**Task Report Cos30018 Option B**

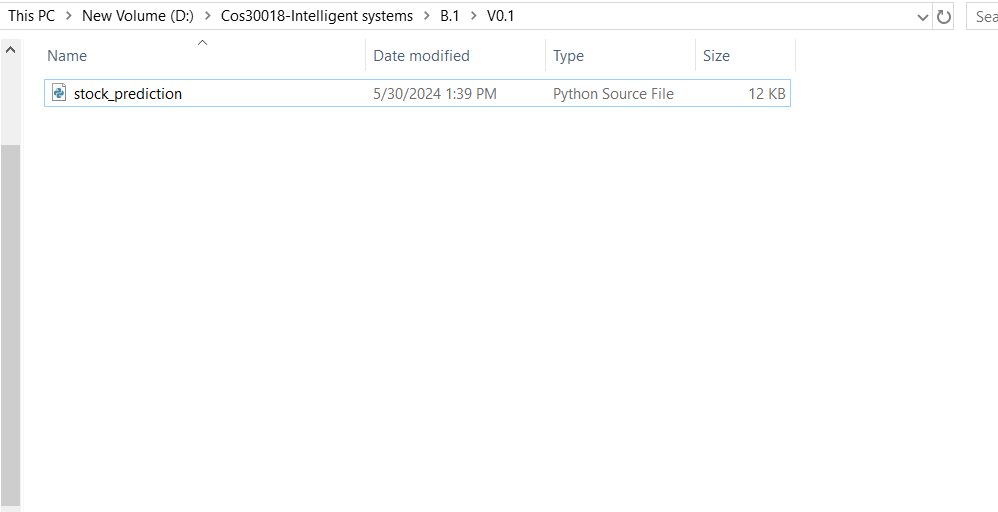
**B.1: Set Up**

**Name: Le Bao Nguyen**

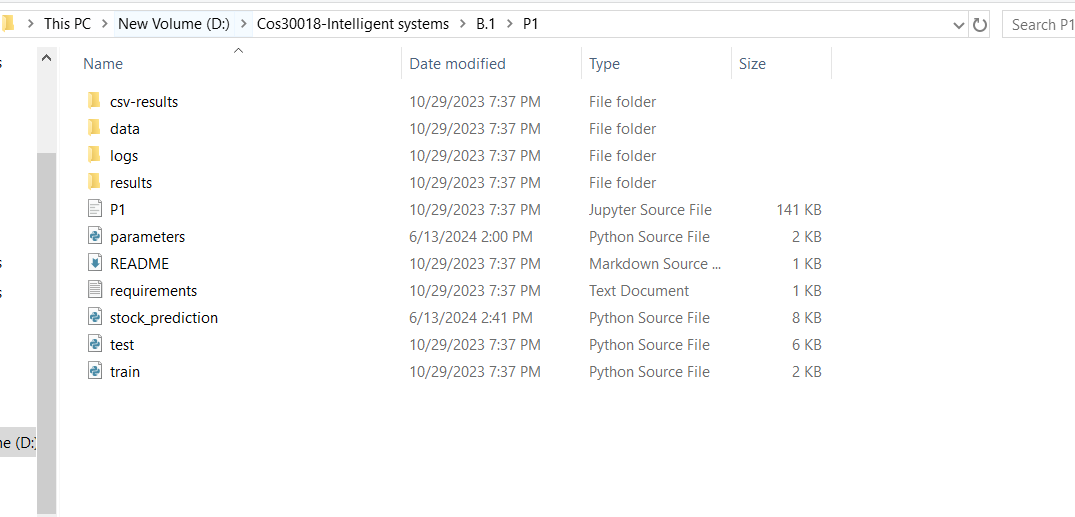
**Student Id: 104169837**

1. Setting up the environment:

* We started downloading the code v0.1 and P1 from the canvas:

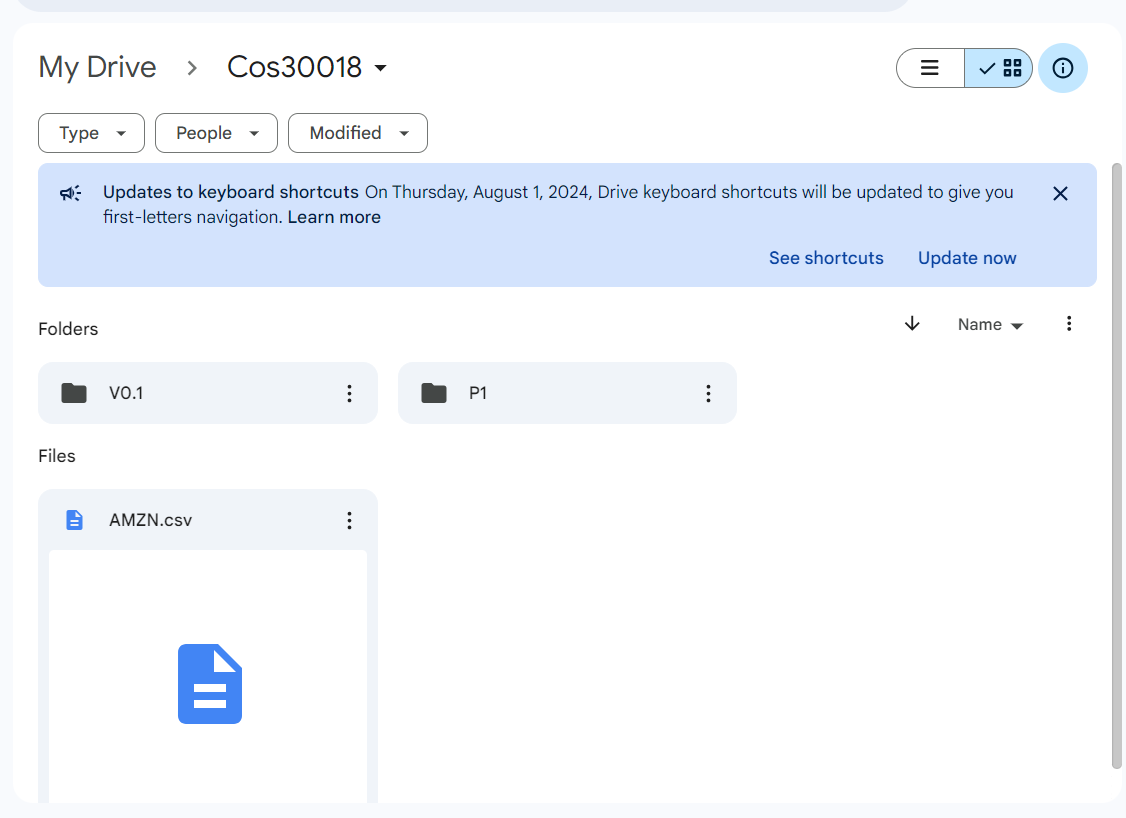


**Figure 1: Downloading the v0.1 code from canvas into the computer.**



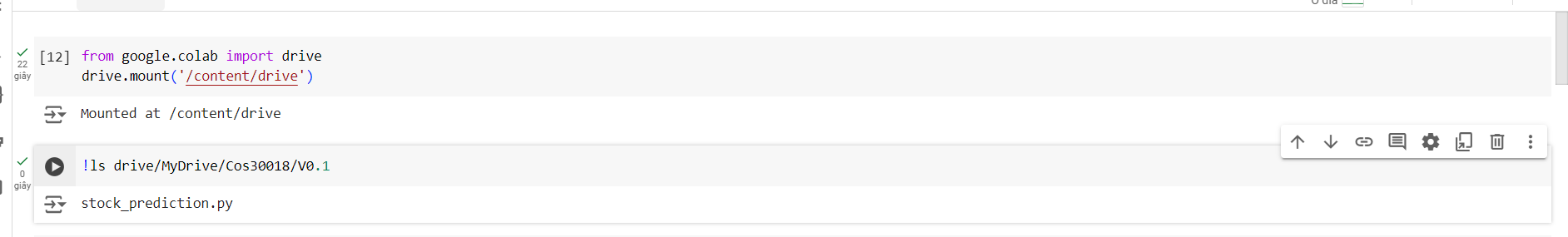
**Figure 2: Downloading the P1 code from canvas into the computer.**

* We upload both v0.1 and P.1 folder into the google drive to link to gg drive (we can still run the code even if it’s not link to the google drive or not):



**Figure 3: Uploading the v0.1 and P1 code into folder in google drive.**

* After that, we open the gg collab and start to link to our gg drive that contain folders and direct to the folder locations:



**Figure 4: Linking the v0.1 and P1 code from google drive to google collab.**

* We import the libraries or extensions that we need to run the code and use “!pip install ….”install the libraries if they are still not available in the gg collab:



**Figure 5: Importing and installing libraries.**

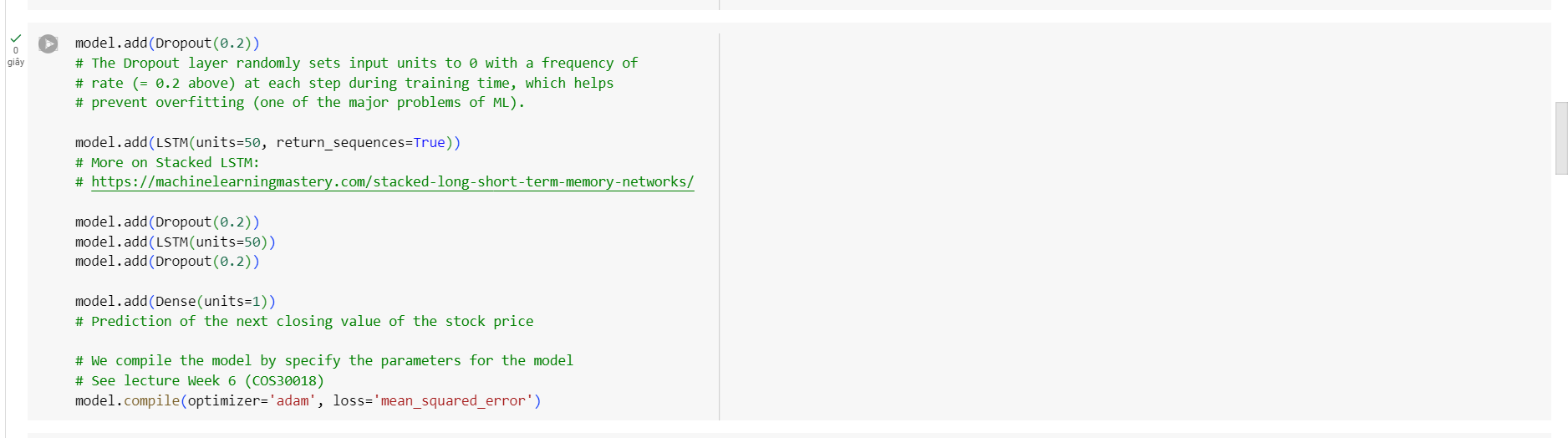
* After completing import the resources, we start to implement code into the gg collab to run it but we implement separately to test out if the code run or not:



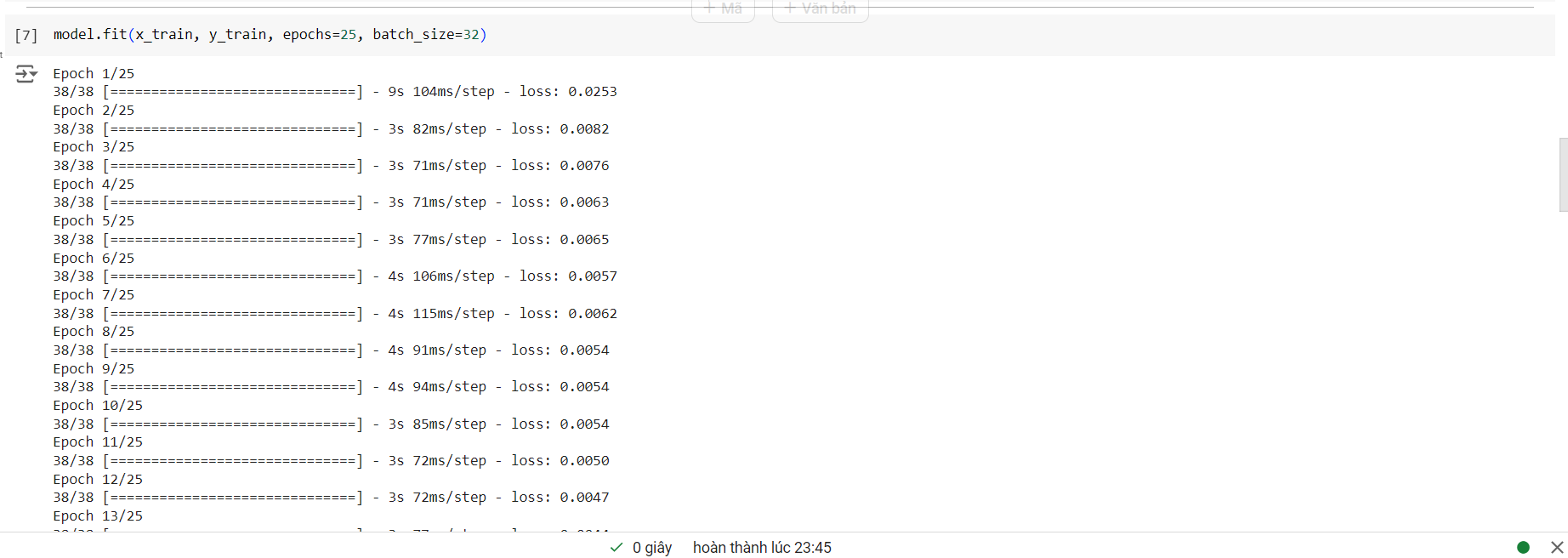
**Figure 6: Implementing the code separately to check if it works or not (1).**



**Figure 7: Implementing the code separately to check if it works or not (2).**



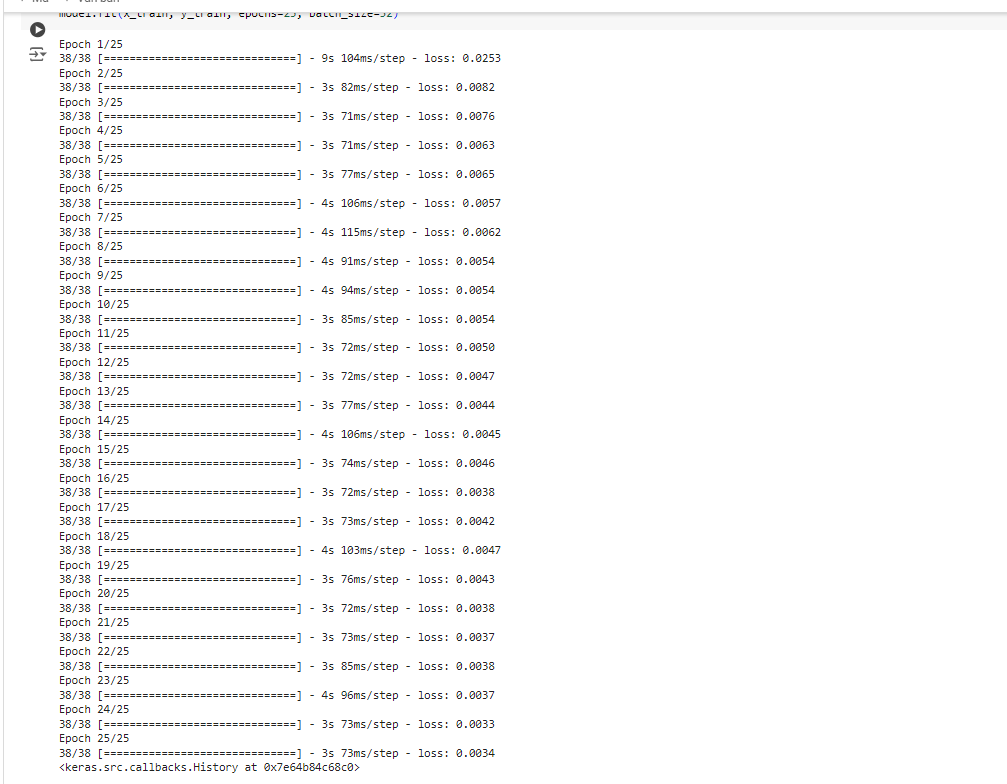
**Figure 8: Implementing the code separately to check if it works or not (3).**



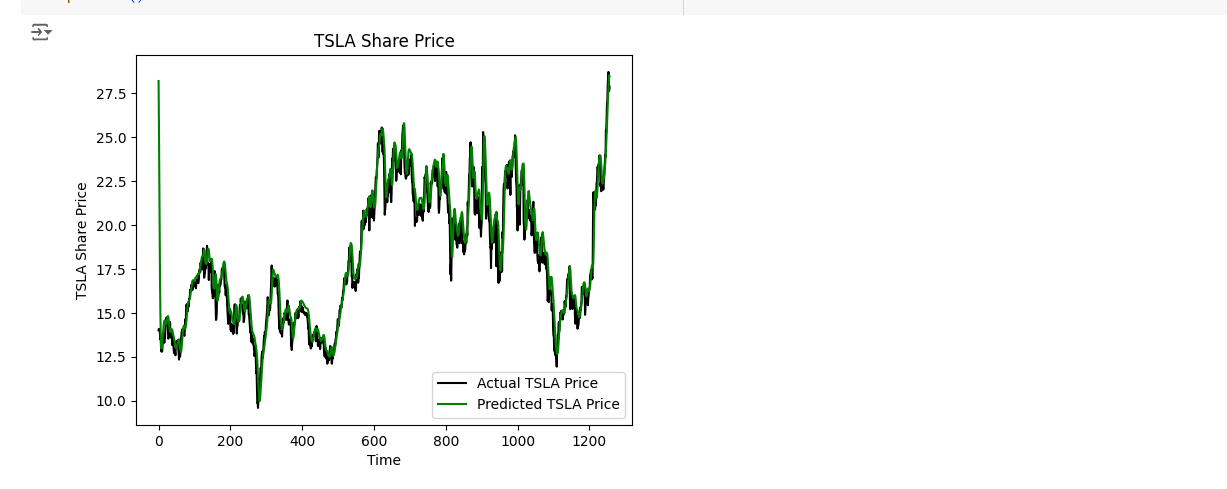
**Figure 9: Implementing the code separately to check if it works or not (4).**

1. Testing the code v0.1 and P1:

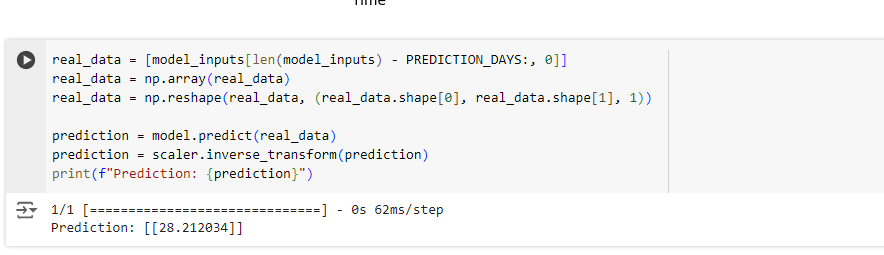
* We run the v0.1 code and train through 25 epochs before creating the charts to show the accurate and predicted data of TSLA stock. After creating the chart, the model predicts the stock price for the next day after the test period.
* The Sequential API from Keras is used to build the model. To avoid overfitting, it is composed of three LSTM layers, each of which is followed by a Dropout layer.



**Figure 10: Training the model through 25 epochs before printing the chart.**



**Figure 11: The prediction chart after training.**

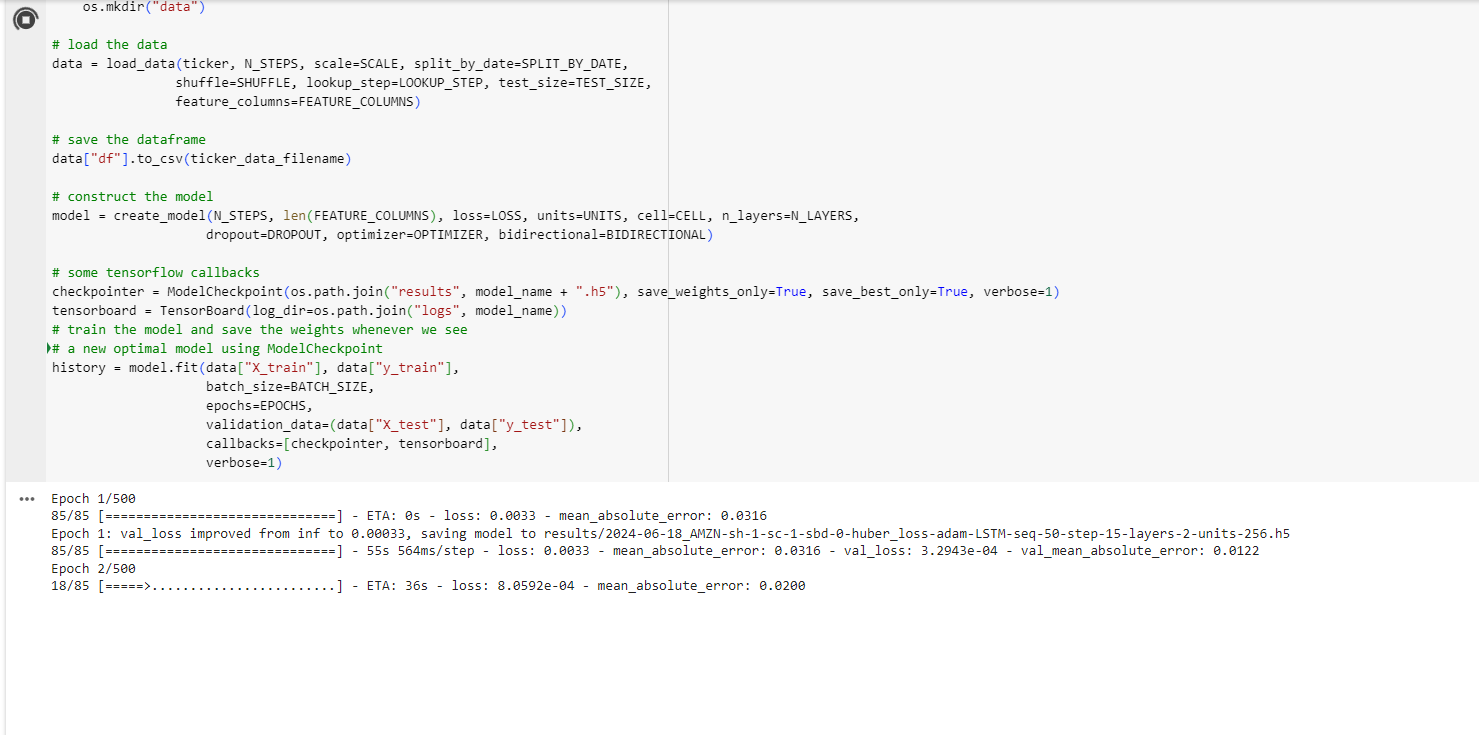


**Figure 12: Printing the prediction for next day.**

* When we run the P1 code, it has to go through 500 epochs to train which take a long time to do it so we decide to reduce it to 15 epochs by editing the epochs in the parameter.py.

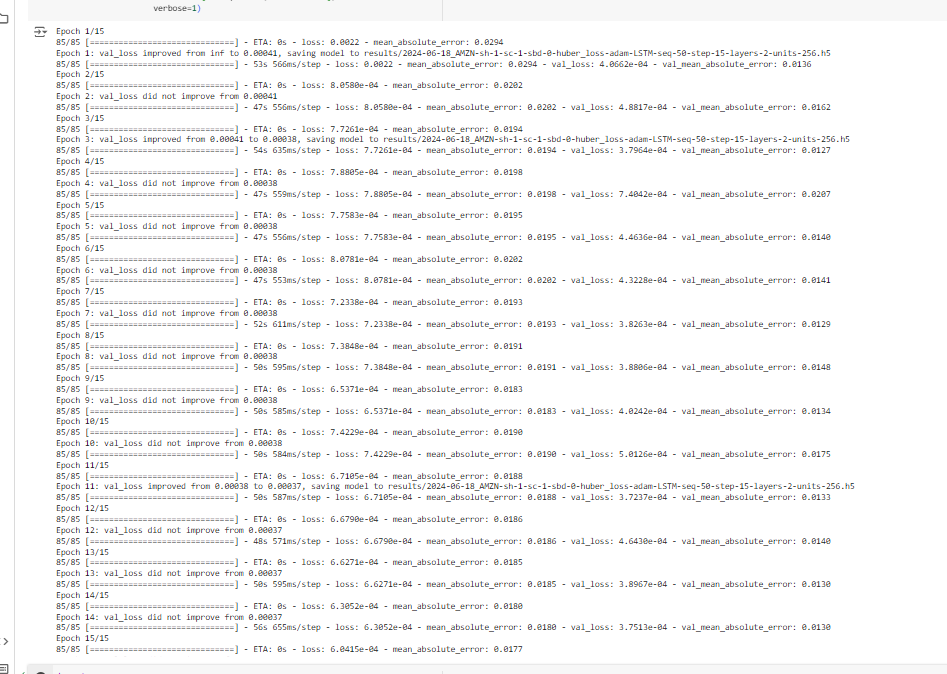


**Figure 13: We need to replace the epochs from 500 to 15 for faster train.**

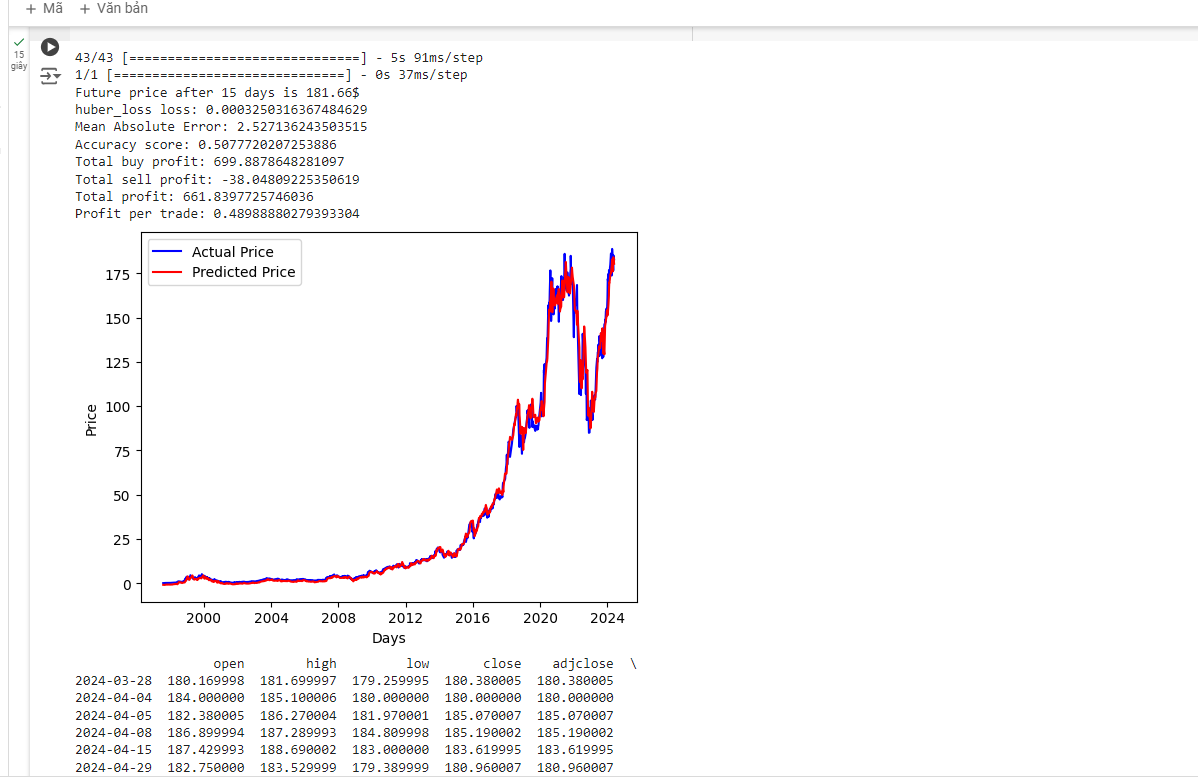


**Figure 14: The training duration when we set to 500 epochs.**

* After we set the epochs to 15 and train the model, we print the chart and results by running the test.py.

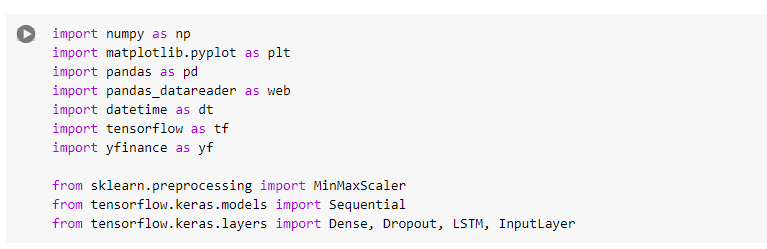


**Figure 15: The training duration when we set to 15 epochs.**



**Figure 16: The prediction chart when after train model set to 15 epochs.**

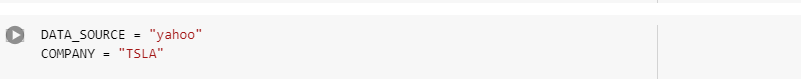
1. The understanding of the code base v0.1:
2. Import libraries.



**Figure 17: Importing the libaries.**

* Before we start running the code, we need to implement the libraries. These are the libraries required for date-time modification (datetime), machine learning (tensorflow, keras), data visualization (matplotlib), and data management (numpy, pandas, pandas\_datareader). The purpose of yfinance is to get historical stock data.

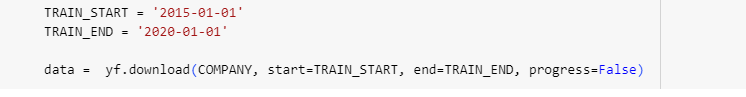
1. Set data source and company.



**Figure 18: Setting the data resources from yahoo and get the TESLA data.**

* To retrieve the stock data for Tesla, “DATA\_SOURCE” is set to "yahoo" (using Yahoo Finance), and “COMPANY” is set to "TSLA" (Tesla).

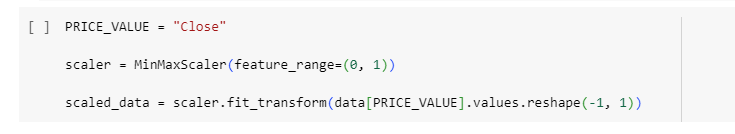
1. Define training period and fetch data.



**Figure 19: Setting the time to train and fetch the TESLA data.**

* The data source, the ticker symbol of the business whose stock values are being examined, and the date ranges for the training and testing datasets are all defined in this section using constants. The stock data will be retrieved from Yahoo Finance, as indicated by the setting of “DATA\_SOURCE” to "yahoo".

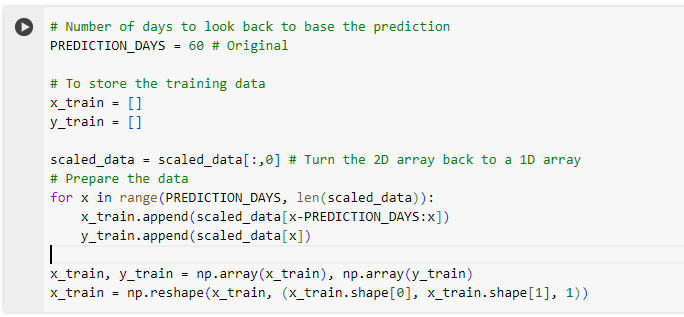
1. Prepare the data.



**Figure 20: Scaling the data.**

* For predicted purposes, the stock's closing price is chosen. Using “MinMaxScaler”, the data is scaled to a range of 0 to 1, which enhances the performance of the model.

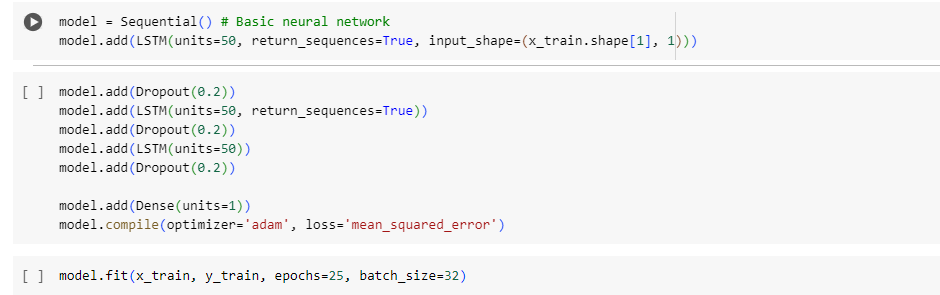
1. Prepare training data.



**Figure 21: Setting the training data part.**

* The model forecasts the price for the following day using data spanning 60 days. Two arrays are created from the data: “x\_train” (input) and “y\_train” (output). As required by the LSTM model, the “x\_train” array is reshaped to be three-dimensional.

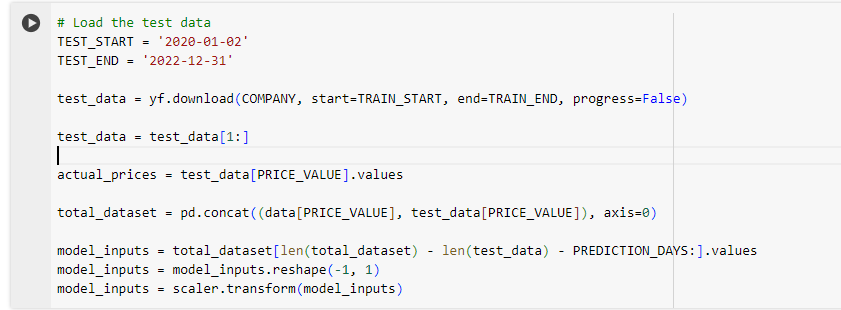
1. Build the model.



**Figure 22: Building the model after training.**

* The Sequential API from Keras is used to build the model. To avoid overfitting, it is composed of three LSTM layers, each of which is followed by a Dropout layer. The anticipated stock price is produced by the last Dense layer. The Mean Squared Error loss function and the Adam optimizer are used to construct the model. After that, it is trained for 25 epochs with a batch size of 32 using the training data.

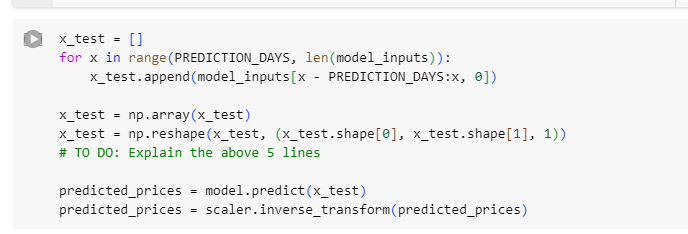
1. Load and prepare test data.



**Figure 23: Loading the test data.**

* Similar to how the training data is retrieved and prepared, as well is the test data. For the duration of the test, “actual\_prices” contains the actual stock prices. The scaled values needed for prediction are stored in the “model\_inputs” array.

1. Prepare test inputs and predict.



**Figure 24: Preparing the test input and begin to predict.**

* In order to create input sequences for the LSTM model to make predictions, the test data is prepared in the same way as the training data. These sequences are initially stored in an empty list called “x\_test”. For the test data, the stock prices are predicted by the trained LSTM model. Predicted prices are generated in a series based on the test sequences that have been developed. The “MinMaxScaler”'s inverse transform function is then used to scale these forecasted prices back to their initial range.

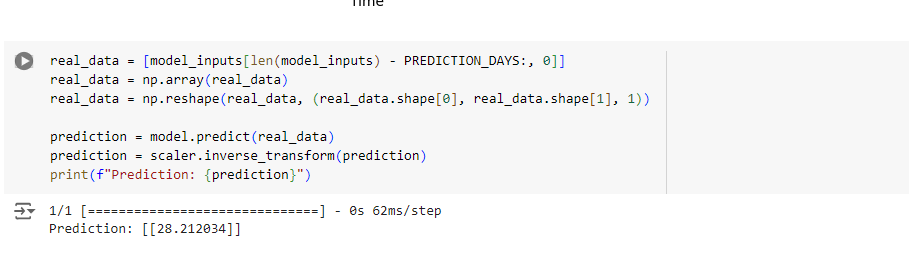
1. Visualize the results.



**Figure 25: Printing the result and prediction chart.**

* The script uses matplotlib.pyplot to plot the actual and predicted stock prices in order to evaluate the effectiveness of the model. Plotting the expected costs is done in green, and plotting the actual prices is done in black. The x- and y-axes of the plot are labeled "TSLA Share Price," in accordance with its title.

1. Predict the next day.



**Figure 26: Printing the prediction for the next day.**

* Finally, following the test period, the model predicts the stock price for the following day. This illustrates how the model may forecast future events using the most recent data.