औद्योगिक प्रशिक्षण के लिए राष्ट्रीय संस्थान
National Institute for Industrial Training

NAME OF THE TECHNOLOGY: PYTHON & MACHINE LEARNING

SUBJECT: STOCK MARKET PREDICTION USING PYTHON & MACHINE LEARNING

Submitted by: Shreya Sarkar

Submitted to: Soumotanu Mazumdar

Submitted on: 06-05-2022

CONTENTS:

STUDENT PROFILE

NAME: SHREYA SARKAR

COLLEGE NAME: B.P.PODDAR INSTITUTE OF MANAGEMENT AND TECHNOLOGY

STREAM: B.TECH ECE (3rd Year)

SEMESTER- 6th,3rd year

EMAIL: shreyasarkar0002@gmail.com

YEAR OF PASSING: 2023

PROJECT: STOCK MARKET PREDICTION USING PYTHON & MACHINE LEARNING

GITHUB PROFILE: https://github.com/tutulshreya

ACKNOWLEDGEMENTS:

The final outcome of this project required a lot of guidance and assistance from many people like **Dwippayan Saha** and **Avik Ghosh** and internet source .I am extremely privileged to have got this all along the completion of my project All that I have done is only due to such supervision and assistance and I would not forget to thank them.

I would like to thank my supervisor **Mr. Soumotanu Mazumdar** Of National Institute for Industrial Training who gave me this golden opportunity to work on this project. I got to learn a lot of research from this project about prediction of stock market in daily life. I would also like to thank him for guiding me in an exemplary manner.

Introduction

There are a lot of complicated financial indicators and also the fluctuation of the stock market is highly violent. However, as the technology is getting advanced, the opportunity to gain a steady fortune from the stock market is increased and it also helps experts to find out the most informative indicators to make a better prediction. The prediction of the market value is of great importance to help in maximizing the profit of stock option purchase while keeping the risk low. Recurrent neural networks (RNN) have proved one of the most powerful models for processing sequential data. Long Short-Term memory is one of the most successful RNNs architectures. LSTM introduces the memory cell, a unit of computation that replaces traditional artificial neurons in the hidden layer of the network. With these memory cells, networks are able to effectively associate memories and input remote in time, hence suit to grasp the structure of data dynamically over time with high prediction capacity

OBJECTIVES:

The main objectives of the proposed system are:

(i) To see in which precision a Machine learning algorithm can predict and how much the epochs can improve our model.

(ii) The main objective here is to obtain the most accurate trained algorithm, to predict future values.

HARDWARE & SOFTWARE REQUIREMENTS:

Hardware Requirements:

Machine: DESKTOP-DL5L3HD

Speed: 2.40 GHz

RAM Storage: 4.00 GB

Software Requirements:

Operating System: Windows 10 Home Single Language

Platform: Google Colab

Future Scopes of Python

(i)Python is one of the fastest growing languages and has undergone a successful span of more than 25 years as far as its adoption is concerned. This success also reveals a promising future scope of python programming language.

(ii) In fact, it has been continuously serving as the best programming language for application development, web development, game development, system administration, scientific and numeric computing, GIS and Mapping etc.

(iii) Python has become the core language as far as the success of these technologies is concerned. Python programming language is undoubtedly dominating the other languages when future technologies like Artificial Intelligence (AI) come into the play.

(iv) The future scope of python programming language can also be predicted by the way it has helped big data technology to grow. Python has been successfully contributing in analysing a large number of data sets across computer clusters through its high-performance toolkits and libraries.

(v) Networking is another field in which python has a brighter scope in the future. Python programming language is used to read, write and configure routers and switches and perform other networking automation tasks in a cost-effective and secure.

LSTM Recurrent Neural Network

Long-Short-Term Memory Recurrent Neural Network belongs to the family of deep learning algorithms. It is a recurrent network because of the feedback connections in its architecture. It has an advantage over traditional neural networks due to its capability to process the entire sequence of data. Its architecture comprises the *cell*, *input gate*, *output gate* and *forget gate*.

