A Mini Project Report

on

STOCK MARKET PREDICTION

carried out as part of the ***AI Lab DS3230***

Submitted

by

Tanisha Kapoor
219309002
Maheeka Dutta
219309031

Submitted
to
Prof. Abhishek Dwivedi

MANIPAL UNIVERSITY
JAIPUR

INSPIRED BY LIFE

Section A

Group No.: 25-A

School of Information Technology
Department of Data Science and Engineering

MANIPAL UNIVERSITY JAIPUR
RAJASTHAN, INDIA

April 2024

ABSTRACT

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on an exchange. The successful prediction of a stock's future price could yield significant profit. The efficient market hypothesis suggests that stock prices reflect all currently available information and any price changes that are not based on newly revealed information thus are inherently unpredictable. Others disagree and those with this viewpoint possess myriad methods and technologies which purportedly allow them to gain future price information.

This project aims to build a model using python libraries to predict future stock market closing values. 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. Long Short-Term Memory (LSTM) is a type of artificial neural network that is often used in time series analysis. It can effectively predict stock market prices by handling data with multiple input and output timesteps.

TABLE OF CONTENTS

INTRODUCTION

Artificial Intelligence, which consists of making computers perform tasks that normally require human intelligence, is currently the dominant trend in scientific research.

This project aims to build a model using python libraries to predict future stock market closing values. 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. LSTM networks are an extension of recurrent neural networks (RNNs) mainly introduced to handle situations where RNNs fail.

1.It fails to store information for a long period of time. At times, a reference to certain information stored quite a long time ago is required to predict the current output. But RNNs are incapable of handling such “long-term dependencies”.

2.There is no finer control over which part of the context needs to be carried forward and how much of the past needs to be ‘forgotten’.

3.Other issues with RNNs are exploding and vanishing gradients which occur during the training process of a network through backtracking.

THEORY

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to address the vanishing gradient problem encountered in traditional RNNs. It was introduced by Hochreiter and Schmidhuber in 1997 and has since become widely used in various fields, including natural language processing, time series analysis, and speech recognition.

The key feature of LSTM networks is their ability to maintain long-term dependencies in sequential data while mitigating the issues of vanishing or exploding gradients. This is achieved through the use of specialized memory cells and gating mechanisms.

Here's a brief overview of how LSTM works:

Memory Cells: LSTMs have memory cells that can store information over long periods of time. These cells allow the network to retain important information from earlier time steps and selectively update or forget it as needed.

Gating Mechanisms: LSTMs use three main gates to control the flow of information: the input gate, forget gate, and output gate.

Overall, LSTM networks are well-suited for modeling sequential data with long-range dependencies, making them a powerful tool for tasks such as time series forecasting, language translation, and sentiment analysis.

PREDICTION METHODS

Fundamental analysis:

Fundamental analysts are concerned with the company that underlies the stock itself. They evaluate a company's past performance as well as the credibility of its accounts. Many performance ratios are created that aid the fundamental analyst with assessing the validity of a stock, such as the P/E ratio. Warren Buffett is perhaps the most famous of all fundamental analysts. He uses the overall market capitalization-to-GDP ratio to indicate the relative value of the stock market in general, hence this ratio has become known as the "Buffett indicator".

Technical analysis:

Technical Analysis is an analysis methodology for analysing and forecasting the direction of prices through the study of past market data, primarily price and volume. The efficacy of technical analysis is disputed by the efficient-market hypothesis, which states that stock market prices are essentially unpredictable, and research on whether technical analysis offers any benefit has produced mixed results.

OBJECTIVE

1.The main objective of this project is to see which algorithm can improve our model.

2.The Apple stock model takes the online published historical stock-price data as input and produces the prediction of the closing price.

3. This project aims to build a model using python libraries to predict future stock market closing values.

EXPERIMENTAL SETUP AND PROCEDURES

SOFTWARE USED:

1. VS Code
2. Jupyter Notebook

HARDWARE USED:

1. Laptop
2. Installations

CODE:

1.1 Loading Data

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import pandas_datareader as data
import yfinance as yf
start = '2010-01-01'
end = '2019-12-31'
df = yf.download('AAPL', start, end)
df.head()
```

1.2 Resetting index and Dropping columns

```
df = df.reset_index()
df.head()
df = df.drop(['Date','Adj Close'], axis=1)
df.head()
```

1.3 Closing price of apple stock

plt.plot(df.Close)

```
Out[14]: [<matplotlib.lines.Line2D at 0x292f4a75f98>]
```

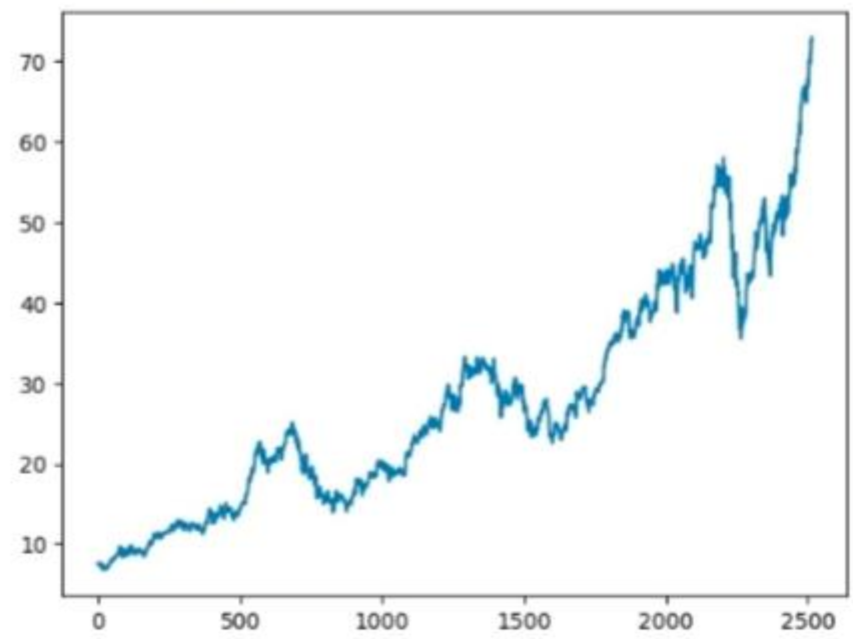


Figure 2.1 Closing price graph

1.4 Moving Average of 100 followed by 200 and Plotting

ma100= df.Close.rolling(100).mean()

ma100

plt.figure(figsize=(12,6))

plt.plot(df.Close)

plt.plot(ma100,'r')

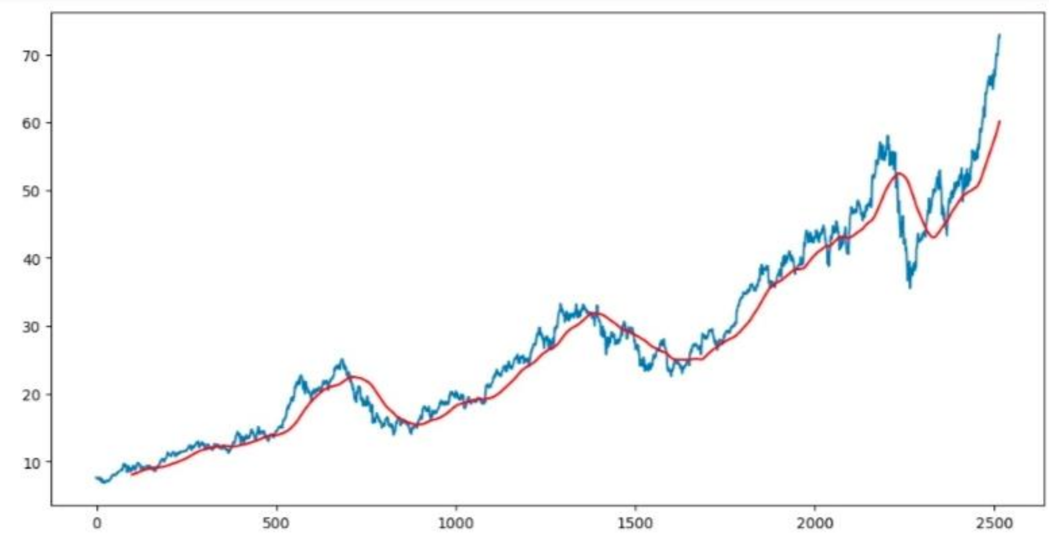


Figure 2.2 Moving Average of 100 Graph

1.5 Plotting both together

