Stock Market Prediction using LSTM by Azeem

In [ ]:
```
import yfinance as yf
import matplotlib.pyplot as plt
import numpy as np

from sklearn.preprocessing import MinMaxScaler
```

In [ ]:
```
stocksymbol='GAIL.NS'
```

In [ ]:
```
d=yf.download(tickers=stocksymbol,period='5y',interval='1d')
```

```
[*********************100%***********************]  1 of 1 completed
```

In [ ]:
```
type(d)
```

Out[ ]:
```
pandas.core.frame.DataFrame
```

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

Out[ ]:

Date	Open	High	Low	Close	Adj Close	Volume
2019-07-19	95.300003	95.300003	91.400002	91.733330	72.365166	14154360
2019-07-22	91.133331	92.500000	89.900002	92.000000	72.575539	10270485
2019-07-23	91.966667	93.866669	91.833336	92.400002	72.891075	7824837
2019-07-24	92.733330	92.733330	90.133331	91.099998	71.865555	8352232
2019-07-25	91.099998	91.666664	89.433334	89.833336	70.866333	12123903

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

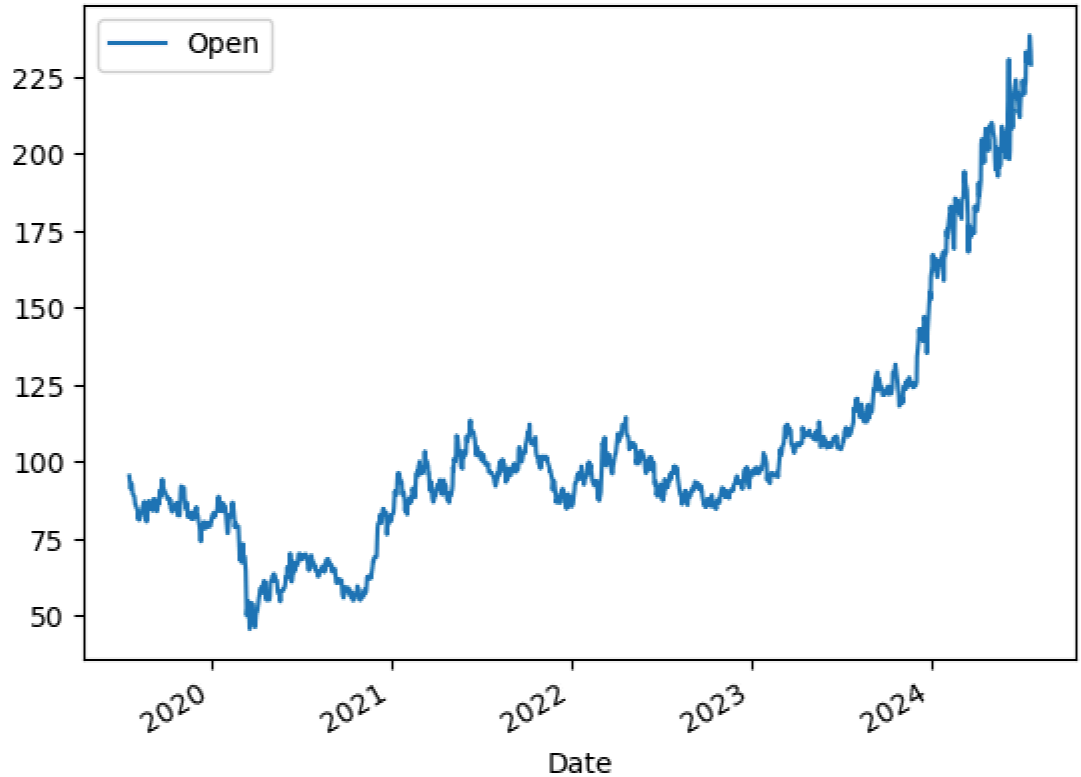
Out[ ]:

Date	Open	High	Low	Close	Adj Close	Volume
2024-07-12	229.699997	233.800003	224.699997	228.710007	228.710007	26631708
2024-07-15	229.600006	238.000000	228.089996	237.160004	237.160004	32695974
2024-07-16	238.199997	239.110001	233.000000	233.410004	233.410004	19213934
2024-07-18	233.509995	233.869995	227.759995	228.860001	228.860001	14554717
2024-07-19	228.860001	228.860001	219.020004	219.759995	219.759995	17258299

