import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

dataset\_train = pd.read\_csv('Google\_Stock\_Price\_Train.csv')

training\_set = dataset\_train.iloc[:, 1:2].values

# normalise the data using min\_max normalisation

from sklearn.preprocessing import MinMaxScaler

sc = MinMaxScaler(feature\_range = (0, 1))

training\_set\_scaled = sc.fit\_transform(training\_set)

#create a data structure with 60 timesteps and one output as an Array of x\_train and y\_train

X\_train = []

y\_train = []

for i in range(60, 1258):

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)

#reshaping of x\_train data

X\_train = np.reshape(X\_train, (X\_train.shape[0], X\_train.shape[1], 1))

# have imported the Keras library and its packages.

from keras.models import Sequential

from keras.layers import Dense

from keras.layers import LSTM

from keras.layers import Dropout

#initialise RNN

regressor = Sequential()

#add first layer of LSTM & dropout regularisation

regressor.add(LSTM(units = 50, return\_sequences = True, input\_shape = (X\_train.shape[1], 1)))

regressor.add(Dropout(0.2))

#add second layer of LSTM & dropout regularisation

regressor.add(LSTM(units = 50, return\_sequences = True))

regressor.add(Dropout(0.2))

#add 3rd layer of LSTM & dropout regularisation

regressor.add(LSTM(units = 50, return\_sequences = True))

regressor.add(Dropout(0.2))

#add 4th layer of LSTM & dropout regularisation

regressor.add(LSTM(units = 50))

regressor.add(Dropout(0.2))

#add output layer

regressor.add(Dense(units = 1))

#compile our RNN model.

regressor.compile(optimizer = 'adam', loss = 'mean\_squared\_error')

#using a training dataset to fit the RNN model

regressor.fit(X\_train, y\_train, epochs = 100, batch\_size = 32)

#load test set

dataset\_test = pd.read\_csv('Google\_Stock\_Price\_Test.csv')

real\_stock\_price = dataset\_test.iloc[:, 1:2].values

#Getting the predicted stock price of 2017

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, 80):

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 = regressor.predict(X\_test)

predicted\_stock\_price = sc.inverse\_transform(predicted\_stock\_price)

#plot results

plt.plot(real\_stock\_price, color = 'red', label = 'Real Google Stock Price')

plt.plot(predicted\_stock\_price, color = 'blue', label = 'Predicted Google Stock Price')

plt.title('Google Stock Price Prediction')

plt.xlabel('Time')

plt.ylabel('Google Stock Price')

plt.legend()

plt.show()