#Nile Pallavi Roll NO: 4217 Div:B
 # DL Practical NO.4

```
import pandas as pd
import numpy as np
train_df = pd.read_csv(r'C:\Users\Pallavi
Nile\Downloads\DL\pract_4\Google_Stock_Price_Train.csv')
print(train_df)
test_df = pd.read_csv(r'C:\Users\Pallavi
Nile\Downloads\DL\pract_4\Google_Stock_Price_Test.csv')
print(test_df)
print( test_df.info())

#Data preprocessing
from sklearn.preprocessing import MinMaxScaler
# Convert 'Close' column to string type and remove commas
train_df['Close'] = train_df['Close'].astype(str).str.replace(',', '').astype(float)
test_df['Close'] = test_df['Close'].astype(str).str.replace(',', '').astype(float)
# Normalize the training and testing data separately
train_scaler = MinMaxScaler()
train_df['Normalized Close'] = train_scaler.fit_transform(train_df['Close'].
values.reshape(-1, 1))
test_scaler = MinMaxScaler()
test_df['Normalized Close'] = test_scaler.fit_transform(test_df['Close'].values.
reshape(-1, 1))

# Convert the data to the appropriate format for RNN
x_train = train_df['Normalized Close'].values[:-1].reshape(-1, 1, 1)
y_train = train_df['Normalized Close'].values[1:].reshape(-1, 1, 1)
x_test = test_df['Normalized Close'].values[:-1].reshape(-1, 1, 1)
y_test = test_df['Normalized Close'].values[1:].reshape(-1, 1, 1)

print("x_train shape: ",x_train.shape)
print("y_train shape: ",y_train.shape)
print("x_test shape: ",x_test.shape)
print("y_test shape: ",y_test.shape)

print(train_df)
```

```
print(test_df)
print(test_df.info())

from keras.models import Sequential
from keras.layers import LSTM, Dense
model = Sequential()
model.add(LSTM(4, input_shape=(1, 1)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
model.summary()

#model training
model.fit(x_train, y_train, epochs=100, batch_size=1, verbose=1)

#model evaluation
test_loss = model.evaluate(x_test, y_test)
print('Testing loss: ', test_loss)

#model testing
y_pred = model.predict(x_test)
# Inverse transform the normalized values to get the actual values
y_test_actual = test_scaler.inverse_transform(y_test.reshape(-1, 1))
y_pred_actual = test_scaler.inverse_transform(y_pred.reshape(-1, 1))
i=1
print("Actual value: {:.2f}".format(y_test_actual[i][0]))
print("Predicted value: {:.2f}".format(y_pred_actual[i][0]))
```

Output:

```
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Date             20 non-null     object
 1   Open             20 non-null     float64
 2   High             20 non-null     float64
 3   Low              20 non-null     float64
 4   Close            20 non-null     float64
 5   Volume           20 non-null     object
 6   Normalized Close  20 non-null     float64
```

dtypes: float64(5), object(2)
memory usage: 1.2+ KB
None
Model: "sequential_3"

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_2 (LSTM)               (None, 4)                 96

 dense_5 (Dense)             (None, 1)                 5

=================================================================
Total params: 101
Trainable params: 101
Non-trainable params: 0
_________________________________________________________________
Epoch 1/100
1257/1257 [==============================] - 3s 1ms/step - loss: 0.0280
Epoch 2/100
1257/1257 [==============================] - 2s 1ms/step - loss: 0.0014
Epoch 3/100
1257/1257 [==============================] - 1s 1ms/step - loss: 7.5121e-04
Epoch 4/100
1257/1257 [==============================] - 1s 1ms/step - loss: 7.5840e-04
Epoch 5/100
1257/1257 [==============================] - 1s 1ms/step - loss: 7.2355e-04
Epoch 6/100
1257/1257 [==============================] - 1s 1ms/step - loss: 8.0052e-04
Epoch 7/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5993e-04
Epoch 8/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.7384e-04
Epoch 9/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5678e-04
Epoch 10/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5570e-04
Epoch 11/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6202e-04
```

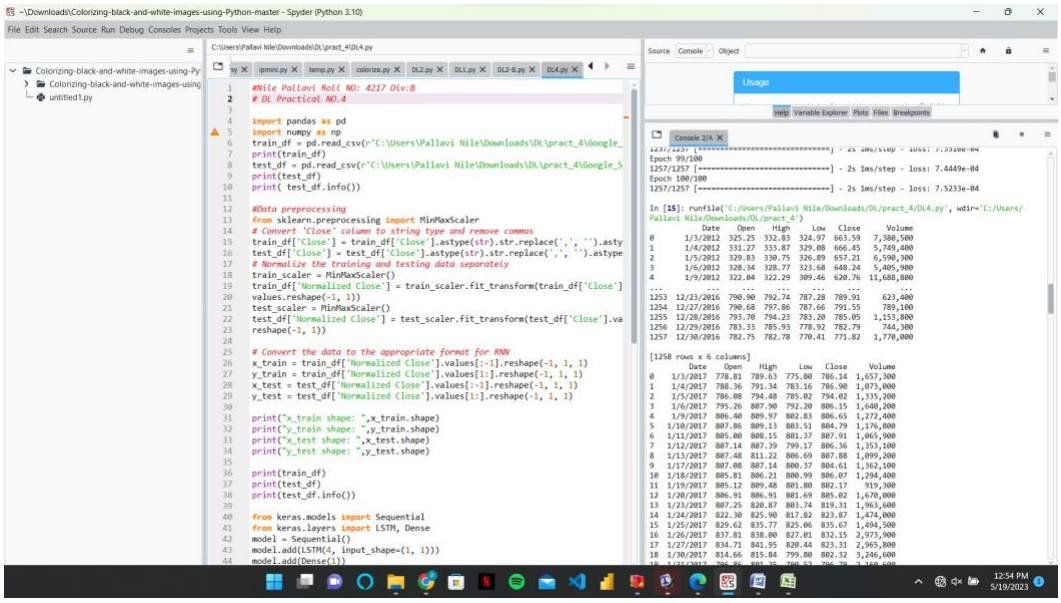
Epoch 12/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6457e-04
Epoch 13/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5565e-04
Epoch 14/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5255e-04
Epoch 15/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6610e-04
Epoch 16/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6055e-04
Epoch 17/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5997e-04
Epoch 18/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5733e-04
Epoch 19/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6517e-04
Epoch 20/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4971e-04
Epoch 21/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6198e-04
Epoch 22/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4764e-04
Epoch 23/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6130e-04
Epoch 24/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6126e-04
Epoch 25/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4789e-04
Epoch 26/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.7258e-04
Epoch 27/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5755e-04
Epoch 28/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6110e-04
Epoch 29/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6231e-04
Epoch 30/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4707e-04

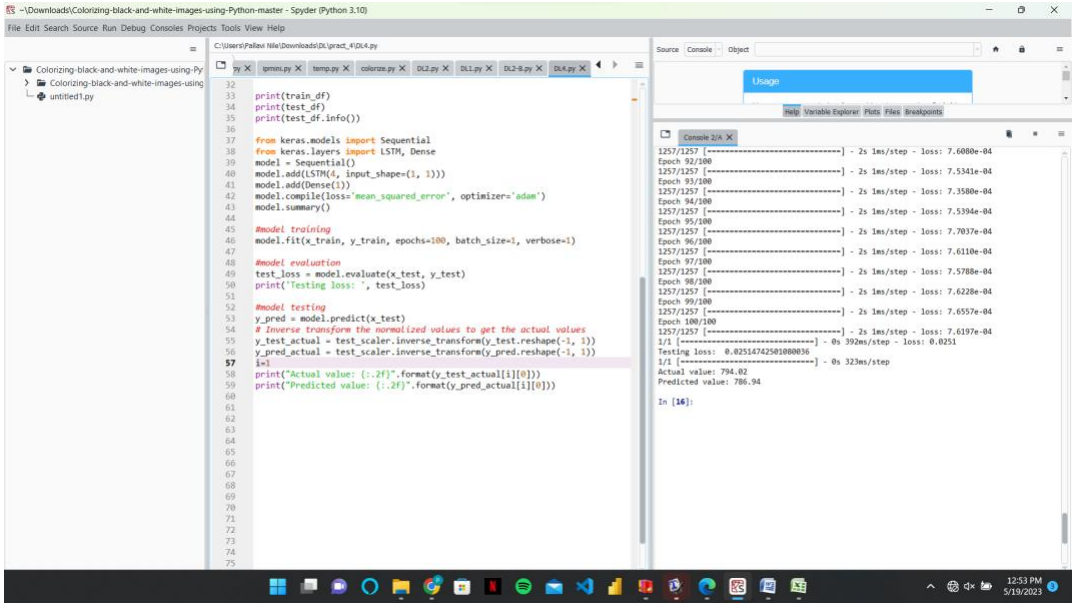
Epoch 31/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5658e-04
Epoch 32/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6453e-04
Epoch 33/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5216e-04
Epoch 34/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4584e-04
Epoch 35/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5470e-04
Epoch 36/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4857e-04
Epoch 37/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6248e-04
Epoch 38/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6770e-04
Epoch 39/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6173e-04
Epoch 40/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5110e-04
Epoch 41/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6038e-04
Epoch 42/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4611e-04
Epoch 43/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6135e-04
Epoch 44/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6493e-04
Epoch 45/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4718e-04
Epoch 46/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4901e-04
Epoch 47/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5801e-04
Epoch 48/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5971e-04
Epoch 49/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5674e-04

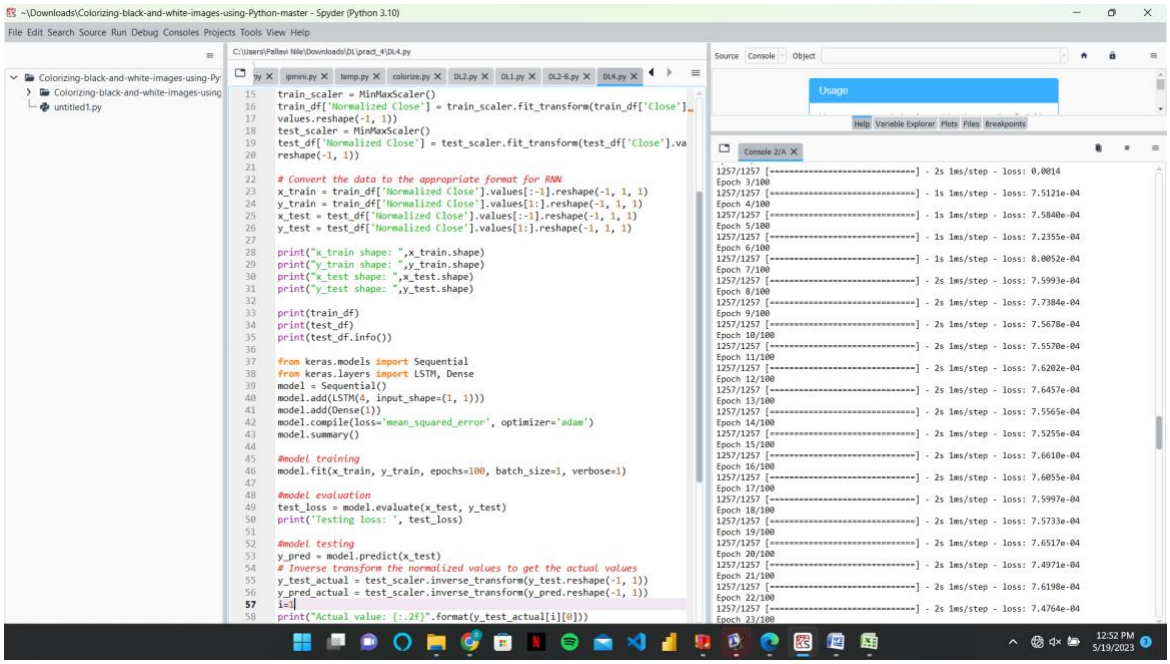
Epoch 50/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5868e-04
Epoch 51/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5481e-04
Epoch 52/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5386e-04
Epoch 53/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5523e-04
Epoch 54/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6254e-04
Epoch 55/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5177e-04
Epoch 56/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5200e-04
Epoch 57/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5651e-04
Epoch 58/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4635e-04
Epoch 59/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4019e-04
Epoch 60/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4784e-04
Epoch 61/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.7016e-04
Epoch 62/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6614e-04
Epoch 63/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4779e-04
Epoch 64/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5891e-04
Epoch 65/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5930e-04
Epoch 66/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5890e-04
Epoch 67/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5841e-04
Epoch 68/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5539e-04

Epoch 69/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5397e-04
Epoch 70/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6236e-04
Epoch 71/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5471e-04
Epoch 72/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5061e-04
Epoch 73/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4939e-04
Epoch 74/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6361e-04
Epoch 75/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4909e-04
Epoch 76/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5398e-04
Epoch 77/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5492e-04
Epoch 78/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5263e-04
Epoch 79/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5514e-04
Epoch 80/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5153e-04
Epoch 81/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5168e-04
Epoch 82/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4175e-04
Epoch 83/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5439e-04
Epoch 84/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4195e-04
Epoch 85/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6719e-04
Epoch 86/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5232e-04
Epoch 87/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4816e-04

```
Epoch 88/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6150e-04
Epoch 89/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6817e-04
Epoch 90/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.4801e-04
Epoch 91/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6080e-04
Epoch 92/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5341e-04
Epoch 93/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.3580e-04
Epoch 94/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5394e-04
Epoch 95/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.7037e-04
Epoch 96/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6110e-04
Epoch 97/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.5788e-04
Epoch 98/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6228e-04
Epoch 99/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6557e-04
Epoch 100/100
1257/1257 [==============================] - 2s 1ms/step - loss: 7.6197e-04
1/1 [==============================] - 0s 392ms/step - loss: 0.0251

Testing loss:  0.025147425010800036

1/1 [==============================] - 0s 323ms/step

Actual value: 794.02
Predicted value: 786.94
```


~\Downloads\Colorizing-black-and-white-images-using-Python-master - Spyder (Python 3.10)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Colorizing-black-and-white-images-using-Py
Colorizing-black-and-white-images-using
untitled1.py
#Nile Pallavi Roll NO: 4217 Div:B
# DL Practical NO.4
import pandas as pd
import numpy as np
train_df = pd.read_csv(r"C:\Users\Pallavi Nile\Downloads\DL\pract_4\Google_
print(train_df)
test_df = pd.read_csv(r"C:\Users\Pallavi Nile\Downloads\DL\pract_4\Google_S
print(test_df)
print(test_df.info())
#Data preprocessing
from sklearn.preprocessing import MinMaxScaler
# Convert 'Close' column to string type and remove commas
train_df['Close'] = train_df['Close'].astype(str).str.replace(',', '').asty
test_df['Close'] = test_df['Close'].astype(str).str.replace(',', '').astype
# Normalize the training and testing data separately
train_scaler = MinMaxScaler()
train_df['Normalized Close'] = train_scaler.fit_transform(train_df['Close']
values.reshape(-1, 1))
test_scaler = MinMaxScaler()
test_df['Normalized Close'] = test_scaler.fit_transform(test_df['Close'].va
reshape(-1, 1))
# Convert the data to the appropriate format for RNN
x_train = train_df['Normalized Close'].values[:-1].reshape(-1, 1, 1)
y_train = train_df['Normalized Close'].values[1:].reshape(-1, 1, 1)
x_test = test_df['Normalized Close'].values[:-1].reshape(-1, 1, 1)
y_test = test_df['Normalized Close'].values[1:].reshape(-1, 1, 1)
print("x_train shape: ",x_train.shape)
print("y_train shape: ",y_train.shape)
print("x_test shape: ",x_test.shape)
print("y_test shape: ",y_test.shape)
print(train_df)
print(test_df)
print(test_df.info())
from keras.models import Sequential
from keras.layers import LSTM, Dense
model = Sequential()
model.add(LSTM(4, input_shape=(1, 1)))
model.add(Dense(1))
Usage
Help Variable Explorer Plots Files Breakpoints
Epoch 99/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.4449e-04
Epoch 100/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5233e-04
In [15]: runfile('C:/Users/Pallavi Nile/Downloads/DL/pract_4/DL4.py', wdir='C:/Users/Pallavi Nile/Downloads/DL/pract_4')
Date Open High Low Close Volume
0 1/3/2012 325.25 332.83 324.97 663.59 7,380,500
1 1/4/2012 331.27 333.87 329.08 666.45 5,749,400
2 1/5/2012 329.83 330.75 326.89 657.21 6,590,300
3 1/6/2012 328.34 328.77 323.68 648.24 5,405,900
4 1/9/2012 322.04 322.29 309.46 620.76 11,688,800
1253 12/23/2016 790.90 792.74 787.28 789.91 623,400
1254 12/27/2016 790.68 797.86 787.66 791.55 789,100
1255 12/28/2016 793.70 794.23 783.20 785.05 1,153,800
1256 12/29/2016 783.33 785.93 778.92 782.79 744,300
1257 12/30/2016 782.75 782.78 770.41 771.82 1,770,000
[1258 rows x 6 columns]
Date Open High Low Close Volume
0 1/3/2017 778.81 789.63 775.80 786.14 1,657,300
1 1/4/2017 788.36 791.34 783.16 786.90 1,073,000
2 1/5/2017 786.08 794.48 785.02 794.02 1,335,200
3 1/6/2017 795.26 807.90 792.20 806.15 1,640,200
4 1/9/2017 806.40 809.97 802.83 806.65 1,272,400
5 1/10/2017 807.86 809.13 803.51 804.79 1,176,800
6 1/11/2017 805.00 808.15 801.37 807.91 1,065,900
7 1/12/2017 807.14 807.39 799.17 806.36 1,353,100
8 1/13/2017 807.48 811.22 806.69 807.88 1,099,200
9 1/17/2017 807.08 807.14 800.37 804.61 1,362,100
10 1/18/2017 805.81 806.21 800.99 806.07 1,294,400
11 1/19/2017 805.12 809.48 801.80 802.17 919,300
12 1/20/2017 806.91 806.91 801.69 805.02 1,670,000
13 1/23/2017 807.25 820.87 803.74 819.31 1,963,600
14 1/24/2017 822.30 825.90 817.82 823.87 1,474,000
15 1/25/2017 829.62 835.77 825.06 835.67 1,494,500
16 1/26/2017 837.81 838.00 827.01 832.15 2,973,900
17 1/27/2017 834.71 841.95 820.44 823.31 2,965,800
18 1/30/2017 814.66 815.84 799.80 802.32 3,246,600
12:54 PM
5/19/2023


~\Downloads\Colorizing-black-and-white-images-using-Python-master - Spyder (Python 3.10)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Colorizing-black-and-white-images-using-Py
Colorizing-black-and-white-images-using
untitled1.py
print(train_df)
print(test_df)
print(test_df.info())
from keras.models import Sequential
from keras.layers import LSTM, Dense
model = Sequential()
model.add(LSTM(4, input_shape=(1, 1)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
model.summary()
#model training
model.fit(x_train, y_train, epochs=100, batch_size=1, verbose=1)
#model evaluation
test_loss = model.evaluate(x_test, y_test)
print('Testing loss: ', test_loss)
#model testing
y_pred = model.predict(x_test)
# Inverse transform the normalized values to get the actual values
y_test_actual = test_scaler.inverse_transform(y_test.reshape(-1, 1))
y_pred_actual = test_scaler.inverse_transform(y_pred.reshape(-1, 1))
i=1
print("Actual value: {:.2f}".format(y_test_actual[i][0]))
print("Predicted value: {:.2f}".format(y_pred_actual[i][0]))
Usage
Help Variable Explorer Plots Files Breakpoints
1257/1257 [==========] - 2s 1ms/step - loss: 7.6080e-04
Epoch 92/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5341e-04
Epoch 93/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.3580e-04
Epoch 94/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5394e-04
Epoch 95/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.7037e-04
Epoch 96/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6110e-04
Epoch 97/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5788e-04
Epoch 98/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6228e-04
Epoch 99/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6557e-04
Epoch 100/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6197e-04
1/1 [==========] - 0s 392ms/step - loss: 0.0251
Testing loss: 0.025147250100036
1/1 [==========] - 0s 323ms/step
Actual value: 794.02
Predicted value: 786.94
In [16]:
12:53 PM
5/19/2023


~\Downloads\Colorizing-black-and-white-images-using-Python-master - Spyder (Python 3.10)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\Pallavi Nile\Downloads\DL\pract_4\DL4.py
Colorizing-black-and-white-images-using-Py
Colorizing-black-and-white-images-using
untitled1.py
ipmini.py
temp.py
colorize.py
DL2.py
DL1.py
DL2-8.py
DL4.py
train_scaler = MinMaxScaler()
train_df['Normalized Close'] = train_scaler.fit_transform(train_df['Close']
values.reshape(-1, 1))
test_scaler = MinMaxScaler()
test_df['Normalized Close'] = test_scaler.fit_transform(test_df['Close'].va
reshape(-1, 1))
# Convert the data to the appropriate format for RNN
x_train = train_df['Normalized Close'].values[:-1].reshape(-1, 1, 1)
y_train = train_df['Normalized Close'].values[1:].reshape(-1, 1, 1)
x_test = test_df['Normalized Close'].values[:-1].reshape(-1, 1, 1)
y_test = test_df['Normalized Close'].values[1:].reshape(-1, 1, 1)
print("x_train shape: ",x_train.shape)
print("y_train shape: ",y_train.shape)
print("x_test shape: ",x_test.shape)
print("y_test shape: ",y_test.shape)
print(train_df)
print(test_df)
print(test_df.info())
from keras.models import Sequential
from keras.layers import LSTM, Dense
model = Sequential()
model.add(LSTM(4, input_shape=(1, 1)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
model.summary()
#model training
model.fit(x_train, y_train, epochs=100, batch_size=1, verbose=1)
#model evaluation
test_loss = model.evaluate(x_test, y_test)
print('Testing loss: ', test_loss)
#model testing
y_pred = model.predict(x_test)
# Inverse transform the normalized values to get the actual values
y_test_actual = test_scaler.inverse_transform(y_test.reshape(-1, 1))
y_pred_actual = test_scaler.inverse_transform(y_pred.reshape(-1, 1))
i=1
print("Actual value: {:.2f}".format(y_test_actual[i][0]))
Source
Console
Object
Usage
Help
Variable Explorer
Plots
Files
Breakpoints
Console 2/A
1257/1257 [==========] - 2s 1ms/step - loss: 0.0014
Epoch 3/100
1257/1257 [==========] - 1s 1ms/step - loss: 7.5121e-04
Epoch 4/100
1257/1257 [==========] - 1s 1ms/step - loss: 7.5840e-04
Epoch 5/100
1257/1257 [==========] - 1s 1ms/step - loss: 7.2355e-04
Epoch 6/100
1257/1257 [==========] - 1s 1ms/step - loss: 8.0052e-04
Epoch 7/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5993e-04
Epoch 8/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.7384e-04
Epoch 9/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5678e-04
Epoch 10/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5570e-04
Epoch 11/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6202e-04
Epoch 12/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6457e-04
Epoch 13/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5565e-04
Epoch 14/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5255e-04
Epoch 15/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6610e-04
Epoch 16/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6055e-04
Epoch 17/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5997e-04
Epoch 18/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.5733e-04
Epoch 19/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6517e-04
Epoch 20/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.4971e-04
Epoch 21/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.6198e-04
Epoch 22/100
1257/1257 [==========] - 2s 1ms/step - loss: 7.4764e-04
Epoch 23/100
12:52 PM
5/19/2023