COS30018 - Option B - Task 1: Setup

Cody Cronin-Sporys

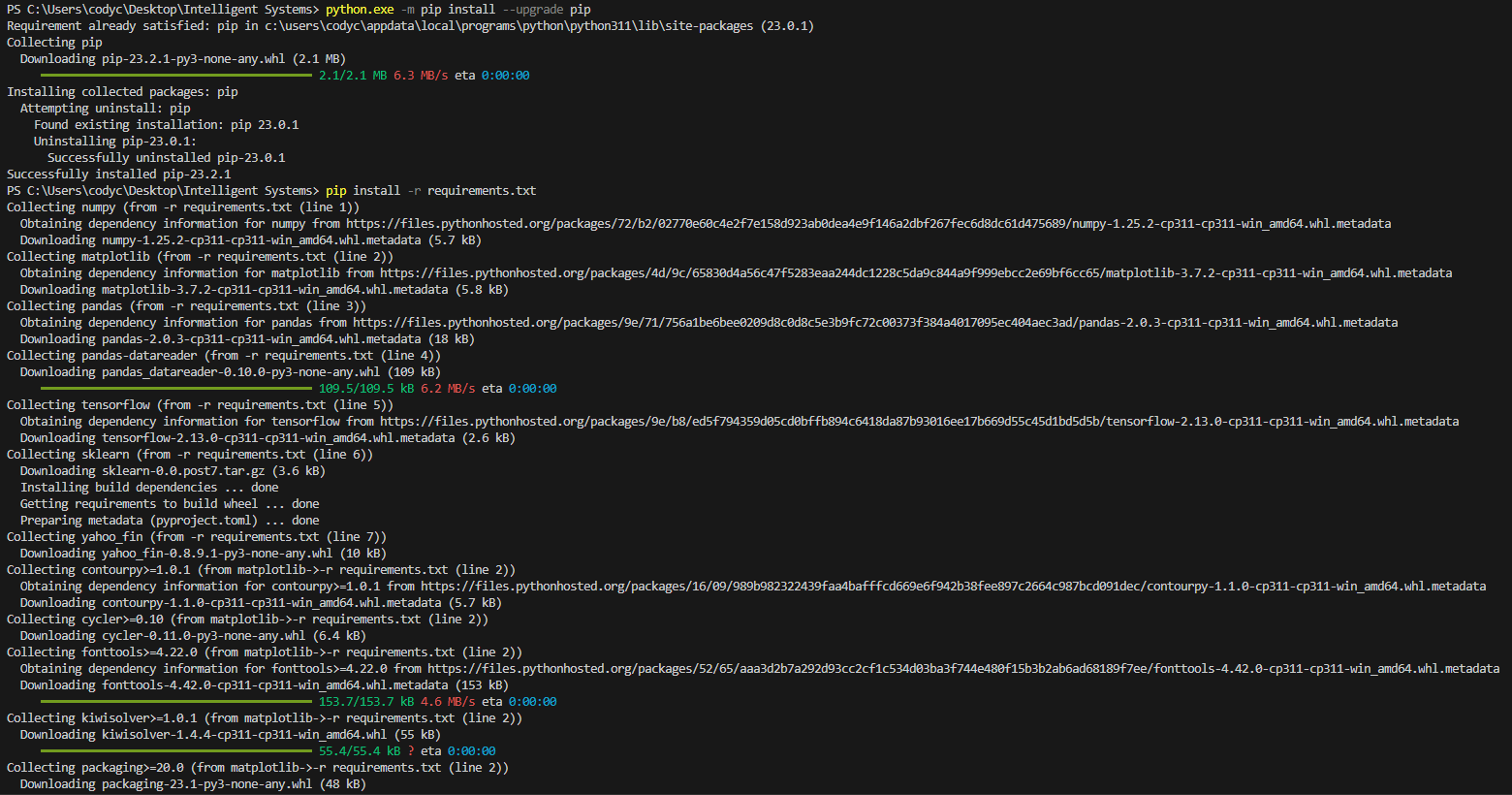
103610020

# The v0.1 Process:

The first step was to download the code file (v0.1) and follow along with the tutorial. I already had python setup in vs code so the first problem I came across was importing all the libraries. I read the information on requirements files, so I wrote down all the missing libraries into a text file. I also check the P1 requirements file and added those libraries as well before running pip install requirements.txt.

A screenshot of a computer

Description automatically generated



After first updating pip, the libraries installed successfully. Despite this, the code was still showing the same errors for all the libraries.

A screen shot of a computer program

Description automatically generated

After reloading and making sure the workspace and code file were trusted, most of the issues resolved themselves. A few of the libraries (like yfinace) were still showing errors because I didn’t have them added to requirements, so I updated and re-ran the requirements file.

This fixed all the imports except the bottom 2. Both “from tensorflow.keras…” were still showing issues however the code seemed to be able to run. I looked around for fixes but couldn’t find any so for now I’ll leave it as is and come back if it causes problems.

After running the code, it completed 25 steps of epoch in the console, then produced this graph and a prediction value.

A screenshot of a computer program

Description automatically generated

A graph with green lines and numbers

Description automatically generated



The graph shows predicted prices, with one day predicted price being generated from 60 days of existing data. The prices being predicted in this graph are in the past, so we already have real data to compare with, which is also shown on the graph. This allows us to check the accuracy of the prediction.

The single prediction value shown in the console is an actual future prediction on data we do not have yet.

# Understanding of v0.1 code:

In the code, The first step is obtaining the raw data which is done in this code section:

A black screen with white text

Description automatically generated

Next we need to manipulate the data so that it’s usable for training our model. We do this through scaling and reshaping the data. We also store a limited amount of the data based on the amount of days we are predicting from.

A black screen with text

Description automatically generated with medium confidence

A screen shot of a computer code

Description automatically generated

Next we need to build the model. In this code we are using a layered LSTM model.

A computer screen with text and images

Description automatically generated

We also setup various parameters for the model, such as the optimizer and loss.





Then we train the model using the data we prepared earlier.

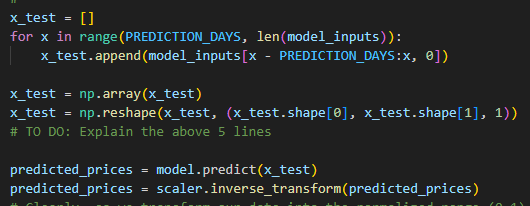


Once the model has been trained, we start testing it. The first test is comparing it against existing data so we can judge it’s accuracy. The first step is again downloading and setting up the data for our model.

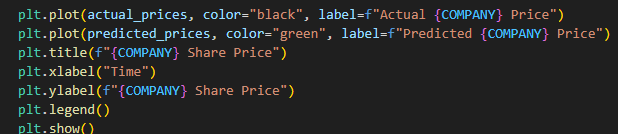
A computer screen with text

Description automatically generated

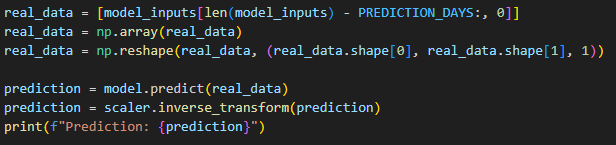
We then use this data and model to prepare a test and predict the next day’s prices. The prediction also needs to have its transformations reverse so it’s readable.



To visualise this data and the prediction we setup a plot with various labels and the two datasets (actual and predicted).



The final thing we do is a similar prediction, but this time it’s not on existing data. Instead of showing the results in a graph/ plot we just print the result to console.



# The P1 Process:

For the P1 code base I downloaded a zipped of the pythoncode-tutorials repo and extracted the stock-prediction folder. I then opened this folder in visual studio code. Since all the libraries were installed for the v0.1 code base there’s no need to do it again here.

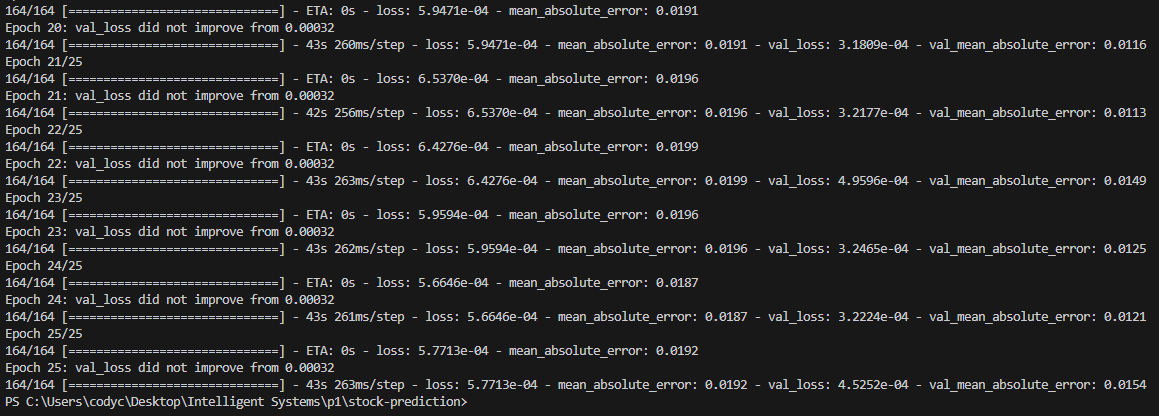
A screenshot of a computer

Description automatically generated

The first step is to edit the parameters.py if needed then start running the train.py code. After starting it with the default parameters I realised the batch size and number of steps was much larger than v0.1 so I reduced it down to save time, since this demonstration doesn’t need super accurate results.

A black background with blue text

Description automatically generated



Now that the training is done we can run test.py to see the results of the model. This will show a similar graph to v0.1 and also some extra information on the prediction in the console.

A graph with red and blue lines

Description automatically generated

A screen shot of a computer

Description automatically generated