In [ ]:
```
op=d[['Open']]
```

In [ ]:
```
op.plot()
```

Out[ ]:
```
<Axes: xlabel='Date'>
```



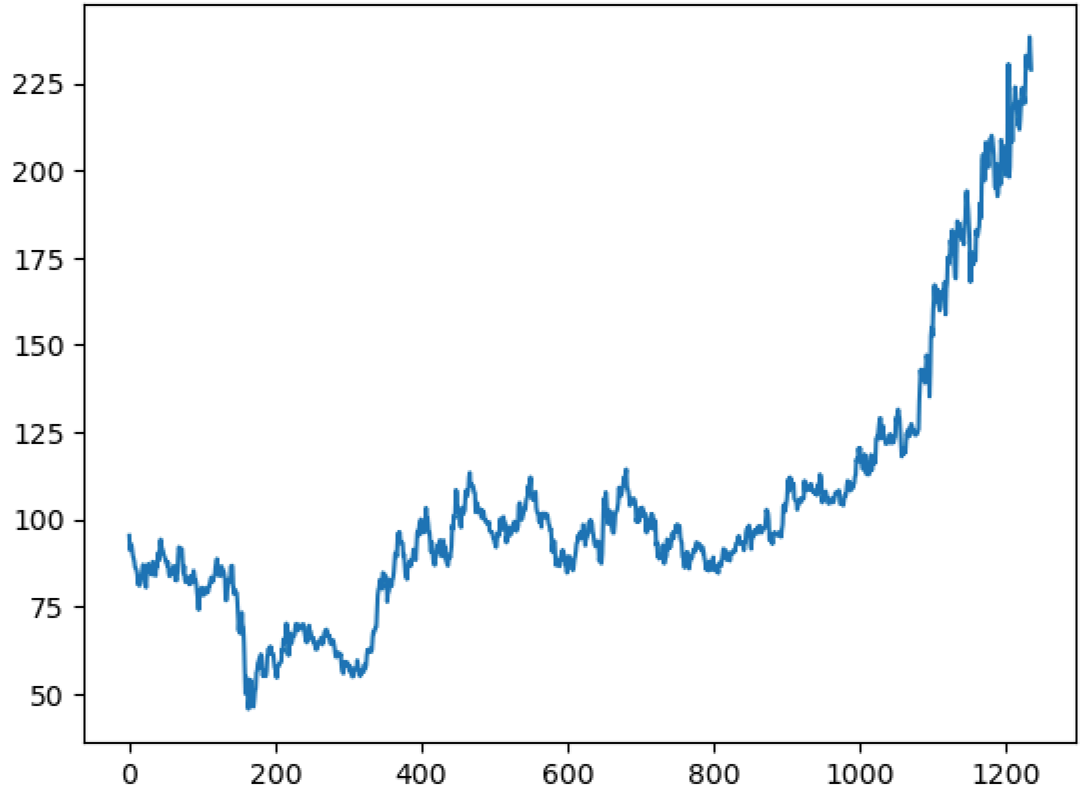
In [ ]:
```
ds=op.values
```

In [ ]:
```
ds
```

Out[ ]:
```
array([[ 95.30000305],
       [ 91.1333313 ],
       [ 91.96666718],
       ...,
       [238.19999695],
       [233.50999451],
       [228.86000061]])
```

In [ ]:
```
plt.plot(ds)
```

Out[ ]:
```
[<matplotlib.lines.Line2D at 0x21280528d50>]
```



In [ ]:
```
normal=MinMaxScaler(feature_range=(0,1))
ds_scale=normal.fit_transform(np.array(ds).reshape(-1,1))
```

In [ ]:
```
len(ds_scale),len(ds)
```

Out[ ]:
```
(1235, 1235)
```

Test and Train Data

In [ ]:
```
train_size=int(len(ds_scale)*0.70)
test_size=len(ds_scale)-train_size
```

Splitting Data

In [ ]:
```
ds_train, ds_test= ds_scale[0:train_size,:],ds_scale[train_size:len(ds_scale),:1]
```

In [ ]:
```
len(ds_train),len(ds_test)
```

Out[ ]:
```
(864, 371)
```

In [ ]:
```
def create_ds(dataset,step):
    Xtrain, Ytrain = [], []
    for i in range(len(dataset)-step-1):
        a = dataset[i:(i+step), 0]
        Xtrain.append(a)
        Ytrain.append(dataset[i + step, 0])
    return np.array(Xtrain), np.array(Ytrain)
```

In [ ]:
```
time_stamp = 100
X_train, y_train = create_ds(ds_train,time_stamp)
X_test, y_test = create_ds(ds_test,time_stamp)
```

In [ ]:
```
X_train.shape,y_train.shape,X_test.shape,y_test.shape
```

Out[ ]:
```
((763, 100), (763,), (270, 100), (270,))
```

In [ ]:
```
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)
```

In [ ]:
```
from tensorflow import keras
```

In [ ]:
```
from keras.models import Sequential
from keras.layers import Dense, LSTM
```

Creating LSTM

In [ ]:
```
model=Sequential()
model.add(LSTM(units=50,return_sequences=True,input_shape=(X_train.shape[1],1)))
model.add(LSTM(units=50,return_sequences=True))
model.add(LSTM(units=50))
model.add(Dense(units=1,activation='linear'))
model.summary()
```

```
c:\Users\Azeem ul Hassan\anaconda3\Lib\site-packages\keras\src\layers\rnn\rnn.py:204:
UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using
Sequential models, prefer using an `Input(shape)` object as the first layer in the mo
del instead.
  super().__init__(**kwargs)
```

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 100, 50)	10,400
lstm_1 (LSTM)	(None, 100, 50)	20,200
lstm_2 (LSTM)	(None, 50)	20,200
dense (Dense)	(None, 1)	51

Total params: 50,851 (198.64 KB)

Trainable params: 50,851 (198.64 KB)

Non-trainable params: 0 (0.00 B)

