Suggested code may be subject to a license | Matt602/kaggle_breast_cancer_wisconsin | SuperMindu/study | friha438/MSc_MT

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout
import math
from sklearn.metrics import mean_squared_error
```

```
data = pd.read_csv('/content/IBM_2006-01-01_to_2018-01-01.csv',index_col='Date',parse_dates=['Date'])
data.head()
```

	Open	High	Low	Close	Volume	Name
Date						
2006-01-03	82.45	82.55	80.81	82.06	11715200	IBM
2006-01-04	82.20	82.50	81.33	81.95	9840600	IBM
2006-01-05	81.40	82.90	81.00	82.50	7213500	IBM
2006-01-06	83.95	85.03	83.41	84.95	8197400	IBM
2006-01-09	84.10	84.25	83.38	83.73	6858200	IBM

Next steps: Generate code with data View recommended plots

```
mytrain = data[:'2016'].iloc[:,1:2].values# selecting the all rows and selecting column
mytest = data['2017':].iloc[:,1:2].values#trie to predict all rows and columns after 2017
```

```
#scaling the training set

sc = MinMaxScaler(feature_range=(0,1))       #minmaxsacler(feature_range=(start,stop))
mytrain_scaled = sc.fit_transform(mytrain)   #instances.fit_transform(data)
```

```
mytrain_scaled #view the scales values
```

```
array([[0.06065089],
       [0.06029868],
       [0.06311637],
       ...,
       [0.66074951],
       [0.65546633],
       [0.6534235 ]])
```

```
len(mytrain_scaled)
```

```
2769
```

```
I_train = []
O_train = []
for i in range(60,len(mytrain_scaled)): #(60,2769)
  I_train.append(mytrain_scaled[i-60:i,0])
  O_train.append(mytrain_scaled[i,0])
```

```
I_train[0]
```

```
array([0.06065089, 0.06029868, 0.06311637, 0.0781206 , 0.07262609,
       0.07171034, 0.07657087, 0.07058326, 0.0669907 , 0.06494787,
       0.075796  , 0.07361229, 0.06417301, 0.05621302, 0.05783319,
       0.05409975, 0.05431107, 0.05515638, 0.05543815, 0.05677656,
       0.05846717, 0.05388842, 0.04811214, 0.04233587, 0.04402649,
       0.0490279 , 0.04832347, 0.05297267, 0.05614258, 0.05290223,
       0.05325444, 0.04909834, 0.04994365, 0.04797126, 0.05431107,
       0.05212736, 0.04726684, 0.04895745, 0.04656241, 0.04839391,
       0.04416737, 0.0485348 , 0.04719639, 0.04825303, 0.05395886,
       0.05663567, 0.05853762, 0.05959425, 0.06375035, 0.06917442,
       0.06889265, 0.06670893, 0.06910397, 0.07783883, 0.07565511,
       0.07276698, 0.06889265, 0.0656523 , 0.06656805, 0.06769513])
```

```
O_train[0]
```

```
0.06875176105945335
```

```
I_train= np.array(I_train)
O_train =np.array(O_train)
```

```
I_train.shape
```

```
(2709, 60)
```

```
I_train = I_train.reshape(2709, 60,1)
```

```
I_train.shape
```

```
(2709, 60, 1)
```

```
model = Sequential()

#first LSTM layer
model.add(LSTM(units=50, return_sequences=True, input_shape=(60,1)))
model.add(Dropout(0.2))

# second lstm layer
model.add(LSTM(units=50, return_sequences=True))
model.add(Dropout(0.2))

#third lstm layer
model.add(LSTM(units=50, return_sequences=True))
model.add(Dropout(0.2))

#4th lstm layer
model.add(LSTM(units=50))
model.add(Dropout(0.2))

# the o/p layer
model.add(Dense(units=1))
```

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

```
model.fit(I_train, O_train, epochs=50, batch_size=32)
```

```
Epoch 1/50
85/85 [==============================] - 16s 136ms/step - loss: 0.0016
Epoch 2/50
85/85 [==============================] - 12s 141ms/step - loss: 0.0016
Epoch 3/50
85/85 [==============================] - 11s 130ms/step - loss: 0.0016
Epoch 4/50
85/85 [==============================] - 11s 123ms/step - loss: 0.0015
Epoch 5/50
85/85 [==============================] - 11s 134ms/step - loss: 0.0015
Epoch 6/50
85/85 [==============================] - 12s 137ms/step - loss: 0.0014
Epoch 7/50
85/85 [==============================] - 13s 153ms/step - loss: 0.0016
Epoch 8/50
85/85 [==============================] - 11s 133ms/step - loss: 0.0015
Epoch 9/50
85/85 [==============================] - 11s 125ms/step - loss: 0.0015
Epoch 10/50
85/85 [==============================] - 11s 128ms/step - loss: 0.0015
Epoch 11/50
85/85 [==============================] - 13s 150ms/step - loss: 0.0014
Epoch 12/50
85/85 [==============================] - 12s 144ms/step - loss: 0.0014
Epoch 13/50
85/85 [==============================] - 12s 143ms/step - loss: 0.0014
Epoch 14/50
85/85 [==============================] - 12s 143ms/step - loss: 0.0015
Epoch 15/50
85/85 [==============================] - 13s 150ms/step - loss: 0.0013
Epoch 16/50
85/85 [==============================] - 13s 150ms/step - loss: 0.0014
Epoch 17/50
85/85 [==============================] - 12s 137ms/step - loss: 0.0014
Epoch 18/50
```

```
85/85 [==============================] - 13s 158ms/step - loss: 0.0014
Epoch 19/50
85/85 [==============================] - 11s 123ms/step - loss: 0.0013
Epoch 20/50
85/85 [==============================] - 11s 134ms/step - loss: 0.0014
Epoch 21/50
85/85 [==============================] - 13s 152ms/step - loss: 0.0013
Epoch 22/50
85/85 [==============================] - 11s 135ms/step - loss: 0.0013
Epoch 23/50
85/85 [==============================] - 11s 133ms/step - loss: 0.0014
Epoch 24/50
85/85 [==============================] - 11s 134ms/step - loss: 0.0013
Epoch 25/50
85/85 [==============================] - 11s 124ms/step - loss: 0.0014
Epoch 26/50
85/85 [==============================] - 11s 128ms/step - loss: 0.0013
Epoch 27/50
85/85 [==============================] - 12s 139ms/step - loss: 0.0013
Epoch 28/50
85/85 [==============================] - 13s 154ms/step - loss: 0.0012
Epoch 29/50
85/85 [==============================] - 12s 142ms/step - loss: 0.0013
```

```
model.save('IBM.h5')
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `m
  saving_api.save_model(
```

```
723, 83.24620825, 81.77694404, 83.20275117, 83.43996357, 83.38335394, 82.69282925, 80.80822433, 80.30044039, 83.6842126, 82.82960781, 80.1242
```

```
WARNING:tensorflow:5 out of the last 5 calls to <function Model.make_predict_function.<locals>.predict_function at 0x7be50e93af80> trigg
1/1 [==============================] - 2s 2s/step
[[-0.03790367]]
The stock price is 79.97947
```