Name: 莊晴雯
Student ID: 108062281

BASIC

In order to design an algorithm to forecast the stock price of TSMC based on MTK, the main idea I used to implement this is to use regression. I first gather the available stock price data of both MTK and TSMC starting from January 2021 up until October 2021. I first declared my own global variables that I am going to used later namely:

- trainingData => list => store the date and stock prices of MTK and TSMC from 01/04 – 10/12.
- testData => list => store the date and stock prices of MTK and TSMC from 10/15 – 11/11.
- xTrain => list => store the stock price of MTK from trainingData.
- yTrain => list => store the stock price of TSMC from trainingData.
- xTest => list => store the stock price of MTK from testData.
- yTest => list => store the stock price of TSMC from testData.
- dates => list => to get the dates from testData to be output to the csv file.
- totalData => variable type int => to store the length of the total data (trainingData + testData)

In order to process the data I currently have, I used the following functions:

- SplitData()
  Splits the total datas from input_datalist to trainingData and testingData. In order to classify which data belongs to the trainingData and testingData respectively, I separated the data according to the date. I first convert the available dates from the csv file which is currently in string format, then split the string using Python's split method to get the day, month, and year, then convert the date to np.datetime64 format. I then input the starting date for the testData which is 2021/10/15, get the index from the list of dates using Python's index method, then set the index as the ending index for the trainingData and starting index of the testData.
- PreprocessData()
  This function is used to classify which data belongs to xTrain, yTrain, and xTest. Using a for loop, I first split the data for MTK and TSMC and store them into a temporary list, and convert them to int before appending them to xTrain and yTrain, then convert them to np.array. I then get the dates of the stock prices that needs to be predicted for the testData which would later be 10/15 – 11/11.
- Regression()
  To implement the regression, I used np.polyfit and used degree 3 since it is the best fit for my current data after a few trials. Polyfit is the implementation of least squares polynomial fit and it will then return the coefficients that minimizes the squared error. Hence, the function Regression() will return a list of coefficients.
- CountLoss()
  Since I did not use gradient descent to implement this model, CountLoss is not really necessary for me and hence, I used this function to calculate the RMSE. Assuming that the real data of TSMC for 10/15 – 11/11 is already inside input.csv, this function can be used to calculate the RMSE of the actual data and the predicted data using numpy's square, subtract, and mean.
- MakePrediction(model)
  This function takes the model returned by Regression, which are the coefficients of the function ax^3 + bx^2 + cx + d, and then uses these data to predict the TSMC stock price based on MTK stock price in xTest, and then the result will be appended to yTest.

Name: 莊晴雯
Student ID: 108062281

After all these functions, I first called SplitData() to split the input data to training data and testing data based on the dates, then called PreprocessData() to process the data and determine which would be xTrain, yTrain, xTest. Then I called the Regression function which will return a list of coefficients of ax^3 + bx^2 + cx + d and store the list into a variable, then I called MakePrediction to generate the price of TSMC by passing the list of coefficients.

After all these steps, I finally got the predicted price of TSMC which is currently stored in yTest, and I formatted the data to be an array in the form [[date, price] [date, price] …], plot the graph, and write the data to the output csv file.

END OF BASIC PART

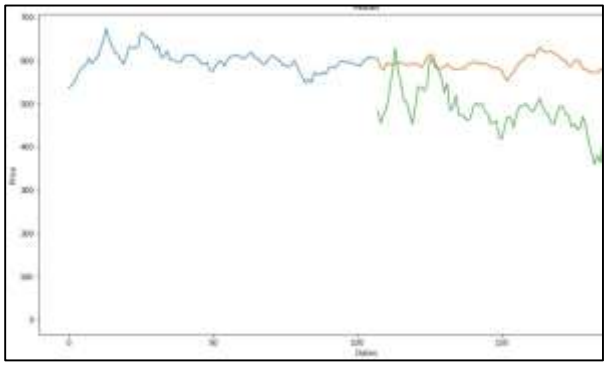
BONUS PART PLEASE REFER TO THE NEXT PAGE

Name: 莊晴雯
Student ID: 108062281

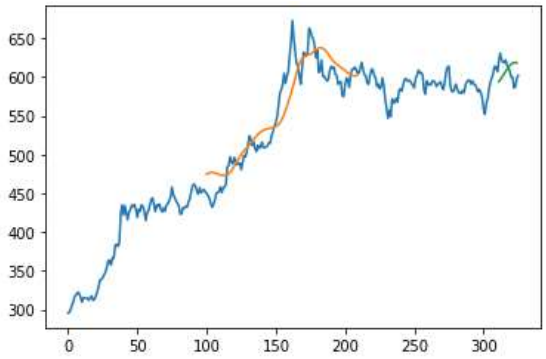
BONUS

In order to implement the bonus part of this assignment, I implemented Long Short-Term Memory (LSTM) to make predictions based on past data since LSTM is very capable of learning order dependence in sequence prediction problems. I did try to use multiple regression to implement this model at first, but I found out that multiple regression's model is not as good as LSTM.

Multiple Regression



LSTM



As seen on the picture above, LSTM has a much better fit on the training data and testing data compared to Multiple Regression. Hence, I decided to go with LSTM. In order to implement this model, I divided the process into four big methods:

1. Collect and preprocess the data.
   I decided to use TSMC's stock price data staring from June 2020. I did realize that there has been a major increase in TSMC's stock price in the past few years and the stock price seems to be more stable in the past few months. Hence, I decided to only take the data starting from June 2020 and not go too far. I first used MinMaxScaler and fit_transform() from sklearn so that we can scale and learn the scaling parameters of the data. I took 65% of the entire data to be my training data and the rest to be the training data, then separated those data to xTrain, yTrain, xTest, and yTest through the preprocessData() function I created.
2. Create the LSTM Model.
   To create the LSTM model, I made use of the built-in function provided by the tensorflow module and fit the xTrain, yTrain, xTest and yTest as the validation data. As the model runs based on the epochs, the loss value and the validation loss keeps on decreasing and our goal is to minimize both of these values.
3. Predict the test data.
   In this step we are basically just shifting both train and test predictions that are going to be used for plotting. After plotting, the graph above referring to LSTM was generated. Based on the plot, we can clearly see how much the training and testing data fits the original data.
4. Predict the future 20 days.
   From the given testData values, we pass it to out model and do the prediction to get the yPredict value, which is the predicted value of TSMC for the next 20 days. However, we do have a condition that we might need to take into account, that is, if the length of xList (testing datas) is greater than 100, we need to shift one position to the right and take the new output as the input.