STOCK MARKET PREDICTION USING LINEAR REGRESSION

723721243007 : ARIVUMATHY M

Phase 4 Submission Document

Project: Stock Market Prediction

Introduction:

Stock market prediction is a challenging and highly sought-after field within the realm of finance and data analytics. It involves using historical stock price and market data, along with various analytical and machine learning techniques, to forecast future movements in stock prices or market trends. Accurate stock market predictions are of great interest to investors, traders, financial institutions, and policymakers, as they can influence investment decisions, risk management, and overall market stability.

Here's an introduction to stock market prediction:

- The stock market is a critical component of the global financial system, where shares of publicly traded companies are bought and sold. It's characterized by volatility and influenced by a multitude of factors, including economic indicators, corporate performance, geopolitical events, and investor sentiment.

- Accurate predictions can help investors and traders make informed decisions about buying, selling, or holding stocks, potentially maximizing returns and minimizing losses.
- Financial institutions, including banks and investment firms, rely on stock market predictions to optimize their portfolio management and investment strategies.
- Governments and regulatory bodies may use these predictions to monitor market stability and implement policies to mitigate systemic risk.

Content for Project Phase 4:

Innovating stock price prediction by exploring regression techniques like Linear Regression, Moving average, LSTM for improved Prediction accuracy.

Technical analysis, on the other hand, entails reading charts and analysing statistical data to identify stock market trends. Here we'll concentrate on the technical analysis. To build a model capable of estimating stock prices, we will use the dataset of Microsoft stock prices from 1986 to 2020.

Data Source

A good data source for prediction using deep learning should be Accurate, Complete, Covering the geographic area of interest, Accessible.

Dataset Link:
(https://www.kaggle.com/datasets/prasoonkottarathil/microsoft-lifetime-stocks-dataset)

The dataset contains several variables, including date, open, high, low, close, and volume. The columns Open and Close represent the opening and closing prices of the stock on a given day. The maximum and minimum share prices for the day are represented by High and Low. The number of shares purchased or sold during the day is referred to as volume. Another thing to keep in mind is that the market is closed on weekends and public holidays.

Date	Open	High	Low	Close	Adj Close	Volume
13-03-1986	0.088542	0.101563	0.088542	0.097222	0.062549	1031788800
14-03-1986	0.097222	0.102431	0.097222	0.100694	0.064783	308160000
17-03-1986	0.100694	0.103299	0.100694	0.102431	0.065899	133171200
18-03-1986	0.102431	0.103299	0.098958	0.099826	0.064224	67766400
19-03-1986	0.099826	0.100694	0.097222	0.09809	0.063107	47894400
20-03-1986	0.09809	0.09809	0.094618	0.095486	0.061432	58435200
21-03-1986	0.095486	0.097222	0.091146	0.092882	0.059756	59990400
24-03-1986	0.092882	0.092882	0.08941	0.090278	0.058081	65289600
25-03-1986	0.090278	0.092014	0.08941	0.092014	0.059198	32083200
26-03-1986	0.092014	0.095486	0.091146	0.094618	0.060873	22752000
27-03-1986	0.094618	0.096354	0.094618	0.096354	0.06199	16848000
31-03-1986	0.096354	0.096354	0.09375	0.095486	0.061432	12873600
01-04-1986	0.095486	0.095486	0.094618	0.094618	0.060873	11088000
02-04-1986	0.094618	0.097222	0.094618	0.095486	0.061432	27014400
03-04-1986	0.096354	0.098958	0.096354	0.096354	0.06199	23040000
04-04-1986	0.096354	0.097222	0.096354	0.096354	0.06199	26582400
07-04-1986	0.096354	0.097222	0.092882	0.094618	0.060873	16560000
08-04-1986	0.094618	0.097222	0.094618	0.095486	0.061432	10252800
09-04-1986	0.095486	0.09809	0.095486	0.097222	0.062549	12153600
10-04-1986	0.097222	0.098958	0.095486	0.09809	0.063107	13881600
11-04-1986	0.098958	0.101563	0.098958	0.099826	0.064224	17222400
14-04-1986	0.099826	0.101563	0.099826	0.100694	0.064783	12153600
15-04-1986	0.100694	0.100694	0.097222	0.100694	0.064783	9302400
16-04-1986	0.100694	0.105035	0.099826	0.104167	0.067016	31910400
17-04-1986	0.104167	0.105035	0.104167	0.105035	0.067575	22003200
18-04-1986	0.105035	0.105035	0.100694	0.101563	0.065341	21628800

Data Collection and Preprocessing:

- ✓ Importing the dataset: Obtain a comprehensive dataset containing relevant features such as etc.
- ✓ Data preprocessing: Clean the data by handling missing values, outliers, and categorical variables. Standardize or normalize numerical features.
- ✓ The date column has been formatted as per the coding requirement.

Exploratory Data Analysis (EDA):

- ✓ Visualize and analyse the dataset to gain insights into the relationships between variables.
- ✓ Identify correlations and patterns that can inform feature selection and engineering.
- ✓ Present various data visualizations to gain insights into the dataset.
- ✓ Explore correlations between features and the target variable (Stock market prediction).

FEATURE SELECTION:

The open and close prices of a stock are two important data points in stock market trading and analysis. They represent the price of a stock at the beginning and end of a trading session or a specific time frame, typically a day.

1. Open Price: The open price of a stock is the price at which the first trade of the day occurs when the stock market opens for trading. It marks the starting point for the stock's price movement during the trading session. This price is crucial because it can indicate the sentiment and activity of investors at the opening of the trading day. If the open price is significantly higher than the previous day's close, it may suggest bullish sentiment, while a lower open price may indicate bearish sentiment.

2. Close Price: The close price of a stock is the last price at which a trade is executed before the market closes for the day. It signifies the final price for that trading session and is used to calculate important metrics like the daily price change and the stock's performance for the day. The close price is often more indicative of overall market sentiment and is widely used for technical analysis.

In addition to open and close prices, this data includes the high and low prices of the stock during the trading session. Together, these price points provide valuable information about a stock's performance and can be used for various types of analysis, such as technical analysis and chart pattern recognition.

The maximum (high) and minimum (low) prices of a stock in the stock market are crucial because they provide valuable information about the price range and the volatility of a stock during a particular trading session or time frame. Here's why these prices are important:

Price Range: The high and low prices of a stock indicate the range of prices at which the stock traded during a specific time period, such as a day, week, or month. This range is important because it shows the extent of price fluctuations. A wide price range suggests high volatility, while a narrow range indicates lower volatility. Traders and investors often use this information to assess the level of price movement in a stock.

Volatility Measurement: High and low prices are used to calculate various volatility metrics, such as average true range (ATR) and standard deviation. Volatility measures help traders and investors gauge the potential risk and reward associated with a particular stock. More volatile stocks can provide opportunities for higher returns but also come with greater risk.

In summary, the maximum and minimum prices in the stock market are essential for assessing a stock's price range, volatility, support and resistance levels, and for making informed trading decisions. They provide valuable insights into a stock's price behavior and help traders and investors manage risk and identify potential opportunities in the market.

Adjusted Closing Price: The adjusted closing price of a stock is the stock's closing price on a given trading day adjusted for factors such as dividends, stock splits, and other corporate actions. It reflects the true economic performance of the stock by accounting for events that could otherwise distort the price chart. The adjusted closing price is used in various financial analyses, including technical analysis, as it provides a more accurate picture of a stock's historical performance over time.

In conclusion that all the features in the dataset have been considered important as they reflect to real time effects on the stock price.

Program:

 # Importing required packages

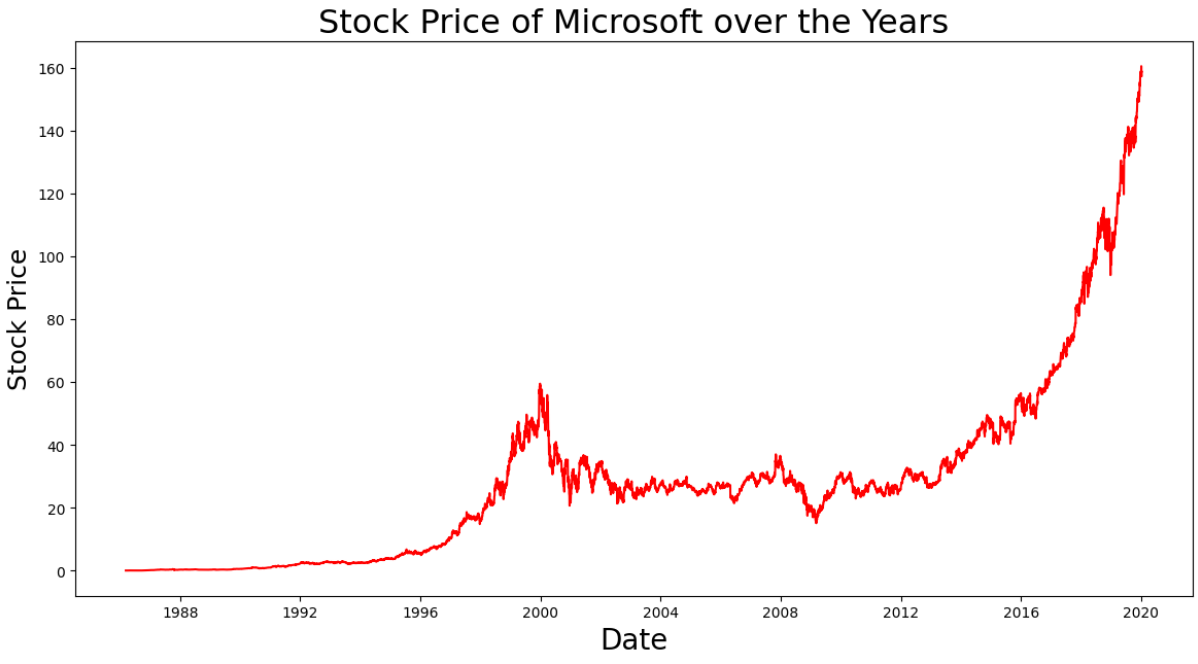
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os

import matplotlib.pyplot as plt
%matplotlib inline
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 20,10
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0, 1))

import sys
import warnings
import datetime as dt
from sklearn.linear_model import LinearRegression
from keras.models import Sequential
from keras.layers import Dense, Dropout, LSTM
from math import floor,ceil,sqrt
from sklearn.linear_model import LinearRegression
from pmdarima.arima import auto_arima
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import MinMaxScaler
```

```
from prophet import Prophet
from sklearn.metrics import r2_score
if not sys.warnoptions:
    warnings.simplefilter("ignore")
```

DATA LOADING:

#importing required Dataset

```
df =pd.read_csv("C:/Downloads/MSFT.csv" , low_memory = False)
df['Date'] = pd.to_datetime(df.Date,format='%m/%d/%Y %H:%M:%S')
df.index = df['Date']
plt.figure(figsize=(14,7))
plt.plot(df['Close'], label='Close Price history',color='r')
plt.xlabel('Date',size=20)
plt.ylabel('Stock Price',size=18)
plt.title('Stock Price of Microsoft over the Years',size=23)
```

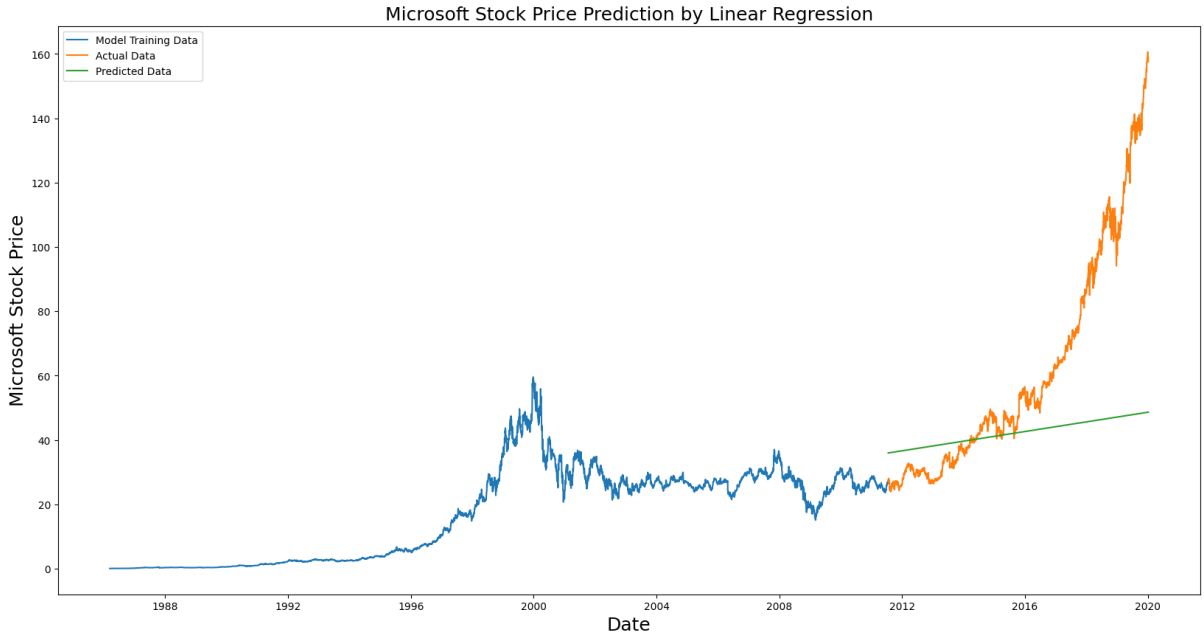


Performing Linear Regression:

