DL Assignment : 3

Name : Shrikrushna Zirape

Roll No : 41283 (BE-2)

Problem Statement : Recurrent neural network (RNN) Use the Google stock pries dataset and design a time series analysis and prediction system using RNN

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

Loading the dataset

In [2]:
```
train_df = pd.read_csv('data/Google_Stock_Price_Train.csv')
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1258 entries, 0 to 1257
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Date    1258 non-null   object 
 1   Open    1258 non-null   float64
 2   High    1258 non-null   float64
 3   Low     1258 non-null   float64
 4   Close   1258 non-null   object 
 5   Volume  1258 non-null   object 
dtypes: float64(3), object(3)
memory usage: 59.1+ KB
```

In [4]:
```
test_df = pd.read_csv('data/Google_Stock_Price_Test.csv')
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
```

Choosing column 'open' for predicition

In [5]:
```
train = train_df.loc[:,["Open"]].values
train.shape
```

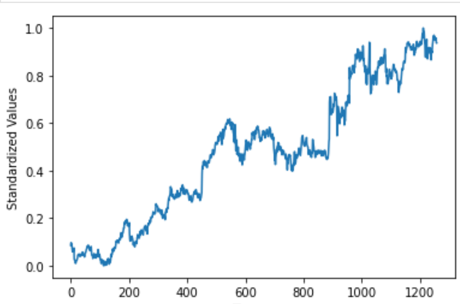
Out[5]:
```
(1258, 1)
```

Feature Scaling

In [6]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
```

In [7]:
```
train_scaled = scaler.fit_transform(train)
```

In [8]:
```
plt.plot(train_scaled)
plt.ylabel("Standardized Values")
plt.xlabel("Time->")
plt.show()
```



Create data structure to train model

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Taking reference of past 60 days to predict future stock price

x_train will have data of 60 days prior to current date and y_train will have price on current date

In [9]:
```
x_train = []
y_train = []
time = 60
for i in range(60,train_scaled.shape[0]):
    x_train.append(train_scaled[i-60:i,0])
    y_train.append(train_scaled[i,0])
x_train = np.array(x_train)
y_train = np.array(y_train)
```

In [10]:
```
x_train.shape,y_train.shape
```

Out[10]:
```
((1198, 60), (1198,))
```

In [11]:
```
x_train = np.reshape(x_train,newshape=(x_train.shape[0],x_train.shape[1],1))
x_train.shape
```

Out[11]:
```
(1198, 60, 1)
```

Build model

In [12]:
```
from keras.models import Sequential
from keras.layers import Dense, SimpleRNN,Dropout
```

In [13]:
```
model = Sequential()

model.add(SimpleRNN(units=50,activation = "tanh", return_sequences = True, input_shape = (x_train.shape[1], 1)))
model.add(Dropout(0.2))

model.add(SimpleRNN(units=50,activation = "tanh", return_sequences = True))
model.add(Dropout(0.2))

model.add(SimpleRNN(units=50,activation = "tanh", return_sequences = True))
model.add(Dropout(0.2))

model.add(SimpleRNN(units=50))
model.add(Dropout(0.2))

model.add(Dense(units=1))

model.compile(optimizer='adam',loss='mse')
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 simple_rnn (SimpleRNN)      (None, 60, 50)            2600

 dropout (Dropout)           (None, 60, 50)            0

 simple_rnn_1 (SimpleRNN)    (None, 60, 50)            5050

 dropout_1 (Dropout)         (None, 60, 50)            0

 simple_rnn_2 (SimpleRNN)    (None, 60, 50)            5050

 dropout_2 (Dropout)         (None, 60, 50)            0

 simple_rnn_3 (SimpleRNN)    (None, 50)                5050

 dropout_3 (Dropout)         (None, 50)                0

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 17,801
Trainable params: 17,801
Non-trainable params: 0
_________________________________________________________________
```

In [14]:
```
model.fit(x_train,y_train,epochs=100,batch_size=30,validation_split=0.05)
```

```
Epoch 1/100
38/38 [==============================] - 11s 128ms/step - loss: 0.3165 - val_loss: 0.0253
Epoch 2/100
38/38 [==============================] - 4s 96ms/step - loss: 0.2319 - val_loss: 8.3071e-04
Epoch 3/100
38/38 [==============================] - 4s 97ms/step - loss: 0.1569 - val_loss: 0.0020
Epoch 4/100
38/38 [==============================] - 4s 108ms/step - loss: 0.1157 - val_loss: 0.0229
Epoch 5/100
38/38 [==============================] - 4s 104ms/step - loss: 0.0880 - val_loss: 0.0052
Epoch 6/100
38/38 [==============================] - 4s 96ms/step - loss: 0.0781 - val_loss: 0.0396
Epoch 7/100
38/38 [==============================] - 4s 96ms/step - loss: 0.0662 - val_loss: 0.0299
Epoch 8/100
38/38 [==============================] - 4s 98ms/step - loss: 0.0582 - val_loss: 0.0065
Epoch 9/100
38/38 [==============================] - 4s 97ms/step - loss: 0.0459 - val_loss: 0.0163
Epoch 10/100
38/38 [==============================] - 4s 96ms/step - loss: 0.0404 - val_loss: 0.0100
Epoch 11/100
38/38 [==============================] - 4s 97ms/step - loss: 0.0359 - val_loss: 0.0042
Epoch 12/100
38/38 [==============================] - 4s 100ms/step - loss: 0.0351 - val_loss: 6.8119e-04
Epoch 13/100
38/38 [==============================] - 4s 98ms/step - loss: 0.0289 - val_loss: 0.0066
Epoch 14/100
38/38 [==============================] - 4s 119ms/step - loss: 0.0260 - val_loss: 0.0029
Epoch 15/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0238 - val_loss: 0.0028
Epoch 16/100
38/38 [==============================] - 4s 105ms/step - loss: 0.0214 - val_loss: 0.0022
Epoch 17/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0197 - val_loss: 0.0058
Epoch 18/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0214 - val_loss: 0.0047
Epoch 19/100
38/38 [==============================] - 4s 111ms/step - loss: 0.0180 - val_loss: 0.0037
Epoch 20/100
38/38 [==============================] - 4s 114ms/step - loss: 0.0149 - val_loss: 0.0091
Epoch 21/100
38/38 [==============================] - 4s 106ms/step - loss: 0.0161 - val_loss: 0.0012
Epoch 22/100
38/38 [==============================] - 4s 111ms/step - loss: 0.0140 - val_loss: 0.0050
Epoch 23/100
38/38 [==============================] - 4s 109ms/step - loss: 0.0124 - val_loss: 0.0106
Epoch 24/100
38/38 [==============================] - 5s 122ms/step - loss: 0.0128 - val_loss: 0.0076
Epoch 25/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0112 - val_loss: 0.0011
Epoch 26/100
38/38 [==============================] - 4s 115ms/step - loss: 0.0102 - val_loss: 0.0040
Epoch 27/100
38/38 [==============================] - 4s 119ms/step - loss: 0.0101 - val_loss: 0.0091
Epoch 28/100
38/38 [==============================] - 5s 133ms/step - loss: 0.0092 - val_loss: 0.0017
Epoch 29/100
38/38 [==============================] - 4s 109ms/step - loss: 0.0087 - val_loss: 0.0021
Epoch 30/100
38/38 [==============================] - 4s 106ms/step - loss: 0.0084 - val_loss: 0.0032
Epoch 31/100
38/38 [==============================] - 4s 101ms/step - loss: 0.0087 - val_loss: 0.0073
Epoch 32/100
38/38 [==============================] - 4s 105ms/step - loss: 0.0080 - val_loss: 0.0032
Epoch 33/100
38/38 [==============================] - 4s 97ms/step - loss: 0.0077 - val_loss: 0.0062
Epoch 34/100
38/38 [==============================] - 4s 96ms/step - loss: 0.0075 - val_loss: 0.0012
Epoch 35/100
38/38 [==============================] - 4s 103ms/step - loss: 0.0070 - val_loss: 0.0020
Epoch 36/100
38/38 [==============================] - 4s 106ms/step - loss: 0.0072 - val_loss: 9.8018e-04
Epoch 37/100
38/38 [==============================] - 4s 108ms/step - loss: 0.0062 - val_loss: 0.0013
Epoch 38/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0063 - val_loss: 0.0049
Epoch 39/100
38/38 [==============================] - 5s 127ms/step - loss: 0.0059 - val_loss: 5.9050e-04
Epoch 40/100
38/38 [==============================] - 5s 127ms/step - loss: 0.0058 - val_loss: 6.0321e-04
Epoch 41/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0053 - val_loss: 0.0013
Epoch 42/100
38/38 [==============================] - 4s 105ms/step - loss: 0.0052 - val_loss: 0.0025
Epoch 43/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0049 - val_loss: 0.0029
Epoch 44/100
38/38 [==============================] - 5s 122ms/step - loss: 0.0052 - val_loss: 0.0020
Epoch 45/100
38/38 [==============================] - 5s 123ms/step - loss: 0.0047 - val_loss: 0.0020
Epoch 46/100
38/38 [==============================] - 4s 112ms/step - loss: 0.0048 - val_loss: 0.0024
Epoch 47/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0044 - val_loss: 9.3178e-04
Epoch 48/100
38/38 [==============================] - 4s 114ms/step - loss: 0.0045 - val_loss: 0.0013
Epoch 49/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0041 - val_loss: 0.0016
38/38 [==============================] - 4s 114ms/step - loss: 0.0040 - val_loss: 6.2601e-04
```

```
Epoch 51/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0043 - val_loss: 8.6406e-04
Epoch 52/100
38/38 [==============================] - 5s 126ms/step - loss: 0.0044 - val_loss: 8.9583e-04
Epoch 53/100
38/38 [==============================] - 5s 123ms/step - loss: 0.0038 - val_loss: 8.3582e-04
Epoch 54/100
38/38 [==============================] - 5s 125ms/step - loss: 0.0046 - val_loss: 9.5669e-04
Epoch 55/100
38/38 [==============================] - 5s 133ms/step - loss: 0.0037 - val_loss: 0.0013
Epoch 56/100
38/38 [==============================] - 5s 129ms/step - loss: 0.0037 - val_loss: 8.7909e-04
Epoch 57/100
38/38 [==============================] - 4s 111ms/step - loss: 0.0037 - val_loss: 0.0017
Epoch 58/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0037 - val_loss: 6.1774e-04
Epoch 59/100
38/38 [==============================] - 4s 115ms/step - loss: 0.0036 - val_loss: 0.0014
Epoch 60/100
38/38 [==============================] - 4s 117ms/step - loss: 0.0038 - val_loss: 6.3705e-04
Epoch 61/100
38/38 [==============================] - 4s 108ms/step - loss: 0.0033 - val_loss: 6.0918e-04
Epoch 62/100
38/38 [==============================] - 4s 100ms/step - loss: 0.0031 - val_loss: 6.8135e-04
Epoch 63/100
38/38 [==============================] - 4s 106ms/step - loss: 0.0032 - val_loss: 8.8936e-04
Epoch 64/100
38/38 [==============================] - 4s 103ms/step - loss: 0.0030 - val_loss: 6.1725e-04
Epoch 65/100
38/38 [==============================] - 4s 109ms/step - loss: 0.0030 - val_loss: 5.9824e-04
Epoch 66/100
38/38 [==============================] - 4s 114ms/step - loss: 0.0029 - val_loss: 8.0693e-04
Epoch 67/100
38/38 [==============================] - 5s 124ms/step - loss: 0.0030 - val_loss: 0.0013
Epoch 68/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0029 - val_loss: 6.3012e-04
Epoch 69/100
38/38 [==============================] - 4s 103ms/step - loss: 0.0028 - val_loss: 7.2763e-04
Epoch 70/100
38/38 [==============================] - 4s 104ms/step - loss: 0.0030 - val_loss: 6.3387e-04
Epoch 71/100
38/38 [==============================] - 4s 107ms/step - loss: 0.0027 - val_loss: 6.8628e-04
Epoch 72/100
38/38 [==============================] - 4s 108ms/step - loss: 0.0025 - val_loss: 5.9166e-04
Epoch 73/100
38/38 [==============================] - 4s 104ms/step - loss: 0.0028 - val_loss: 5.9803e-04
Epoch 74/100
38/38 [==============================] - 4s 101ms/step - loss: 0.0027 - val_loss: 7.5142e-04
Epoch 75/100
38/38 [==============================] - 4s 108ms/step - loss: 0.0026 - val_loss: 9.3893e-04
Epoch 76/100
38/38 [==============================] - 4s 105ms/step - loss: 0.0028 - val_loss: 6.0789e-04
Epoch 77/100
38/38 [==============================] - 5s 120ms/step - loss: 0.0026 - val_loss: 5.9871e-04
Epoch 78/100
38/38 [==============================] - 4s 115ms/step - loss: 0.0026 - val_loss: 0.0028
Epoch 79/100
38/38 [==============================] - 4s 104ms/step - loss: 0.0027 - val_loss: 7.2219e-04
Epoch 80/100
38/38 [==============================] - 4s 101ms/step - loss: 0.0026 - val_loss: 8.7359e-04
Epoch 81/100
38/38 [==============================] - 4s 107ms/step - loss: 0.0025 - val_loss: 7.5039e-04
Epoch 82/100
38/38 [==============================] - 4s 117ms/step - loss: 0.0024 - val_loss: 0.0018
Epoch 83/100
38/38 [==============================] - 5s 125ms/step - loss: 0.0023 - val_loss: 0.0010
Epoch 84/100
38/38 [==============================] - 4s 109ms/step - loss: 0.0025 - val_loss: 5.9867e-04
Epoch 85/100
38/38 [==============================] - 4s 102ms/step - loss: 0.0024 - val_loss: 0.0011
Epoch 86/100
38/38 [==============================] - 4s 98ms/step - loss: 0.0023 - val_loss: 6.9074e-04
Epoch 87/100
38/38 [==============================] - 4s 105ms/step - loss: 0.0021 - val_loss: 6.9190e-04
Epoch 88/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0022 - val_loss: 7.5086e-04
Epoch 89/100
38/38 [==============================] - 4s 103ms/step - loss: 0.0024 - val_loss: 0.0010
Epoch 90/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0021 - val_loss: 9.7637e-04
Epoch 91/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0020 - val_loss: 7.7010e-04
Epoch 92/100
38/38 [==============================] - 5s 120ms/step - loss: 0.0020 - val_loss: 6.0084e-04
Epoch 93/100
38/38 [==============================] - 4s 111ms/step - loss: 0.0022 - val_loss: 8.4450e-04
Epoch 94/100
38/38 [==============================] - 5s 129ms/step - loss: 0.0022 - val_loss: 8.6722e-04
Epoch 95/100
38/38 [==============================] - 4s 107ms/step - loss: 0.0021 - val_loss: 6.1268e-04
Epoch 96/100
38/38 [==============================] - 4s 100ms/step - loss: 0.0019 - val_loss: 0.0011
Epoch 97/100
38/38 [==============================] - 4s 118ms/step - loss: 0.0022 - val_loss: 6.0908e-04
Epoch 98/100
38/38 [==============================] - 5s 125ms/step - loss: 0.0022 - val_loss: 0.0012
Epoch 99/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0019 - val_loss: 6.1296e-04
38/38 [==============================] - 4s 104ms/step - loss: 0.0020 - val_loss: 6.0433e-04
```

Out[14]: <keras.callbacks.History at 0x1cf08309850>

Prepare test dataset

In [15]:
```
data = pd.concat((train_df['Open'],test_df['Open']),axis=0)
```

In [16]:
```
test_input = data.iloc[len(data) - len(test_df) - time : ].values
test_input.shape
```

Out[16]: (80,)

In [17]:
```
test_input = test_input.reshape(-1,1)
test_input.shape
```

Out[17]: (80, 1)

In [18]:
```
test_scaled = scaler.transform(test_input)
```

Create test data set

In [19]:
```
x_test = []
for i in range(time,test_scaled.shape[0]):
    x_test.append(test_scaled[i - time: i,0 ])
x_test = np.array(x_test)
x_test.shape
```

Out[19]: (20, 60)

In [20]:
```
x_test = np.reshape(x_test,newshape=(x_test.shape[0],x_test.shape[1],1))
x_test.shape
```

Out[20]: (20, 60, 1)

In [21]:
```
y_test = test_df.loc[:,"Open"].values
```

Model Prediction

In [22]:
```
y_pred = model.predict(x_test)
```

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

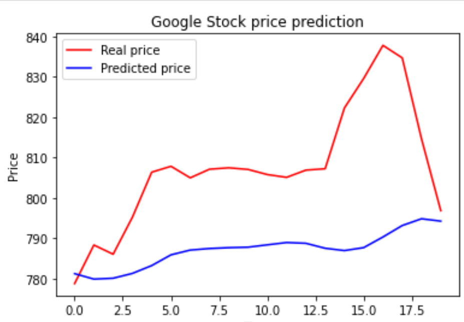
In [23]:
```
y_pred = scaler.inverse_transform(y_pred)
```

In [24]:
```
output = model.evaluate(x=x_test,y=y_test)
```

```
1/1 [==============================] - 1s 1s/step - loss: 650790.0000
```

In [25]:
```
plt.plot(y_test, color = 'red', label = 'Real price')
plt.plot(y_pred, color = 'blue', label = 'Predicted price')

plt.title('Google Stock price prediction')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```



In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js