Deep Learning Practical Assignment 4

Name – Wale Shubhangi Ramesh
Roll No. – 4279 Batch – B8

[1]:
```
import pandas as pd
import numpy as np
```

[2]:
```
train_df = pd.read_csv(r"D:\DL Practical\Google_Stock_Price_Train.csv") #Path
↪where the CSV file is stored.
```

[3]:
```
train_df
```

[3]:

	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]

[4]:
```
test_df = pd.read_csv(r"D:\DL Practical\Google_Stock_Price_Test.csv") #Path
↪where the CSV file is stored.
```

[5]:
```
test_df
```

[5]:

	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
19	1/31/2017	796.86	801.25	790.52	796.79	2,160,600

[6]:
```
test_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 6 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
dtypes: float64(4), object(2)
memory usage: 1.1+ KB
```

Data Preprocessing

[7]:
```
from sklearn.preprocessing import MinMaxScaler
```

[8]:
```
# 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)
```

[9]:
```
# 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))
```

[10]:
```
# 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)
```

[11]:
```
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)
```

```
x_train shape:  (1257, 1, 1)
y_train shape:  (1257, 1, 1)
x_test shape:  (19, 1, 1)
y_test shape:  (19, 1, 1)
```

[12]:
```
train_df
```

[12]:

	Date	Open	High	Low	Close	Volume	Normalized Close
0	1/3/2012	325.25	332.83	324.97	663.59	7,380,500	0.237573
1	1/4/2012	331.27	333.87	329.08	666.45	5,749,400	0.241514
2	1/5/2012	329.83	330.75	326.89	657.21	6,590,300	0.228781
3	1/6/2012	328.34	328.77	323.68	648.24	5,405,900	0.216419
4	1/9/2012	322.04	322.29	309.46	620.76	11,688,800	0.178548
...	...	...	...	...	...	...	...
1253	12/23/2016	790.90	792.74	787.28	789.91	623,400	0.411656
1254	12/27/2016	790.68	797.86	787.66	791.55	789,100	0.413916
1255	12/28/2016	793.70	794.23	783.20	785.05	1,153,800	0.404958
1256	12/29/2016	783.33	785.93	778.92	782.79	744,300	0.401844
1257	12/30/2016	782.75	782.78	770.41	771.82	1,770,000	0.386726

[1258 rows x 7 columns]

[13]:
```
test_df
```

[13]:

	Date	Open	High	Low	Close	Volume	Normalized Close
0	1/3/2017	778.81	789.63	775.80	786.14	1,657,300	0.000000
1	1/4/2017	788.36	791.34	783.16	786.90	1,073,000	0.015344
2	1/5/2017	786.08	794.48	785.02	794.02	1,335,200	0.159095
3	1/6/2017	795.26	807.90	792.20	806.15	1,640,200	0.403998
4	1/9/2017	806.40	809.97	802.83	806.65	1,272,400	0.414092
5	1/10/2017	807.86	809.13	803.51	804.79	1,176,800	0.376539
6	1/11/2017	805.00	808.15	801.37	807.91	1,065,900	0.439532
7	1/12/2017	807.14	807.39	799.17	806.36	1,353,100	0.408237

8	1/13/2017	807.48	811.22	806.69	807.88	1,099,200	0.438926
9	1/17/2017	807.08	807.14	800.37	804.61	1,362,100	0.372905
10	1/18/2017	805.81	806.21	800.99	806.07	1,294,400	0.402382
11	1/19/2017	805.12	809.48	801.80	802.17	919,300	0.323642
12	1/20/2017	806.91	806.91	801.69	805.02	1,670,000	0.381183
13	1/23/2017	807.25	820.87	803.74	819.31	1,963,600	0.669695
14	1/24/2017	822.30	825.90	817.82	823.87	1,474,000	0.761761
15	1/25/2017	829.62	835.77	825.06	835.67	1,494,500	1.000000
16	1/26/2017	837.81	838.00	827.01	832.15	2,973,900	0.928932
17	1/27/2017	834.71	841.95	820.44	823.31	2,965,800	0.750454
18	1/30/2017	814.66	815.84	799.80	802.32	3,246,600	0.326671
19	1/31/2017	796.86	801.25	790.52	796.79	2,160,600	0.215021

[14]:
```
test_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
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
```

Building our Model

[15]:
```
from keras.models import Sequential
from keras.layers import LSTM, Dense
```

[16]:
```
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: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 lstm_4 (LSTM)               (None, 4)                 96        

 dense_4 (Dense)             (None, 1)                 5         
```

```
=================================================================
Total params: 101
Trainable params: 101
Non-trainable params: 0
_________________________________________________________________
```

Training our Model

[17]:
```
model.fit(x_train, y_train, epochs=100, batch_size=1, verbose=1)
```

```
Epoch 1/100
1257/1257 [==============================] - 4s 2ms/step - loss: 0.0310
Epoch 2/100
1257/1257 [==============================] - 3s 2ms/step - loss: 0.0014
Epoch 3/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.7640e-04
Epoch 4/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.8601e-04
Epoch 5/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5519e-04
Epoch 6/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5621e-04
Epoch 7/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6944e-04
Epoch 8/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6748e-04
Epoch 9/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6328e-04
Epoch 10/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5948e-04
Epoch 11/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5432e-04
Epoch 12/100
1257/1257 [==============================] - 3s 3ms/step - loss: 7.6786e-04
Epoch 13/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6365e-04
Epoch 14/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4921e-04
Epoch 15/100
1257/1257 [==============================] - 4s 3ms/step - loss: 7.7051e-04
Epoch 16/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5754e-04
Epoch 17/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6044e-04
Epoch 18/100
1257/1257 [==============================] - 3s 3ms/step - loss: 7.5942e-04
Epoch 19/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6386e-04
Epoch 20/100
```

```
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5305e-04
Epoch 21/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.7052e-04
Epoch 22/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5623e-04
Epoch 23/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5279e-04
Epoch 24/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5948e-04
Epoch 25/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5198e-04
Epoch 26/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.7338e-04
Epoch 27/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6739e-04
Epoch 28/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6597e-04
Epoch 29/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6504e-04
Epoch 30/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6284e-04
Epoch 31/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5100e-04
Epoch 32/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6783e-04
Epoch 33/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6847e-04
Epoch 34/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5156e-04
Epoch 35/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.3298e-04
Epoch 36/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.8577e-04
Epoch 37/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6031e-04
Epoch 38/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5284e-04
Epoch 39/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4893e-04
Epoch 40/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5578e-04
Epoch 41/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5905e-04
Epoch 42/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5671e-04
Epoch 43/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5353e-04
Epoch 44/100
```

```
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6960e-04
Epoch 45/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4974e-04
Epoch 46/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6117e-04
Epoch 47/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5874e-04
Epoch 48/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5113e-04
Epoch 49/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5599e-04
Epoch 50/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5120e-04
Epoch 51/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6484e-04
Epoch 52/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5089e-04
Epoch 53/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5940e-04
Epoch 54/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.7353e-04
Epoch 55/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5936e-04
Epoch 56/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5985e-04
Epoch 57/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4112e-04
Epoch 58/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5930e-04
Epoch 59/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5598e-04
Epoch 60/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5615e-04
Epoch 61/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6379e-04
Epoch 62/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6190e-04
Epoch 63/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5898e-04
Epoch 64/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4999e-04
Epoch 65/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5289e-04
Epoch 66/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4483e-04
Epoch 67/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6378e-04
Epoch 68/100
```

```
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6579e-04
Epoch 69/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5623e-04
Epoch 70/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6837e-04
Epoch 71/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6128e-04
Epoch 72/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5361e-04
Epoch 73/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5751e-04
Epoch 74/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4904e-04
Epoch 75/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5084e-04
Epoch 76/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5798e-04
Epoch 77/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6144e-04
Epoch 78/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5390e-04
Epoch 79/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6827e-04
Epoch 80/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5229e-04
Epoch 81/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4967e-04
Epoch 82/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6215e-04
Epoch 83/100
1257/1257 [==============================] - 3s 3ms/step - loss: 7.4721e-04
Epoch 84/100
1257/1257 [==============================] - 3s 3ms/step - loss: 7.5734e-04
Epoch 85/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6813e-04
Epoch 86/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6794e-04
Epoch 87/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4982e-04
Epoch 88/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6006e-04
Epoch 89/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5863e-04
Epoch 90/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6074e-04
Epoch 91/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6040e-04
Epoch 92/100
```

```
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5800e-04
Epoch 93/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6033e-04
Epoch 94/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4986e-04
Epoch 95/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.4980e-04
Epoch 96/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5753e-04
Epoch 97/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5469e-04
Epoch 98/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.6439e-04
Epoch 99/100
1257/1257 [==============================] - 3s 3ms/step - loss: 7.5264e-04
Epoch 100/100
1257/1257 [==============================] - 3s 2ms/step - loss: 7.5342e-04
```

[17]: <keras.callbacks.History at 0x27733114c70>

Evaluating our Model

[18]:
```
test_loss = model.evaluate(x_test, y_test)
print('Testing loss: ', test_loss)
```

```
1/1 [==============================] - 1s 573ms/step - loss: 0.0245
Testing loss:  0.02449163608253002
```

Testing our Model

[19]:
```
y_pred = model.predict(x_test)
```

```
1/1 [==============================] - 1s 501ms/step
```

[20]:
```
# 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))
```

[21]:
```
i=1
```

[22]:
```
print("Actual value: {:.2f}".format(y_test_actual[i][0]))
print("Predicted value: {:.2f}".format(y_pred_actual[i][0]))
```

```
Actual value: 794.02
Predicted value: 787.65
```