```
def Linear_Regression_Prediction(df):
    Shape=df.shape[0]
    df_new=df[['Close']]
    df_new.head()
    train_data_set=df_new.iloc[:ceil(Shape*0.75)]
    valid_data_set=df_new.iloc[ceil(Shape*0.75):]
    print("**********STOCK PRICE PREDICTION BY LINEAR
REGRESSION******")
    print('Shape of Training dataset Set',train_data_set.shape)
    print('Shape of Validation dataset Set',valid_data_set.shape)
    train=train_data_set.reset_index()
    valid=valid_data_set.reset_index()
    x_train = train['Date'].map(dt.datetime.toordinal)
    y_train = train[['Close']]
    x_valid = valid['Date'].map(dt.datetime.toordinal)
    y_valid = valid[['Close']]
    #implement linear regression
    Model = LinearRegression()
    Model.fit(np.array(x_train).reshape(-1,1),y_train)
    preds = Model.predict(np.array(x_valid).reshape(-1,1))
    RMS=np.sqrt(np.mean(np.power((np.array(valid_data_set['Close'])-
preds),2)))
    r2= r2_score(y_valid,preds)
    preds = Model.predict(np.array(x_valid).reshape(-1,1))
    print('(R2 Score)R2 value on validation set:',r2)
    print('(Root Mean Square Error)RMSE value on validation set:',RMS)
    valid_data_set['Predictions'] = preds
    plt.plot(train_data_set['Close'])
    plt.plot(valid_data_set[['Close', 'Predictions']])
    plt.xlabel('Date',size=18)
    plt.ylabel('Microsoft Stock Price',size=18)
    plt.title('Microsoft Stock Price Prediction by Linear Regression',size=18)
    plt.legend(['Model Training Data','Actual Data','Predicted Data'])

Linear_Regression_Prediction(df)
```

Output :-

**********STOCK PRICE PREDICTION BY LINEAR REGRESSION******
Shape of Training dataset Set (6394, 1)
Shape of Validation dataset Set (2131, 1)
(R2 Score)R2 value on validation set: -0.13947302794145022
(Root Mean Square Error)RMSE value on validation set: 39.7286430309166



Performing Moving Average

```
def Moving_Average_Prediction(df):
    shape=df.shape[0]
    df_new=df[['Close']]
    df_new.head()
    train__data_set=df_new.iloc[:ceil(shape*0.75)]
    valid__data_set=df_new.iloc[ceil(shape*0.75):]
    print("***** Microsoft Stock Price Prediction Using Moving Averages*********")
```

```
    print('Shape of Training Data Set of Microsft Stock
Prices',train__data_set.shape)

    print('Shape of Validating Data Set of Microsft Stock
Prices',valid__data_set.shape)

    preds = []

    for i in range(0,valid__data_set.shape[0]):

        a = train__data_set['Close'][len(train__data_set)-
valid__data_set.shape[0]+i:].sum() + sum(preds)

        b = a/(valid__data_set.shape[0])

        preds.append(b)

    RMS=np.sqrt(np.mean(np.power((np.array(valid__data_set['Close'])-
preds),2)))

    r2= r2_score(valid__data_set,preds)

    print('(Root Mean Square Error) RMSE value on validation set:',RMS)

    print('(R2 Score)R2 value on validation set:',r2)

    valid__data_set['Predictions'] = preds

    plt.plot(train__data_set['Close'])

    plt.plot(valid__data_set[['Close', 'Predictions']])

    plt.xlabel('Date',size=18)

    plt.ylabel('Stock Price',size=18)

    plt.title('Stock Price Prediction by Moving Averages',size=18)

    plt.legend(['Model Training Data','Actual Data','Predicted Data'])

    Moving_Average_Prediction(df)
```

Output:

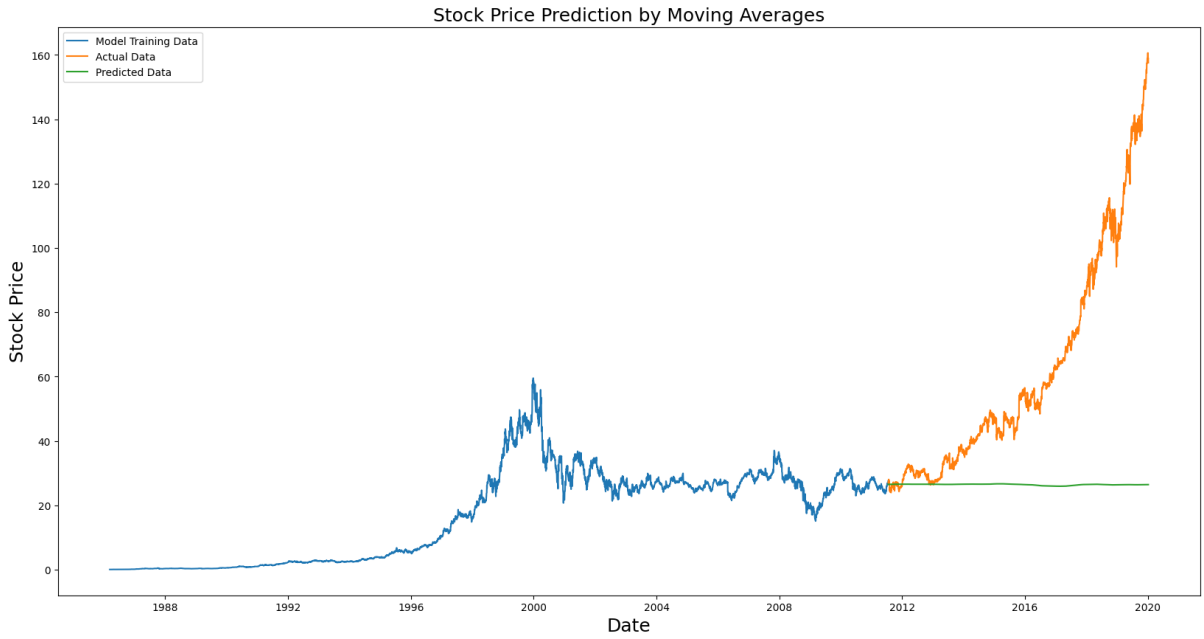
***** Microsoft Stock Price Prediction Using Moving Averages*********

Shape of Training Data Set of Microsft Stock Prices (6394, 1)

Shape of Validating Data Set of Microsft Stock Prices (2131, 1)

(Root Mean Square Error) RMSE value on validation set: 49.41170132172666

(R2 Score)R2 value on validation set: -1.0672822471010206



Performing LSTM

```
def LSTM_Prediction(df):
    shape=df.shape[0]
    df_new=df[['Close']]
    df_new.head()
    dataset = df_new.values
    train=df_new[ceil(shape*0.25):ceil(shape*0.80)]
    valid=df_new[ceil(shape*0.75):]
    print("******** Microsoft STOCK PRICE PREDICTION BY LONG SHORT TERM MEMORY (LSTM) *******")
    print('Shape of Training data Set',train.shape)
    print('Shape of Validation data Set',valid.shape)
    scaler = MinMaxScaler(feature_range=(0, 1))
    scaled_data = scaler.fit_transform(dataset)
    X_train, Y_train = [], []
```

```
    for i in range(40,len(train)):
        X_train.append(scaled_data[i-40:i,0])
        Y_train.append(scaled_data[i,0])
    X_train, Y_train = np.array(X_train), np.array(Y_train)
    Y_train = np.reshape(X_train, (X_train.shape[0],X_train.shape[1],1))
    Model = Sequential()
    Model.add(LSTM(units=50, return_sequences=True,
input_shape=(X_train.shape[1],1)))
    Model.add(LSTM(units=50))
    Model.add(Dense(1))
    Model.compile(loss='mean_squared_error', optimizer='adam')
    Model.fit(X_train, Y_train, epochs=1, batch_size=1, verbose=2)
    inputs = df_new[len(df_new) - len(valid) - 40:].values
    inputs = inputs.reshape(-1,1)
    inputs  = scaler.transform(inputs)
    X_test = []
    for i in range(40,inputs.shape[0]):
        X_test.append(inputs[i-40:i,0])
    X_test = np.array(X_test)
    X_test = np.reshape(X_test, (X_test.shape[0],X_test.shape[1],1))
    closing_price = Model.predict(X_test)
    closing_price = scaler.inverse_transform(closing_price)
    RMS=np.sqrt(np.mean(np.power((valid-closing_price),2)))
    r2 = r2_score(valid-closing_price ,closing_price )
    print('(R2 score)R2 value on validation set:', r2)
    print('(Root Neab Square Error) RMSE value on validation set:',RMS)
    valid['Predictions'] = closing_price
    plt.plot(train['Close'])
```

