Stock Market Prediction And Forecasting Using LSTM

In [177…

```
!pip install scikit-learn
```

```
Requirement already satisfied: scikit-learn in c:\python311\lib\site-packages
(1.3.0)
Requirement already satisfied: numpy>=1.17.3 in c:\python311\lib\site-packages
(from scikit-learn) (1.25.1)
Requirement already satisfied: scipy>=1.5.0 in c:\python311\lib\site-packages
(from scikit-learn) (1.11.2)
Requirement already satisfied: joblib>=1.1.1 in c:\python311\lib\site-packages
(from scikit-learn) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\python311\lib\site-pa
ckages (from scikit-learn) (3.2.0)
[notice] A new release of pip is available: 23.1.2 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [ ]:

```
pip install tensorflow
```

In [ ]:

```
pip install --upgrade tensorflow
```

In [126…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
import tensorflow
```

In [127…

```
df=pd.read_csv("C:\\Users\\ravip\\OneDrive\\Documents\\sem6\\EC460-deep learning
```

In [128…

```
df.head()
```

Out[128]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-12-13	42.622501	43.142502	42.387501	42.737499	41.019913	127594400
1	2018-12-14	42.250000	42.270000	41.320000	41.369999	39.707378	162814800
2	2018-12-17	41.362499	42.087502	40.682499	40.985001	39.337849	177151600
3	2018-12-18	41.345001	41.882500	41.097500	41.517502	39.848949	135366000
4	2018-12-19	41.500000	41.862499	39.772499	40.222500	38.605984	196189200

In [129…

```
df.tail()
```

Out[129]:

	Date	Open	High	Low	Close	Adj Close	Volume
1187	2023-09-01	189.490005	189.919998	188.279999	189.460007	189.210739	45732600
1188	2023-09-05	188.279999	189.979996	187.610001	189.699997	189.450409	45280000
1189	2023-09-06	188.399994	188.850006	181.470001	182.910004	182.669342	81755800
1190	2023-09-07	175.179993	178.210007	173.539993	177.559998	177.326385	112488800
1191	2023-09-08	178.350006	180.240005	177.789993	178.179993	177.945557	65551300

In [130…
```
df['Date'] = pd.to_datetime(df['Date'])
print(type(df['Date'][0]))
```

```
<class 'pandas._libs.tslibs.timestamps.Timestamp'>
```

In [131…
```
df1=df.reset_index()['Close']
```

In [132…
```
df1
```

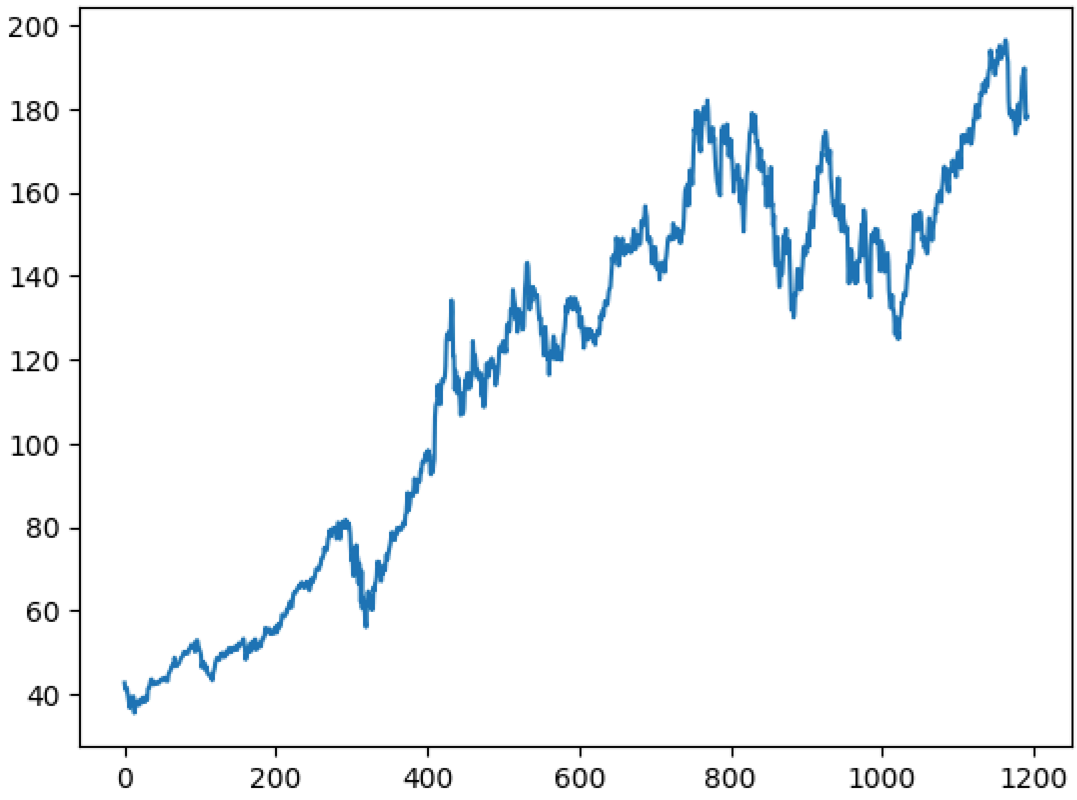
Out[132]:
```
0        42.737499
1        41.369999
2        40.985001
3        41.517502
4        40.222500
           ...
1187    189.460007
1188    189.699997
1189    182.910004
1190    177.559998
1191    178.179993
Name: Close, Length: 1192, dtype: float64
```

In [133…
```
df_final=df1
```

Dataset of stock prices from Apple of 1192 Days from 2018-12-13 to 2023-09-08

In [134…
```
plt.plot(df1)
```

Out[134]:
```
[<matplotlib.lines.Line2D at 0x1c68b67cfa0>]
```



In [ ]:
```
### LSTM are sensitive to the scale of the data. so we apply MinMax scaler
```

In [ ]:

In [135…
```
df1
```

Out[135]:
```
0        42.737499
1        41.369999
2        40.985001
3        41.517502
4        40.222500
           ...
1187    189.460007
1188    189.699997
1189    182.910004
1190    177.559998
1191    178.179993
Name: Close, Length: 1192, dtype: float64
```

In [136…
```
scaler=MinMaxScaler(feature_range=(0,1))
df1=scaler.fit_transform(np.array(df1).reshape(-1,1))
```

In [137…
```
print(df1)
```

```
[[0.04468543]
 [0.0361865 ]
 [0.03379376]
 ...
 [0.9158497 ]
 [0.88259971]
 [0.88645295]]
```

Splitting dataset into train and test data

In [138…
```
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]
```

In [139…
```
training_size,test_size
print(training_size,test_size)
print(training_size/(training_size+test_size))
print(test_size/(training_size+test_size))
```

```
774 418
0.6493288590604027
0.35067114093959734
```

In [140…
```
train_data
```

```
Out[140]: array([[0.04468543],
                 [0.0361865 ],
                 [0.03379376],
                 [0.03710322],
                 [0.02905486],
                 [0.02274669],
                 [0.01326889],
                 [0.00720933],
                 [0.02327496],
                 [0.02169014],
                 [0.02181444],
                 [0.02416059],
                 [0.02444026],
                 [0.        ],
                 [0.00943117],
                 [0.00891843],
                 [0.01329997],
                 [0.01727753],
                 [0.01803887],
                 [0.01569271],
                 [0.01213467],
                 [0.01690465],
                 [0.01981013],
                 [0.02123956],
                 [0.02273116],
                 [0.01726201],
                 [0.01822532],
                 [0.01632975],
                 [0.02419166],
                 [0.02192321],
                 [0.01940614],
                 [0.03582915],
                 [0.0376781 ],
                 [0.0378024 ],
                 [0.04515156],
                 [0.049704  ],
                 [0.04979724],
                 [0.04466991],
                 [0.04384643],
                 [0.04232375],
                 [0.04459222],
                 [0.04348905],
                 [0.04445239],
                 [0.04386196],
                 [0.04465435],
                 [0.04636348],
                 [0.04485635],
                 [0.04782399],
                 [0.04978169],
                 [0.04993707],
                 [0.05077608],
                 [0.04810365],
                 [0.05093146],
                 [0.05229876],
                 [0.05180155],
                 [0.05023228],
                 [0.04709373],
                 [0.04773077],
                 [0.05703763],
                 [0.06016066],
```

```
       [0.06140365],
       [0.06454218],
       [0.06825561],
       [0.07120772],
       [0.06889265],
       [0.07142524],
       [0.08219262],
       [0.07591554],
       [0.07232641],
       [0.06929661],
       [0.0719069 ],
       [0.07229533],
       [0.07420642],
       [0.07621075],
       [0.08053014],
       [0.08259661],
       [0.08312488],
       [0.08516026],
       [0.08997686],
       [0.0890446 ],
       [0.09078478],
       [0.08819004],
       [0.08806574],
       [0.08862509],
       [0.08865617],
       [0.09468467],
       [0.09581889],
       [0.0968599 ],
       [0.10144341],
       [0.10094623],
       [0.0980252 ],
       [0.09650254],
       [0.09698419],
       [0.09086247],
       [0.10616678],
       [0.10403814],
       [0.10807787],
       [0.10299715],
       [0.09426516],
       [0.09432729],
       [0.09094016],
       [0.08543992],
       [0.06763412],
       [0.07220211],
       [0.07571355],
       [0.07440841],
       [0.07273038],
       [0.06354779],
       [0.06900142],
       [0.06306614],
       [0.05821849],
       [0.0571464 ],
       [0.05599663],
       [0.05467597],
       [0.05610541],
       [0.05108685],
       [0.04833673],
       [0.05818741],
       [0.06269323],
       [0.06685725],
```

```
       [0.07451716],
       [0.07829275],
       [0.08175758],
       [0.08079427],
       [0.0807321 ],
       [0.07854135],
       [0.08032814],
       [0.08741317],
       [0.08651201],
       [0.08898247],
       [0.08792591],
       [0.08761517],
       [0.08293843],
       [0.08951073],
       [0.08941751],
       [0.0865897 ],
       [0.09222977],
       [0.09406316],
       [0.09667345],
       [0.09639377],
       [0.08985255],
       [0.09174811],
       [0.09484003],
       [0.09254051],
       [0.09494881],
       [0.09791645],
       [0.09681328],
       [0.0950265 ],
       [0.09861562],
       [0.09384564],
       [0.10103945],
       [0.10355649],
       [0.10329236],
       [0.1007287 ],
       [0.10184739],
       [0.10486162],
       [0.10346327],
       [0.11008218],
       [0.10291945],
       [0.0960675 ],
       [0.07947358],
       [0.08516026],
       [0.08832987],
       [0.09515077],
       [0.09135968],
       [0.09056726],
       [0.10375848],
       [0.09409425],
       [0.09252498],
       [0.09992076],
       [0.10590265],
       [0.10591818],
       [0.10946069],
       [0.10918103],
       [0.09392333],
       [0.09990522],
       [0.09628502],
       [0.09841363],
       [0.10382063],
       [0.10340113],
```

```
       [0.09867776],
       [0.10410031],
       [0.11045509],
       [0.110424  ],
       [0.11183791],
       [0.11576886],
       [0.1264741 ],
       [0.12569723],
       [0.11895402],
       [0.12074081],
       [0.1219838 ],
       [0.12520005],
       [0.12238779],
       [0.1173692 ],
       [0.11890741],
       [0.11729151],
       [0.12249654],
       [0.12072528],
       [0.1190628 ],
       [0.12706452],
       [0.12802783],
       [0.11928032],
       [0.12217027],
       [0.13178787],
       [0.13186556],
       [0.12773262],
       [0.13181896],
       [0.13657338],
       [0.14608226],
       [0.14555398],
       [0.14469944],
       [0.14322337],
       [0.14463728],
       [0.14639301],
       [0.15276331],
       [0.15190877],
       [0.15691178],
       [0.15753329],
       [0.16219449],
       [0.16603223],
       [0.15708269],
       [0.15703608],
       [0.16558163],
       [0.17655103],
       [0.17916129],
       [0.17858642],
       [0.17875731],
       [0.18215999],
       [0.18326318],
       [0.18646387],
       [0.18609094],
       [0.18999083],
       [0.18714752],
       [0.19199516],
       [0.19407717],
       [0.19281864],
       [0.18800206],
       [0.18616865],
       [0.18581128],
       [0.19294292],
```

```
       [0.18971117],
       [0.19522692],
       [0.19431022],
       [0.18950918],
       [0.1821911 ],
       [0.18574912],
       [0.19171546],
       [0.19968613],
       [0.1937975 ],
       [0.19622133],
       [0.19977935],
       [0.20085143],
       [0.20658472],
       [0.2139028 ],
       [0.21475739],
       [0.21371637],
       [0.21415141],
       [0.21325027],
       [0.2203353 ],
       [0.22075479],
       [0.22951788],
       [0.22934695],
       [0.23201937],
       [0.23532884],
       [0.24573889],
       [0.24120196],
       [0.24488431],
       [0.24269358],
       [0.2501515 ],
       [0.26015756],
       [0.26124515],
       [0.27154642],
       [0.26489643],
       [0.26281443],
       [0.268874  ],
       [0.27429657],
       [0.27094049],
       [0.27269622],
       [0.27507343],
       [0.27364397],
       [0.25910103],
       [0.27268067],
       [0.283013  ],
       [0.28228274],
       [0.25997111],
       [0.25865043],
       [0.27448301],
       [0.27852273],
       [0.28436474],
       [0.2763164 ],
       [0.27867806],
       [0.27566381],
       [0.28745671],
       [0.28383648],
       [0.2839608 ],
       [0.27471606],
       [0.28189431],
       [0.27673589],
       [0.26547131],
       [0.24236726],
```

```
       [0.22667452],
       [0.2337751 ],
       [0.20405212],
       [0.20380352],
       [0.24334612],
       [0.22860118],
       [0.2494523 ],
       [0.23419464],
       [0.22815059],
       [0.1926322 ],
       [0.2224173 ],
       [0.20701977],
       [0.16475815],
       [0.21096627],
       [0.15540468],
       [0.17195196],
       [0.16233433],
       [0.15939777],
       [0.13525272],
       [0.12768601],
       [0.16266062],
       [0.16054754],
       [0.18062181],
       [0.16399683],
       [0.17498174],
       [0.17417379],
       [0.15338482],
       [0.15963082],
       [0.15416169],
       [0.18688336],
       [0.18215999],
       [0.19247682],
       [0.19545997],
       [0.20363263],
       [0.22507417],
       [0.22100339],
       [0.22451485],
       [0.21847079],
       [0.20935037],
       [0.1960504 ],
       [0.20806079],
       [0.20639828],
       [0.21873495],
       [0.21904571],
       [0.21191403],
       [0.22613075],
       [0.23556189],
       [0.22821275],
       [0.23456752],
       [0.24140395],
       [0.24617393],
       [0.25100603],
       [0.26093442],
       [0.26851666],
       [0.26292321],
       [0.25708114],
       [0.26001773],
       [0.25717436],
       [0.26843895],
       [0.26561119],
```

```
       [0.27507343],
       [0.27137554],
       [0.27454517],
       [0.27118909],
       [0.27333321],
       [0.27355075],
       [0.2730691 ],
       [0.27914422],
       [0.28145926],
       [0.28422491],
       [0.27987447],
       [0.29413776],
       [0.29718307],
       [0.31354391],
       [0.32729448],
       [0.30097418],
       [0.30548001],
       [0.31199017],
       [0.32611362],
       [0.32535231],
       [0.32556985],
       [0.32244683],
       [0.3366635 ],
       [0.34856513],
       [0.33851245],
       [0.34593931],
       [0.32852194],
       [0.34118488],
       [0.34587715],
       [0.34480506],
       [0.34480506],
       [0.35993849],
       [0.35813615],
       [0.37162256],
       [0.37417071],
       [0.37521169],
       [0.37246159],
       [0.38228122],
       [0.38642966],
       [0.3789562 ],
       [0.37774428],
       [0.39036061],
       [0.38192384],
       [0.38361741],
       [0.35610075],
       [0.3546713 ],
       [0.3683131 ],
       [0.35863335],
       [0.36974256],
       [0.37688975],
       [0.43947423],
       [0.45611473],
       [0.46063611],
       [0.46310654],
       [0.48697191],
       [0.46963225],
       [0.47966938],
       [0.45883377],
       [0.4814251 ],
       [0.49385499],
```

```
       [0.49321795],
       [0.49135345],
       [0.49728874],
       [0.49818989],
       [0.51414678],
       [0.55202687],
       [0.56127157],
       [0.55485464],
       [0.56540452],
       [0.55600443],
       [0.55474591],
       [0.5810506 ],
       [0.61299541],
       [0.59571788],
       [0.53033668],
       [0.53083389],
       [0.48024425],
       [0.5082115 ],
       [0.48440825],
       [0.475148  ],
       [0.49603022],
       [0.49714891],
       [0.47595592],
       [0.46483117],
       [0.44307886],
       [0.46321532],
       [0.47396715],
       [0.44481909],
       [0.45165552],
       [0.47688818],
       [0.49354423],
       [0.48813721],
       [0.49882692],
       [0.50491759],
       [0.48148722],
       [0.50311525],
       [0.48235736],
       [0.49429004],
       [0.49360639],
       [0.50603628],
       [0.55221332],
       [0.53170398],
       [0.53226335],
       [0.52928015],
       [0.51877689],
       [0.49988349],
       [0.50939235],
       [0.50541479],
       [0.49845404],
       [0.49404143],
       [0.4941036 ],
       [0.50373673],
       [0.47017603],
       [0.49578161],
       [0.45563308],
       [0.45507371],
       [0.4654527 ],
       [0.49348207],
       [0.51883905],
       [0.51672599],
```

```
       [0.50199656],
       [0.49982133],
       [0.52169792],
       [0.51995774],
       [0.5202685 ],
       [0.52673205],
       [0.52107643],
       [0.5126241 ],
       [0.51641522],
       [0.50833577],
       [0.48664563],
       [0.49484936],
       [0.50019422],
       [0.50367457],
       [0.51896337],
       [0.5417722 ],
       [0.54400959],
       [0.5431395 ],
       [0.53885117],
       [0.54817359],
       [0.55208899],
       [0.53593014],
       [0.54500396],
       [0.53984559],
       [0.53593014],
       [0.57384129],
       [0.57340625],
       [0.57893754],
       [0.5662591 ],
       [0.57601651],
       [0.59870112],
       [0.59298338],
       [0.59926044],
       [0.62859498],
       [0.61728374],
       [0.61013659],
       [0.6037352 ],
       [0.5833502 ],
       [0.59329405],
       [0.56588617],
       [0.59273473],
       [0.59975764],
       [0.58067772],
       [0.57955908],
       [0.59254828],
       [0.58024273],
       [0.56924224],
       [0.57353057],
       [0.59963332],
       [0.62971362],
       [0.64338658],
       [0.66731405],
       [0.66880568],
       [0.6619692 ],
       [0.63108092],
       [0.59919832],
       [0.61274685],
       [0.61802959],
       [0.61150388],
       [0.63294542],
```

```
       [0.62902998],
       [0.62996228],
       [0.62436877],
       [0.62051553],
       [0.61889968],
       [0.62039121],
       [0.60684267],
       [0.59223752],
       [0.5852147 ],
       [0.58620902],
       [0.56215721],
       [0.56128713],
       [0.55811749],
       [0.53102033],
       [0.53269839],
       [0.57328197],
       [0.55668808],
       [0.53767032],
       [0.52567547],
       [0.53369276],
       [0.50224516],
       [0.53164181],
       [0.52474327],
       [0.53704883],
       [0.53126894],
       [0.54966516],
       [0.55948479],
       [0.5544507 ],
       [0.52816146],
       [0.52480539],
       [0.5459362 ],
       [0.54065351],
       [0.52542687],
       [0.52853434],
       [0.53238763],
       [0.53350632],
       [0.52424607],
       [0.53822969],
       [0.54351238],
       [0.56153573],
       [0.56346235],
       [0.57396562],
       [0.58925438],
       [0.60566182],
       [0.59472355],
       [0.61454915],
       [0.59963332],
       [0.61498424],
       [0.61287118],
       [0.61709729],
       [0.60634547],
       [0.60876929],
       [0.59907399],
       [0.61386559],
       [0.61635153],
       [0.61430059],
       [0.6092665 ],
       [0.60864497],
       [0.59609085],
       [0.6028029 ],
```

```
       [0.57365485],
       [0.57520863],
       [0.58540114],
       [0.58832217],
       [0.5674399 ],
       [0.56159789],
       [0.54208293],
       [0.55575583],
       [0.57116886],
       [0.56383523],
       [0.55501002],
       [0.55401565],
       [0.57029878],
       [0.5586147 ],
       [0.56899364],
       [0.56775068],
       [0.5674399 ],
       [0.55768245],
       [0.55351845],
       [0.5514675 ],
       [0.55631515],
       [0.54686846],
       [0.56147356],
       [0.56153573],
       [0.56675626],
       [0.56918008],
       [0.56284086],
       [0.57054738],
       [0.59000014],
       [0.5847796 ],
       [0.5879492 ],
       [0.5981417 ],
       [0.58987591],
       [0.60131138],
       [0.61175244],
       [0.61001226],
       [0.60820997],
       [0.60634547],
       [0.61672442],
       [0.6263576 ],
       [0.63027304],
       [0.63219966],
       [0.64891788],
       [0.66172064],
       [0.67756877],
       [0.66930288],
       [0.6809248 ],
       [0.67713368],
       [0.68421871],
       [0.70603313],
       [0.70186913],
       [0.68887992],
       [0.66439302],
       [0.6873883 ],
       [0.68272709],
       [0.69142807],
       [0.70236634],
       [0.70503881],
       [0.69124163],
       [0.68011683],
```

```
[0.68421871],
[0.68558601],
[0.68347295],
[0.69490842],
[0.69236027],
[0.69304392],
[0.68732618],
[0.68701541],
[0.68397015],
[0.68558601],
[0.70441728],
[0.70572246],
[0.71827658],
[0.71249672],
[0.68869348],
[0.69080654],
[0.70006683],
[0.70951358],
[0.70895416],
[0.70112337],
[0.69602706],
[0.70261499],
[0.73070646],
[0.72268923],
[0.72691535],
[0.73400038],
[0.73804015],
[0.75289386],
[0.74307424],
[0.73661074],
[0.70491448],
[0.70851916],
[0.69963174],
[0.70528737],
[0.70379574],
[0.68682898],
[0.66743838],
[0.67048364],
[0.68552389],
[0.69161451],
[0.69217383],
[0.68254065],
[0.661037  ],
[0.66675473],
[0.65848885],
[0.66563599],
[0.64382157],
[0.65606502],
[0.66159632],
[0.66961355],
[0.66718973],
[0.66663041],
[0.65855096],
[0.65482205],
[0.67253459],
[0.67924673],
[0.68987433],
[0.70360931],
[0.70671678],
[0.70808407],
```

```
       [0.70317431],
       [0.70286354],
       [0.70708975],
       [0.70416872],
       [0.72728832],
       [0.7100729 ],
       [0.70485237],
       [0.71144019],
       [0.72057617],
       [0.71728226],
       [0.71927099],
       [0.71405046],
       [0.71634996],
       [0.69838877],
       [0.69807801],
       [0.71125375],
       [0.71131587],
       [0.71753081],
       [0.73300606],
       [0.76022745],
       [0.77688355],
       [0.77980458],
       [0.78222841],
       [0.78552231],
       [0.75363963],
       [0.77495693],
       [0.80640453],
       [0.80311062],
       [0.79683347],
       [0.78490078],
       [0.80652886],
       [0.84294834],
       [0.86718668],
       [0.86395488],
       [0.89434595],
       [0.87128856],
       [0.86252547],
       [0.89341375],
       [0.84966049],
       [0.84269978],
       [0.83406101],
       [0.85419746],
       [0.87066703],
       [0.87464459],
       [0.89981513],
       [0.89335154],
       [0.89391096],
       [0.88657727],
       [0.88266192],
       [0.9102562 ],
       [0.89589969],
       [0.86619226],
       [0.84804464],
       [0.84910117]])
```

