NAME- RAJ KAMAL SHAKYA

LGM-VIP INTERNSHIP

BEGINNER LEVEL TASK -2

Stock Market Prediction And Forecasting Using Stacked LSTM

1. IMPORT LIBRARY

In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import MinMaxScaler
```

1. LOAD DATA

In [ ]:
```
data=pd.read_csv("NSE.csv")
```

In [ ]:
```
data.head()
```

Out[ ]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2018-10-08	208.00	222.25	206.85	216.00	215.15	4642146.0	10062.83
1	2018-10-05	217.00	218.60	205.90	210.25	209.20	3519515.0	7407.06
2	2018-10-04	223.50	227.80	216.15	217.25	218.20	1728786.0	3815.79
3	2018-10-03	230.00	237.50	225.75	226.45	227.60	1708590.0	3960.27
4	2018-10-01	234.55	234.60	221.05	230.30	230.90	1534749.0	3486.05

In [ ]:
```
data.tail()
```

Out[ ]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
1230	2013-10-14	160.85	161.45	157.70	159.3	159.45	1281419.0	2039.09
1231	2013-10-11	161.15	163.45	159.00	159.8	160.05	1880046.0	3030.76
1232	2013-10-10	156.00	160.80	155.85	160.3	160.15	3124853.0	4978.80
1233	2013-10-09	155.70	158.20	154.15	155.3	155.55	2049580.0	3204.49
1234	2013-10-08	157.00	157.80	155.20	155.8	155.80	1720413.0	2688.94

In [ ]:
```
# sort with date
data['Date']=pd.to_datetime(data['Date'])
print(type(data.Date[0]))
```

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

In [ ]:
```
df=data.sort_values(by='Date')
df.head()
```

Out[ ]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
1234	2013-10-08	157.00	157.80	155.20	155.8	155.80	1720413.0	2688.94
1233	2013-10-09	155.70	158.20	154.15	155.3	155.55	2049580.0	3204.49
1232	2013-10-10	156.00	160.80	155.85	160.3	160.15	3124853.0	4978.80
1231	2013-10-11	161.15	163.45	159.00	159.8	160.05	1880046.0	3030.76
1230	2013-10-14	160.85	161.45	157.70	159.3	159.45	1281419.0	2039.09

In [ ]:
```
df.reset_index(inplace=True)
```

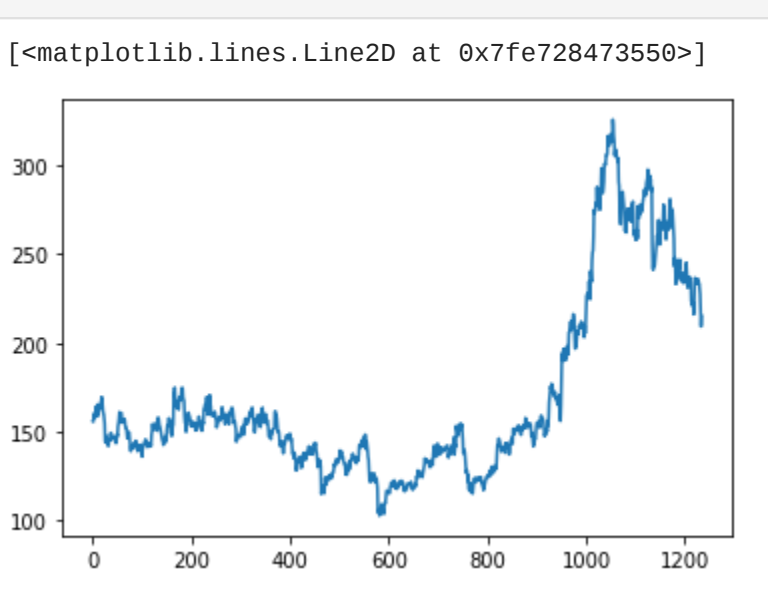
In [ ]:
```
df.head()
```

Out[ ]:

	index	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	1234	2013-10-08	157.00	157.80	155.20	155.8	155.80	1720413.0	2688.94
1	1233	2013-10-09	155.70	158.20	154.15	155.3	155.55	2049580.0	3204.49
2	1232	2013-10-10	156.00	160.80	155.85	160.3	160.15	3124853.0	4978.80
3	1231	2013-10-11	161.15	163.45	159.00	159.8	160.05	1880046.0	3030.76
4	1230	2013-10-14	160.85	161.45	157.70	159.3	159.45	1281419.0	2039.09

In [ ]:
```
plt.plot(df['Close'])
```

Out[ ]: [<matplotlib.lines.Line2D at 0x7fe728473550>]



In [ ]:
```
df1=df['Close']
```

1. PREPARE DATA

In [ ]:
```
## LSTM are sensitive to the scale of the data,therefore applying MinMax scaler
scaler=MinMaxScaler(feature_range=(0,1))
df1=scaler.fit_transform(np.array(df1).reshape(-1,1))
df1
```

Out[ ]:
```
array([[0.23823398],
       [0.2371134 ],
       [0.25773196],
       ...,
       [0.51792918],
       [0.47758853],
       [0.50425818]])
```

In [ ]:
```
##splitting dataset into train and test split
training_size=int(len(df1)*0.70)
test_size=len(df1)-training_size
train_data,test_data=df1[0:training_size,:],df1[training_size:len(df1),:1]
```

In [ ]:
```
training_size,test_size
```

Out[ ]: (864, 371)

In [ ]:
```
# 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 np.array(dataX), np.array(dataY)
```

In [ ]:
```
# 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 [ ]:
```
print(X_train.shape), print(y_train.shape)
```

```
(763, 100)
(763,)
```