In [ ]:
```
model.compile(loss='mean_squared_error',optimizer='adam')
model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=100,batch_size=64)
```

```
Epoch 1/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 16s 385ms/step - loss: 0.0224 - val_loss: 0.0819
Epoch 2/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 323ms/step - loss: 0.0037 - val_loss: 0.0224
Epoch 3/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 323ms/step - loss: 0.0024 - val_loss: 0.0258
Epoch 4/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 163ms/step - loss: 0.0013 - val_loss: 0.0150
Epoch 5/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 161ms/step - loss: 9.7883e-04 - val_loss: 0.0051
Epoch 6/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 181ms/step - loss: 8.5168e-04 - val_loss: 0.0040
Epoch 7/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 158ms/step - loss: 9.0616e-04 - val_loss: 0.0041
Epoch 8/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 165ms/step - loss: 9.2887e-04 - val_loss: 0.0043
Epoch 9/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 158ms/step - loss: 8.0452e-04 - val_loss: 0.0038
Epoch 10/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 158ms/step - loss: 7.7905e-04 - val_loss: 0.0038
Epoch 11/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 158ms/step - loss: 8.4984e-04 - val_loss: 0.0039
Epoch 12/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 168ms/step - loss: 6.9650e-04 - val_loss: 0.0036
Epoch 13/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 168ms/step - loss: 7.5905e-04 - val_loss: 0.0040
Epoch 14/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 166ms/step - loss: 6.8679e-04 - val_loss: 0.0045
Epoch 15/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 203ms/step - loss: 7.3482e-04 - val_loss: 0.0022
Epoch 16/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 214ms/step - loss: 6.8421e-04 - val_loss: 0.0036
Epoch 17/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 245ms/step - loss: 7.0366e-04 - val_loss: 0.0053
Epoch 18/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 6s 314ms/step - loss: 7.3917e-04 - val_loss: 0.0036
Epoch 19/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 258ms/step - loss: 6.5950e-04 - val_loss: 0.0047
Epoch 20/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 204ms/step - loss: 6.3462e-04 - val_loss: 0.0022
Epoch 21/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 196ms/step - loss: 6.5293e-04 - val_loss: 0.0053
Epoch 22/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 191ms/step - loss: 6.3904e-04 - val_loss: 0.0034
Epoch 23/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 177ms/step - loss: 5.8732e-04 - val_loss: 0.0035
Epoch 24/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 222ms/step - loss: 6.1154e-04 - val_loss: 0.0047
Epoch 25/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 263ms/step - loss: 6.3784e-04 - val_loss: 0.0044
Epoch 26/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 200ms/step - loss: 5.8196e-04 - val_loss: 0.0047
Epoch 27/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 211ms/step - loss: 5.0717e-04 - val_loss: 0.0022
Epoch 28/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 389ms/step - loss: 5.4950e-04 - val_loss: 0.0056
Epoch 29/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 284ms/step - loss: 5.9448e-04 - val_loss: 0.0027
Epoch 30/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 7s 410ms/step - loss: 5.5364e-04 - val_loss: 0.0054
```

```
Epoch 31/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 212ms/step - loss: 5.4896e-04 - val_loss: 0.0024
Epoch 32/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 186ms/step - loss: 5.4673e-04 - val_loss: 0.0056
Epoch 33/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 180ms/step - loss: 5.4275e-04 - val_loss: 0.0050
Epoch 34/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 184ms/step - loss: 5.1164e-04 - val_loss: 0.0039
Epoch 35/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 204ms/step - loss: 5.0863e-04 - val_loss: 0.0053
Epoch 36/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 220ms/step - loss: 5.0817e-04 - val_loss: 0.0032
Epoch 37/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 216ms/step - loss: 4.7580e-04 - val_loss: 0.0040
Epoch 38/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 169ms/step - loss: 4.7589e-04 - val_loss: 0.0061
Epoch 39/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 171ms/step - loss: 4.8864e-04 - val_loss: 0.0050
Epoch 40/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 171ms/step - loss: 4.8246e-04 - val_loss: 0.0044
Epoch 41/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 181ms/step - loss: 4.2887e-04 - val_loss: 0.0034
Epoch 42/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 220ms/step - loss: 4.3482e-04 - val_loss: 0.0045
Epoch 43/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 177ms/step - loss: 4.2297e-04 - val_loss: 0.0048
Epoch 44/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 188ms/step - loss: 4.6553e-04 - val_loss: 0.0041
Epoch 45/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 7s 597ms/step - loss: 4.3904e-04 - val_loss: 0.0043
Epoch 46/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 6s 466ms/step - loss: 3.8525e-04 - val_loss: 0.0035
Epoch 47/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 287ms/step - loss: 4.5814e-04 - val_loss: 0.0057
Epoch 48/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 6s 400ms/step - loss: 4.4302e-04 - val_loss: 0.0032
Epoch 49/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 367ms/step - loss: 4.4647e-04 - val_loss: 0.0065
Epoch 50/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 386ms/step - loss: 4.2523e-04 - val_loss: 0.0037
Epoch 51/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 6s 428ms/step - loss: 3.8757e-04 - val_loss: 0.0052
Epoch 52/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 317ms/step - loss: 3.6367e-04 - val_loss: 0.0057
Epoch 53/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 272ms/step - loss: 3.6159e-04 - val_loss: 0.0067
Epoch 54/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 223ms/step - loss: 3.4770e-04 - val_loss: 0.0039
Epoch 55/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 218ms/step - loss: 3.6452e-04 - val_loss: 0.0066
Epoch 56/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 250ms/step - loss: 3.5833e-04 - val_loss: 0.0042
Epoch 57/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 218ms/step - loss: 3.1410e-04 - val_loss: 0.0056
Epoch 58/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 279ms/step - loss: 3.8718e-04 - val_loss: 0.0061
Epoch 59/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 8s 454ms/step - loss: 3.4767e-04 - val_loss: 0.0044
Epoch 60/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 400ms/step - loss: 3.0757e-04 - val_loss: 0.0028
```

```
Epoch 61/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 6s 485ms/step - loss: 3.5993e-04 - val_loss: 0.0064
Epoch 62/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 9s 347ms/step - loss: 3.3325e-04 - val_loss: 0.0048
Epoch 63/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 6s 397ms/step - loss: 3.1219e-04 - val_loss: 0.0047
Epoch 64/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 369ms/step - loss: 3.1529e-04 - val_loss: 0.0045
Epoch 65/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 405ms/step - loss: 2.8375e-04 - val_loss: 0.0037
Epoch 66/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 358ms/step - loss: 2.4896e-04 - val_loss: 0.0040
Epoch 67/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 355ms/step - loss: 2.9334e-04 - val_loss: 0.0033
Epoch 68/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 360ms/step - loss: 2.7430e-04 - val_loss: 0.0052
Epoch 69/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 411ms/step - loss: 2.7154e-04 - val_loss: 0.0064
Epoch 70/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 330ms/step - loss: 2.9646e-04 - val_loss: 0.0043
Epoch 71/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 239ms/step - loss: 2.6945e-04 - val_loss: 0.0032
Epoch 72/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 181ms/step - loss: 2.7846e-04 - val_loss: 0.0035
Epoch 73/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 208ms/step - loss: 2.4245e-04 - val_loss: 0.0031
Epoch 74/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 266ms/step - loss: 2.4062e-04 - val_loss: 0.0020
Epoch 75/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 5s 250ms/step - loss: 2.7908e-04 - val_loss: 0.0022
Epoch 76/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 261ms/step - loss: 2.8565e-04 - val_loss: 0.0015
Epoch 77/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 237ms/step - loss: 3.0162e-04 - val_loss: 0.0027
Epoch 78/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 166ms/step - loss: 2.5437e-04 - val_loss: 0.0050
Epoch 79/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 243ms/step - loss: 2.3094e-04 - val_loss: 0.0031
Epoch 80/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 202ms/step - loss: 2.1505e-04 - val_loss: 0.0052
Epoch 81/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 176ms/step - loss: 2.7510e-04 - val_loss: 0.0027
Epoch 82/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 159ms/step - loss: 2.7141e-04 - val_loss: 0.0026
Epoch 83/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 168ms/step - loss: 2.4836e-04 - val_loss: 0.0023
Epoch 84/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 198ms/step - loss: 2.6811e-04 - val_loss: 0.0014
Epoch 85/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 190ms/step - loss: 2.3914e-04 - val_loss: 0.0017
Epoch 86/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 213ms/step - loss: 2.4569e-04 - val_loss: 0.0015
Epoch 87/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 177ms/step - loss: 2.7262e-04 - val_loss: 0.0017
Epoch 88/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 242ms/step - loss: 2.5003e-04 - val_loss: 0.0027
Epoch 89/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 169ms/step - loss: 2.4510e-04 - val_loss: 0.0028
Epoch 90/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 219ms/step - loss: 2.2411e-04 - val_loss: 0.0021
```

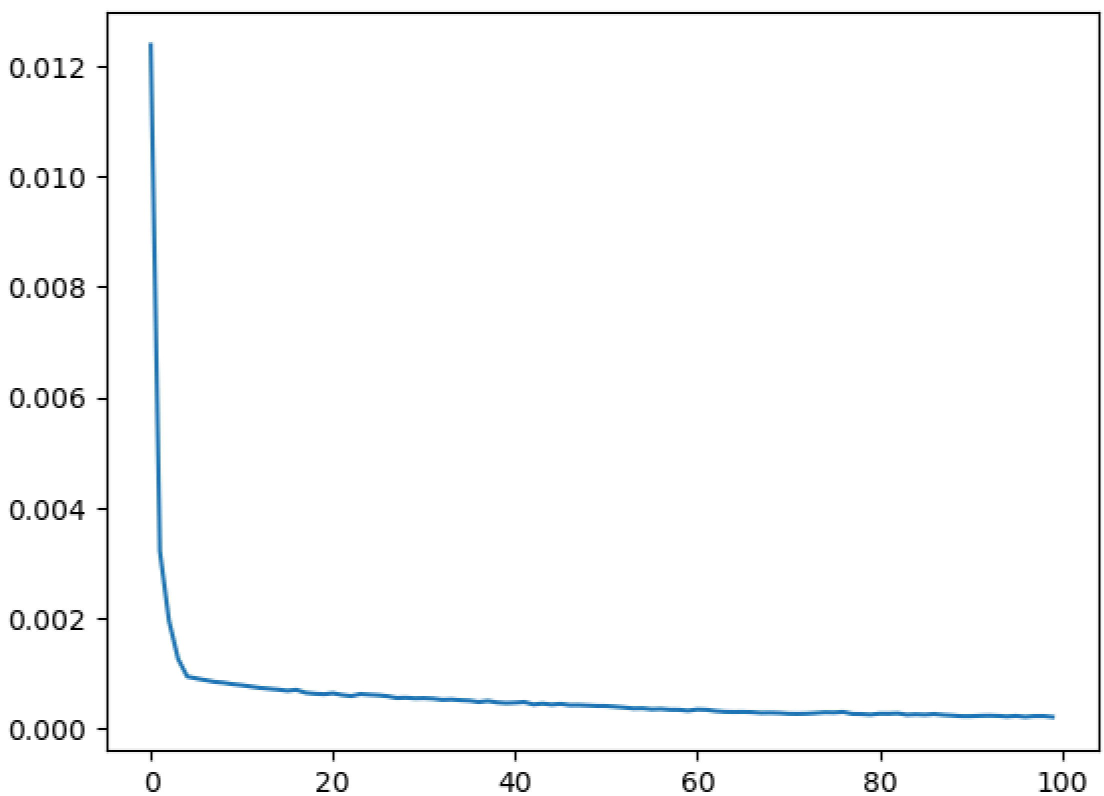
```
Epoch 91/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 4s 350ms/step - loss: 1.9231e-04 - val_loss: 0.0021
Epoch 92/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 179ms/step - loss: 2.0734e-04 - val_loss: 0.0028
Epoch 93/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 231ms/step - loss: 2.2923e-04 - val_loss: 0.0023
Epoch 94/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 277ms/step - loss: 2.2304e-04 - val_loss: 0.0022
Epoch 95/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 203ms/step - loss: 2.0720e-04 - val_loss: 0.0031
Epoch 96/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 174ms/step - loss: 2.2161e-04 - val_loss: 0.0024
Epoch 97/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 173ms/step - loss: 2.3186e-04 - val_loss: 0.0017
Epoch 98/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 184ms/step - loss: 2.3621e-04 - val_loss: 0.0010
Epoch 99/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 177ms/step - loss: 2.2916e-04 - val_loss: 0.0011
Epoch 100/100
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 172ms/step - loss: 2.2927e-04 - val_loss: 0.0019
```

Out[ ]: `<keras.src.callbacks.history.History at 0x21280619710>`

In [ ]:

```
loss=model.history.history['loss']
plt.plot(loss)
```

Out[ ]: `[<matplotlib.lines.Line2D at 0x21289c96310>]`



In [ ]:

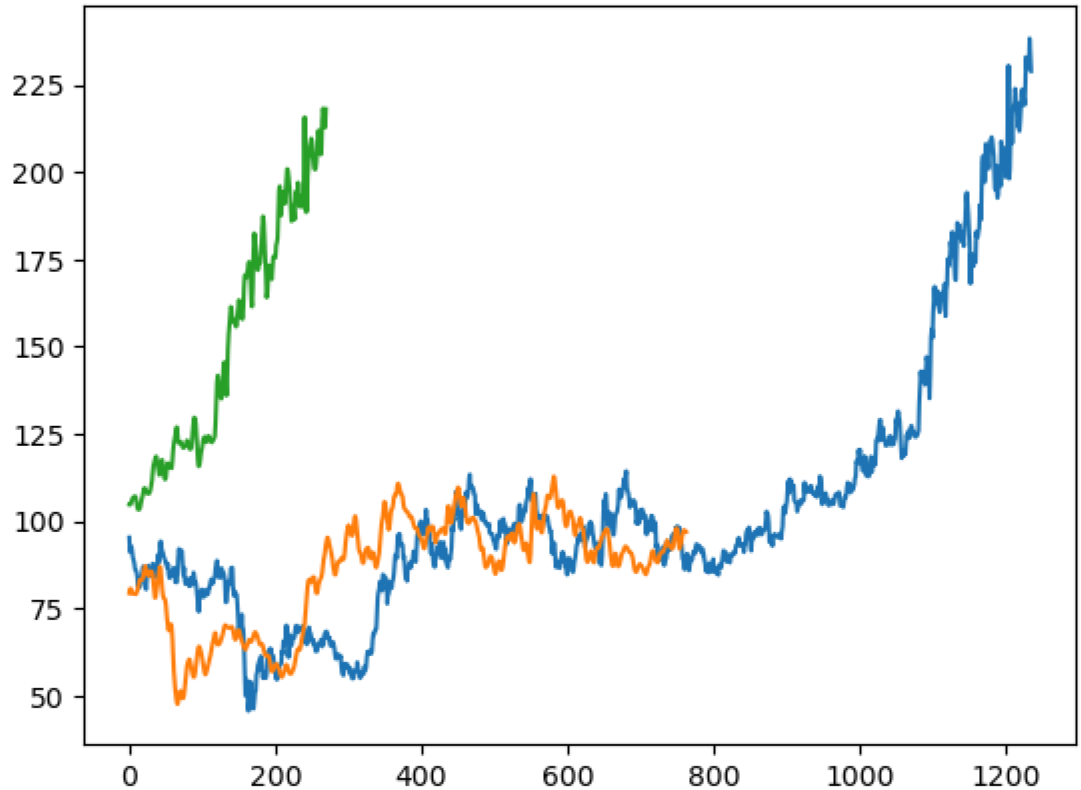
```
train_predict=model.predict(X_train)
test_predict=model.predict(X_test)
```

```
24/24 ━━━━━━━━━━━━━━━━━━━━ 3s 97ms/step
9/9 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step
```

In [ ]:
```
train_predict = normal.inverse_transform(train_predict)
test_predict = normal.inverse_transform(test_predict)
```

In [ ]:
```
plt.plot(normal.inverse_transform(ds_scale))
plt.plot(train_predict)
plt.plot(test_predict)
```

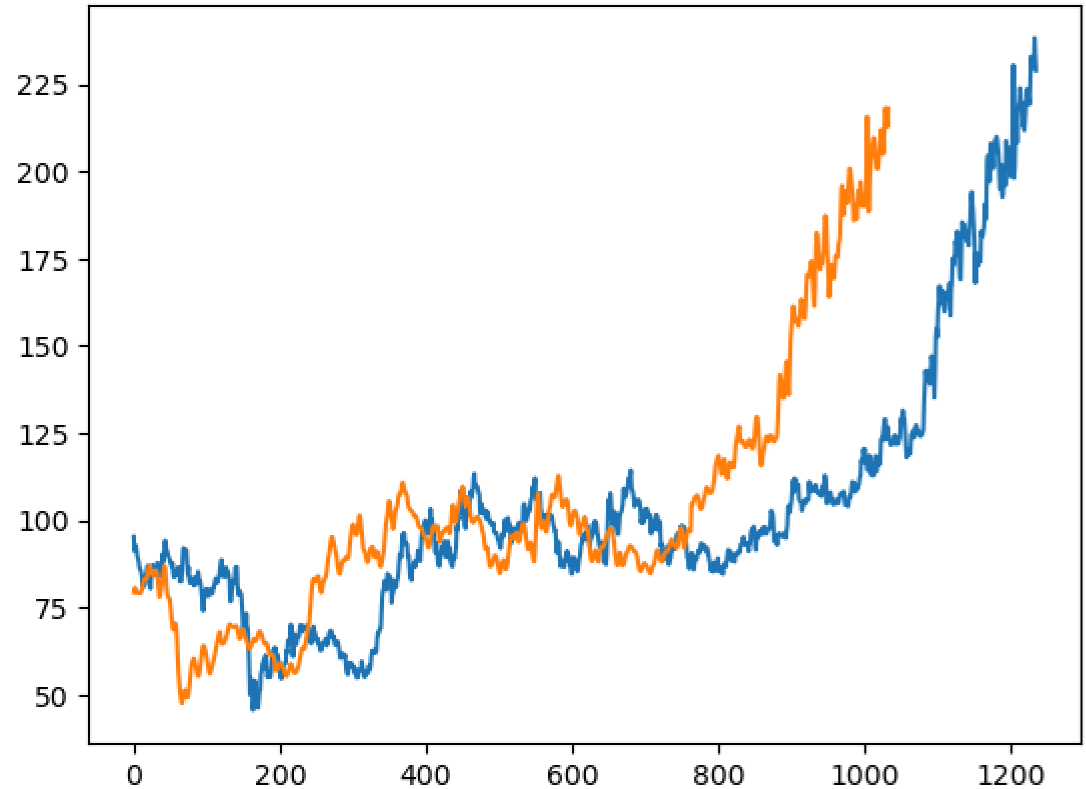
Out[ ]:
```
[<matplotlib.lines.Line2D at 0x2129485d110>]
```



In [ ]:
```
test=np.vstack((train_predict,test_predict))
```

In [ ]:
```
plt.plot(normal.inverse_transform(ds_scale))
plt.plot(test)
```

Out[ ]:
```
[<matplotlib.lines.Line2D at 0x2128c744190>]
```



In [ ]:
```
len(ds_test)
```
Out[ ]:
```
371
```

In [ ]:
```
future=ds_test[270:]
```

In [ ]:
```
future=future.reshape(1,-1)
```

In [ ]:
```
tmp=list(future)
```

In [ ]:
```
future.shape
```
Out[ ]:
```
(1, 101)
```

In [ ]:
```
tmp=tmp[0].tolist()
```

In [ ]:
```
lst_output=[]
n_steps=100
i=0
while(i<30):

    if(len(tmp)>100):
        future = np.array(tmp[1:])
        future=future.reshape(1,-1)
        future = future.reshape((1, n_steps, 1))
        yhat = model.predict(future, verbose=0)
        tmp.extend(yhat[0].tolist())
        tmp = tmp[1:]
        lst_output.extend(yhat.tolist())
        i=i+1
    else:
```

```
        future = future.reshape((1, n_steps,1))
        yhat = model.predict(future, verbose=0)
        tmp.extend(yhat[0].tolist())
        lst_output.extend(yhat.tolist())
        i=i+1


print(lst_output)
```

```
[[0.8826842308044434], [0.8259167671203613], [0.7638117671012878], [0.712570488452911
4], [0.6778731346130371], [0.6601383090019226], [0.6564286351203918], [0.661698877811
4319], [0.6700546145439148], [0.675944447517395], [0.6751735210418701], [0.6655706763
267517], [0.6471671462059021], [0.6218885183334351], [0.5928829908370972], [0.5636612
772941589], [0.5372721552848816], [0.515731930732727], [0.4998249411582947], [0.48923
6056804657], [0.4828683137893677], [0.4792052209377289], [0.47663578391075134], [0.47
37176299095154], [0.46936631202697754], [0.46296465396881104], [0.45438718795776367],
[0.4439403712749481], [0.4322397708892822], [0.4200502336025238]]
```

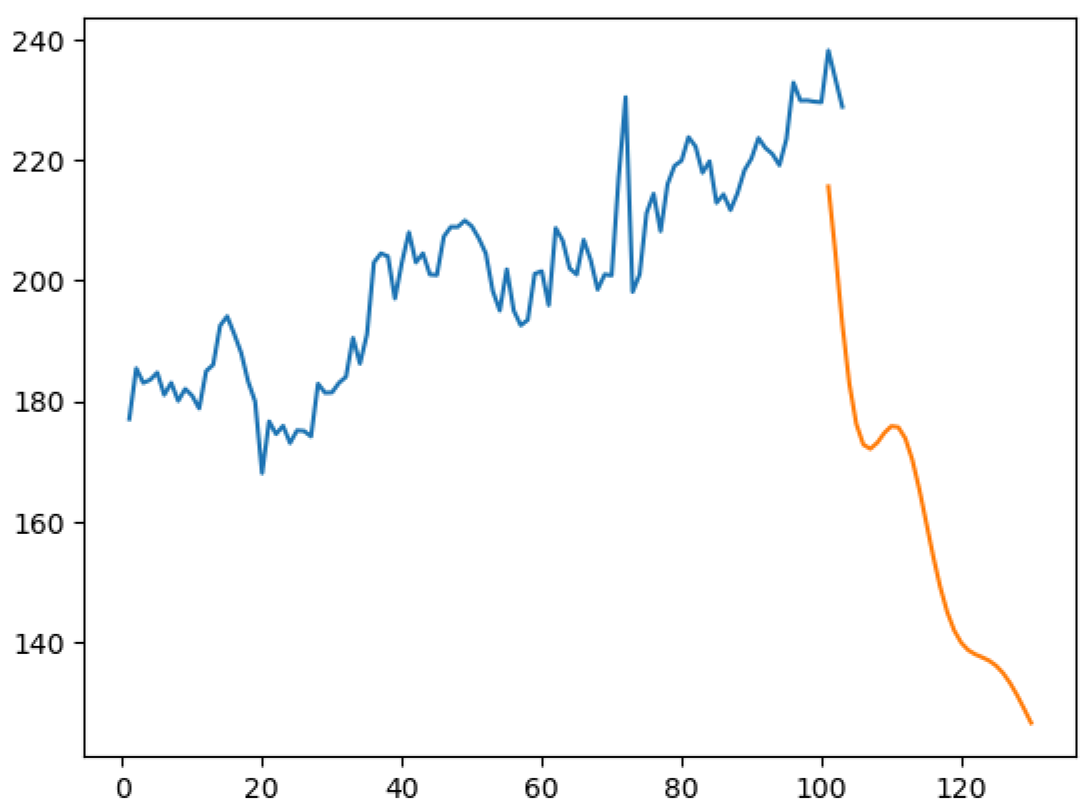
In [ ]:
```
len(ds_scale)
```

Out[ ]:
```
1235
```

In [ ]:
```
plot_new=np.arange(1,104)
plot_pred=np.arange(101,131)
```

In [ ]:
```
plt.plot(plot_new, normal.inverse_transform(ds_scale[1132:]))
plt.plot(plot_pred, normal.inverse_transform(lst_output))
```

Out[ ]:
```
[<matplotlib.lines.Line2D at 0x21294aceb50>]
```



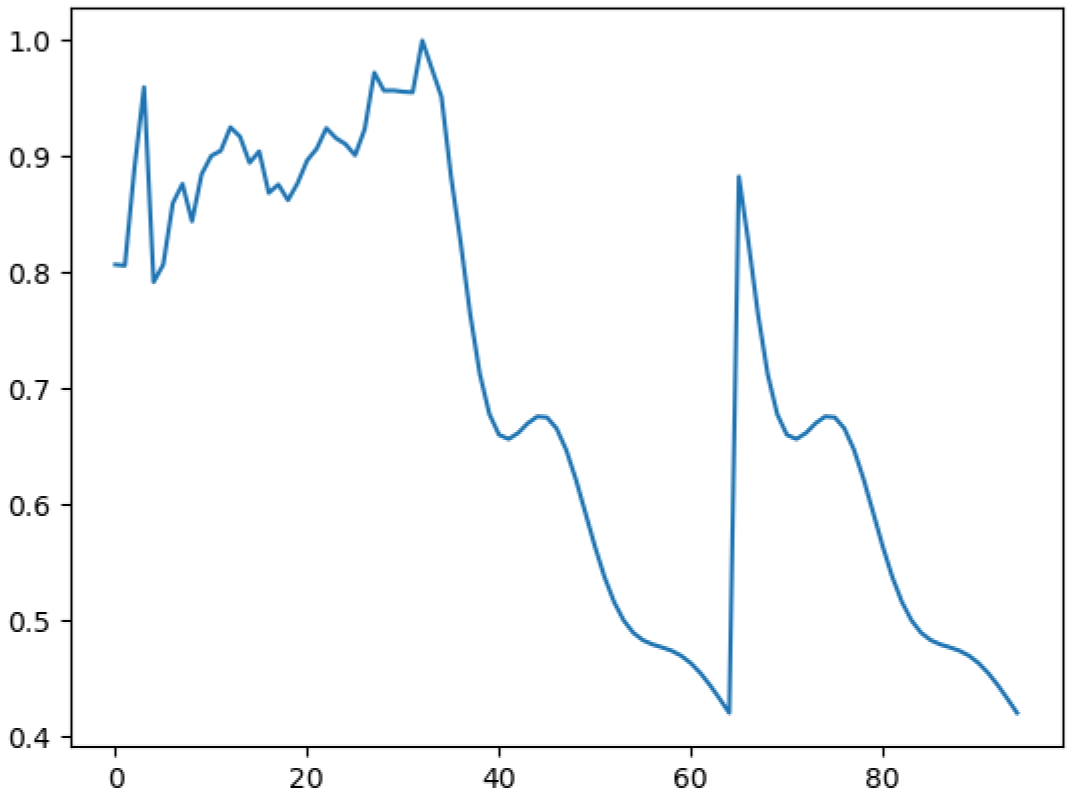
In [ ]:
```
dsnew=ds_scale.tolist()
```

In [ ]:
```
len(dsnew)
```

Out[ ]:
```
1235
```

In [ ]:
```
dsnew.extend(lst_output)
plt.plot(dsnew[1200:])
```

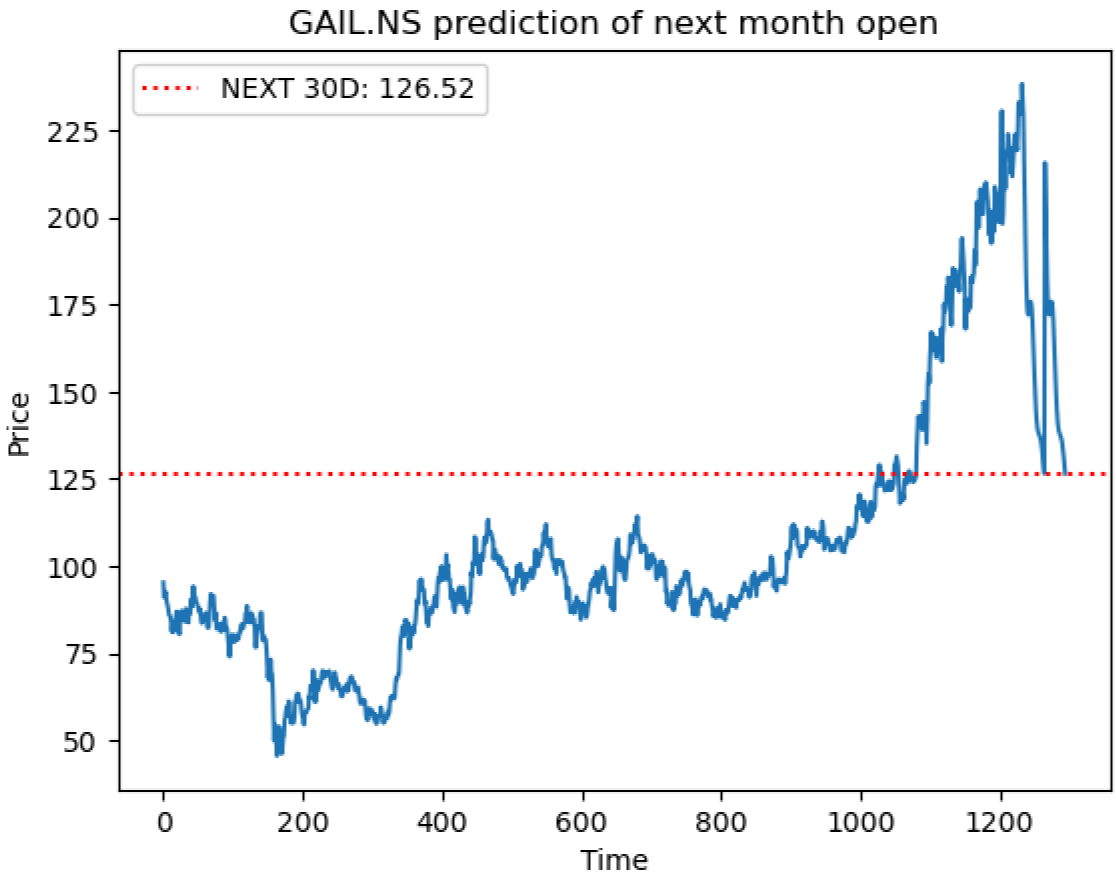
Out[ ]:
```
[<matplotlib.lines.Line2D at 0x21292788d50>]
```



In [ ]:
```
final_graph = normal.inverse_transform(dsnew).tolist()
```

In [ ]:
```
plt.plot(final_graph,)
plt.ylabel("Price")
plt.xlabel("Time")
plt.title("{0} prediction of next month open".format(stocksymbol))
plt.axhline(y=final_graph[len(final_graph)-1], color = 'red', linestyle = ':', label =
plt.legend()
```

Out[ ]:
```
<matplotlib.legend.Legend at 0x21294c173d0>
```


GAIL.NS prediction of next month open
NEXT 30D: 126.52
Price
Time
225
200
175
150
125
100
75
50
0
200
400
600
800
1000
1200