```
#plotting both ma100 and ma200on closing prices
plt.figure(figsize=(12,6))
plt.plot(df.Close)
plt.plot(ma100,'r')
plt.plot(ma200,'g')
```

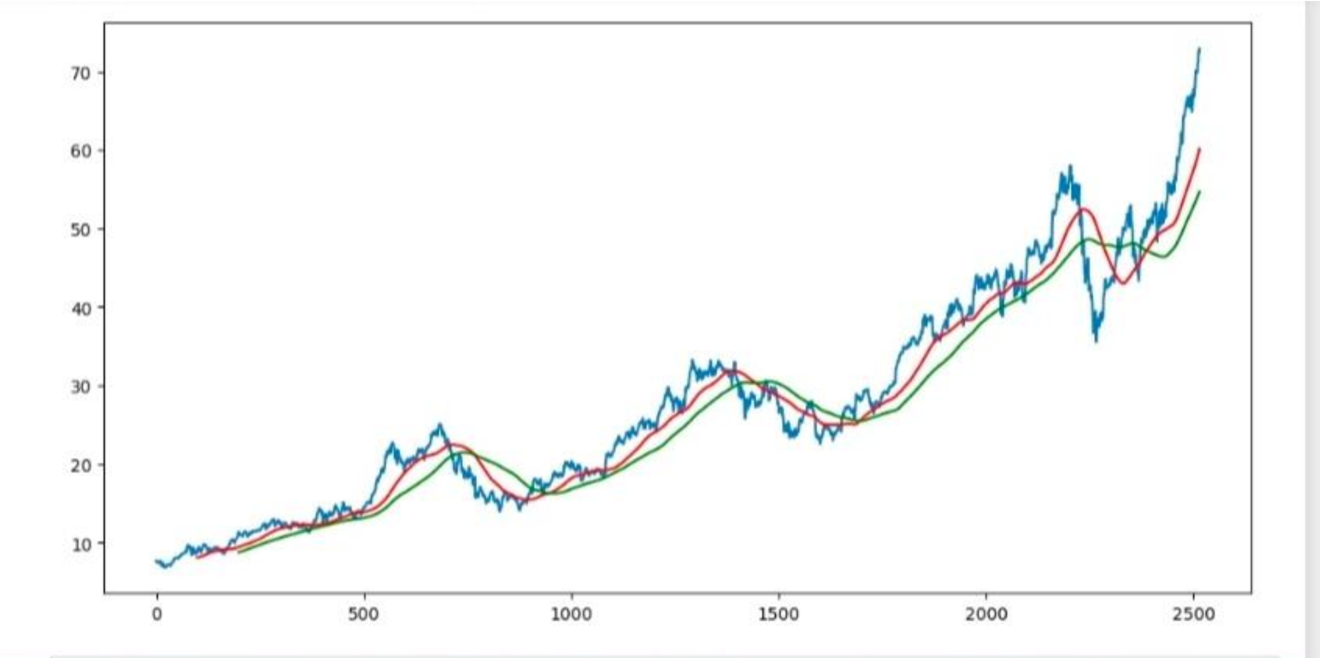


Figure 2.3 Ma100,Ma200 Graph

1.6 Splitting data into train and test set

```
train=pd.DataFrame(df['Close'][0:int(len(df)*0.70)])#going till 70% of the tot vals
test = pd.DataFrame(df['Close'][int(len(df)*0.70): int(len(df))])
print(train.shape)
print(test.shape)
```

1.7 Scaling data btw 0 and 1

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

1.8 LSTM Model

```
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))
x_train=[]
y_train=[]

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

#converting it to numpy arr to be able to use data in LSTM model
x_train,y_train = np.array(x_train),np.array(y_train)


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)
```

```
In [26]: train.head()
```

Out[26]:

	Close
0	7.643214
1	7.656429
2	7.534643
3	7.520714
4	7.570714

```
In [27]: test.head()
```

Out[27]:

	Close
1760	29.182501
1761	28.955000
1762	29.037500
1763	29.004999
1764	29.152500

```
In [82]: #scaling data btw 0 and 1
         from sklearn.preprocessing import MinMaxScaler
         scaler = MinMaxScaler(feature_range=(0,1))
```

Figure 2.4 Test and Train Data

```
In [83]: train_arr=scaler.fit_transform(train)
         train_arr

Out[83]: array([[0.02971782],
                [0.03021854],
                [0.02560389],
                ...,
                [0.84388656],
                [0.85089656],
                [0.84616011]])

In [32]: x_train=[]
         y_train=[]

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

         #converting it to numpy arr to be able to use data in LSTM model
         x_train,y_train = np.array(x_train),np.array(y_train)

In [84]: x_train.shape

Out[84]: (1660, 100, 1)

In [ ]: #ml model

In [33]: from keras.layers import Dense, Dropout , LSTM
         from keras.models import Sequential
```

Figure 2.5 Data Transform

