```
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
import math
from sklearn.metrics import mean_squared_error
```

```
df=pd.read_csv('/content/AAPL.csv')
```

```
df.head(5)
```

| | Unnamed: 0 | symbol | date | close | high | low | open | volume | adjC |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | AAPL | 2015-05-27 00:00:00+00:00 | 132.045 | 132.260 | 130.05 | 130.34 | 45833246 | 121.68 |
| **1** | 1 | AAPL | 2015-05-28 00:00:00+00:00 | 131.780 | 131.950 | 131.10 | 131.86 | 30733309 | 121.43 |
| **2** | 2 | AAPL | 2015-05-29 00:00:00+00:00 | 130.280 | 131.450 | 129.90 | 131.23 | 50884452 | 120.05 |
| **3** | 3 | AAPL | 2015-06-01 00:00:00+00:00 | 130.535 | 131.390 | 130.05 | 131.20 | 32112797 | 120.29 |
| **4** | 4 | AAPL | 2015-06-02 00:00:00+00:00 | 129.960 | 130.655 | 129.32 | 129.86 | 33667627 | 119.76 |

```
df.tail()
```

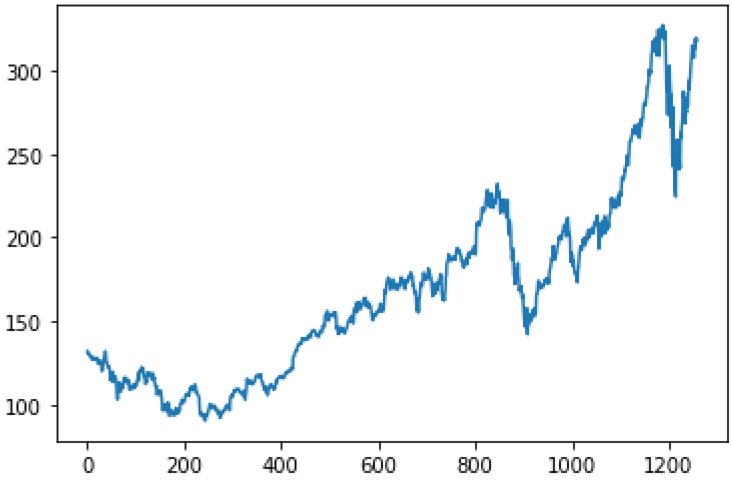
| | Unnamed: 0 | symbol | date | close | high | low | open | volume | ad |
|---|---|---|---|---|---|---|---|---|---|
| **1253** | 1253 | AAPL | 2020-05-18 00:00:00+00:00 | 314.96 | 316.50 | 310.3241 | 313.17 | 33843125 | |
| **1254** | 1254 | AAPL | 2020-05-19 00:00:00+00:00 | 313.14 | 318.52 | 313.0100 | 315.03 | 25432385 | |
| **1255** | 1255 | AAPL | 2020-05-20 00:00:00+00:00 | 319.23 | 319.52 | 316.2000 | 316.68 | 27876215 | |
| **1256** | 1256 | AAPL | 2020-05-21 00:00:00+00:00 | 316.85 | 320.89 | 315.8700 | 318.66 | 25672211 | |
| **1257** | 1257 | AAPL | 2020-05-22 00:00:00+00:00 | 318.89 | 319.23 | 315.3500 | 315.77 | 20450754 | |

```
df1=df.reset_index()['close']
df1
```

```
0       132.045
1       131.780
2       130.280
3       130.535
4       129.960
         ...
1253    314.960
1254    313.140
1255    319.230
1256    316.850
1257    318.890
Name: close, Length: 1258, dtype: float64
```

```
import matplotlib.pyplot as plt
plt.plot(df1)
```

```
[<matplotlib.lines.Line2D at 0x7f63606a1a50>]
```



```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
df1=scaler.fit_transform(np.array(df1).reshape(-1,1))
```

```
print(df1)
```

```
[[0.17607447]
 [0.17495567]
 [0.16862282]
 ...
 [0.96635143]
 [0.9563033 ]
 [0.96491598]]
```

```
training_size=int(len(df1)*0.65)
test_size=len(df1)-training_size
train_data,test_data=df1[0:training_size,:],df1[training_size:len(df1),:1]
```

```
training_size,test_size
```

```
(817, 441)
```

```
train_data
```

```
array([[0.17607447],
       [0.17495567],
       [0.16862282],
       [0.1696994 ],
       [0.16727181],
       [0.16794731],
       [0.16473866],
       [0.16174111],
       [0.1581525 ],
       [0.15654817],
       [0.16271215],
       [0.1614878 ],
       [0.1554927 ],
       [0.15443722],
       [0.15730811],
       [0.15604154],
       [0.15849025],
       [0.15308621],
       [0.15735033],
       [0.15490163],
       [0.15946129],
       [0.15688592],
       [0.1537195 ],
       [0.14434687],
       [0.14812547],
       [0.15308621],
       [0.15241071],
       [0.15055307],
       [0.14924428],
       [0.13607194],
       [0.12551718],
       [0.13906949],
       [0.14911762],
       [0.14890653],
       [0.15401503],
       [0.16115005],
       [0.16583636],
       [0.17618002],
       [0.17060711],
       [0.14725998],
       [0.14700667],
       [0.14422021],
       [0.13691632],
       [0.13949168],
       [0.13784514],
       [0.13522756],
       [0.13071012],
       [0.11863548],
       [0.10259225],
       [0.1058009 ],
       [0.10466098],
       [0.10630752],
       [0.12403952],
       [0.09773706],
       [0.10512539],
       [0.10474542],
       [0.10816516],
       [0.11323144]
```

```
def create_dataset(dataset, time_step=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-time_step-1):
        a = dataset[i:(i+time_step), 0]   ###i=0, 0,1,2,3-----99   100
        dataX.append(a)
        dataY.append(dataset[i + time_step, 0])
    return np.array(dataX), np.array(dataY)
```

```
time_step = 100
X_train, y_train = create_dataset(train_data, time_step)
X_test, ytest = create_dataset(test_data, time_step)
```

```
print(X_train.shape), print(y_train.shape)
```

```
(716, 100)
(716,)
(None, None)
```

```
print(X_test.shape), print(ytest.shape)
```

```
(340, 100)
(340,)
(None, None)
```

```
X_train =X_train.reshape(X_train.shape[0],X_train.shape[1] , 1)
X_test = X_test.reshape(X_test.shape[0],X_test.shape[1] , 1)
```

```
model=Sequential()
model.add(LSTM(50,return_sequences=True,input_shape=(100,1)))
model.add(LSTM(50,return_sequences=True))
model.add(LSTM(50))
model.add(Dense(1))
model.compile(loss='mean_squared_error',optimizer='adam')
```

```
model.summary()
```

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

 lstm_1 (LSTM)               (None, 100, 50)           20200

 lstm_2 (LSTM)               (None, 50)                20200

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 50,851
Trainable params: 50,851
```

```
    Non-trainable params: 0
    _________________________________________________________________
```

```
model.fit(X_train,y_train,validation_data=(X_test,ytest),epochs=100,batch_size=64,verbose=
```

```
    Epoch 1/100
    12/12 [==============================] - 11s 342ms/step - loss: 0.0213 - val_l
    Epoch 2/100
    12/12 [==============================] - 3s 264ms/step - loss: 0.0049 - val_lo
    Epoch 3/100
    12/12 [==============================] - 3s 237ms/step - loss: 0.0015 - val_lo
    Epoch 4/100
    12/12 [==============================] - 2s 209ms/step - loss: 8.5310e-04 - va
    Epoch 5/100
    12/12 [==============================] - 2s 207ms/step - loss: 7.7258e-04 - va
    Epoch 6/100
    12/12 [==============================] - 2s 207ms/step - loss: 6.4711e-04 - va
    Epoch 7/100
    12/12 [==============================] - 3s 261ms/step - loss: 5.9850e-04 - va
    Epoch 8/100
    12/12 [==============================] - 3s 267ms/step - loss: 5.9243e-04 - va
    Epoch 9/100
    12/12 [==============================] - 3s 248ms/step - loss: 6.0387e-04 - va
    Epoch 10/100
    12/12 [==============================] - 2s 206ms/step - loss: 6.6368e-04 - va
    Epoch 11/100
    12/12 [==============================] - 3s 250ms/step - loss: 6.4377e-04 - va
    Epoch 12/100
    12/12 [==============================] - 2s 204ms/step - loss: 6.0519e-04 - va
    Epoch 13/100
    12/12 [==============================] - 2s 206ms/step - loss: 5.5982e-04 - va
    Epoch 14/100
    12/12 [==============================] - 2s 204ms/step - loss: 5.5725e-04 - va
    Epoch 15/100
    12/12 [==============================] - 2s 206ms/step - loss: 5.5570e-04 - va
    Epoch 16/100
    12/12 [==============================] - 2s 206ms/step - loss: 5.4922e-04 - va
    Epoch 17/100
    12/12 [==============================] - 2s 206ms/step - loss: 5.4728e-04 - va
    Epoch 18/100
    12/12 [==============================] - 2s 206ms/step - loss: 5.4264e-04 - va
    Epoch 19/100
    12/12 [==============================] - 2s 205ms/step - loss: 5.2259e-04 - va
    Epoch 20/100
    12/12 [==============================] - 2s 205ms/step - loss: 5.2435e-04 - va
    Epoch 21/100
    12/12 [==============================] - 2s 204ms/step - loss: 5.1984e-04 - va
    Epoch 22/100
    12/12 [==============================] - 2s 204ms/step - loss: 4.9914e-04 - va
    Epoch 23/100
    12/12 [==============================] - 2s 203ms/step - loss: 5.0503e-04 - va
    Epoch 24/100
    12/12 [==============================] - 2s 208ms/step - loss: 5.2338e-04 - va
    Epoch 25/100
    12/12 [==============================] - 4s 309ms/step - loss: 4.8210e-04 - va
    Epoch 26/100
    12/12 [==============================] - 2s 206ms/step - loss: 4.8098e-04 - va
    Epoch 27/100
    12/12 [==============================] - 2s 206ms/step - loss: 4.8822e-04 - va
    Epoch 28/100
    12/12 [==============================] - 2s 204ms/step - loss: 4.6871e-04 - va
    Epoch 29/100
    12/12 [==============================] - 2s 206ms/step - loss: 4.7354e-04 - va
```

```
train_predict=model.predict(X_train)
test_predict=model.predict(X_test)
```

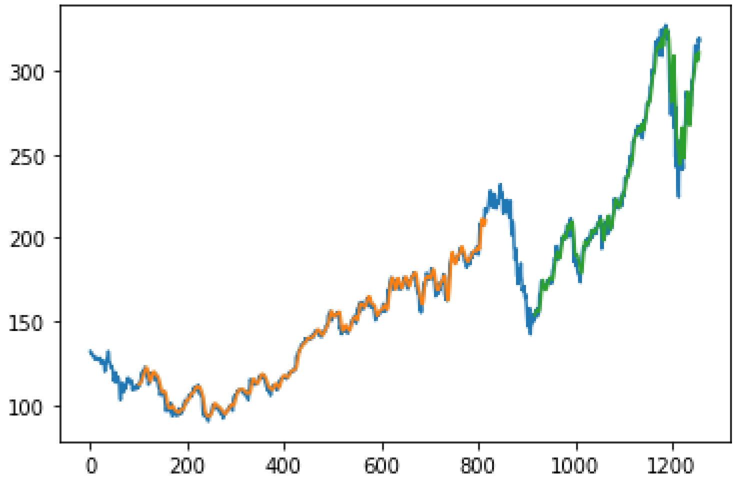
```
    23/23 [==============================] - 4s 48ms/step
    11/11 [==============================] - 1s 58ms/step
```

```
train_predict=scaler.inverse_transform(train_predict)
test_predict=scaler.inverse_transform(test_predict)
```

```
math.sqrt(mean_squared_error(y_train,train_predict))
```

```
    142.54191734152107
```

```
look_back=100
trainPredictPlot = np.empty_like(df1)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(train_predict)+look_back, :] = train_predict
# shift test predictions for plotting
testPredictPlot = np.empty_like(df1)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(train_predict)+(look_back*2)+1:len(df1)-1, :] = test_predict
# plot baseline and predictions
plt.plot(scaler.inverse_transform(df1))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
```



● ✕