datareader as data
import sklearn as sk
from sklearn.preprocessing import MaxAbsScaler
import yfinance as yf
```

```
symbol = 'SBIN.NS'
ticker = yf.Ticker(symbol)
```

```
stock = ticker.history(start ='2019-01-01', end ='2020-01-01')
```

```
stock
```

SBI Stock Prediction Table

Stock Trend Prediction

Select Stock

State Bank of India

Start Date

2019/01/01

End Date

2020/01/01

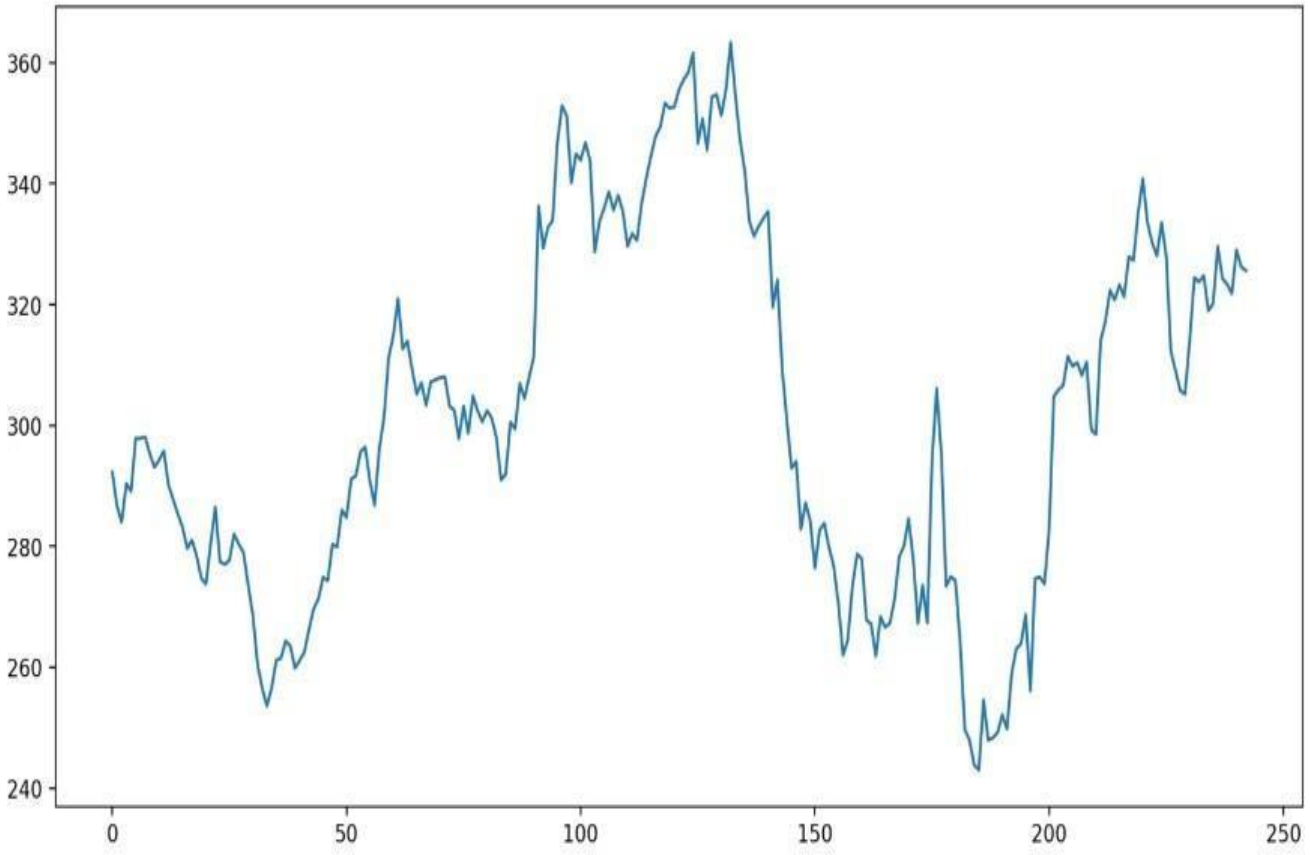
Date from 2010-2019

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2019-01-01	290.2488	293.3708	286.6878	292.2976	11,837,127	0	0
1	2019-01-02	291.8098	295.1269	285.956	286.7365	25,559,853	0	0
2	2019-01-03	287.8097	288.3463	283.0292	284.0048	17,548,347	0	0
3	2019-01-04	284.9804	291.7122	284.395	290.3951	19,514,041	0	0
4	2019-01-07	293.7123	294.1513	288.0049	289.078	14,579,399	0	0
5	2019-01-08	288.5902	298.5416	287.5658	297.8587	22,949,961	0	0
6	2019-01-09	298.5416	299.6148	293.4196	297.8587	21,109,911	0	0
7	2019-01-10	297.6636	300.0051	296.1025	298.1026	16,295,468	0	0

Code

```
st.subheader('Closing Price vs Time chart')
fig = plt.figure(figsize =(12, 6))
plt.plot(stock.Close)
st.pyplot(fig)
```

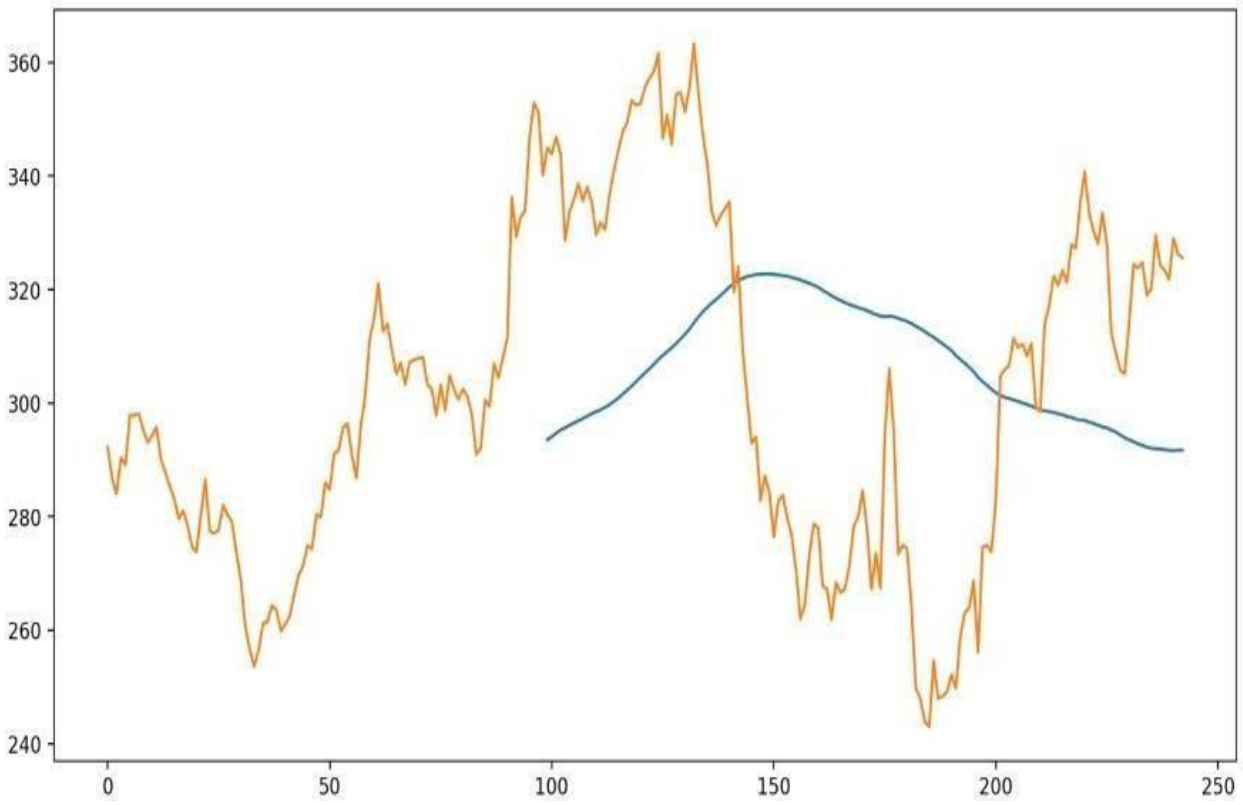
Output Of Closing Price Vs Time Chart Graph



Code

```
st.subheader('Closing Price vs Time chart with 100MA')
ma100 = stock.Close.rolling(100).mean()
fig = plt.figure(figsize =(12, 6))
plt.plot(ma100)
plt.plot(stock.Close)
st.pyplot(fig)
```

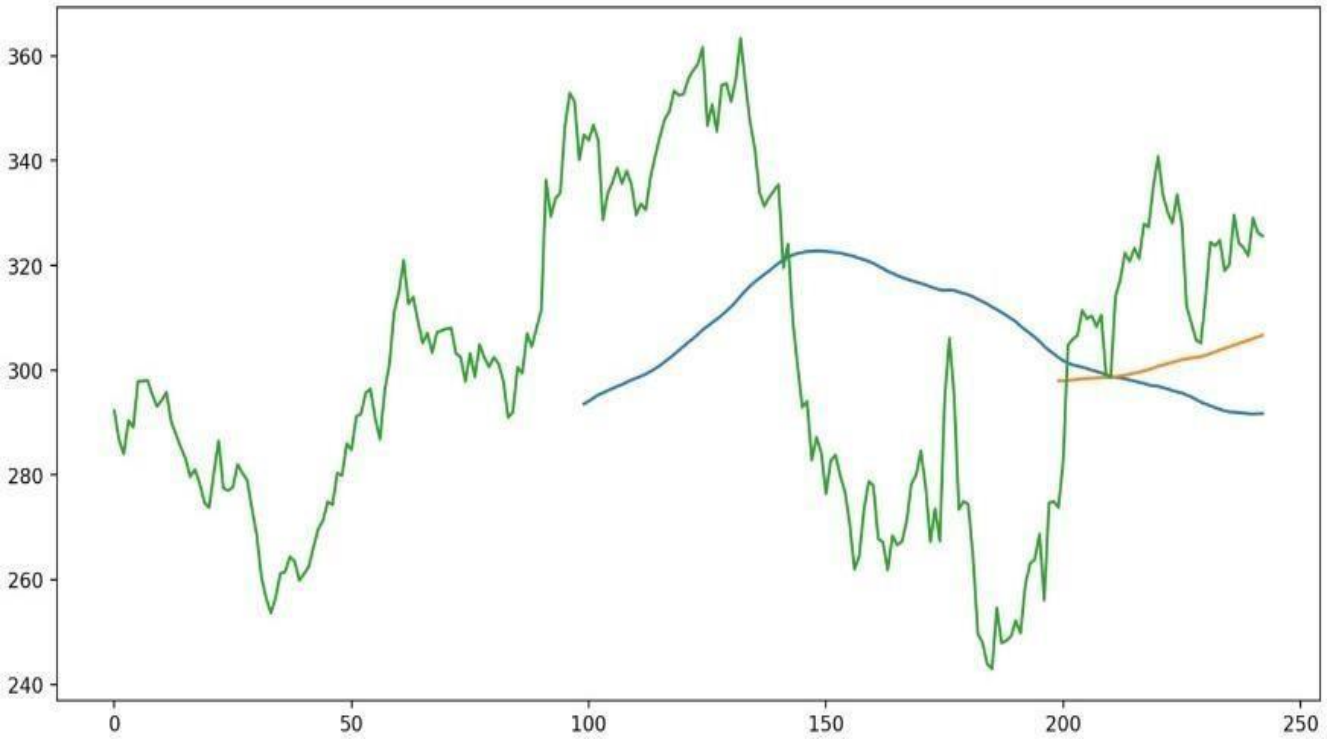
Output Graph Of Closing Price Vs Time Chart With 100MA



Code

```
st.subheader('Closing Price vs Time chart with 100MA & 200MA')
ma100 = stock.Close.rolling(100).mean()
ma200 = stock.Close.rolling(200).mean()
fig = plt.figure(figsize =(12, 6))
plt.plot(ma100)
plt.plot(ma200)
plt.plot(stock.Close)
st.pyplot(fig)
```

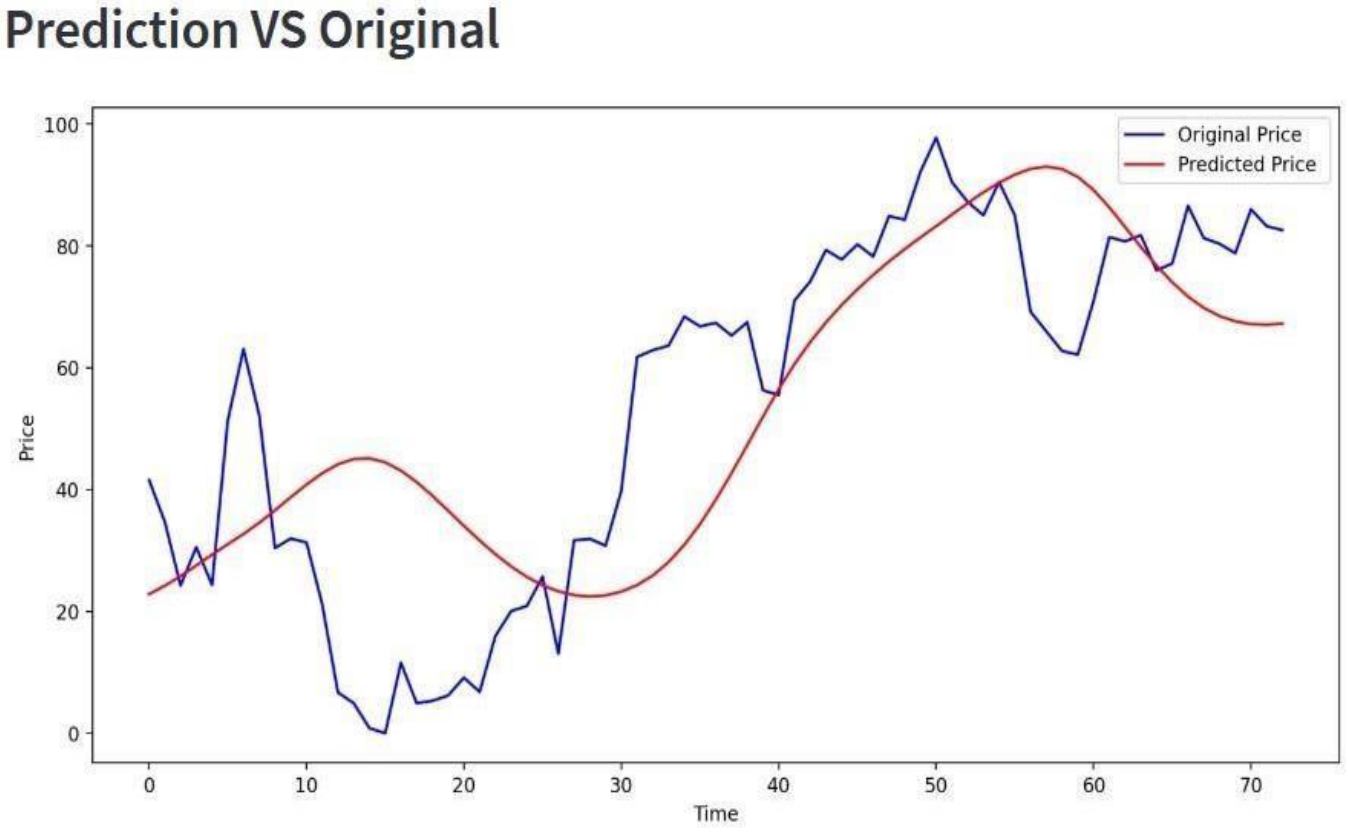
Output Graph Of Closing Price Vs Time Chart With 100MA & 200MA



Code

```
st.subheader('Prediction VS Original')
fig2= plt.figure(figsize=(12, 6))
plt.plot(y_test, 'b', label = 'Original Price')
plt.plot(y_predicted, 'r', label = 'Predicted Price ')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
st.pyplot(fig2)
```

Output Graph Of Prediction Vs Original



AIM OF THE PROJECT

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 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. The paper focuses on the use of Regression and LSTM based Machine learning to predict stock values. Factors considered are open, close, low, high and volume.