Out[ ]: (None, None)

In [ ]:
```
print(X_test.shape), print(ytest.shape)
```

```
(270, 100)
(270,)
```

Out[ ]: (None, None)

In [ ]:
```
# 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)
```

1. MODEL BUILDING

In [ ]:
```
### Create the Stacked LSTM model
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
```

In [ ]:
```
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
_________________________________________________________________
```

In [ ]:
```
model.fit(X_train,y_train,validation_split=0.1,epochs=60,batch_size=64,verbose=1)
```

```
Epoch 1/60
11/11 [==============================] - 8s 295ms/step - loss: 0.0091 - val_loss: 0.0019
Epoch 2/60
11/11 [==============================] - 2s 176ms/step - loss: 0.0029 - val_loss: 0.0018
Epoch 3/60
11/11 [==============================] - 2s 170ms/step - loss: 0.0022 - val_loss: 0.0011
Epoch 4/60
11/11 [==============================] - 2s 172ms/step - loss: 0.0015 - val_loss: 6.2189e-04
Epoch 5/60
11/11 [==============================] - 2s 171ms/step - loss: 0.0011 - val_loss: 5.3674e-04
Epoch 6/60
11/11 [==============================] - 2s 174ms/step - loss: 0.0011 - val_loss: 6.2505e-04
Epoch 7/60
11/11 [==============================] - 2s 170ms/step - loss: 0.0010 - val_loss: 8.4115e-04
Epoch 8/60
11/11 [==============================] - 2s 174ms/step - loss: 9.6495e-04 - val_loss: 5.9642e-04
Epoch 9/60
11/11 [==============================] - 2s 173ms/step - loss: 9.5185e-04 - val_loss: 4.7372e-04
Epoch 10/60
11/11 [==============================] - 2s 173ms/step - loss: 9.5183e-04 - val_loss: 7.2890e-04
Epoch 11/60
11/11 [==============================] - 2s 176ms/step - loss: 9.3416e-04 - val_loss: 8.4711e-04
Epoch 12/60
11/11 [==============================] - 2s 176ms/step - loss: 9.4325e-04 - val_loss: 9.5590e-04
Epoch 13/60
11/11 [==============================] - 2s 175ms/step - loss: 8.9423e-04 - val_loss: 7.4650e-04
Epoch 14/60
11/11 [==============================] - 2s 178ms/step - loss: 8.1313e-04 - val_loss: 4.4684e-04
Epoch 15/60
11/11 [==============================] - 2s 174ms/step - loss: 7.8776e-04 - val_loss: 4.4898e-04
Epoch 16/60
11/11 [==============================] - 2s 170ms/step - loss: 7.8323e-04 - val_loss: 4.1582e-04
Epoch 17/60
11/11 [==============================] - 2s 171ms/step - loss: 7.3928e-04 - val_loss: 4.2275e-04
Epoch 18/60
11/11 [==============================] - 2s 170ms/step - loss: 7.2076e-04 - val_loss: 6.5585e-04
Epoch 19/60
11/11 [==============================] - 2s 171ms/step - loss: 7.0410e-04 - val_loss: 5.3706e-04
Epoch 20/60
11/11 [==============================] - 2s 171ms/step - loss: 6.9747e-04 - val_loss: 5.1224e-04
Epoch 21/60
11/11 [==============================] - 2s 172ms/step - loss: 6.7135e-04 - val_loss: 3.8551e-04
Epoch 22/60
11/11 [==============================] - 2s 171ms/step - loss: 6.6599e-04 - val_loss: 4.7411e-04
Epoch 23/60
11/11 [==============================] - 2s 168ms/step - loss: 6.1339e-04 - val_loss: 4.2646e-04
Epoch 24/60
11/11 [==============================] - 2s 169ms/step - loss: 6.0368e-04 - val_loss: 3.5996e-04
Epoch 25/60
11/11 [==============================] - 2s 171ms/step - loss: 5.9184e-04 - val_loss: 3.8579e-04
Epoch 26/60
11/11 [==============================] - 2s 175ms/step - loss: 5.6898e-04 - val_loss: 5.3197e-04
Epoch 27/60
11/11 [==============================] - 2s 171ms/step - loss: 6.2604e-04 - val_loss: 5.3961e-04
Epoch 28/60
11/11 [==============================] - 2s 170ms/step - loss: 6.2292e-04 - val_loss: 3.6026e-04
Epoch 29/60
11/11 [==============================] - 2s 171ms/step - loss: 5.9219e-04 - val_loss: 3.4618e-04
Epoch 30/60
11/11 [==============================] - 2s 172ms/step - loss: 5.3796e-04 - val_loss: 4.9693e-04
Epoch 31/60
11/11 [==============================] - 2s 175ms/step - loss: 5.4462e-04 - val_loss: 3.6071e-04
Epoch 32/60
11/11 [==============================] - 2s 177ms/step - loss: 5.2240e-04 - val_loss: 3.4403e-04
Epoch 33/60
11/11 [==============================] - 2s 175ms/step - loss: 5.5494e-04 - val_loss: 5.0003e-04
Epoch 34/60
11/11 [==============================] - 2s 171ms/step - loss: 5.0637e-04 - val_loss: 3.6468e-04
Epoch 35/60
11/11 [==============================] - 2s 174ms/step - loss: 4.9271e-04 - val_loss: 3.2515e-04
Epoch 36/60
11/11 [==============================] - 2s 169ms/step - loss: 4.8640e-04 - val_loss: 4.6487e-04
Epoch 37/60
11/11 [==============================] - 2s 173ms/step - loss: 4.8982e-04 - val_loss: 4.1832e-04
Epoch 38/60
11/11 [==============================] - 2s 171ms/step - loss: 4.7346e-04 - val_loss: 2.8406e-04
Epoch 39/60
11/11 [==============================] - 2s 175ms/step - loss: 4.9858e-04 - val_loss: 3.6149e-04
Epoch 40/60
11/11 [==============================] - 2s 170ms/step - loss: 4.5629e-04 - val_loss: 2.8142e-04
Epoch 41/60
11/11 [==============================] - 2s 172ms/step - loss: 4.5875e-04 - val_loss: 4.5174e-04
Epoch 42/60
11/11 [==============================] - 2s 171ms/step - loss: 4.8847e-04 - val_loss: 2.6880e-04
Epoch 43/60
11/11 [==============================] - 2s 173ms/step - loss: 4.4822e-04 - val_loss: 4.4395e-04
Epoch 44/60
11/11 [==============================] - 2s 172ms/step - loss: 4.8107e-04 - val_loss: 3.5326e-04
Epoch 45/60
11/11 [==============================] - 2s 171ms/step - loss: 4.6134e-04 - val_loss: 2.5127e-04
Epoch 46/60
11/11 [==============================] - 2s 171ms/step - loss: 4.2663e-04 - val_loss: 2.5461e-04
Epoch 47/60
11/11 [==============================] - 2s 171ms/step - loss: 4.1843e-04 - val_loss: 4.6376e-04
Epoch 48/60
11/11 [==============================] - 2s 172ms/step - loss: 4.1889e-04 - val_loss: 2.8639e-04
Epoch 49/60
11/11 [==============================] - 2s 173ms/step - loss: 4.0924e-04 - val_loss: 3.0457e-04
Epoch 50/60
11/11 [==============================] - 2s 169ms/step - loss: 3.9453e-04 - val_loss: 2.3208e-04
Epoch 51/60
11/11 [==============================] - 2s 170ms/step - loss: 4.0058e-04 - val_loss: 2.2754e-04
Epoch 52/60
11/11 [==============================] - 2s 172ms/step - loss: 3.9583e-04 - val_loss: 3.7254e-04
Epoch 53/60
11/11 [==============================] - 2s 167ms/step - loss: 3.7187e-04 - val_loss: 2.6971e-04
Epoch 54/60
11/11 [==============================] - 2s 169ms/step - loss: 3.6223e-04 - val_loss: 2.6407e-04
Epoch 55/60
11/11 [==============================] - 2s 176ms/step - loss: 3.5833e-04 - val_loss: 3.4166e-04
Epoch 56/60
11/11 [==============================] - 2s 174ms/step - loss: 3.8704e-04 - val_loss: 2.1048e-04
Epoch 57/60
11/11 [==============================] - 2s 172ms/step - loss: 3.3931e-04 - val_loss: 3.1227e-04
Epoch 58/60
11/11 [==============================] - 2s 172ms/step - loss: 3.4760e-04 - val_loss: 2.0441e-04
Epoch 59/60
11/11 [==============================] - 2s 173ms/step - loss: 3.3888e-04 - val_loss: 2.3699e-04
Epoch 60/60
11/11 [==============================] - 2s 174ms/step - loss: 3.2169e-04 - val_loss: 2.0129e-04
```

Out[ ]: <keras.callbacks.History at 0x7fe6e3416950>

In [ ]:
```
## Lets do the prediction and check performance metrics
test_predict=model.predict(X_test)
```

In [ ]:
```
## Transform back to original form
test_predict1=scaler.inverse_transform(test_predict)
```

In [ ]:
```
test_predict1
```

Out[ ]:
```
array([[191.15317],
       [191.65982],
       [194.30145],
       [197.8249 ],
       [201.13991],
       [203.94894],
       [206.39359],
       [206.84453],
       [206.99156],
       [206.71843],
       [206.91104],
       [208.10168],
       [208.95831],
       [208.90826],
       [205.66905],
       [200.40524],
       [197.56657],
       [195.93913],
       [196.40733],
       [198.7499 ],
       [201.8168 ],
       [204.15546],
       [206.4668 ],
       [208.32845],
       [209.09029],
       [208.96622],
       [208.58614],
       [208.80568],
       [208.91556],
       [208.66585],
       [208.41602],
       [207.4331 ],
       [205.73248],
       [203.77975],
       [203.83875],
       [204.2525 ],
       [205.68806],
       [210.01881],
       [216.66734],
       [222.80745],
       [226.79152],
       [227.75845],
       [226.27705],
       [226.33035],
       [224.76794],
       [222.27472],
       [224.37515],
       [227.61977],
       [229.70071],
       [233.83832],
       [239.33066],
       [244.00299],
       [248.83432],
       [256.72717],
       [262.87482],
       [265.45334],
       [265.04285],
       [264.32755],
       [263.2069 ],
       [265.65634],
       [266.48996],
       [268.41965],
       [269.27658],
       [267.9147 ],
       [266.68063],
       [264.83688],
       [266.84015],
       [272.03027],
       [280.45764],
       [283.7222 ],
       [281.89484],
       [278.6038 ],
       [277.09482],
       [278.59393],
       [284.45584],
       [290.31403],
       [293.46716],
       [294.96274],
       [296.25504],
       [296.73508],
       [296.92197],
       [300.4506 ],
       [302.81467],
       [302.86038],
       [301.96692],
       [301.61685],
       [302.1189 ],
       [304.6874 ],
       [306.48468],
       [306.88263],
       [306.44604],
       [310.60794],
       [314.25607],
       [313.56766],
       [310.32474],
       [304.72308],
       [301.76526],
       [301.75613],
       [302.7656 ],
       [303.09903],
       [303.3266 ],
       [302.89392],
       [303.30106],
       [299.5891 ],
       [294.82367],
       [287.4244 ],
       [281.33072],
       [276.48993],
       [278.1877 ],
       [282.87717],
       [286.67758],
       [290.78384],
       [292.57364],
       [291.35547],
       [287.78656],
       [281.81207],
       [275.15536],
       [271.65625],
       [269.6528 ],
       [271.5277 ],
       [275.6763 ],
       [279.90875],
       [282.6939 ],
       [283.35574],
       [282.74826],
       [281.04306],
       [277.51218],
       [275.58038],
       [273.54742],
       [274.43478],
       [277.20526],
       [279.87488],
       [281.72147],
       [280.04813],
       [273.39935],
       [267.2912 ],
       [264.27396],
       [262.6243 ],
       [261.94034],
       [263.38464],
       [266.07117],
       [266.96515],
       [272.062  ],
       [277.1085 ],
       [278.3913 ],
       [277.7706 ],
       [276.17905],
       [276.07904],
       [275.83252],
       [274.77014],
       [273.57672],
       [273.29233],
       [274.93372],
       [277.36813],
       [280.25577],
       [282.93903],
       [284.43716],
       [283.74432],
       [282.0107 ],
       [282.44946],
       [284.52322],
       [286.78046],
       [290.0325 ],
       [290.80316],
       [287.84863],
       [284.77856],
       [285.25797],
       [285.8485 ],
       [287.01547],
       [285.91516],
       [285.05185],
       [274.67682],
       [260.1534 ],
       [249.36647],
       [246.27109],
       [247.47635],
       [250.2553 ],
       [254.13402],
       [258.44733],
       [262.8996 ],
       [266.87378],
       [269.84338],
       [269.9591 ],
       [270.13263],
       [271.4377 ],
       [270.58813],
       [265.91336],
       [263.28082],
       [263.34204],
       [265.00568],
       [266.7    ],
       [266.70142],
       [267.01202],
       [268.10513],
       [271.7701 ],
       [273.5682 ],
       [274.47397],
       [271.90866],
       [266.48718],
       [260.95993],
       [258.77908],
       [259.98538],
       [263.44012],
       [266.3613 ],
       [267.46793],
       [268.99008],
       [268.76114],
       [272.77164],
       [276.43686],
       [276.13095],
       [272.09537],
       [270.56668],
       [270.85355],
       [270.18378],
       [267.7394 ],
       [263.4874 ],
       [255.41649],
       [249.92543],
       [245.48851],
       [241.64949],
       [241.02792],
       [243.87216],
       [248.65648],
       [250.4027 ],
       [250.31178],
       [250.20851],
       [250.0946 ],
       [251.10239],
       [249.06895],
       [245.57101],
       [242.18553],
       [241.21284],
       [241.31801],
       [241.4263 ],
       [241.64473],
       [240.81073],
       [239.83818],
       [239.93541],
       [241.08035],
       [243.34656],
       [246.01242],
       [246.5675 ],
       [244.0961 ],
       [239.47232],
       [236.87282],
       [235.34792],
       [235.05551],
       [235.91226],
       [236.67213],
       [237.74199],
       [235.17108],
       [230.89307],
       [226.86891],
       [224.81305],
       [224.27672],
       [223.15984],
       [223.58377],
       [227.82721],
       [233.87186],
       [238.47842],
       [240.46294],
       [240.07784],
       [238.0091 ],
       [236.30452],
       [234.77267],
       [233.36247],
       [232.51135],
       [231.4586 ],
       [229.70702],
       [225.67807]], dtype=float32)
```

In [ ]:
```
## Calculate RMSE performance metrics
import math
from sklearn.metrics import mean_squared_error
math.sqrt(mean_squared_error(ytest,test_predict))
```

Out[ ]: 0.042101229896561884