```
    plt.plot(valid[['Close','Predictions']])
    plt.xlabel('Date',size=20)
    plt.ylabel('Stock Price',size=20)
    plt.title('Microsoft Stock Price Prediction by Long Short Term Memory (LSTM)',size=20)
    plt.legend(['Model Training Data','Actual Data','Predicted Data'])
LSTM_Prediction(df)
```

Output:

******** Microsoft STOCK PRICE PREDICTION BY LONG SHORT TERM MEMORY (LSTM) *******

Shape of Training data Set (4688, 1)

Shape of Validation data Set (2131, 1)

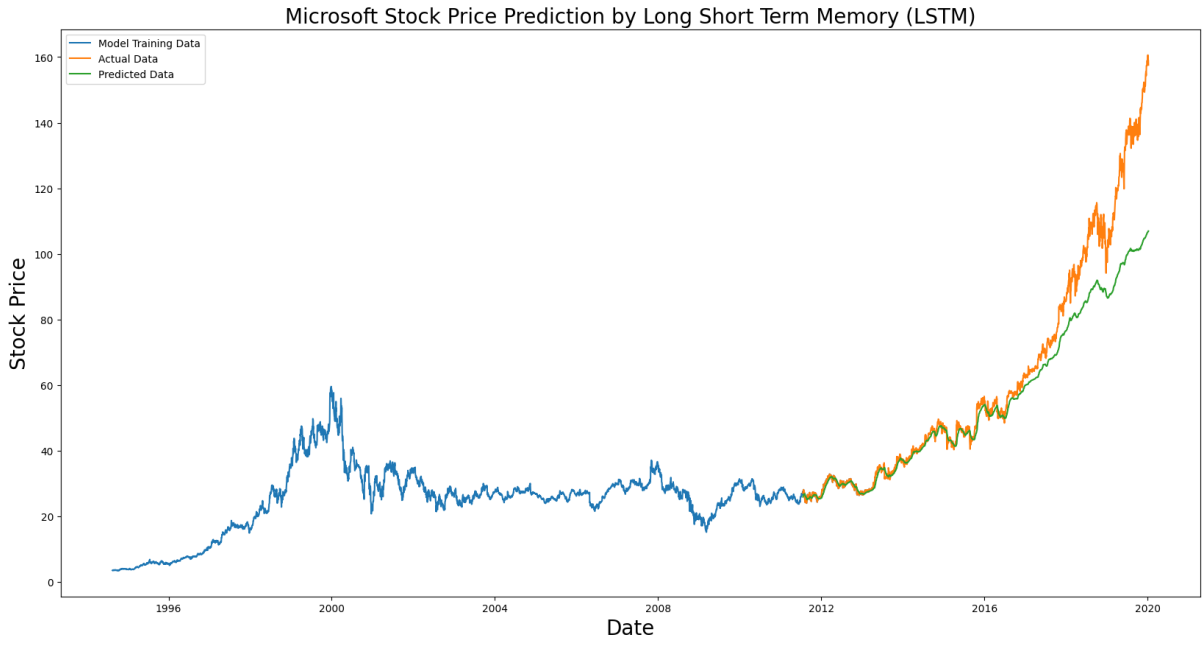
4648/4648 - 109s - loss: 2.1202e-04 - 109s/epoch - 23ms/step

67/67 [==============================] - 3s 22ms/step

(R2 score)R2 value on validation set: -18.290772388082388

(Root Neab Square Error) RMSE value on validation set: Close 13.399323

dtype: float64



Conclusion:

All the features in the dataset have been considered important as they reflect to real time effects on the stock price. Thus, various algorithms like linear regression, moving average and LSTM has been implemented and various performances metrices like r2 value, Root Mean Square Value, etc, are analysed and compared to find the better model.