In [141…

```
import numpy
# convert an array of values into a dataset matrix
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 numpy.array(dataX), numpy.array(dataY)
```

In [142…

```
# reshape into X=t,t+1,t+2,t+3 and Y=t+4
time_step = 100
X_train, y_train = create_dataset(train_data, time_step)
X_test, ytest = create_dataset(test_data, time_step)
```

In [143…

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

```
(673, 100)
(673,)
```

In [144…

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

```
(317, 100)
(317,)
```

Out[144]: (None, None)

In [145…

```
# reshape input to be [samples, time steps, features] which is required for LSTM
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)
```

Building the LSTM Model

In [146…

```
import tensorflow
from tensorflow.keras import layers
```

In [147…

```
### Create the Stacked LSTM model
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.layers import LSTM
```

In [148…

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, Input

model = Sequential()
model.add(Input(shape=(100, 1)))# Input layer specifying input shape
model.add(LSTM(50, return_sequences=True))
#model.add(layers.BatchNormalization())
model.add(LSTM(50, return_sequences=True))
#model.add(layers.BatchNormalization())
model.add(LSTM(50))
#model.add(layers.BatchNormalization())
model.add(Dense(1))

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

In [149…

```
model.summary()
```

Model: "sequential_7"

Layer (type)	Output Shape
lstm_17 (LSTM)	(None, 100, 50)
lstm_18 (LSTM)	(None, 100, 50)
lstm_19 (LSTM)	(None, 50)
dense_5 (Dense)	(None, 1)

Total params: 50,851 (198.64 KB)

Trainable params: 50,851 (198.64 KB)

Training and testing the model

In [150…

```
model.fit(X_train,y_train,validation_data=(X_test,ytest),epochs=10,batch_size=64
```

```
Epoch 1/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 10s 308ms/step - loss: 0.0711 - val_loss: 0.0456
Epoch 2/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 209ms/step - loss: 0.0114 - val_loss: 0.0051
Epoch 3/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 215ms/step - loss: 0.0050 - val_loss: 0.0051
Epoch 4/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 217ms/step - loss: 0.0018 - val_loss: 0.0034
Epoch 5/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 216ms/step - loss: 0.0016 - val_loss: 0.0025
Epoch 6/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 210ms/step - loss: 0.0013 - val_loss: 0.0025
Epoch 7/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 212ms/step - loss: 0.0013 - val_loss: 0.0025
Epoch 8/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 218ms/step - loss: 0.0013 - val_loss: 0.0024
Epoch 9/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 215ms/step - loss: 0.0014 - val_loss: 0.0024
Epoch 10/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 214ms/step - loss: 0.0011 - val_loss: 0.0023
```

Out[150]: <keras.src.callbacks.history.History at 0x1c68b71efe0>

In [151…

```
import tensorflow as tf
```

In [152…

```
tf.__version__
```

Out[152]: '2.16.1'

In [153…

```
### Lets Do the prediction and check performance metrics
train_predict=model.predict(X_train)
test_predict=model.predict(X_test)
```

```
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 83ms/step
10/10 ━━━━━━━━━━━━━━━━━━━━ 1s 51ms/step
```

In [154…

```
##Transformback to original form
train_predict=scaler.inverse_transform(train_predict)
```

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

In [155…

```
### Calculate RMSE performance metrics
import math
from sklearn.metrics import mean_squared_error,mean_absolute_error
math.sqrt(mean_squared_error(y_train,train_predict))
```

Out[155]: 108.68561425762088

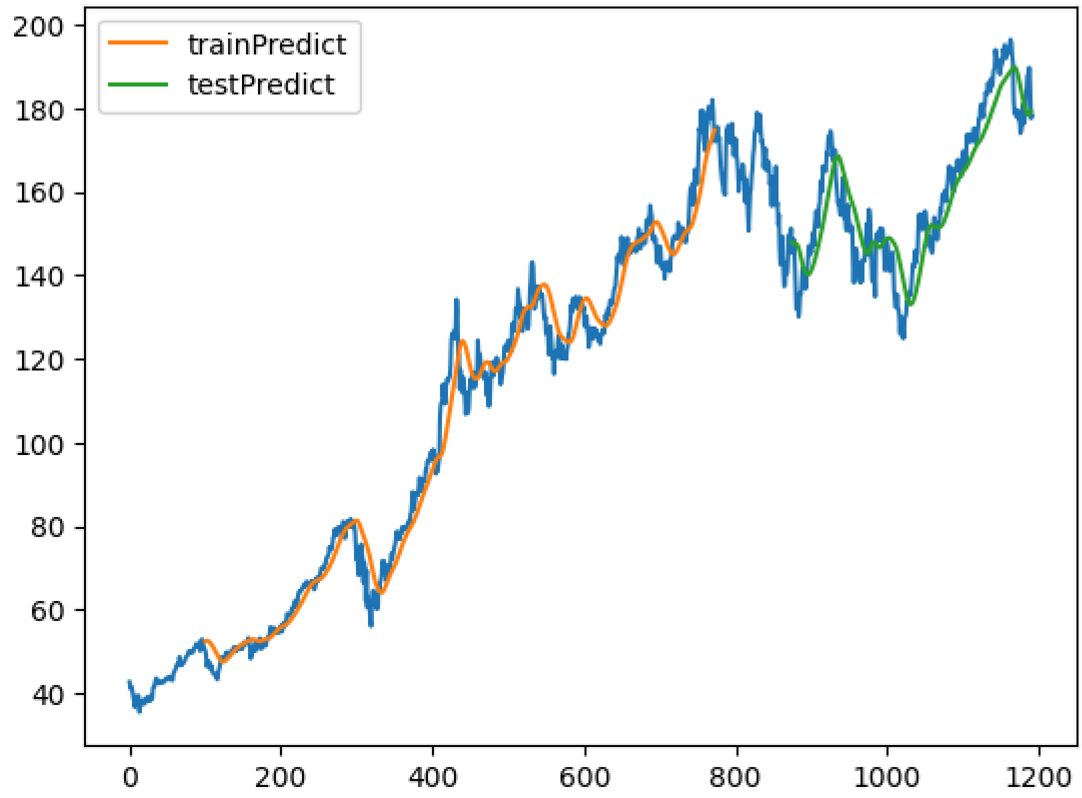
In [156…

```
### Test Data RMSE
math.sqrt(mean_squared_error(ytest,test_predict))
```

Out[156]: 158.31699144942678

In [157…

```
### Plotting
# shift train predictions for plotting
look_back=100
trainPredictPlot = numpy.empty_like(df1)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(train_predict)+look_back, :] = train_predict
# shift test predictions for plotting
testPredictPlot = numpy.empty_like(df1)
testPredictPlot[:, :] = numpy.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,label='trainPredict')
plt.plot(testPredictPlot,label='testPredict')
plt.legend()
plt.show()
```



In [158…

```
len(test_data)
```

Out[158]: 418

Predicting the stock price for nxt 30 days

In [159…
```
x_input=test_data[341:].reshape(1,-1)
x_input.shape
```

Out[159]: (1, 77)