FUTURE SCOPE

Enhanced Predictive Models: Machine learning algorithms have already been applied to stock market prediction with varying degrees of success. We can explore advanced machine learning techniques such as deep learning, reinforcement learning, and ensemble methods to improve the accuracy and reliability of stock market predictions.
Integration of Alternative Data Sources: Traditionally, stock market prediction models have relied on historical stock price data and financial indicators. The future scope lies in integrating alternative data sources such as social media sentiment analysis, news sentiment analysis, satellite imagery, and web scraping. Incorporating these diverse data sources can provide additional insights and improve the predictive capabilities of machine learning models.

CONCLUSION

Two techniques have been utilized in this paper : LSTM and Regression, on the Yahoo finance dataset. Both the techniques have shown an improvement in the accuracy of predictions, thereby yielding positive results.
It has led to the conclusion that it is possible to predict stock market with more accuracy and efficiency using machine learning techniques.

REFERENCES

[1] F. a. o. Eugene, "Efficient capital markets: a review of theory and empirical work," Journal offinance, vol. 25, no. 2, pp. 383-417, 1970.

[2] Z. A. Farhath, B. Arputhamary and L. Arockiam, "A Survey on ARIMA Forecasting Using TimeSeries Model,"Int. J. Comput. Sci. Mobile Comput, vol. 5, pp. 104-109, 2016.

[3] S. Wichaidit and S. Kittitornkun, "Predicting SET50 stock prices using CARIMA (cross correlation ARIMA)," in2015 International Computer Science and Engineering Conference (ICSEC), IEEE,2015, pp. 1-4.

[4] D. Mondal, G. Maji, T. Goto, N. C. Debnath and S. Sen, "A Data Warehouse Based Modelling Technique forStock Market Analysis," International Journal of Engineering & Technology, vol. 3,no. 13, pp. 165-170, 2018.

[5] G. Maji, S. Sen and A. Sarkar, "Share Market Sectoral Indices Movement Forecast with Lagged Correlation and Association Rule Mining," in International Conference on Computer Information Stock Price Prediction Using LSTM on Indian Share Market A. Ghosh et al.70Systems and Industrial Management, Bialystok, Poland, Sprigner, 2017, pp.327-340.

[6] M. Roondiwala, H. Patel and S. Varma, "Predicting stock prices using LSTM," InternationalJournal of Scienceand Research (IJSR), vol. 6, no. 4, pp. 1754-1756, 2017.

[7] T. Kim and H. Y. Kim, "Forecasting stock prices with a feature fusion LSTM-CNN model using differentrepresentations of the same data," PloS one, vol. 14, no. 2, p. e0212320, April 2019.

[8] S. Selvin, R. Vinayakumar, E. A. Gopalkrishnan, V. K. Menon and K. P. Soman, "Stock price prediction using LSTM, RNN and CNN-sliding window model," in International Conference on Advances in Computing, Communications and Informatics, 2017.

[9] Hu, Z., Zhao, Y. and Khushi, M., 2021. A survey of forex and stock price prediction using deep learning. Applied System Innovation, 4(1), p.9.

[10] Jain, S., Gupta, R. and Moghe, A.A., 2018, December. Stock price prediction on daily stock data using deep neural networks. In 2018 International Conference on Advanced Computation and Telecommunication (ICACAT) (pp. 1-13). IEEE.