

STOCK PRICE PREDICTION

PROJECT

BY

SK.HASEENA BEGUM & J.BHUVANASREE

S180720 &S180073

CSE-2E

PREFACE

About the project:

Stock price prediction using machine learning helps you discover the future value of company stock and other financial assets traded on an exchanges....

The entire idea of predicting stock prices is to gain significant profits...

INTRODUCTION

Stock Market:

Stock market is a place where buying and selling of shares happen for publicly listed companies.

Stock exchange is the mediator that allows buying and selling of shares.

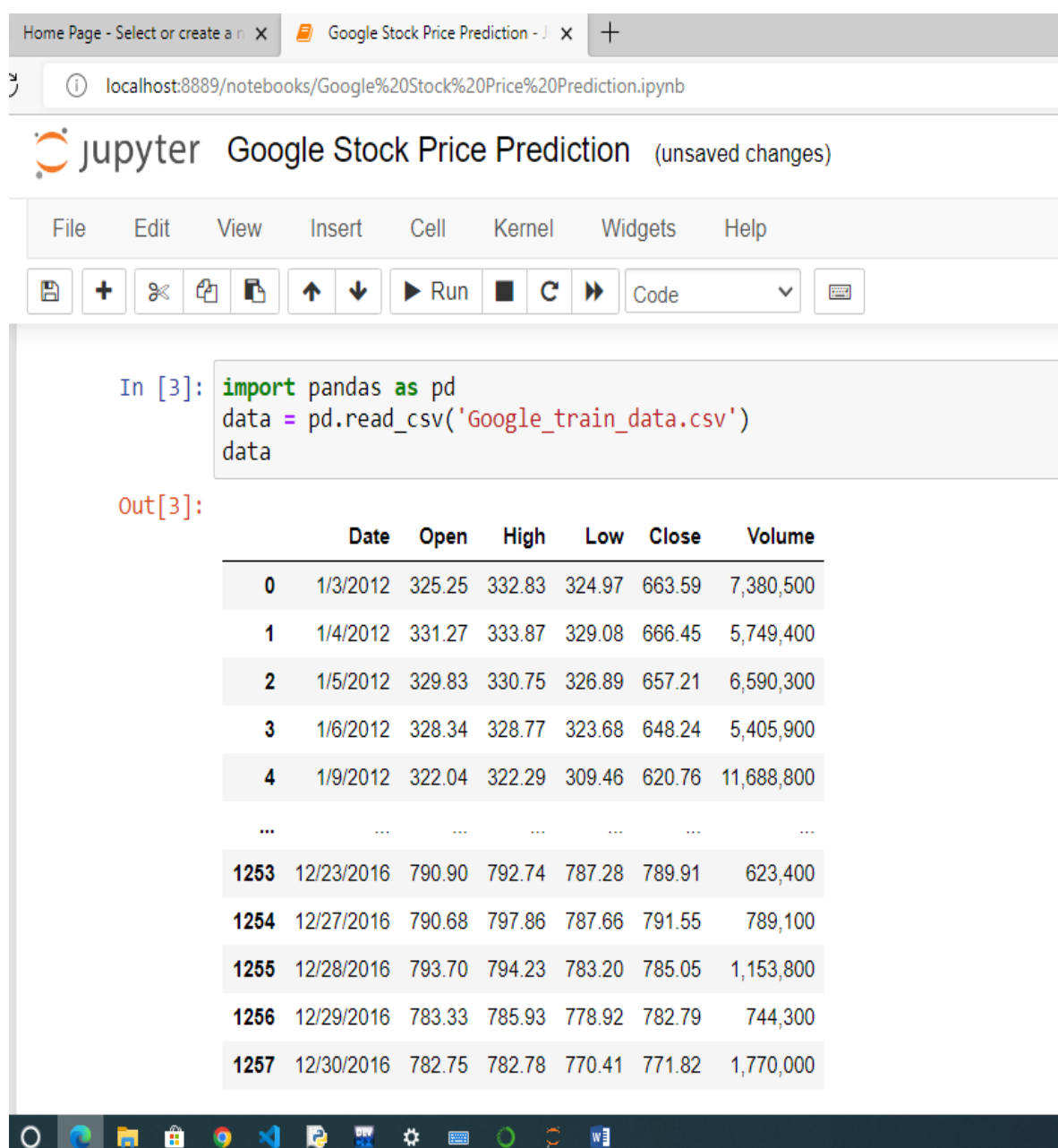
Importance of stock market:

- Helps companies to raise capital
- Helps create personal wealth
- Serves as an indicator of the state of the economy

- Helps to increase investment

Analysis

Taken Columns: High, Open, Close, Low, Volume..



Home Page - Select or create a notebook | Google Stock Price Prediction - Jupyter Notebook | +

localhost:8889/notebooks/Google%20Stock%20Price%20Prediction.ipynb

jupyter Google Stock Price Prediction (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
In [3]: import pandas as pd
data = pd.read_csv('Google_train_data.csv')
data
```

Out[3]:

	Date	Open	High	Low	Close	Volume
0	1/3/2012	325.25	332.83	324.97	663.59	7,380,500
1	1/4/2012	331.27	333.87	329.08	666.45	5,749,400
2	1/5/2012	329.83	330.75	326.89	657.21	6,590,300
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	11,688,800
...
1253	12/23/2016	790.90	792.74	787.28	789.91	623,400
1254	12/27/2016	790.68	797.86	787.66	791.55	789,100
1255	12/28/2016	793.70	794.23	783.20	785.05	1,153,800
1256	12/29/2016	783.33	785.93	778.92	782.79	744,300
1257	12/30/2016	782.75	782.78	770.41	771.82	1,770,000

High: High is the highest price at which the stock trading during the period.

Open: The price at which started trading when the market open on a particular date.

Close: Close refers to price of an individual stock and stock exchange closed market on the day. It represents last buy and sell order executed between two traders.

Low: Lowest price in the period.

Volume: Volume is the total amount of trading activity during the period of the time.

CODE:

Importing the libraries

```
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
```

Load the training dataset:

```
In [3]: data = pd.read_csv('Google_train_data.csv')
data.head()
```

```
Out[3]:
```

	Date	Open	High	Low	Close	Volume
0	1/3/2012	325.25	332.83	324.97	663.59	7,380,500
1	1/4/2012	331.27	333.87	329.08	666.45	5,749,400
2	1/5/2012	329.83	330.75	326.89	657.21	6,590,300
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	11,688,800

Creating x_train and y_train data structures:

```
In [8]: 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)
```

Reshape the dataset

```
In [9]: X_train = np.reshape(X_train,(X_train.shape[0],X_train.shape[1],1))
X_train.shape
```

```
Out[9]: (1089, 60, 1)
```

Building the model by adding layers

```
In [10]: 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')
```

Fitting the model

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

Epoch 1/20
- 12s - loss: 0.0322
Epoch 2/20
```

Preparing the input for the model

```
In [13]: testData = pd.read_csv('Google_test_data.csv')
testData["Close"] = pd.to_numeric(testData.Close, errors='coerce')
testData = testData.dropna()
testData = testData.iloc[:, 4:5]
y_test = testData.iloc[60:, 0].values
#input array for the model
inputClosing = testData.iloc[:, 0].values |
inputClosing_scaled = sc.transform(inputClosing)
inputClosing_scaled.shape
X_test = []
length = len(testData)
timestep = 60
for i in range(timestep, length):
    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
```

```
Out[13]: (192, 60, 1)
```

Predicting the values for stock prices

```
In [15]: y_pred = model.predict(X_test)
          y_pred
```

```
Out[15]: array([[1.1260811],  
                [1.1293991],  
                [1.1416496],  
                [1.1594812],  
                [1.1733905],  
                [1.1727879],  
                [1.1506567]])
```

To plot the data between actual and predicted stock price we use `inverse_transform` function over `y_pred` data.

```
[16]: predicted_price = sc.inverse_transform(y_pred)
```

Plotting the actual and predicted prices for stocks

```
In [17]: plt.plot(y_test, color = 'red', label = 'Actual Stock Price')
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()
```

Output:



Conclusion:

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

References:

<https://www.simplelearn.com/tutorials/machine-learning-tutorial/stock-price-prediction-using-machine-learning>

Directed by
N. Sesa Kumar Sir

THE END