In [160…
```
temp_input=list(x_input)
temp_input=temp_input[0].tolist()
```

In [161…
```
temp_input
```

```
Out[161]: [0.8676838860225016,
 0.8617174962904244,
 0.8453100503798276,
 0.847050222266285,
 0.8541974637857701,
 0.8693618525345934,
 0.8809838599396247,
 0.8806730940954455,
 0.8983235101585996,
 0.9036683682023181,
 0.8951539260149204,
 0.8928544278144699,
 0.8842156556726133,
 0.9013067516367181,
 0.9037305797916275,
 0.9213187842654724,
 0.9183356422264577,
 0.9223132001631598,
 0.9351159723463831,
 0.9283417020454425,
 0.9289010283594359,
 0.9223754117524692,
 0.9412687979681809,
 0.9392799723877499,
 0.930516969730538,
 0.947856626164457,
 0.9552524219388121,
 0.9573654780345984,
 0.9845869637721469,
 0.9752024356415205,
 0.9681795178615504,
 0.971162666115509,
 0.9641397483355387,
 0.9512748640021098,
 0.9479809499039717,
 0.9584842176718007,
 0.9632696561773662,
 0.964201953709904,
 0.9847112875116619,
 0.9830953461405596,
 0.9916098815521173,
 0.9793664356828873,
 0.971970633693588,
 0.9770047258931274,
 0.9824116960870515,
 0.987880877870285,
 0.9799257557819365,
 0.9961467658028129,
 1.0000000000000002,
 0.9947794719107406,
 0.9759481978452345,
 0.9671851019638626,
 0.9101319596682951,
 0.890617041764225,
 0.8965212199069927,
 0.8865151538730638,
 0.8851478599809914,
 0.8840291203437891,
 0.8944081638112067,
 0.8819160642480028,
```

```
 0.8764469756889293,
 0.8604745261378668,
 0.8635198797661909,
 0.8719099982140739,
 0.8805487703559305,
 0.9047248962502112,
 0.8752661239015214,
 0.8891254241326376,
 0.8989450418469582,
 0.9233697282110529,
 0.9453084742700326,
 0.946675768162105,
 0.9565576036806789,
 0.9580491280881063,
 0.9158497019213425,
 0.8825997143015112,
 0.8864529484986983]
```

In [ ]:

In [162…

```
from numpy import array

lst_output = []
n_steps = 100
i = 0
while i < 30:

    if len(temp_input) > 100:
        x_input = array(temp_input[1:])
        print("{} day input {}".format(i, x_input))
        x_input = x_input.reshape((1, n_steps, 1))
        yhat = model.predict(x_input, verbose=0)
        print("{} day output {}".format(i, yhat))
        temp_input.extend(yhat[0].tolist())
        temp_input = temp_input[1:]
        lst_output.extend(yhat.tolist())
        i += 1
    else:
        x_input = array(temp_input)
        x_input = x_input.reshape((1, len(temp_input), -1))
        yhat = model.predict(x_input, verbose=0)
        print(yhat[0])
        temp_input.extend(yhat[0].tolist())
        print(len(temp_input))
        lst_output.extend(yhat.tolist())
        i += 1

print(lst_output)
```

```
[0.8972957]
78
[0.8987118]
79
[0.8996778]
80
[0.9002356]
81
[0.9004485]
82
[0.90038544]
83
[0.90011233]
84
[0.89968675]
85
[0.8991567]
86
[0.89855903]
87
[0.89792156]
88
[0.89726454]
89
[0.8966013]
90
[0.8959409]
91
[0.89528847]
92
[0.89464676]
93
[0.89401674]
94
[0.8933989]
95
[0.8927927]
96
[0.89219743]
97
[0.89161193]
98
[0.8910359]
99
[0.8904683]
100
[0.8899089]
101
24 day input [0.8617175  0.84531005 0.84705022 0.85419746 0.86936185 0.88098386
 0.88067309 0.89832351 0.90366837 0.89515393 0.89285443 0.88421566
 0.90130675 0.90373058 0.92131878 0.91833564 0.9223132  0.93511597
 0.9283417  0.92890103 0.92237541 0.9412688  0.93927997 0.93051697
 0.94785663 0.95525242 0.95736548 0.98458696 0.97520244 0.96817952
 0.97116267 0.96413975 0.95127486 0.94798095 0.95848422 0.96326966
 0.96420195 0.98471129 0.98309535 0.99160988 0.97936644 0.97197063
 0.97700473 0.9824117  0.98788088 0.97992576 0.99614677 1.
 0.99477947 0.9759482  0.9671851  0.91013196 0.89061704 0.89652122
 0.88651515 0.88514786 0.88402912 0.89440816 0.88191606 0.87644698
 0.86047453 0.86351988 0.87191    0.88054877 0.9047249  0.87526612
 0.88912542 0.89894504 0.92336973 0.94530847 0.94667577 0.9565576
```

```
 0.95804913 0.9158497  0.88259971 0.88645295 0.89729571 0.8987118
 0.89967781 0.90023559 0.9004485  0.90038544 0.90011233 0.89968675
 0.89915669 0.89855903 0.89792156 0.89726454 0.89660132 0.8959409
 0.89528847 0.89464676 0.89401674 0.89339888 0.8927927  0.89219743
 0.89161193 0.89103591 0.8904683  0.88990891]
24 day output [[0.8893566]]
25 day input [0.84531005 0.84705022 0.85419746 0.86936185 0.88098386 0.88067309
 0.89832351 0.90366837 0.89515393 0.89285443 0.88421566 0.90130675
 0.90373058 0.92131878 0.91833564 0.9223132  0.93511597 0.9283417
 0.92890103 0.92237541 0.9412688  0.93927997 0.93051697 0.94785663
 0.95525242 0.95736548 0.98458696 0.97520244 0.96817952 0.97116267
 0.96413975 0.95127486 0.94798095 0.95848422 0.96326966 0.96420195
 0.98471129 0.98309535 0.99160988 0.97936644 0.97197063 0.97700473
 0.9824117  0.98788088 0.97992576 0.99614677 1.         0.99477947
 0.9759482  0.9671851  0.91013196 0.89061704 0.89652122 0.88651515
 0.88514786 0.88402912 0.89440816 0.88191606 0.87644698 0.86047453
 0.86351988 0.87191    0.88054877 0.9047249  0.87526612 0.88912542
 0.89894504 0.92336973 0.94530847 0.94667577 0.9565576  0.95804913
 0.9158497  0.88259971 0.88645295 0.89729571 0.8987118  0.89967781
 0.90023559 0.9004485  0.90038544 0.90011233 0.89968675 0.89915669
 0.89855903 0.89792156 0.89726454 0.89660132 0.8959409  0.89528847
 0.89464676 0.89401674 0.89339888 0.8927927  0.89219743 0.89161193
 0.89103591 0.8904683  0.88990891 0.88935661]
25 day output [[0.8888115]]
26 day input [0.84705022 0.85419746 0.86936185 0.88098386 0.88067309 0.89832351
 0.90366837 0.89515393 0.89285443 0.88421566 0.90130675 0.90373058
 0.92131878 0.91833564 0.9223132  0.93511597 0.9283417  0.92890103
 0.92237541 0.9412688  0.93927997 0.93051697 0.94785663 0.95525242
 0.95736548 0.98458696 0.97520244 0.96817952 0.97116267 0.96413975
 0.95127486 0.94798095 0.95848422 0.96326966 0.96420195 0.98471129
 0.98309535 0.99160988 0.97936644 0.97197063 0.97700473 0.9824117
 0.98788088 0.97992576 0.99614677 1.         0.99477947 0.9759482
 0.9671851  0.91013196 0.89061704 0.89652122 0.88651515 0.88514786
 0.88402912 0.89440816 0.88191606 0.87644698 0.86047453 0.86351988
 0.87191    0.88054877 0.9047249  0.87526612 0.88912542 0.89894504
 0.92336973 0.94530847 0.94667577 0.9565576  0.95804913 0.9158497
 0.88259971 0.88645295 0.89729571 0.8987118  0.89967781 0.90023559
 0.9004485  0.90038544 0.90011233 0.89968675 0.89915669 0.89855903
 0.89792156 0.89726454 0.89660132 0.8959409  0.89528847 0.89464676
 0.89401674 0.89339888 0.8927927  0.89219743 0.89161193 0.89103591
 0.8904683  0.88990891 0.88935661 0.88881153]
26 day output [[0.88827324]]
27 day input [0.85419746 0.86936185 0.88098386 0.88067309 0.89832351 0.90366837
 0.89515393 0.89285443 0.88421566 0.90130675 0.90373058 0.92131878
 0.91833564 0.9223132  0.93511597 0.9283417  0.92890103 0.92237541
 0.9412688  0.93927997 0.93051697 0.94785663 0.95525242 0.95736548
 0.98458696 0.97520244 0.96817952 0.97116267 0.96413975 0.95127486
 0.94798095 0.95848422 0.96326966 0.96420195 0.98471129 0.98309535
 0.99160988 0.97936644 0.97197063 0.97700473 0.9824117  0.98788088
 0.97992576 0.99614677 1.         0.99477947 0.9759482  0.9671851
 0.91013196 0.89061704 0.89652122 0.88651515 0.88514786 0.88402912
 0.89440816 0.88191606 0.87644698 0.86047453 0.86351988 0.87191
 0.88054877 0.9047249  0.87526612 0.88912542 0.89894504 0.92336973
 0.94530847 0.94667577 0.9565576  0.95804913 0.9158497  0.88259971
 0.88645295 0.89729571 0.8987118  0.89967781 0.90023559 0.9004485
 0.90038544 0.90011233 0.89968675 0.89915669 0.89855903 0.89792156
 0.89726454 0.89660132 0.8959409  0.89528847 0.89464676 0.89401674
 0.89339888 0.8927927  0.89219743 0.89161193 0.89103591 0.8904683
 0.88990891 0.88935661 0.88881153 0.88827324]
27 day output [[0.88774115]]
```

```
28 day input [0.86936185 0.88098386 0.88067309 0.89832351 0.90366837 0.89515393
 0.89285443 0.88421566 0.90130675 0.90373058 0.92131878 0.91833564
 0.9223132  0.93511597 0.9283417  0.92890103 0.92237541 0.9412688
 0.93927997 0.93051697 0.94785663 0.95525242 0.95736548 0.98458696
 0.97520244 0.96817952 0.97116267 0.96413975 0.95127486 0.94798095
 0.95848422 0.96326966 0.96420195 0.98471129 0.98309535 0.99160988
 0.97936644 0.97197063 0.97700473 0.9824117  0.98788088 0.97992576
 0.99614677 1.         0.99477947 0.9759482  0.9671851  0.91013196
 0.89061704 0.89652122 0.88651515 0.88514786 0.88402912 0.89440816
 0.88191606 0.87644698 0.86047453 0.86351988 0.87191    0.88054877
 0.9047249  0.87526612 0.88912542 0.89894504 0.92336973 0.94530847
 0.94667577 0.9565576  0.95804913 0.9158497  0.88259971 0.88645295
 0.89729571 0.8987118  0.89967781 0.90023559 0.9004485  0.90038544
 0.90011233 0.89968675 0.89915669 0.89855903 0.89792156 0.89726454
 0.89660132 0.8959409  0.89528847 0.89464676 0.89401674 0.89339888
 0.8927927  0.89219743 0.89161193 0.89103591 0.8904683  0.88990891
 0.88935661 0.88881153 0.88827324 0.88774115]
28 day output [[0.88721585]]
29 day input [0.88098386 0.88067309 0.89832351 0.90366837 0.89515393 0.89285443
 0.88421566 0.90130675 0.90373058 0.92131878 0.91833564 0.9223132
 0.93511597 0.9283417  0.92890103 0.92237541 0.9412688  0.93927997
 0.93051697 0.94785663 0.95525242 0.95736548 0.98458696 0.97520244
 0.96817952 0.97116267 0.96413975 0.95127486 0.94798095 0.95848422
 0.96326966 0.96420195 0.98471129 0.98309535 0.99160988 0.97936644
 0.97197063 0.97700473 0.9824117  0.98788088 0.97992576 0.99614677
 1.         0.99477947 0.9759482  0.9671851  0.91013196 0.89061704
 0.89652122 0.88651515 0.88514786 0.88402912 0.89440816 0.88191606
 0.87644698 0.86047453 0.86351988 0.87191    0.88054877 0.9047249
 0.87526612 0.88912542 0.89894504 0.92336973 0.94530847 0.94667577
 0.9565576  0.95804913 0.9158497  0.88259971 0.88645295 0.89729571
 0.8987118  0.89967781 0.90023559 0.9004485  0.90038544 0.90011233
 0.89968675 0.89915669 0.89855903 0.89792156 0.89726454 0.89660132
 0.8959409  0.89528847 0.89464676 0.89401674 0.89339888 0.8927927
 0.89219743 0.89161193 0.89103591 0.8904683  0.88990891 0.88935661
 0.88881153 0.88827324 0.88774115 0.88721585]
29 day output [[0.8866964]]
[[0.8972957134246826], [0.8987118005752563], [0.8996778130531311], [0.900235593
3189392], [0.9004485011100769], [0.9003854393959045], [0.9001123309135437], [0.
8996867537498474], [0.8991566896438599], [0.898559033870697], [0.89792156219482
42], [0.8972645401954651], [0.8966013193130493], [0.895940899848938], [0.895288
4674072266], [0.8946467638015747], [0.8940167427062988], [0.8933988809585571],
[0.8927927017211914], [0.8921974301338196], [0.8916119337081909], [0.8910359144
210815], [0.8904682993888855], [0.8899089097976685], [0.8893566131591797], [0.8
888115286827087], [0.8882732391357422], [0.8877411484718323], [0.88721585273742
68], [0.8866963982582092]]
```

In [ ]:

In [163…:
```python
day_new=np.arange(1,101)
day_pred=np.arange(101,131)
```

In [164…:
```python
import matplotlib.pyplot as plt
```

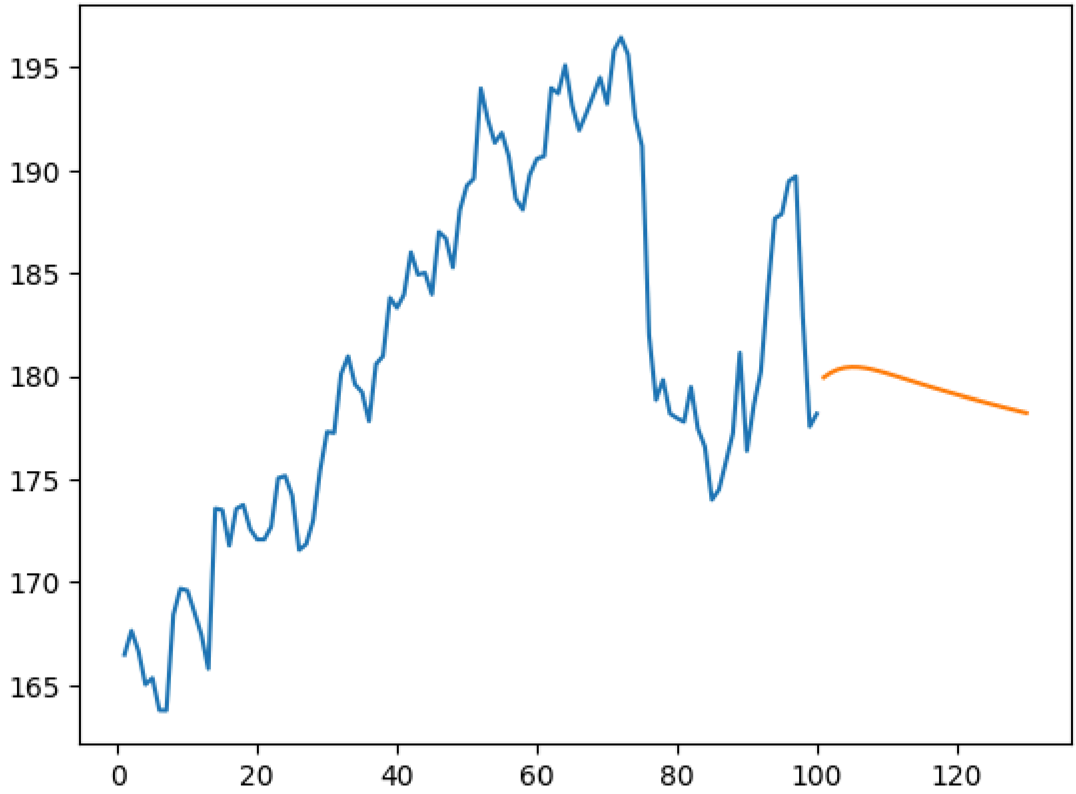
In [165…:
```python
len(df1)
```

Out[165]:
```
1192
```

In [166…:
```python
plt.plot(day_new,scaler.inverse_transform(df1[1092:]))
```

```
plt.plot(day_pred,scaler.inverse_transform(lst_output))
```

