LetsGrowMore

Priyanshi Badaya

Data Science Intern

Stock Market Prediction and Forecasting using stacked LSTM

Data Set Link- https://www.canva.com/link?target=https%3A%2F%2Fraw.githubusercontent.com%2Fmwitiderrick%2Fstockprice%2Fmaster%2FNSE-TATAGLOBAL.csv&design=DAEjrwWV35w

In [ ]:
```
IMPORTING LIBRARIES
```

In [3]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
```

In [4]:
```python
data = pd.read_csv('https://raw.githubusercontent.com/mwitiderrick/stockprice/master/NSE-TATAGLOBAL.csv')
data.head()
```

Out[4]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2018-09-28	234.05	235.95	230.20	233.50	233.75	3069914	7162.35
1	2018-09-27	234.55	236.80	231.10	233.80	233.25	5082859	11859.95
2	2018-09-26	240.00	240.00	232.50	235.00	234.25	2240909	5248.60
3	2018-09-25	233.30	236.75	232.00	236.25	236.10	2349368	5503.90
4	2018-09-24	233.55	239.20	230.75	234.00	233.30	3423509	7999.55

In [5]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2035 entries, 0 to 2034
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Date                  2035 non-null   object 
 1   Open                  2035 non-null   float64
 2   High                  2035 non-null   float64
 3   Low                   2035 non-null   float64
 4   Last                  2035 non-null   float64
 5   Close                 2035 non-null   float64
 6   Total Trade Quantity  2035 non-null   int64  
 7   Turnover (Lacs)       2035 non-null   float64
dtypes: float64(6), int64(1), object(1)
memory usage: 127.3+ KB
```

In [6]:
```python
data.describe()
```

Out[6]:

	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
count	2035.000000	2035.000000	2035.000000	2035.000000	2035.00000	2.035000e+03	2035.000000
mean	149.713735	151.992826	147.293931	149.474251	149.45027	2.335681e+06	3899.980565
std	48.664509	49.413109	47.931958	48.732570	48.71204	2.091778e+06	4570.767877
min	81.100000	82.800000	80.000000	81.000000	80.95000	3.961000e+04	37.040000
25%	120.025000	122.100000	118.300000	120.075000	120.05000	1.146444e+06	1427.460000
50%	141.500000	143.400000	139.600000	141.100000	141.25000	1.783456e+06	2512.030000
75%	157.175000	159.400000	155.150000	156.925000	156.90000	2.813594e+06	4539.015000
max	327.700000	328.750000	321.650000	325.950000	325.75000	2.919102e+07	55755.080000

In [7]:
```python
data['Date'] = pd.to_datetime(data['Date'])
data.dtypes
```

Out[7]:
```
Date                    datetime64[ns]
Open                           float64
High                           float64
Low                            float64
Last                           float64
Close                          float64
Total Trade Quantity             int64
Turnover (Lacs)                float64
dtype: object
```

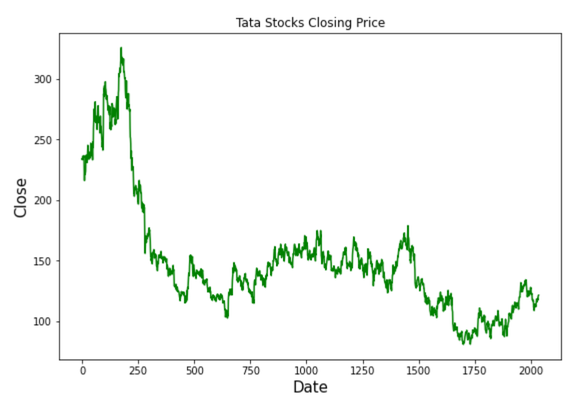
In [8]:
```python
data = data.sort_values('Date')
data.head()
```

Out[8]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
2034	2010-07-21	122.1	123.00	121.05	121.10	121.55	658666	803.56
2033	2010-07-22	120.3	122.00	120.25	120.75	120.90	293312	355.17
2032	2010-07-23	121.8	121.95	120.25	120.35	120.65	281312	340.31
2031	2010-07-26	120.1	121.00	117.10	117.10	117.60	658440	780.01
2030	2010-07-27	117.6	119.50	112.00	118.80	118.65	586100	694.98

In [9]:
```python
plt.figure(figsize = (9,6))
plt.title('Tata Stocks Closing Price')
plt.plot(data['Close'],'g')
plt.xlabel('Date',fontsize=15)
plt.ylabel('Close',fontsize=15)
```

Out[9]:
```
Text(0, 0.5, 'Close')
```



In [11]:
```python
dcorr = data.corr()
top_corr_features = dcorr.index
plt.figure(figsize=(10,7))
#plot heatmap
sns.heatmap(data[top_corr_features].corr(), annot=True, cmap="YlGnBu")
```

Out[11]:
```
<AxesSubplot:>
```

MinMaxScaler

From the original dataset, we can tell that each of our target value are in close proximity to one another. So, we will use MinMaxScaler to scale down all the target variables in the range of (0, 1) for the ease of computation.

In [13]:
```python
data_close = data.reset_index()['Close']
data_close.head()
scaler = MinMaxScaler(feature_range = (0, 1))
data_close = scaler.fit_transform(np.array(data_close).reshape(-1, 1))
```

In [14]:
```python
#### Splitting train, Test data
train_size = int(len(data_close)*0.70)
test_size = len(data_close) - train_size
train, test = data_close[0 : train_size, :], data_close[train_size : len(data_close), :1]
```

In [15]:
```python
def create_matrix(ds, time_step=1):
    dataX, dataY = [], []
    for i in range(len(ds)-time_step-1):
        a = ds[i:(i+time_step),0]
        dataX.append(a)
        dataY.append(ds[i+time_step,0])
    return np.array(dataX), np.array(dataY)
```

In [16]:
```python
step=100 # time step
X_train, y_train = create_matrix(train, step)
X_test, y_test = create_matrix(test, step)
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(1323, 100) (1323,)
(510, 100) (510,)
```

In [17]:
```python
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)
```

LSTM Model

In [18]:
```python
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')
```

In [19]:
```python
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
_________________________________________________________________
```

In [20]:
```python
history = model.fit(X_train, y_train, validation_split=0.1, epochs=77, batch_size=64, verbose=1, shuffle=True).history
```

```
Epoch 1/77
19/19 [==============================] - 21s 402ms/step - loss: 0.0126 - val_loss: 0.0012
Epoch 2/77
19/19 [==============================] - 5s 273ms/step - loss: 0.0022 - val_loss: 0.0010
Epoch 3/77
19/19 [==============================] - 5s 272ms/step - loss: 9.1068e-04 - val_loss: 0.0012
Epoch 4/77
19/19 [==============================] - 5s 275ms/step - loss: 8.2487e-04 - val_loss: 0.0010
Epoch 5/77
19/19 [==============================] - 5s 270ms/step - loss: 7.8200e-04 - val_loss: 0.0010
Epoch 6/77
19/19 [==============================] - 5s 271ms/step - loss: 7.6129e-04 - val_loss: 9.9624e-04
Epoch 7/77
19/19 [==============================] - 5s 275ms/step - loss: 7.4231e-04 - val_loss: 9.4753e-04
Epoch 8/77
19/19 [==============================] - 5s 273ms/step - loss: 7.2508e-04 - val_loss: 9.1960e-04
Epoch 9/77
19/19 [==============================] - 5s 274ms/step - loss: 7.3610e-04 - val_loss: 8.8987e-04
Epoch 10/77
19/19 [==============================] - 5s 274ms/step - loss: 7.1568e-04 - val_loss: 8.5990e-04
Epoch 11/77
19/19 [==============================] - 6s 333ms/step - loss: 7.2330e-04 - val_loss: 8.2346e-04
Epoch 12/77
19/19 [==============================] - 6s 298ms/step - loss: 6.6661e-04 - val_loss: 8.7568e-04
Epoch 13/77
19/19 [==============================] - 6s 331ms/step - loss: 6.8323e-04 - val_loss: 9.1490e-04
Epoch 14/77
19/19 [==============================] - 7s 356ms/step - loss: 6.5528e-04 - val_loss: 8.4697e-04
Epoch 15/77
19/19 [==============================] - 6s 341ms/step - loss: 6.3155e-04 - val_loss: 7.1486e-04
Epoch 16/77
19/19 [==============================] - 6s 335ms/step - loss: 6.3890e-04 - val_loss: 7.0997e-04
Epoch 17/77
19/19 [==============================] - 6s 333ms/step - loss: 6.0083e-04 - val_loss: 6.7898e-04
Epoch 18/77
19/19 [==============================] - 6s 331ms/step - loss: 5.8549e-04 - val_loss: 6.6199e-04
Epoch 19/77
19/19 [==============================] - 6s 316ms/step - loss: 5.8730e-04 - val_loss: 6.7840e-04
Epoch 20/77
19/19 [==============================] - 6s 292ms/step - loss: 5.5518e-04 - val_loss: 6.1268e-04
Epoch 21/77
19/19 [==============================] - 6s 292ms/step - loss: 6.1832e-04 - val_loss: 8.5536e-04
Epoch 22/77
19/19 [==============================] - 6s 302ms/step - loss: 5.8056e-04 - val_loss: 7.4409e-04
Epoch 23/77
19/19 [==============================] - 6s 311ms/step - loss: 5.6961e-04 - val_loss: 5.8850e-04
Epoch 24/77
19/19 [==============================] - 6s 338ms/step - loss: 5.5236e-04 - val_loss: 6.1269e-04
Epoch 25/77
19/19 [==============================] - 6s 334ms/step - loss: 5.8782e-04 - val_loss: 6.6021e-04
Epoch 26/77
19/19 [==============================] - 6s 333ms/step - loss: 5.0381e-04 - val_loss: 5.3167e-04
Epoch 27/77
19/19 [==============================] - 6s 334ms/step - loss: 4.9593e-04 - val_loss: 5.2528e-04
Epoch 28/77
19/19 [==============================] - 6s 335ms/step - loss: 4.8005e-04 - val_loss: 4.9735e-04
Epoch 29/77
19/19 [==============================] - 6s 337ms/step - loss: 4.9021e-04 - val_loss: 5.5060e-04
Epoch 30/77
19/19 [==============================] - 6s 337ms/step - loss: 4.7612e-04 - val_loss: 4.7173e-04
Epoch 31/77
19/19 [==============================] - 6s 343ms/step - loss: 5.3426e-04 - val_loss: 4.7232e-04
Epoch 32/77
19/19 [==============================] - 6s 340ms/step - loss: 4.6261e-04 - val_loss: 5.2266e-04
Epoch 33/77
19/19 [==============================] - 6s 334ms/step - loss: 4.6113e-04 - val_loss: 4.9537e-04
Epoch 34/77
19/19 [==============================] - 6s 343ms/step - loss: 4.3483e-04 - val_loss: 4.4188e-04
Epoch 35/77
19/19 [==============================] - 6s 342ms/step - loss: 4.1188e-04 - val_loss: 4.3008e-04
Epoch 36/77
19/19 [==============================] - 6s 343ms/step - loss: 3.9891e-04 - val_loss: 4.2977e-04
Epoch 37/77
19/19 [==============================] - 6s 343ms/step - loss: 3.8861e-04 - val_loss: 4.7543e-04
Epoch 38/77
19/19 [==============================] - 6s 342ms/step - loss: 3.9535e-04 - val_loss: 3.8665e-04
Epoch 39/77
19/19 [==============================] - 6s 341ms/step - loss: 3.7475e-04 - val_loss: 3.7045e-04
Epoch 40/77
19/19 [==============================] - 7s 368ms/step - loss: 3.7165e-04 - val_loss: 3.5009e-04
Epoch 41/77
19/19 [==============================] - 7s 369ms/step - loss: 3.6045e-04 - val_loss: 3.2425e-04
Epoch 42/77
19/19 [==============================] - 7s 373ms/step - loss: 3.4289e-04 - val_loss: 3.5298e-04
Epoch 43/77
19/19 [==============================] - 8s 442ms/step - loss: 3.2685e-04 - val_loss: 3.1705e-04
Epoch 44/77
19/19 [==============================] - 7s 367ms/step - loss: 3.1805e-04 - val_loss: 3.1081e-04
Epoch 45/77
19/19 [==============================] - 7s 381ms/step - loss: 3.1152e-04 - val_loss: 2.8742e-04
Epoch 46/77
19/19 [==============================] - 9s 492ms/step - loss: 3.1494e-04 - val_loss: 2.7147e-04
Epoch 47/77
19/19 [==============================] - 8s 419ms/step - loss: 2.9894e-04 - val_loss: 3.0867e-04
Epoch 48/77
19/19 [==============================] - 9s 506ms/step - loss: 2.9034e-04 - val_loss: 2.7369e-04
Epoch 49/77
19/19 [==============================] - 8s 404ms/step - loss: 2.9483e-04 - val_loss: 3.5000e-04
Epoch 50/77
19/19 [==============================] - 7s 384ms/step - loss: 2.8363e-04 - val_loss: 2.7601e-04
Epoch 51/77
19/19 [==============================] - 9s 477ms/step - loss: 2.8433e-04 - val_loss: 2.5465e-04
Epoch 52/77
19/19 [==============================] - 6s 332ms/step - loss: 2.7805e-04 - val_loss: 2.7037e-04
Epoch 53/77
19/19 [==============================] - 6s 332ms/step - loss: 2.8520e-04 - val_loss: 2.3266e-04
Epoch 54/77
19/19 [==============================] - 6s 311ms/step - loss: 2.5411e-04 - val_loss: 3.3747e-04
Epoch 55/77
19/19 [==============================] - 6s 296ms/step - loss: 2.7600e-04 - val_loss: 3.3386e-04
Epoch 56/77
19/19 [==============================] - 7s 364ms/step - loss: 2.6383e-04 - val_loss: 2.8304e-04
Epoch 57/77
19/19 [==============================] - 7s 358ms/step - loss: 2.5307e-04 - val_loss: 2.4800e-04
Epoch 58/77
19/19 [==============================] - 7s 367ms/step - loss: 2.3156e-04 - val_loss: 2.2019e-04
Epoch 59/77
19/19 [==============================] - 7s 367ms/step - loss: 2.4225e-04 - val_loss: 1.9924e-04
Epoch 60/77
19/19 [==============================] - 7s 357ms/step - loss: 2.4692e-04 - val_loss: 1.9849e-04
Epoch 61/77
19/19 [==============================] - 7s 361ms/step - loss: 2.6103e-04 - val_loss: 2.3723e-04
Epoch 62/77
19/19 [==============================] - 7s 355ms/step - loss: 2.4236e-04 - val_loss: 2.0474e-04
Epoch 63/77
19/19 [==============================] - 7s 356ms/step - loss: 2.1516e-04 - val_loss: 1.8463e-04
Epoch 64/77
19/19 [==============================] - 7s 365ms/step - loss: 2.2108e-04 - val_loss: 1.8563e-04
Epoch 65/77
19/19 [==============================] - 8s 412ms/step - loss: 2.1391e-04 - val_loss: 1.7637e-04
Epoch 66/77
19/19 [==============================] - 7s 365ms/step - loss: 2.1650e-04 - val_loss: 3.2710e-04
Epoch 67/77
19/19 [==============================] - 7s 355ms/step - loss: 2.2696e-04 - val_loss: 2.5188e-04
Epoch 68/77
19/19 [==============================] - 6s 340ms/step - loss: 2.0352e-04 - val_loss: 1.9597e-04
Epoch 69/77
19/19 [==============================] - 6s 342ms/step - loss: 2.1069e-04 - val_loss: 1.8025e-04
Epoch 70/77
19/19 [==============================] - 7s 348ms/step - loss: 2.0512e-04 - val_loss: 1.7968e-04
Epoch 71/77
19/19 [==============================] - 7s 347ms/step - loss: 1.9982e-04 - val_loss: 1.7075e-04
Epoch 72/77
19/19 [==============================] - 7s 362ms/step - loss: 2.0837e-04 - val_loss: 1.6660e-04
Epoch 73/77
19/19 [==============================] - 7s 337ms/step - loss: 1.8418e-04 - val_loss: 1.5303e-04
Epoch 74/77
19/19 [==============================] - 7s 346ms/step - loss: 1.7076e-04 - val_loss: 1.5163e-04
Epoch 75/77
19/19 [==============================] - 9s 434ms/step - loss: 1.8557e-04 - val_loss: 1.4737e-04
Epoch 76/77
19/19 [==============================] - 8s 431ms/step - loss: 1.7837e-04 - val_loss: 1.4948e-04
Epoch 77/77
19/19 [==============================] - 9s 480ms/step - loss: 1.7465e-04 - val_loss: 1.6115e-04
```

In [21]:
```python
train_predict = model.predict(X_train)
test_predict = model.predict(X_test)
```

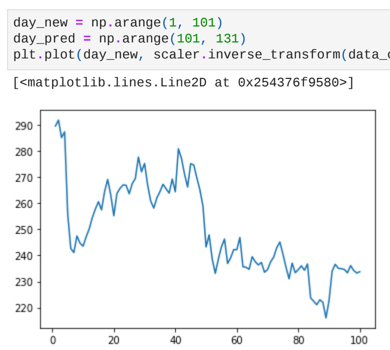
In [22]:
```python
# Reversing the MinMax Scaler
train_predict = scaler.inverse_transform(train_predict)
test_predict = scaler.inverse_transform(test_predict)
```

In [ ]:
```python
import math
from sklearn.metrics import mean_squared_error
math.sqrt(mean_squared_error(Y_train, train_predict))
math.sqrt(mean_squared_error(ytest,test_predict))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [34], in <module>
      1 import math
      2 from sklearn.metrics import mean_squared_error
----> 3 math.sqrt(mean_squared_error(Y_train, train_predict))
      4 math.sqrt(mean_squared_error(ytest,test_predict))

NameError: name 'Y_train' is not defined
```

In [35]:
```python
### Visualise the Predictions
look_back = 100
train_num_pyredict_plot = np.empty_like(data_close)
train_num_pyredict_plot[:, :] = np.nan
train_num_pyredict_plot[look_back : len(train_predict) + look_back, :] = train_predict
test_predict_plot = np.empty_like(data_close)
test_predict_plot[:, :] = np.nan
test_predict_plot[len(train_predict) + (look_back * 2) + 1 : len(data_close) - 1, :] = test_predict
plt.plot(scaler.inverse_transform(data_close))
plt.plot(train_num_pyredict_plot)
plt.plot(test_predict_plot)
plt.show()
```



In [36]:
```python
### Future Prediction Model
x_inum_pyut=test[307:].reshape(1, -1)
x_inum_pyut.shape
temp_inum_pyut = list(x_inum_pyut)
temp_inum_pyut = temp_inum_pyut[0].tolist()
temp_inum_pyut = list(x_inum_pyut)
temp_inum_pyut = temp_inum_pyut[0].tolist()
```

In [37]:
```python
day_new = np.arange(1, 101)
day_pred = np.arange(101, 131)
plt.plot(day_new, scaler.inverse_transform(data_close[1935 : ]))
```

Out[37]:
```
[<matplotlib.lines.Line2D at 0x254376f9580>]
```