The cell remembers values over arbitrary time intervals, and the three gates regulate the flow of information into and out of the cell. The cell of the model is responsible for keeping track of the dependencies between the elements in the input sequence. The input gate controls the extent to which a new value flows into the cell, the forget gate controls the extent to which a value remains in the cell, and the output gate controls the extent to which the value in the cell is used to compute the output activation of the LSTM unit.

However, there are some variants of the LSTM model such as Gated Recurrent Units (GRUs) that do not have the output gate. LSTM Networks are popularly used on time-series data for classification, processing, and making predictions. The reason for its popularity in time-series application is that there can be several lags of unknown duration between important events in a time series.

ADVANTAGES:

(i) Over the years, various machine learning techniques have been used in stock market prediction, but with the increased amount of data and expectation of more accurate prediction, the deep learning models are being used nowadays which have proven their advantage over traditional machine learning methods in terms of accuracy and speed of prediction.

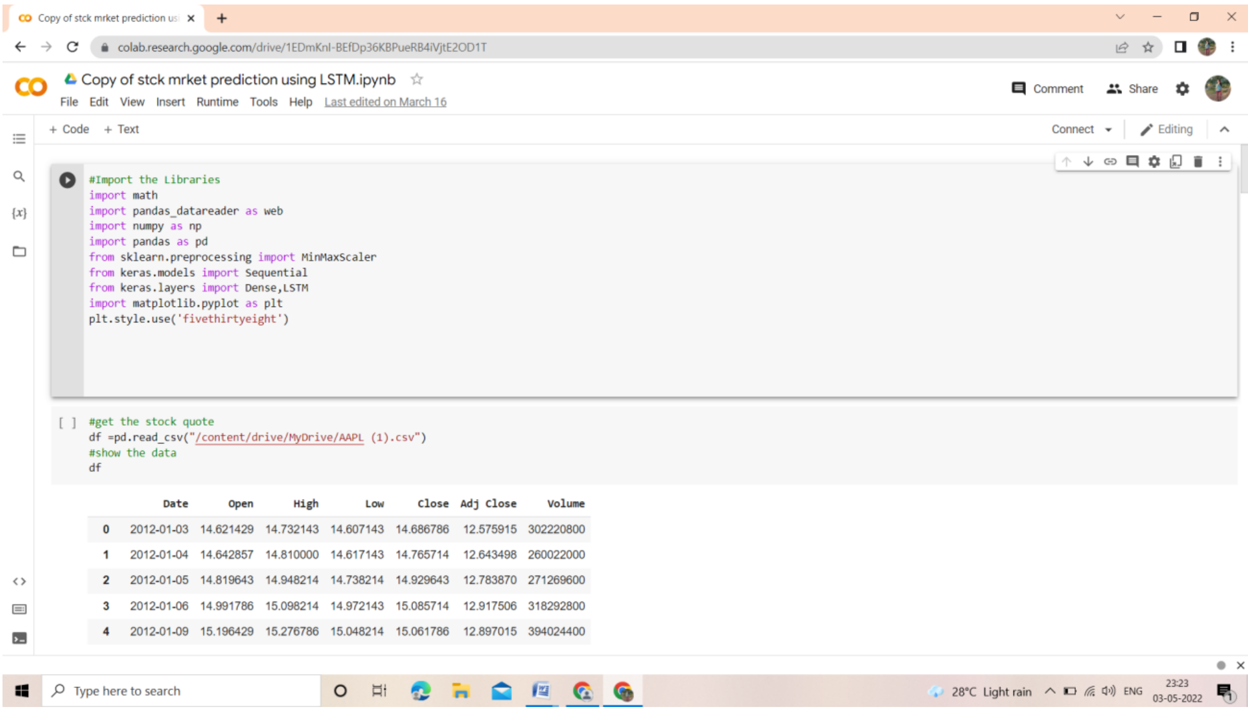
(ii) Long-Short-Term Memory Recurrent Neural Network belongs to the family of deep learning algorithms. It is a recurrent network because of the feedback connections in its architecture. It has an advantage over traditional neural networks due to its capability to process the entire sequence of data.

DISADVANTAGES:

(i) Predicting stock prices is an uncertain task which is modelled using machine learning to predict the return on stocks.

(ii) The stock market is considered to be very dynamic and complex in nature. An accurate prediction of future prices may lead to a higher yield of profit for investors through stock investments.

SOME CODE SNIPPETS & OUTPUTS:




```python
#Import the Libraries
import math
import pandas_datareader as web
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense,LSTM
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
```

```python
#get the stock quote
df =pd.read_csv("/content/drive/MyDrive/AAPL (1).csv")
#show the data
df
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2012-01-03	14.621429	14.732143	14.607143	14.686786	12.575915	302220800
1	2012-01-04	14.642857	14.810000	14.617143	14.765714	12.643498	260022000
2	2012-01-05	14.819643	14.948214	14.738214	14.929643	12.783870	271269600
3	2012-01-06	14.991786	15.098214	14.972143	15.085714	12.917506	318292800
4	2012-01-09	15.196429	15.276786	15.048214	15.061786	12.897015	394024400

Copy of stck mrket prediction using LSTM.ipynb

...	...	...	...	...	...	...	...
1998	2019-12-11	67.202499	67.775002	67.125000	67.692497	66.616829	78756800
1999	2019-12-12	66.945000	68.139999	66.830002	67.864998	66.786591	137310400
2000	2019-12-13	67.864998	68.824997	67.732498	68.787498	67.694427	133587600
2001	2019-12-16	69.250000	70.197502	69.245003	69.964996	68.853210	128186000
2002	2019-12-17	69.892502	70.442497	69.699997	70.102501	68.988541	114158400

2003 rows × 7 columns

```
#get the no of rows and columns in the dataset
df.shape
```

(2003, 7)

```
#visualize the closing price history
plt.figure(figsize=(16,8))
plt.title('Close Price History')
plt.plot(df['Close'])
plt.xlabel('Date',fontsize=18)
plt.ylabel(' Close Price USD($)', fontsize=18)
plt.show()
```

Copy of stck mrket prediction using LSTM.ipynb


```
#creating a new dataframe with only for the 'Close' Column
data=df.filter(['Close'])
#Convert the dataframe to a numpy array
```

```
#Convert the dataframe to a numpy array
dataset=data.values
#get the number of rows to train the model on
training_data_len=math.ceil(len(dataset)*.8)
training_data_len
```

```
1603
```

```
#Scale the data
scaler=MinMaxScaler(feature_range=(0,1))
scaled_data=scaler.fit_transform(dataset)
scaled_data
```

```
array([[0.0131651 ],
       [0.01457063],
       [0.01748986],
       ...,
       [0.97658262],
       [0.99755133],
       [1.        ]])
```

```
#create the training dataset
#create the scaled training dataset
train_data=scaled_data[0:training_data_len,:]
#split the datab into x_train & y_train datasets
x_train=[]
y_train=[]
for i in range (60,len(train_data)):
  x_train.append(train_data[i-60:i,0])
  y_train.append(train_data[i,0])
  if i <=60:
```

```
    print()
```

```
[array([0.0131651 , 0.01457063, 0.01748986, 0.02026915, 0.01984304,
       0.02080338, 0.02036454, 0.0196268 , 0.01862192, 0.02173194,
       0.02453667, 0.02367173, 0.01893356, 0.02345549, 0.01900353,
       0.03569839, 0.03440732, 0.03609271, 0.03973694, 0.04194383,
       0.04175942, 0.04107891, 0.04397904, 0.04670743, 0.0497984 ,
       0.05479095, 0.0652785 , 0.0654375 , 0.07127595, 0.07563886,
       0.0681405 , 0.07102789, 0.07097067, 0.07906688, 0.07791571,
       0.08004628, 0.08387497, 0.08600557, 0.09214292, 0.09661394,
       0.09790501, 0.0983566 , 0.09071196, 0.08886754, 0.08914104,
       0.09632779, 0.09835024, 0.10269409, 0.11293359, 0.12659476,
       0.12403805, 0.12404441, 0.13392141, 0.13701237, 0.1348118 ,
       0.13280208, 0.13070964, 0.13766104, 0.14243104, 0.14442806])]
[0.139492723007876]
```

```
#convert the x_train & y_train into numpy arrays
x_train,y_train=np.array(x_train),np.array(y_train)
```

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

```
(1543, 60, 1)
```

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

Copy of stck mrket prediction using LSTM.ipynb

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

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

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

```
1543/1543 [==============================] - 44s 26ms/step - loss: 7.9555e-04
<keras.callbacks.History at 0x7fd15714cc10>
```

```
#Create the testing dataset
#create a new array containing scaled values from index 1543 to 2003
test_data=scaled_data[training_data_len-60: , :]
#create the data sets x_test & y_tests
x_test=[]
y_test=dataset[training_data_len: , :]
for i in range(60,len(test_data)):
  x_test.append(test_data[i-60:i,0])
```

```
#convert the data into a numpy array
x_test=np.array(x_test)
```

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

Copy of stck mrket prediction using LSTM.ipynb

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

```
#Get the models predicted price values
predictions=model.predict(x_test)
predictions=scaler.inverse_transform(predictions)
```

```
#get the RMSE
rmse=np.sqrt(np.mean(predictions-y_test)**2)
rmse
```

```
0.014158333908080998
```

```
#plot the data
train=data[:training_data_len]
valid=data[training_data_len:]
valid['Predictions'] = predictions
#visualize the data
plt.figure(figsize=(16,8))
plt.title('Model')
plt.xlabel('Date',fontsize=18)
plt.ylabel('Close price USD($)',fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close','Predictions']])
plt.legend(['Train','Val','predictions'],loc='lower right')
plt.show()
```

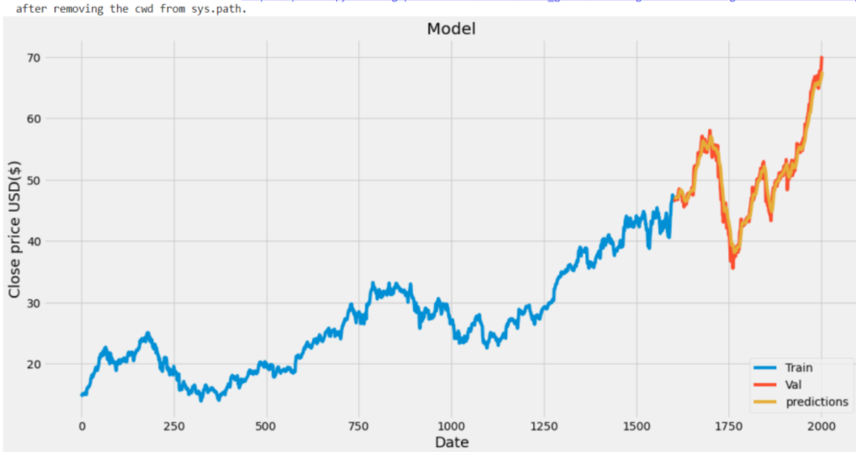
Copy of stck mrket prediction using LSTM.ipynb

File Edit View Insert Runtime Tools Help Last edited on March 16

+ Code + Text

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  after removing the cwd from sys.path.
```



Copy of stck mrket prediction using LSTM.ipynb

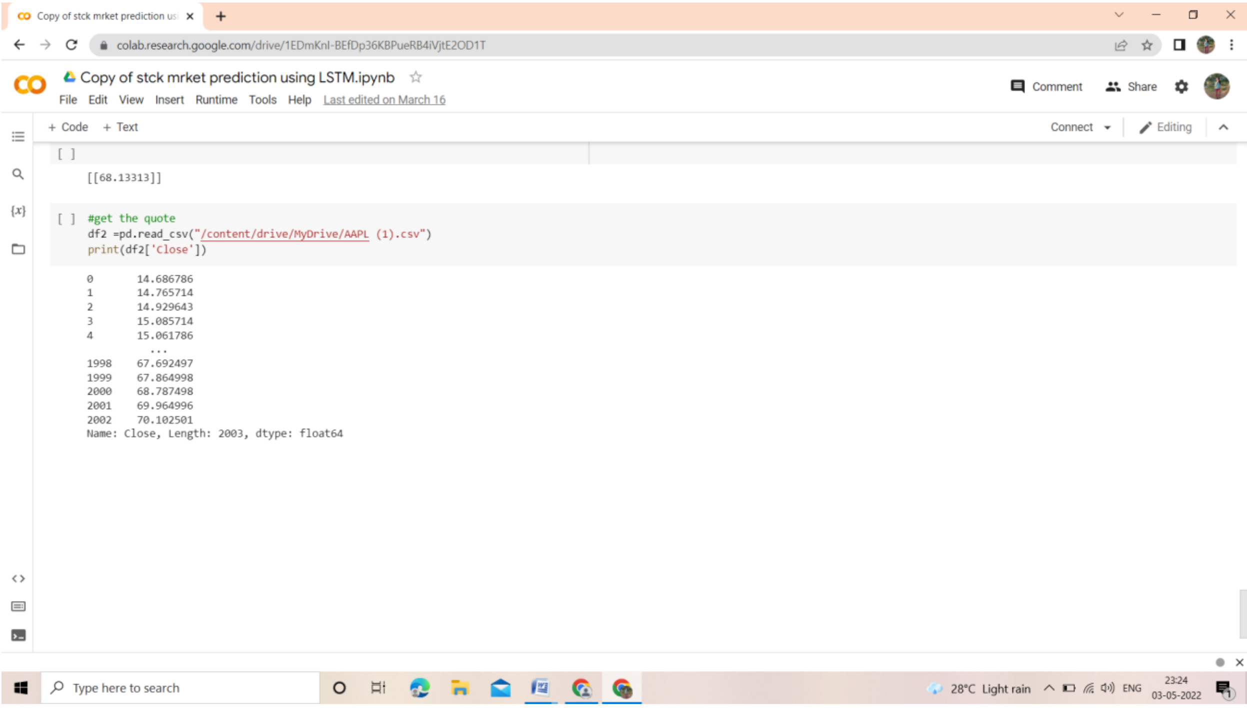
File Edit View Insert Runtime Tools Help Last edited on March 16

```
#Show the valid & predicted prices
valid
```

	Close	Predictions
1603	46.747501	47.206738
1604	46.577499	47.166935
1605	46.907501	47.073898
1606	46.790001	47.021233
1607	47.090000	46.976311
...	...	...
1998	67.692497	66.136826
1999	67.864998	66.405334
2000	68.787498	66.678093
2001	69.964996	67.055069
2002	70.102501	67.596741

400 rows × 2 columns

```
#get the quote
df =pd.read_csv("/content/drive/MyDrive/AAPL (1).csv")
#Create a new dataframe
new_df=df.filter(['Close'])
#get the last 60 days closing price values and convert the dataframe to an array
last_60_days=new_df[-60:].values
```


Copy of stck mrket prediction using LSTM.ipynb
colab.research.google.com/drive/1EDmKnl-BEfDp36KBPueRB4iVjtE2OD1T
File Edit View Insert Runtime Tools Help Last edited on March 16
Comment Share
+ Code + Text
Connect Editing
[[68.13313]]
#get the quote
df2 =pd.read_csv("/content/drive/MyDrive/AAPL (1).csv")
print(df2['Close'])
0 14.686786
1 14.765714
2 14.929643
3 15.085714
4 15.061786
...
1998 67.692497
1999 67.864998
2000 68.787498
2001 69.964996
2002 70.102501
Name: Close, Length: 2003, dtype: float64
Type here to search
28°C Light rain
ENG
23:24
03-05-2022

Conclusion

This paper proposes RNN based on LSTM built to forecast future values, the result of our model has shown some promising result. The testing result conform that our model is capable of tracing the evolution of opening prices for both assets. For our future work we will try to find the best sets for bout data length and number of training epochs that beater suit our assets and maximize our predictions accuracy.

BIBLIOGRAPHY

[1] Moritz, B., & Zimmermann, T. (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. Available at SSRN 2740751.

[2] Olah, C. (2015). Understanding lstm networks–colah's blog. Colah.

From where I have gathered all the contents

Information: Google Scholar

Images: Google

Coding & Snapshots: Self Performed