```
In [85]: test.head()
```

Out[85]:

	Close
1760	29.182501
1761	28.955000
1762	29.037500
1763	29.004999
1764	29.152500

```
In [86]: train.tail(100)
```

Out[86]:

	Close
1660	27.092501
1661	27.202499
1662	27.000000
1663	26.982500
1664	27.045000
...	...
1755	29.264999
1756	29.072500
1757	29.129999
1758	29.315001

Figure 2.6 Data Retrieve

1.9 Final Prediction

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

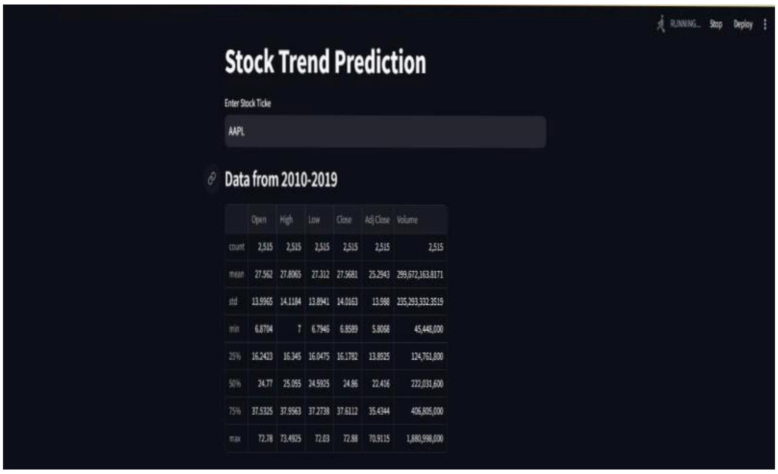
Figure 2.8 Predicted vs Original Price Graph

RESULTS AND DISCUSSIONS

Conclusion

In conclusion, our study demonstrates the effectiveness of Long Short Term Memory-a deep learning neural network in predicting stock market movements based on historical data extracted from yahoo finance. While further refinement is needed to address certain limitations, our findings provide valuable insights into the potential applications of predictive analytics in financial markets.

Web application



app.py 8 ×

C: > Users > PC > OneDrive > Desktop > app.py > ...

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

start = '2010-01-01'
end = '2019-12-31'

st.title('Stock Trend Prediction')

user= st.text_input('Enter Stock Ticke','AAPL')
df = yf.download('AAPL', start ,end)

#describing data
st.subheader('Data from 2010-2019')
st.write(df.describe())

#visualization
st.subheader('Closing Price vs Time Chart')
fig = plt.figure(figsize = (12,6))
plt.plot(df.Close)
st.pyplot(fig)
st.subheader('Closing Price vs Time Chart with 100MA')
ma100 = df.Close.rolling(100).mean()
fig = plt.figure(figsize = (12,6))
plt.plot(ma100)
plt.plot(df.Close)
st.pyplot(fig)

st.subheader('Closing Price vs Time Chart with 100MA & 200MA')
ma100 = df.Close.rolling(100).mean()
ma200 = df.Close.rolling(200).mean()
fig = plt.figure(figsize = (12,6))
plt.plot(ma100, 'r')
plt.plot(ma200, 'g')
plt.plot(df.Close, 'b')
st.pyplot(fig)

#splitting data into train and test
train=pd.DataFrame(df['Close'][0:int(len(df)*0.70)])#going till 70% of the tot vals
test = pd.DataFrame(df['Close'][int(len(df)*0.70): int(len(df))]) #goign till complete length

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

train_arr=scaler.fit_transform(train)
```

REFERENCES

Journal / Conference Papers

[1] Adil MOGHAR ,Mhamed HAMICHE "Stock Market Prediction Using LSTM Recurrent Neural Network", Procedia Computer Science, volume 170, 2020, Pages 1168-1173.

[2] Ayan Maiti,Pushparaj Shetty"**Indian Stock Market Prediction using Deep Learning",2022**

[3] A.J.P.Samarawickrama; T.G.I.Fernando, "A recurrent neural network approach in predicting daily stock prices an application to the Sri Lankan stock market", IEEE, 15-16 December 2017, https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8300345&isnumber=8300323

Web

[1] Stock market research paper, Indian Stock Market Prediction using Deep Learning | IEEE Conference Publication | IEEE Xplore, 3rd April 2024.

[2] https://www.geeksforgeeks.org/stock-price-prediction-using-machine-learning-in-python/