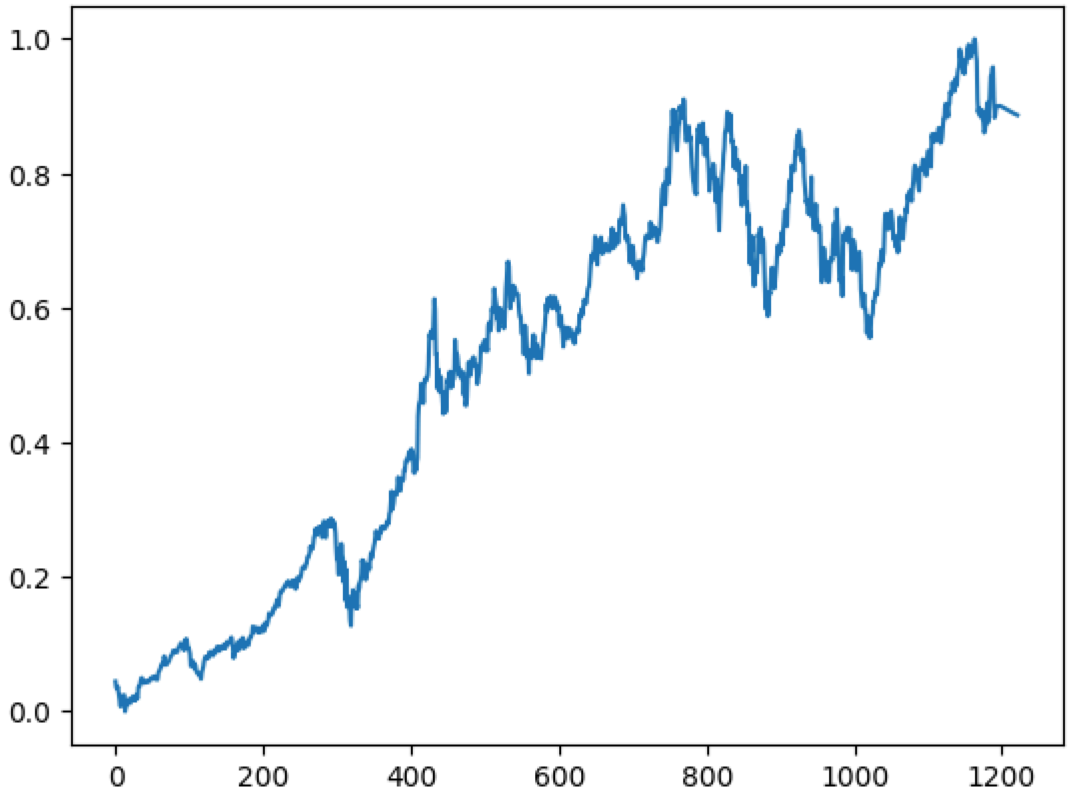
Out[166]: [<matplotlib.lines.Line2D at 0x1c6922065c0>]



In [167…

```
df3=df1.tolist()
df3.extend(lst_output)
plt.plot(df3)
```

Out[167]: [<matplotlib.lines.Line2D at 0x1c68b4bfee0>]



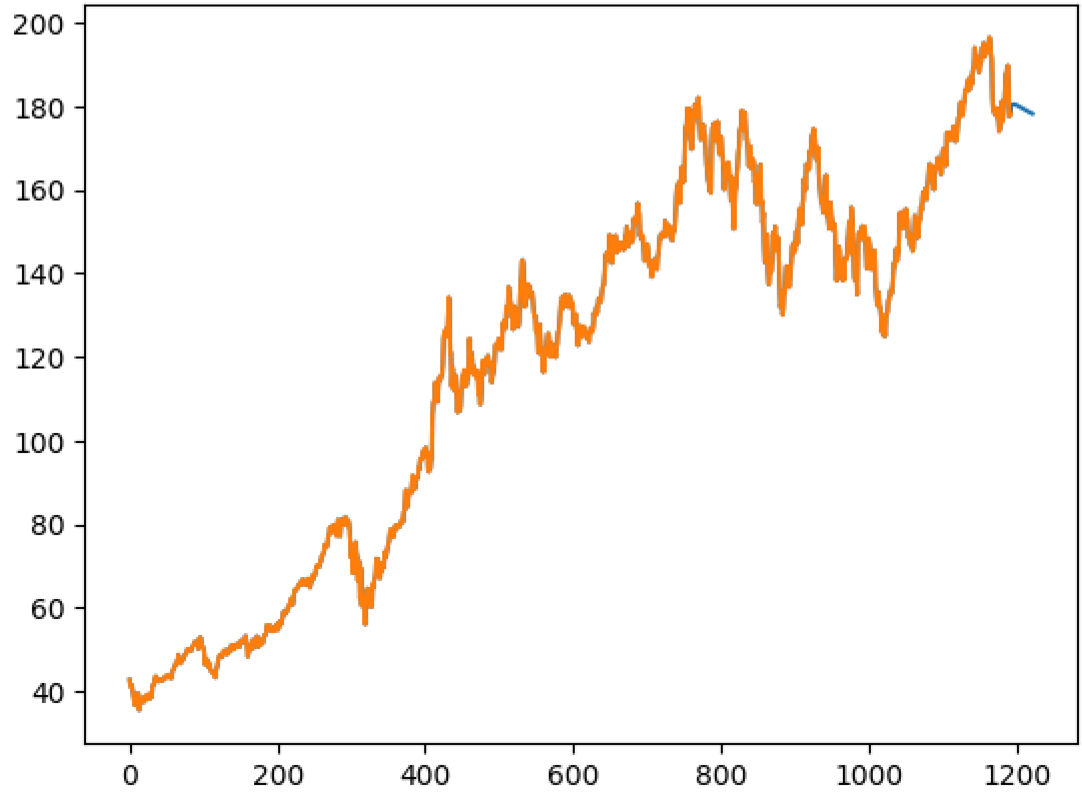
In [168…

```
df3=scaler.inverse_transform(df3).tolist()
```

In [169…

```
plt.plot(df3)
plt.plot(df_final)
```

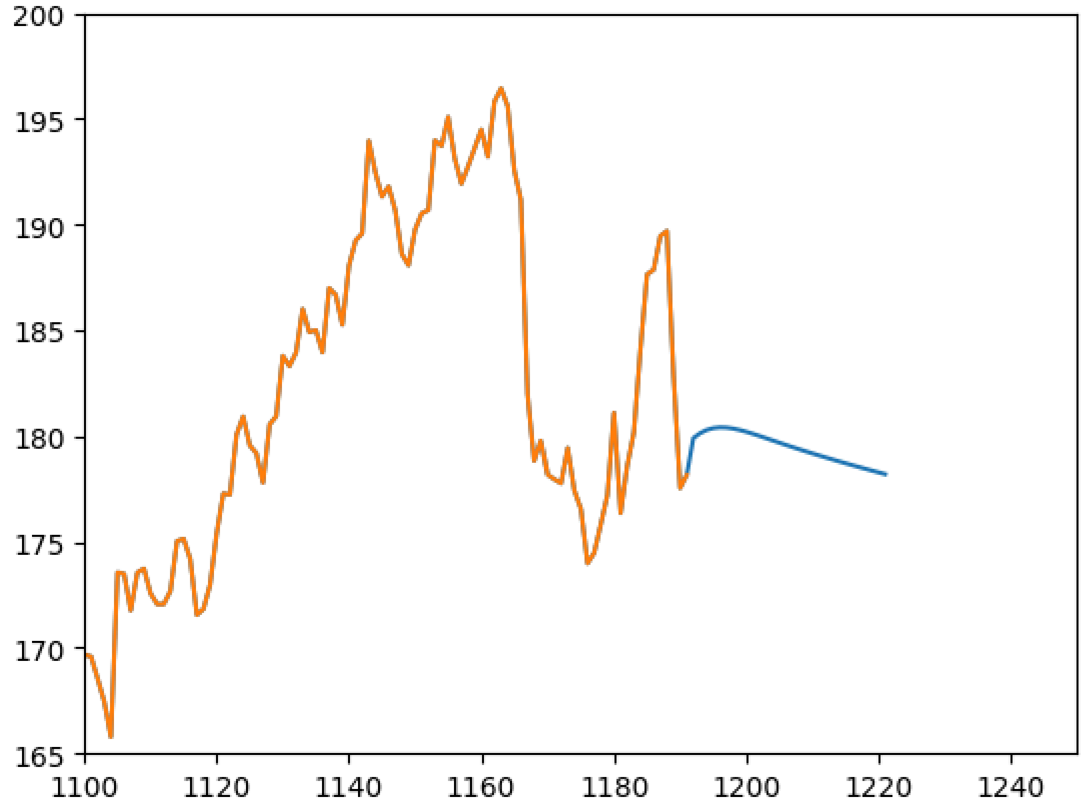
Out[169]: [<matplotlib.lines.Line2D at 0x1c692147760>]



In [170…

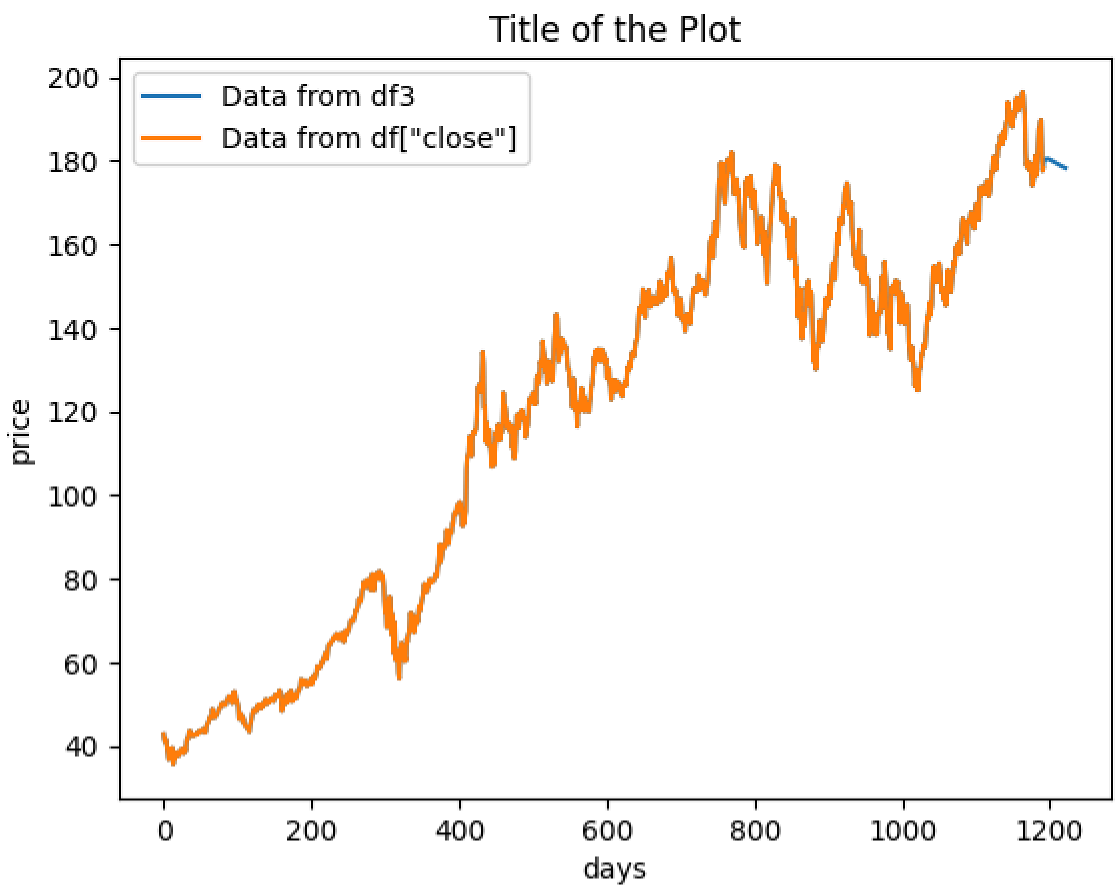
```
plt.xlim(1100,1250)
plt.ylim(165,200)
plt.plot(df3)
plt.plot(df_final)
```

Out[170]: [<matplotlib.lines.Line2D at 0x1c692454a90>]



In [171…

```
import matplotlib.pyplot as plt

# Assuming df3 and df["close"] are two dataframes or arrays that you want to plo

plt.plot(df3, label='Data from df3')
plt.plot(df["Close"], label='Data from df["close"]')

# Add labels and title
plt.xlabel('days')
plt.ylabel('price')
plt.title('Title of the Plot')

# Add legend
plt.legend()

# Show the plot
plt.show()
```



In [172…

```
df4=pd.read_csv("C:\\Users\\ravip\\OneDrive\\Documents\\sem6\\EC460-deep learnin
```

In [173…

```
df4
```

Out[173]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2022-11-11	145.820007	150.009995	144.369995	149.699997	148.867905	93979700
1	2022-11-14	148.970001	150.279999	147.429993	148.279999	147.455780	73374100
2	2022-11-15	152.220001	153.589996	148.559998	150.039993	149.206009	89868300
3	2022-11-16	149.130005	149.869995	147.289993	148.789993	147.962952	64218300
4	2022-11-17	146.429993	151.479996	146.149994	150.720001	149.882233	80389400
...	...	...	...	...	...	...	...
246	2023-11-06	176.380005	179.429993	176.210007	179.229996	178.994186	63841300
247	2023-11-07	179.179993	182.440002	178.970001	181.820007	181.580780	70530000
248	2023-11-08	182.350006	183.449997	181.589996	182.889999	182.649368	49340300
249	2023-11-09	182.960007	184.119995	181.809998	182.410004	182.169998	53763500
250	2023-11-10	183.970001	186.570007	183.529999	186.399994	186.399994	66133400

251 rows × 7 columns

In [174…

```
actual_data=df4[196:226]["Close"]
actual_data
```

Out[174]:
```
196    178.610001
197    180.190002
198    184.119995
199    187.649994
200    187.869995
201    189.460007
202    189.699997
203    182.910004
204    177.559998
205    178.179993
206    179.360001
207    176.300003
208    174.210007
209    175.740005
210    175.009995
211    177.970001
212    179.070007
213    175.490005
214    173.929993
215    174.789993
216    176.080002
217    171.960007
218    170.429993
219    170.690002
220    171.210007
221    173.750000
222    172.399994
223    173.660004
224    174.910004
225    177.490005
Name: Close, dtype: float64
```

In [175…
```
predicted_data=df3[-31:-1]
predicted_data
```

Out[175]:
```
[[178.17999299999997],
 [179.92462094013212],
 [180.15247289721296],
 [180.30790671607016],
 [180.3976549530582],
 [180.4319123480701],
 [180.42176556085775],
 [180.37782172436712],
 [180.30934529648778],
 [180.22405665879438],
 [180.12789235314366],
 [180.02532156936644],
 [179.91960508974265],
 [179.8128911943626],
 [179.70662805418013],
 [179.60165004583737],
 [179.4983983339958],
 [179.39702636723325],
 [179.29761086983868],
 [179.20007511752317],
 [179.10429443331716],
 [179.01008659703444],
 [178.9174036559944],
 [178.82607298054694],
 [178.7360657990837],
 [178.64719989141844],
 [178.5594944386234],
 [178.4728823069458],
 [178.38726759102437],
 [178.30274619622037]]
```

Checking the errors and difference between predicted and Actual values of Stock

In [176…
```
mse=mean_squared_error(actual_data,predicted_data)
mae=mean_absolute_error(actual_data,predicted_data)
rmse=math.sqrt(mse)
print("MSE is {: .9f}" .format(mse))
print("RMSE is {: .9f}" .format(rmse))
print("MAE is {: .9f}" .format(mae))
```

```
MSE is  27.170851042
RMSE is  5.212566646
MAE is  4.473661423
```

In [ ]: