Stock Market Prediction And Forecasting Using Stacked LSTM

Steps included:-

1.Importing packages

2.Data Normalization

3.Incorporating Timesteps Into Data

3.Creating the LSTM Model

4.Making Predictions on the Test Set

5.Plotting the Results

Importing packages

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dropout
```

In [2]:
```
dataset_train = pd.read_csv('https://raw.githubusercontent.com/mwitiderrick/stockprice/master/NSE-TATAGLOBAL.csv')
training_set = dataset_train.iloc[:, 1:2].values
```

In [3]:
```
dataset_train.head()
```

Out[3]:

	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

Data Normalization

Since LSTM is very sensitive to the scale of the data,the scale of the Close value is in a kind of scale, we should always try to transform the value. Here we will use min-max scalar to transform the values from 0 to 1

In [4]:
```
sc = MinMaxScaler(feature_range=(0,1))
training_set_scaled = sc.fit_transform(training_set)
```

Incorporating Timesteps Into Data

In [5]:
```
X_train = []
y_train = []
for i in range(60, 2035):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

Creating the LSTM Model

In [6]:
```
model = Sequential()
model.add(LSTM(units=50,return_sequences=True,input_shape=(X_train.shape[1], 1)))
model.add(Dropout(0.2))
model.add(LSTM(units=50,return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(units=50,return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(units=50))
model.add(Dropout(0.2))
model.add(Dense(units=1))
model.compile(optimizer='adam',loss='mean_squared_error')
model.fit(X_train,y_train,epochs=100,batch_size=32)
```

```
Epoch 1/100
62/62 [==============================] - 9s 57ms/step - loss: 0.0115
Epoch 2/100
62/62 [==============================] - 4s 57ms/step - loss: 0.0035
Epoch 3/100
62/62 [==============================] - 3s 56ms/step - loss: 0.0028
Epoch 4/100
62/62 [==============================] - 4s 57ms/step - loss: 0.0024
Epoch 5/100
62/62 [==============================] - 4s 56ms/step - loss: 0.0023
Epoch 6/100
62/62 [==============================] - 4s 58ms/step - loss: 0.0025
Epoch 7/100
62/62 [==============================] - 4s 59ms/step - loss: 0.0025
Epoch 8/100
62/62 [==============================] - 4s 57ms/step - loss: 0.0020
Epoch 9/100
62/62 [==============================] - 3s 56ms/step - loss: 0.0020
Epoch 10/100
```

Making Predictions on the Test Set

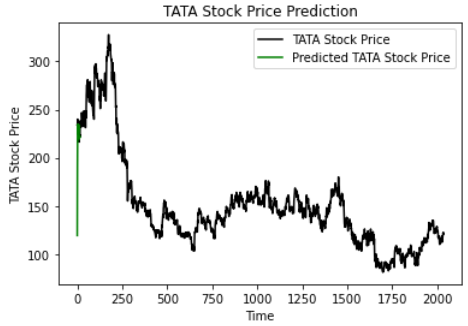
In [8]:
```
dataset_test = pd.read_csv('https://raw.githubusercontent.com/mwitiderrick/stockprice/master/NSE-TATAGLOBAL.csv')
real_stock_price = dataset_test.iloc[:, 1:2].values
```

In [9]:
```
dataset_total = pd.concat((dataset_train['Open'], dataset_test['Open']), axis = 0)
inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 76):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
predicted_stock_price = model.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

```
1/1 [==============================] - 1s 1s/step
```

Plotting the Results

In [10]:
```
plt.plot(real_stock_price, color = 'black', label = 'TATA Stock Price')
plt.plot(predicted_stock_price, color = 'green', label = 'Predicted TATA Stock Price')
plt.title('TATA Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('TATA Stock Price')
plt.legend()
plt.show()
```



In [ ]: