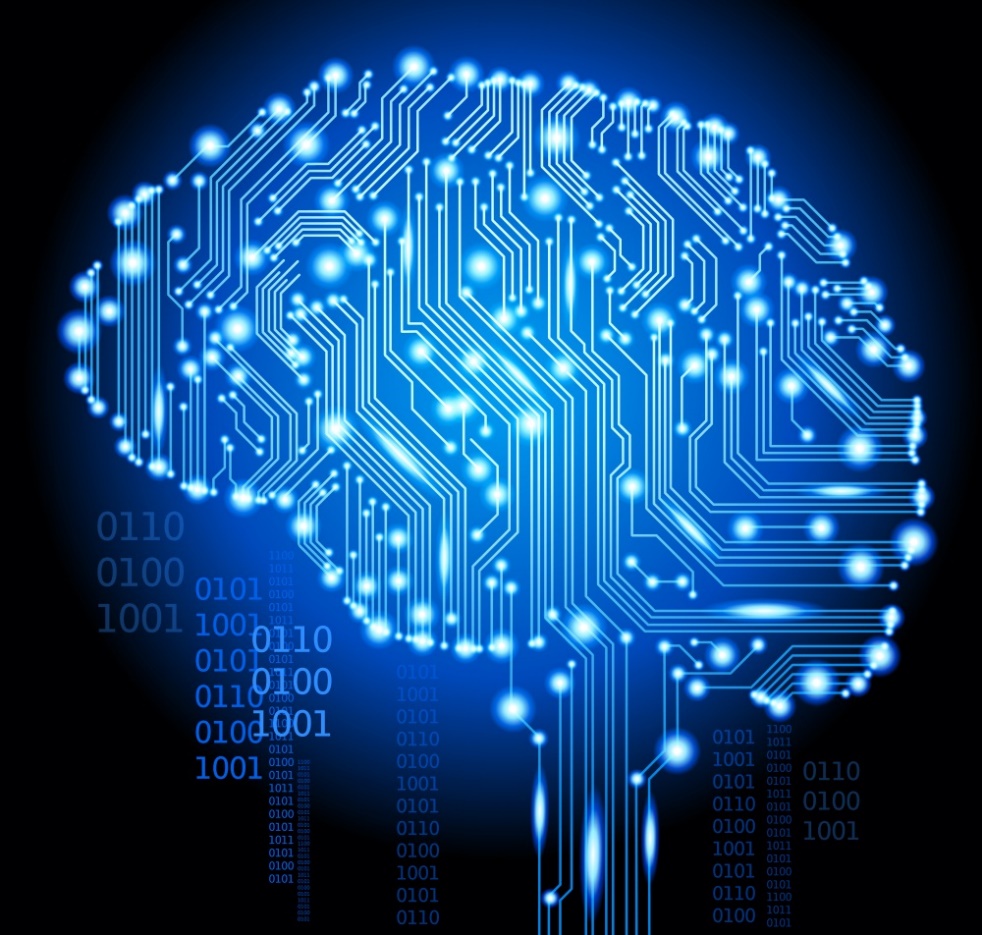


COS30018 – Intelligent Systems

**REPORT**

TASK B1 – SETUP



**TABLE OF CONTENTS**

*Table of contents…………………………………………………………………………………………………………………2*

*I. Setup……………………………………………………………………………………………………………………………….3*

*1. Setup environment on local machine………………………………………………………………………3*

*2. Setup virtual environment………………………………………………………………………………………4*

*3. Setup project………………………………………………………………………………………………………….5*

*II. Testing…………………………………………………………………………………………………………………………….7*

*1. Test* ***P1****…………………………………………………………………………………………………………………..7*

*2. Test* ***v0.1****………………………………………………………………………………………………………………10*

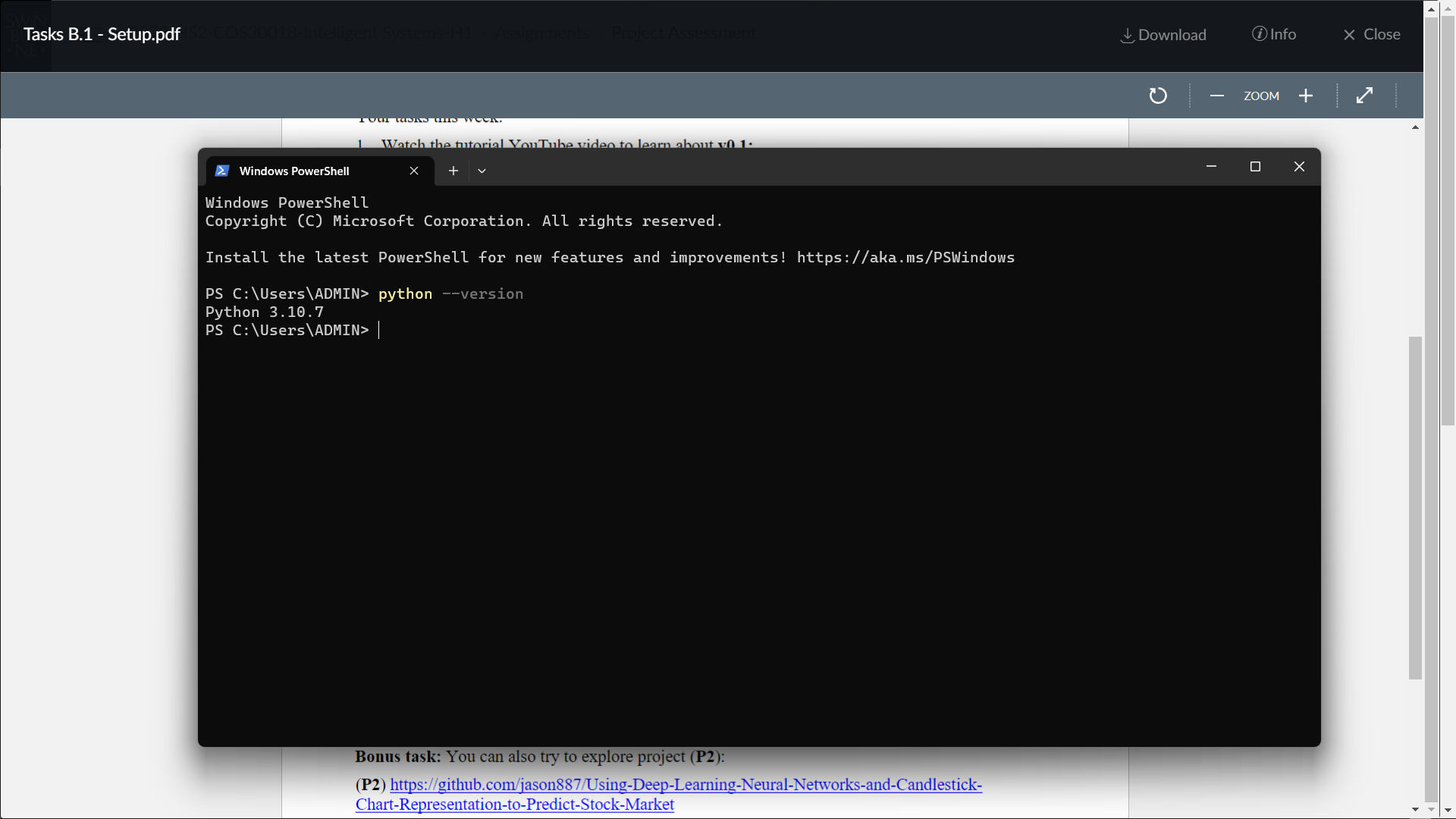
*3. Test* ***P2****…………………………………………………………………………………………………………………11*

*III. Summary…………………………………………………………………………………………………………………….17*

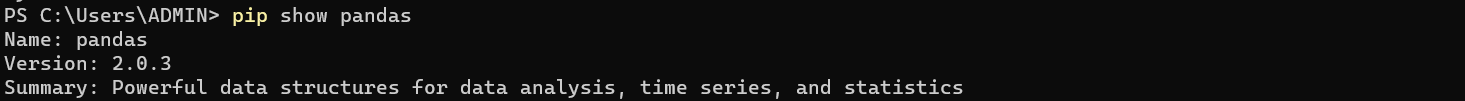
**SETUP**

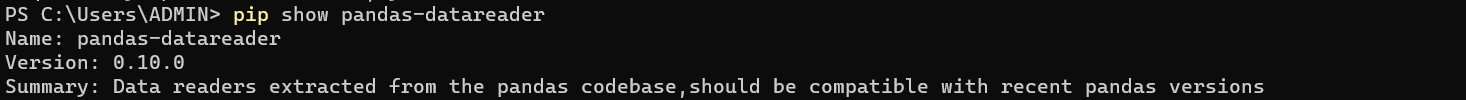
**1. Setup environment on local machine**

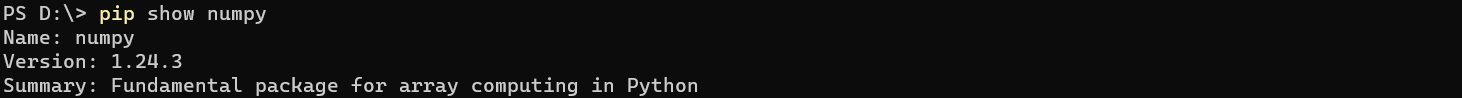
First, I have installed Python, version 3.10.7, from <https://www.python.org/downloads/>

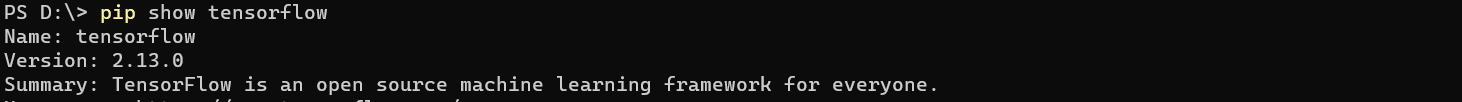
**

Then is the installation of some necessary libraries for the project, including *pandas*, *pandas-datareader*, *numpy*, *tensorflow*, *yfinance* and *scikit-learn* using *pip*:

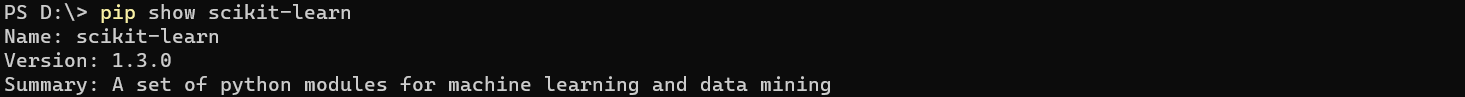








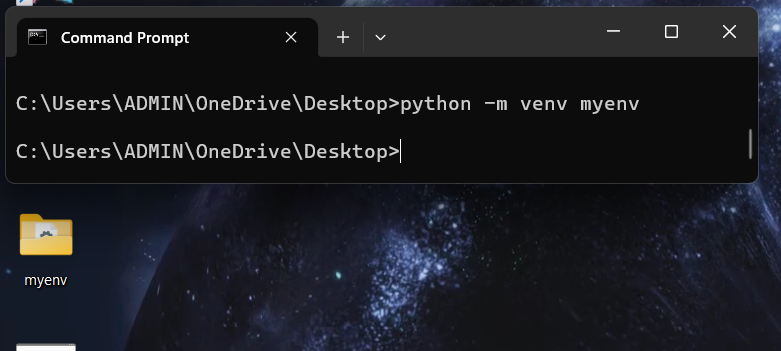




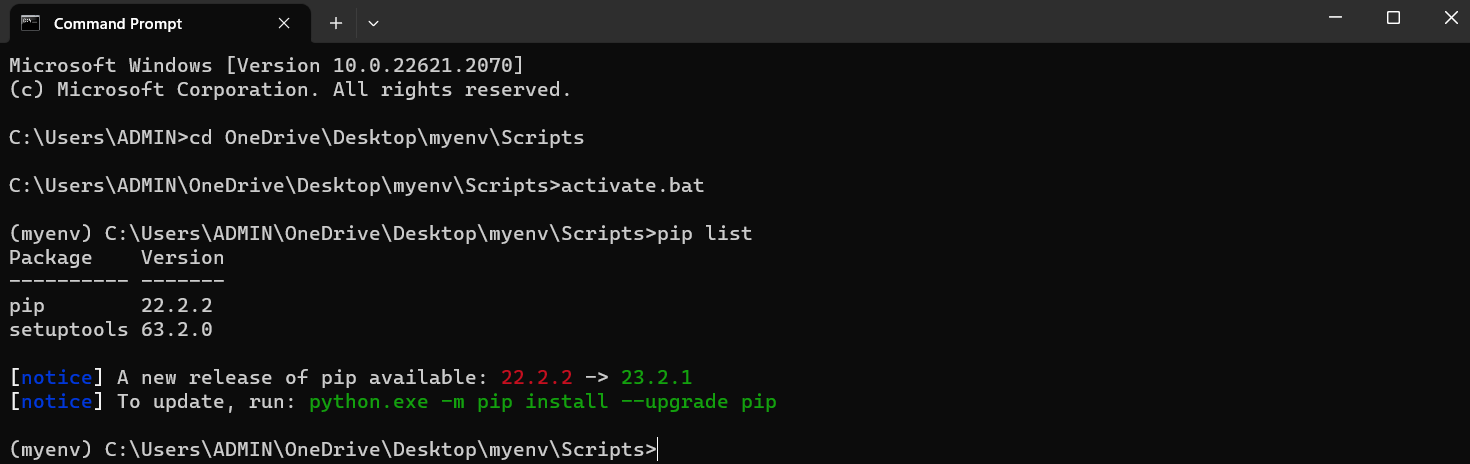
**2. Setup virtual environment**

To create a Python virtual environment, I have used the library of *venv* as a common virtual environment library for Python.

I have created a new virtual environment, and named it “myenv”:



To activate the virtual environment, I change the directory into “myenv\Scripts”, and run the “activate.bat” file:



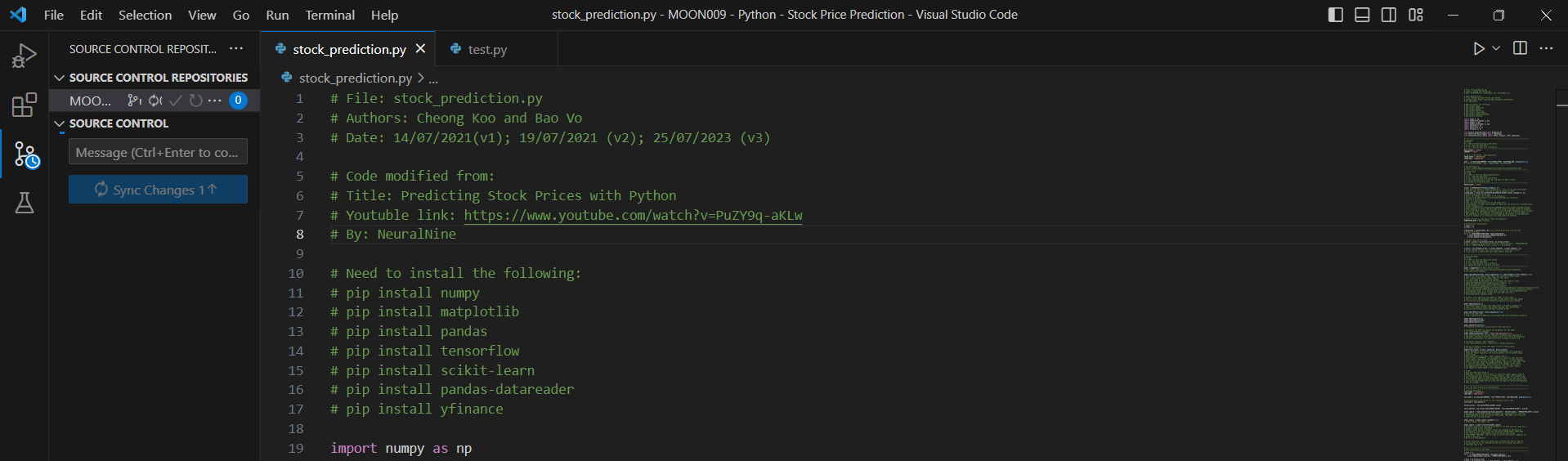
Now, It is possible to install Python packages or libraries without affecting the system-wide Python installation on my local machine.

To deactivate it, I run the deactivate.bat instead.

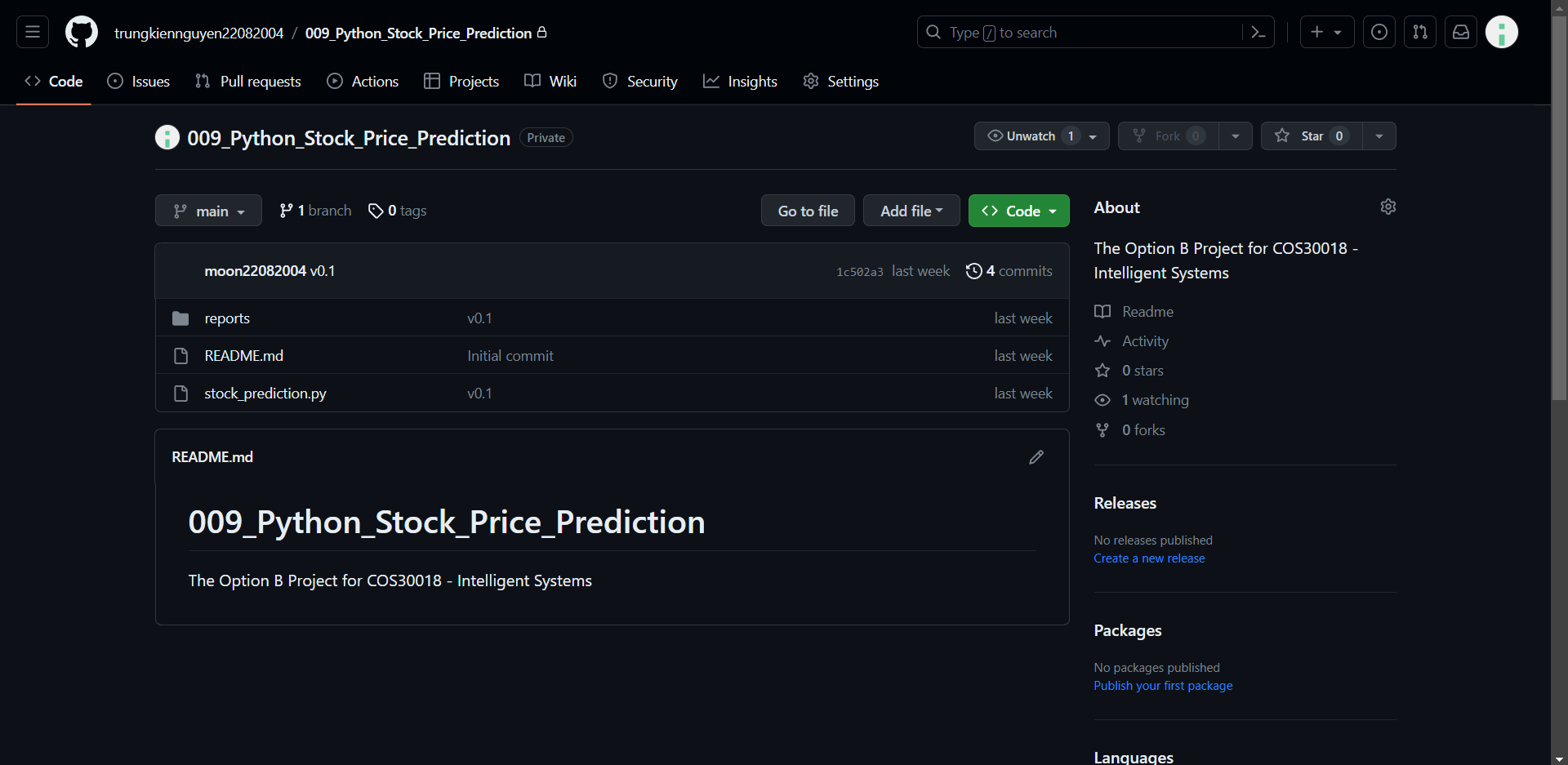


**3. Setup project**

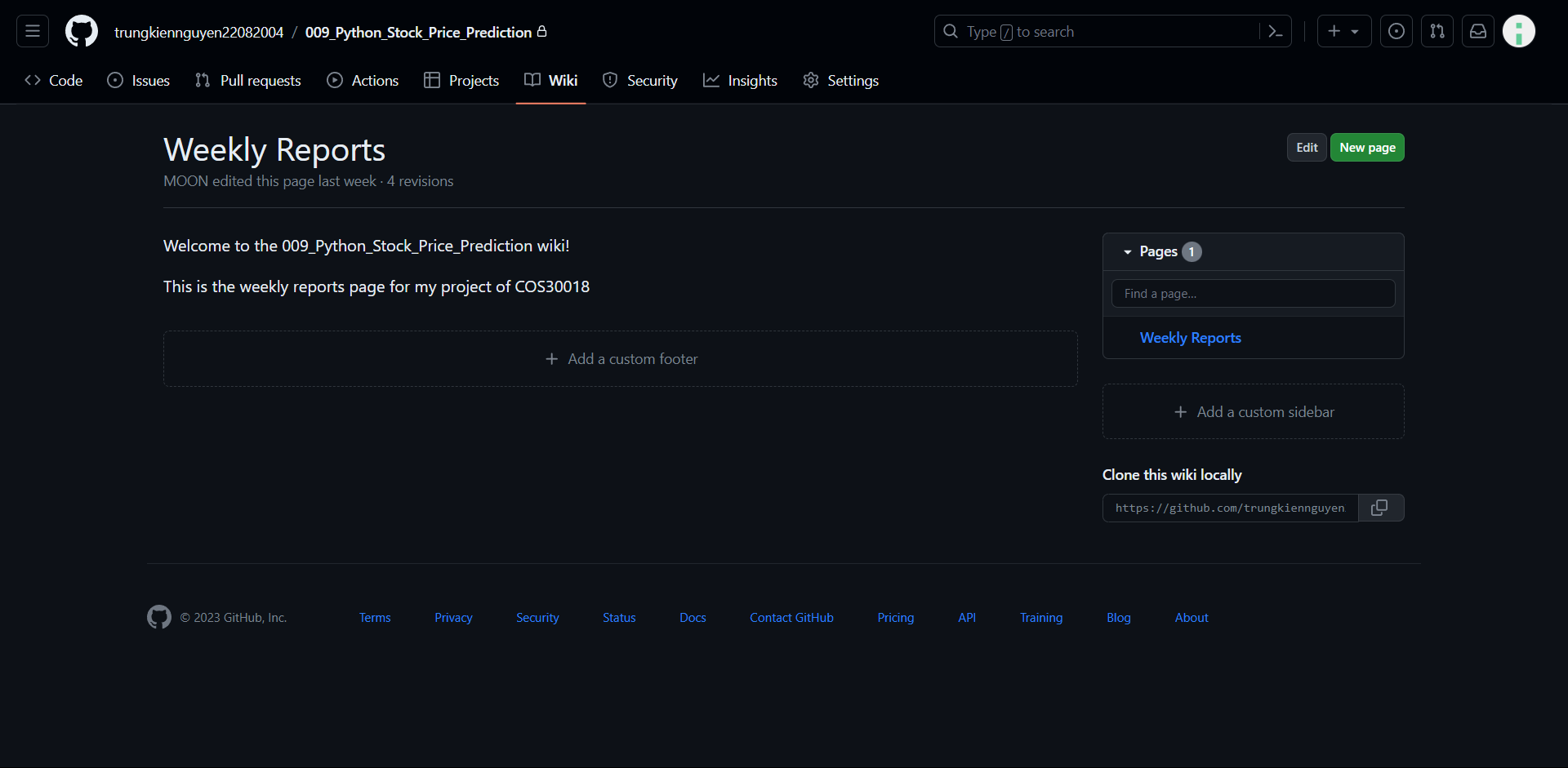
I have setup the GitHub repository for the unit project, clone it to my device, adding “README.md” and “stock\_prediction.py” files to the folder, then commit and push to my repository, through the use of “SOURCE CONTROL REPOSITORIES” tool in Visual Studio Code.



My GitHub account’s username is “trungkiennguyen22082004”, and the project repository names “009\_Python\_Stock\_Price\_Prediction”:



*GitHub repository for the project*



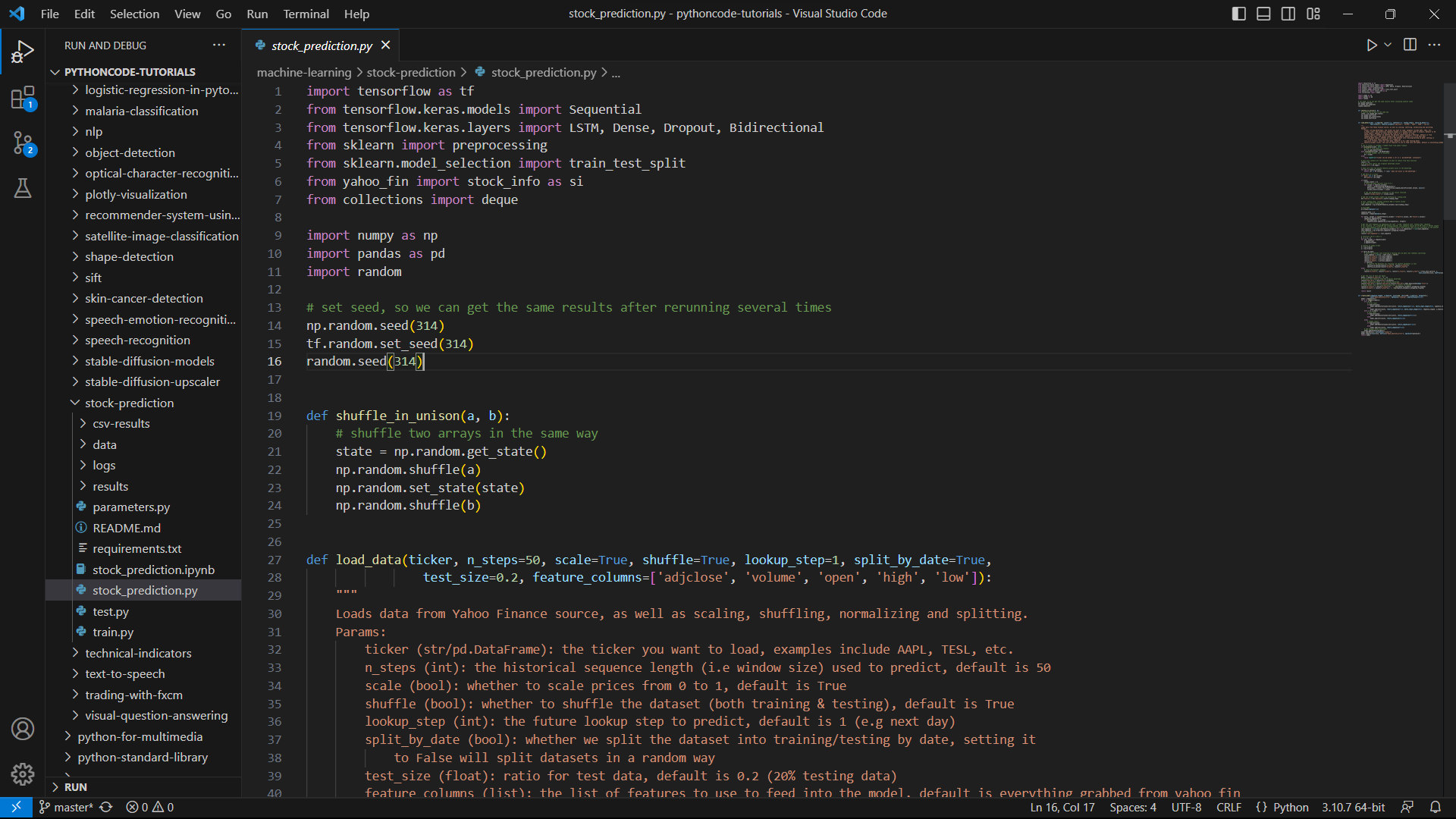
*And its Wiki page for weekly reports*

**TESTING**

To make sure that the Python packages and libraries used for my main project do not conflict with those used in **P1** and **P2**, I test the **P1** and **P2** in two separated virtual environments, while running **v0.1** in my local machine’s environment.

**1. Test P1**

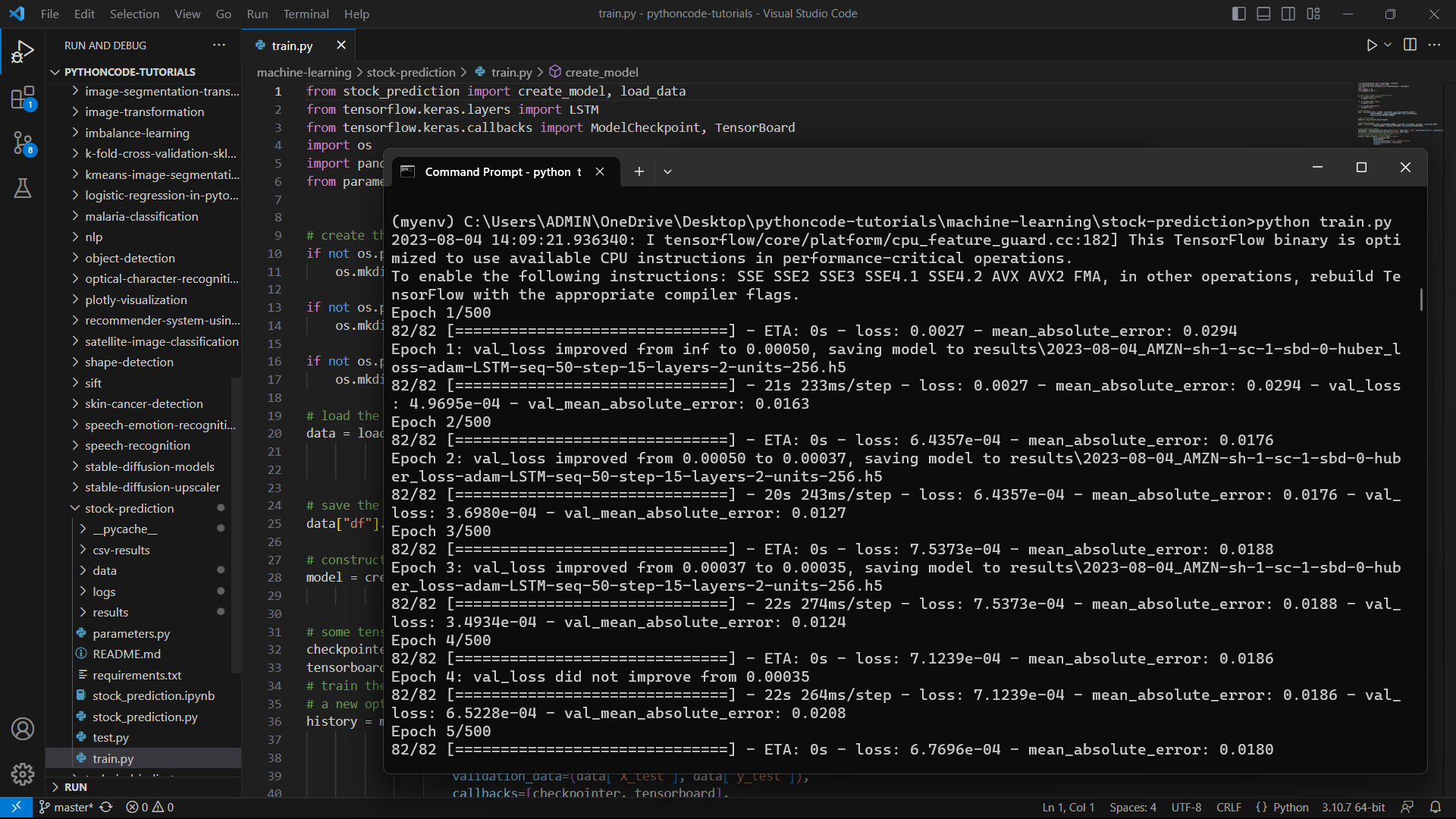
First, I have cloned the project <https://github.com/x4nth055/pythoncode-tutorials>, and open the folder “machine-learning\stock-prediction” (P1):

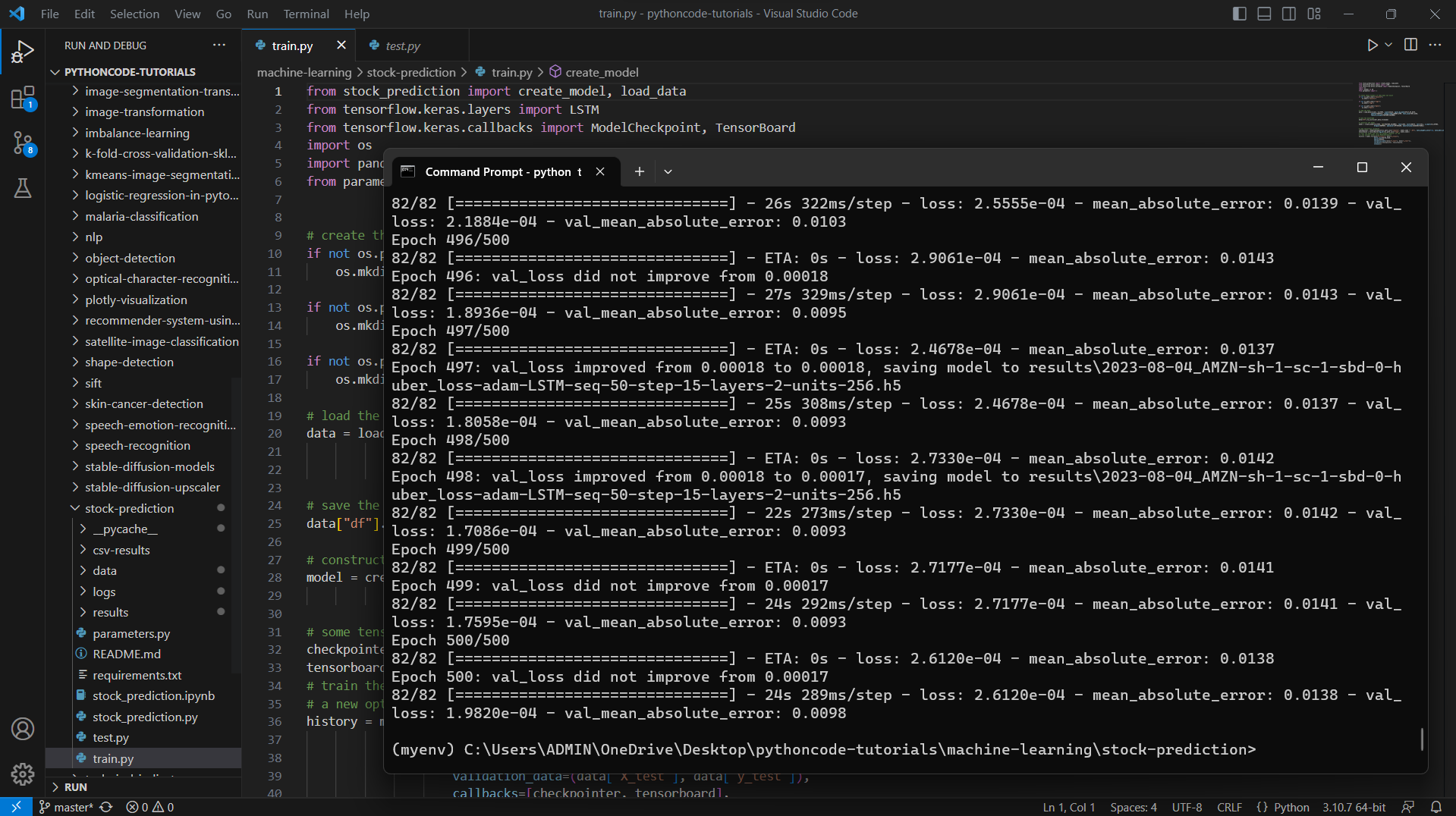


Using Command Prompt, I have activate my created virtual environment “myenv”, then install libraries used for the project as shown in the “requirements.txt” file, including *pandas*, *numpy*, *tensorflow*, *scikit-learn*, *yahoo-fin* and *matplotlib*.

After that, I have read the tutotial in <https://www.thepythoncode.com/article/stock-price-prediction-in-python-using-tensorflow-2-and-keras> as mentioned in the “README.md” file of the project.

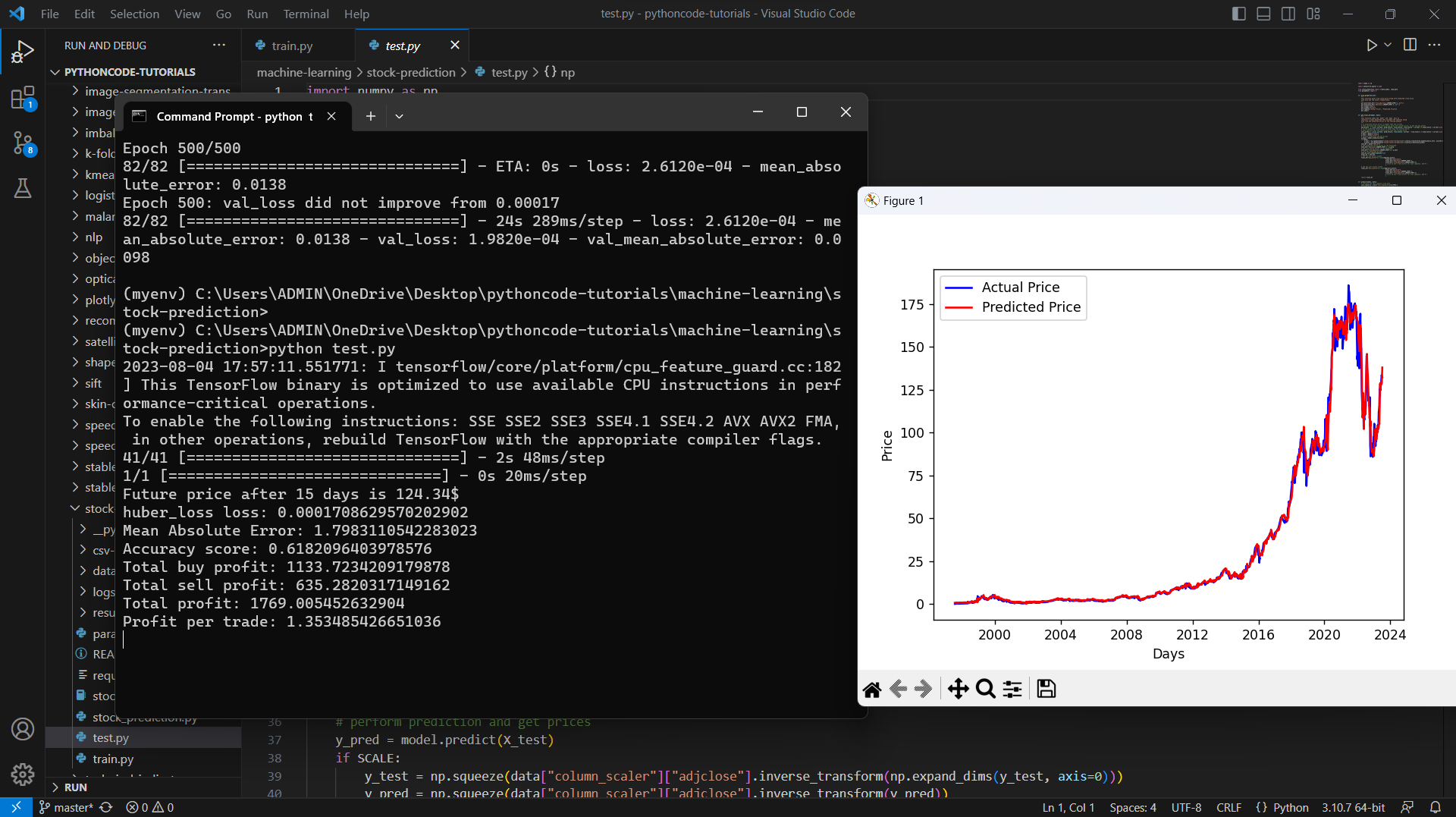
By running “train.py” file, the training has been started using the parameters specified in “parameters.py” file:

**

**

*It takes a total of 190 minutes to finish the training process*

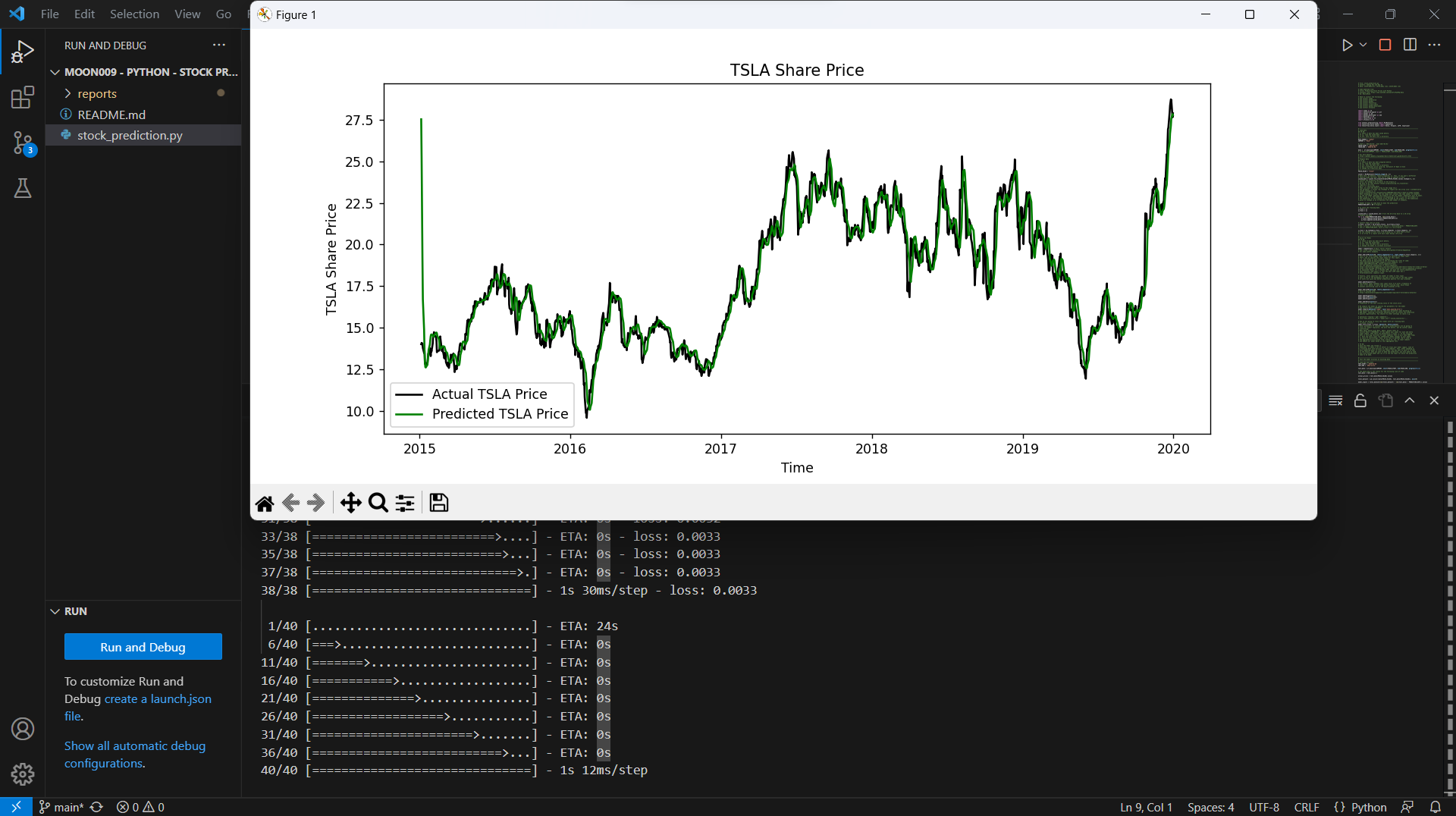
Following that, I use the file “test.py” to evaluate and test the model:

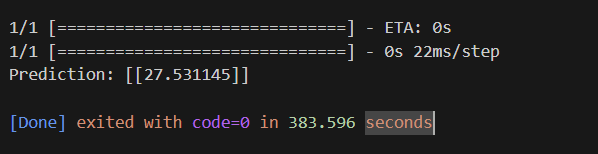


The result is a line graph showing of Amazon (AMZN)’s stock price in terms of prediction based on the actual one, from 2000 to 2024, using the data from Yahoo Finance.

**2. Test v0.1**

With the project that I have setup, I have run the project by running the file “stock\_prediction.py” in the local Python environment on my device:





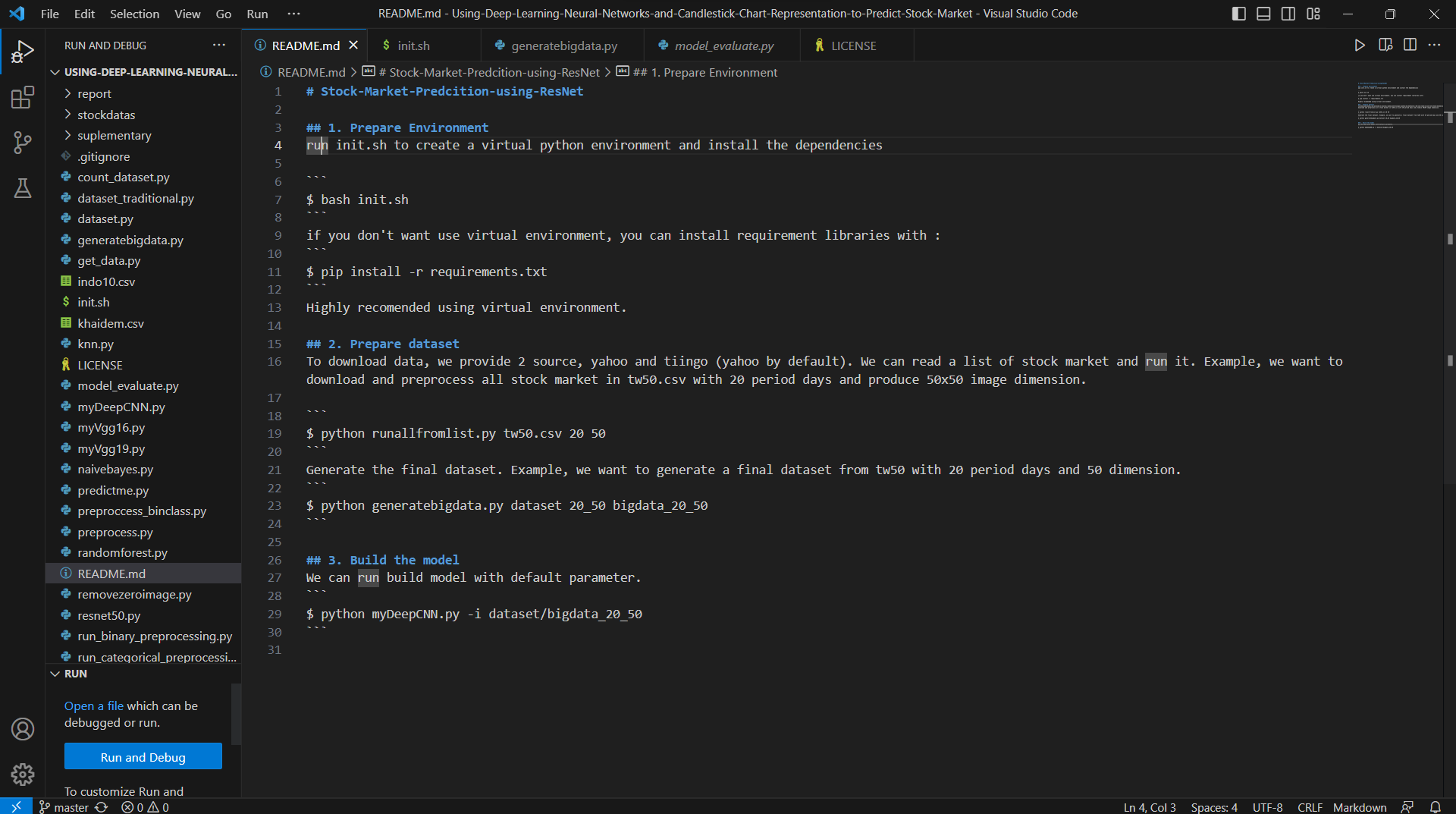
*It takes less than 1 minutes to finish the training process and give the output*

From the data from Yahoo Finance, the output is the line chart of actual and predicted stock price of Tesla (TSLA) between 2015 and 2020.

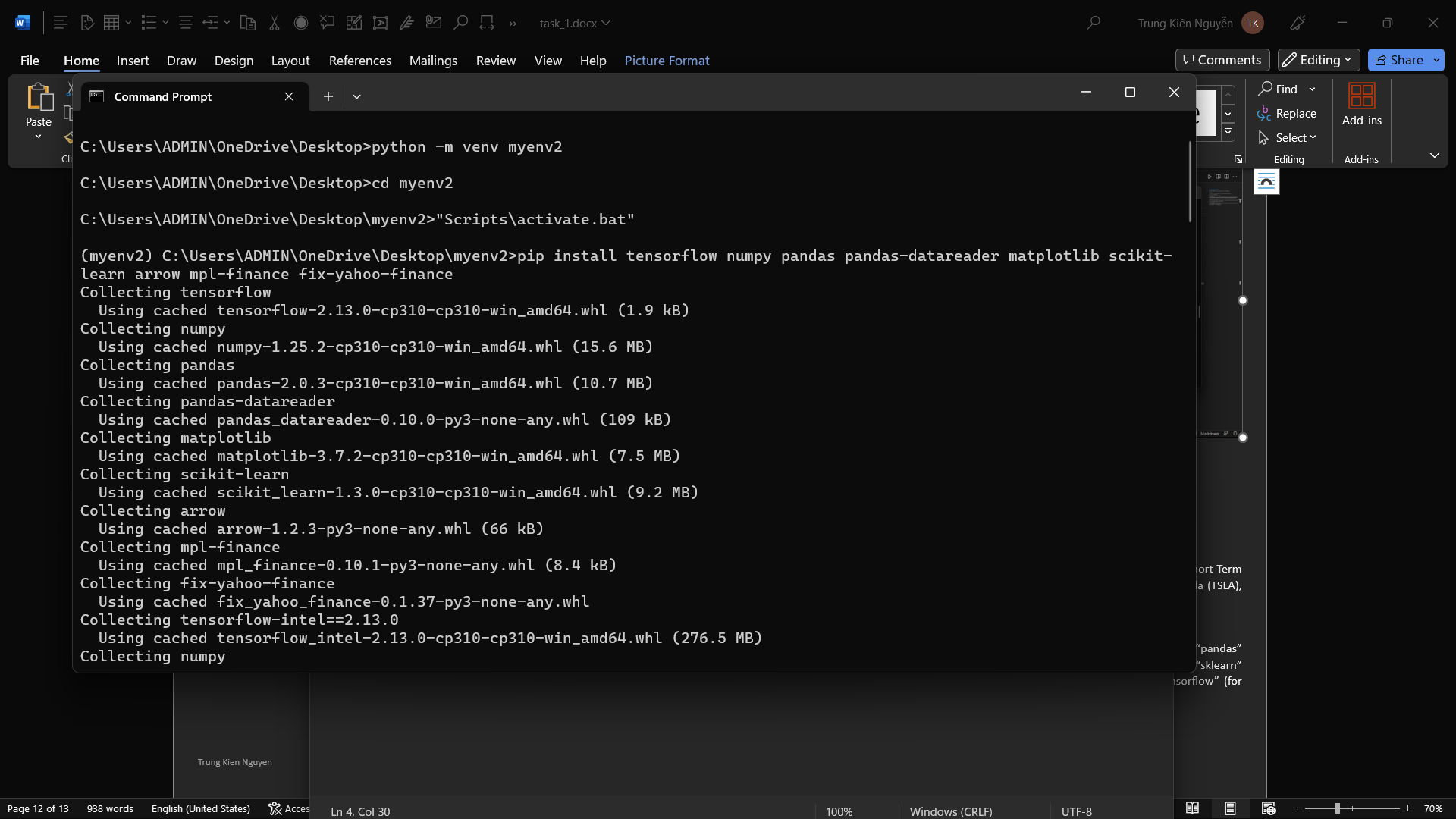
**3. Test P2 (Attempt)**

**3.1 Prepare Environment**

To start, I have cloned the project (P2) “Stock-Market-Predcition-using-ResNet” by the user “jason887” from <https://github.com/jason887/Using-Deep-Learning-Neural-Networks-and-Candlestick-Chart-Representation-to-Predict-Stock-Market>:



As the setup instructions seem to aim to Linux OS, as well as I can’t find the “requirements.txt” file for the project, I have to create a new Python virtual environment directly and install the necessary libraries for the project, instead of trying to run the “init.sh” file:

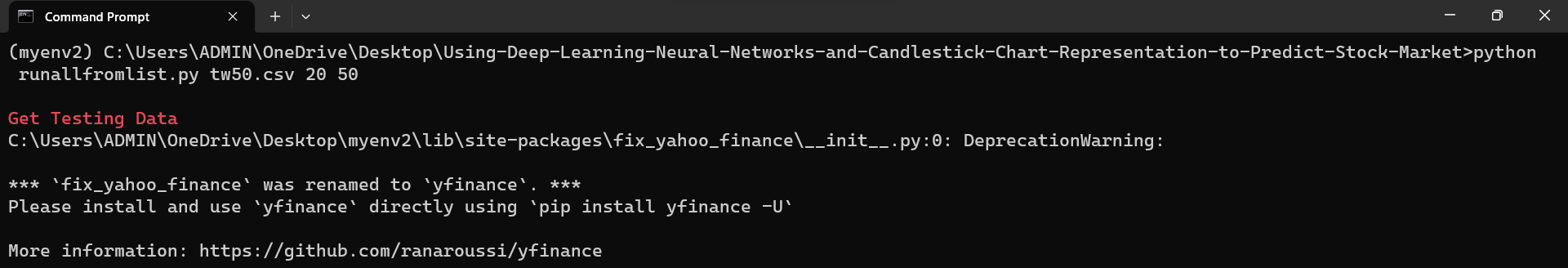


*Create and setup a new virtual environment of “myenv2” for the project*

**3.2 Prepare dataset**

First, I followed the “README.md” instruction, running the file “runallfromlist.py”, downloading and preprocessing all stock market in “tw50.csv” with 20 period days and produce 50x50 image dimension by this command: “python runallfromlist.py tw50.csv 20 50”. There are some problems shown up:

* Deprecation Warning

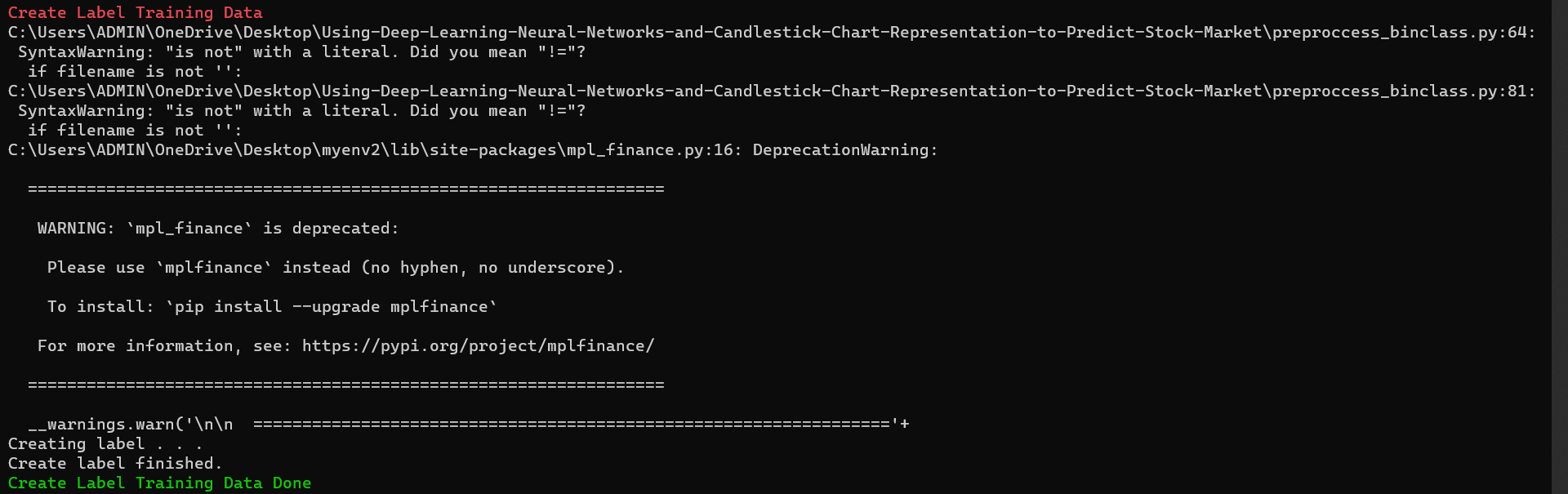
*“fix\_yahoo\_finance” was renamed to “yfinance”*

*Fix:*

* + Install “yfinance”:

**

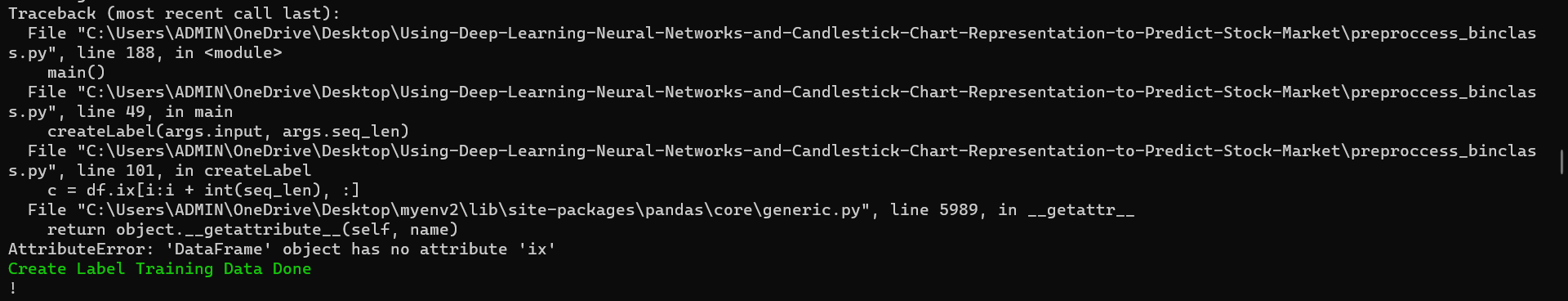
* + Change the importing code (from “import fix\_yahoo\_finance as yf” to “import yfinance as yf”) in the files “get\_data.py” and “predictme.py”.
* Deprecation Warning



*“mpl\_financ” is deprecated*

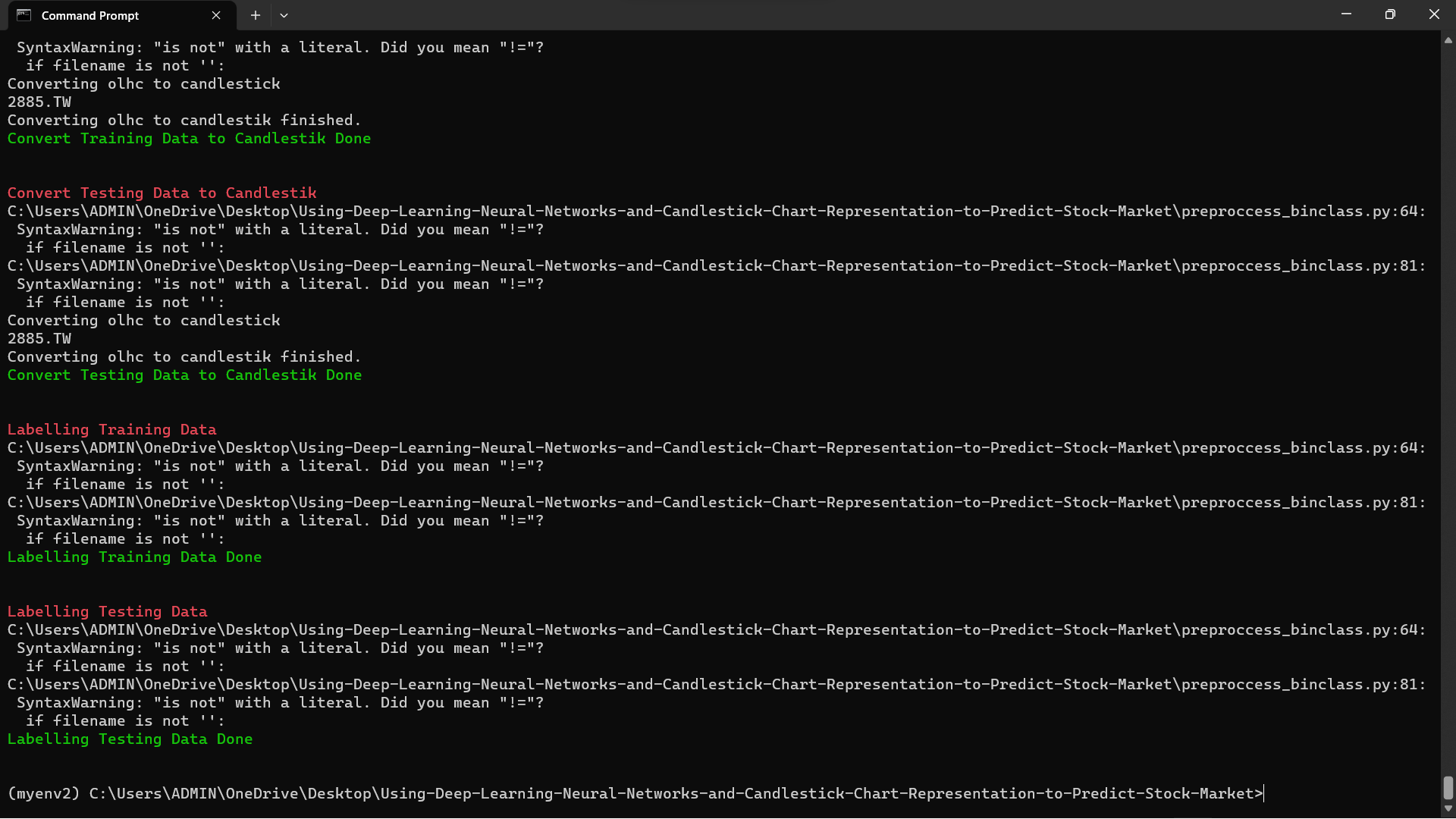
*Fix:*

* + Install “mplfinance” using “pip”, then, change the importing code (from “from mpl\_finance import …” to “from mplfinance import ...”) in the files “predictme.py”, “preprocess\_binclass.py”, “preprocess.py”.
    - “preprocess\_binclass.py”: from “from mpl\_finance import candlestick2\_ochl, volume\_overlay” to “from mpl\_finance import volume\_overlay” and “from mpl\_finance.original\_flavor import candlestick2\_ochl”.
* Attribute Error:

*“DataFrame” object has no attribute “ix”*

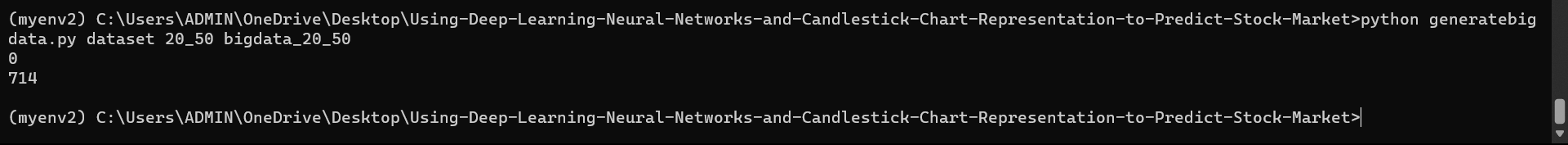
*Fix:*

* + *In “preprocess\_binclass.py” file, line 101 and 147, change from “df.ix[...] to df.loc[...]”*

**

*The process now finishes in 9 minutes.*

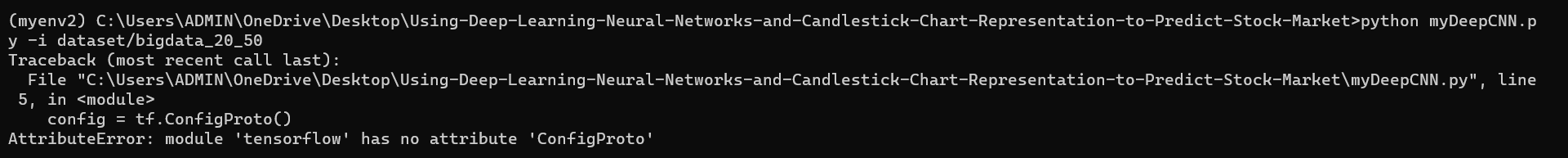
Second, I have generated the final dataset from “tw50” with 20 period days and 50 dimension:

****

**3.3. Build the model**

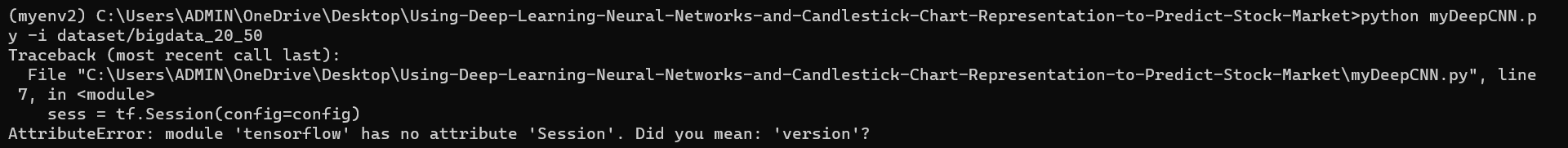
In this stage, I have run “myDeepCNN.py” with default parameters to build the model. There are some problems have shown up:

* Attribute Error

*“tensorflow” has no attribute “ConfigProto”*

*Fix:* “ConfigProto” is disappeared in “tensorflow” 2.0, so I change the code in the file “myDeepCNN.py”, line 5, from “config = tf.ConfigProto()” to “config = tf.compat.v1.ConfigProto()”.

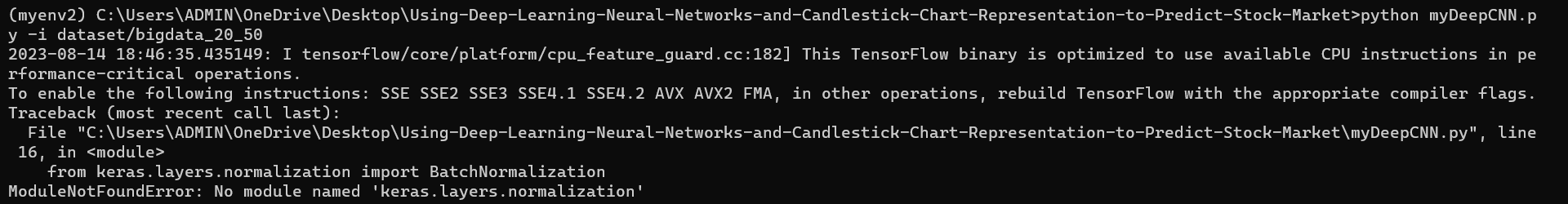
* Attribute Error



*“tensorflow” has no attribute “Session”*

*Fix:* “Session” is disappeared in “tensorflow” 2.0, so I change the code in the file “myDeepCNN.py”, line 7, from “sess = tf.Session(config=config)” to “sess = tf.compat.v1.Session(config=config)”.

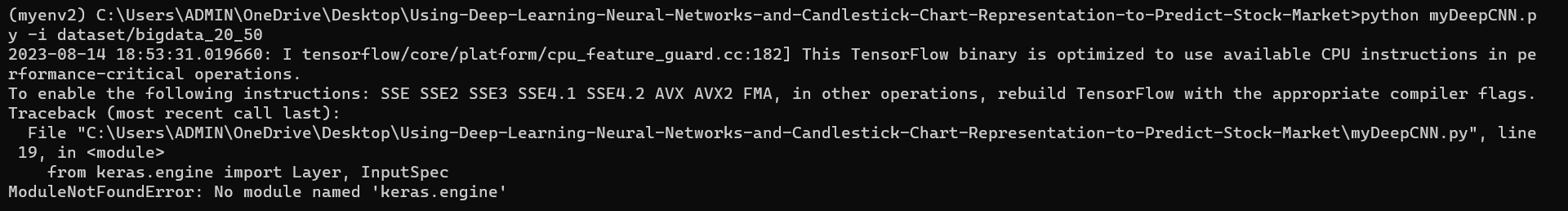
* Module Not Found Error:



*No module named “keras.layers.normalization”*

*Fix:* Change the code in the file “myDeepCNN.py”, line 16, from “from keras.layers.normalization import BatchNormalization” to “from keras.layers import BatchNormalization”.

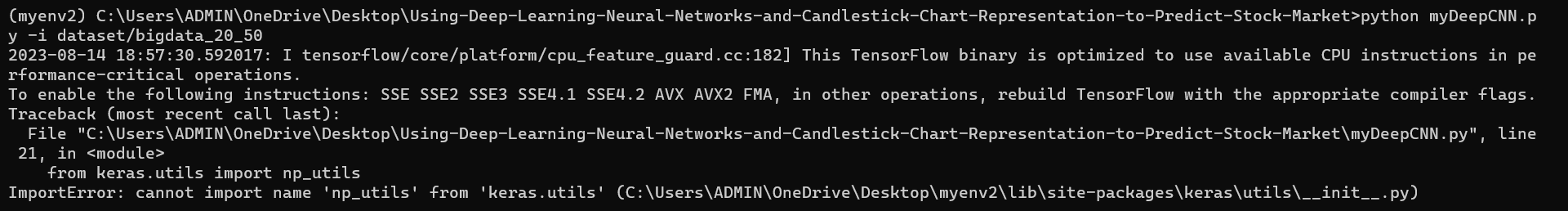
* Module Not Found Error:



*No module named “keras.engine”*

*Fix:* Change the code in the file “myDeepCNN.py”, line 19, from “from keras.engine import Layer, InputSpec” to “from keras.layers import Layer, InputSpec”.

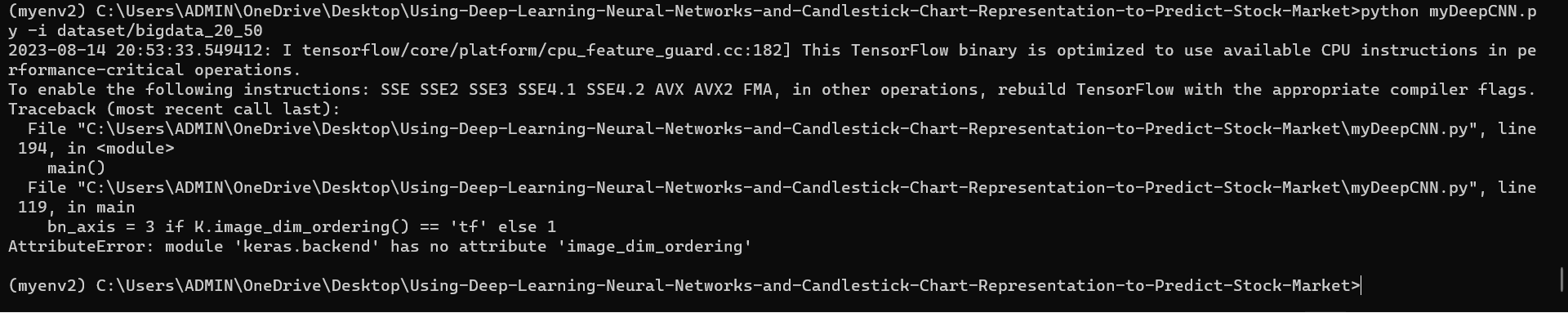
* Import Error:



*Cannot import name “np\_utils” from “keras.utils”*

*Fix:* Change the code in the file “myDeepCNN.py”, line 21, from “from keras.utils import np\_utils” to “from keras.src.utils import np\_utils”.

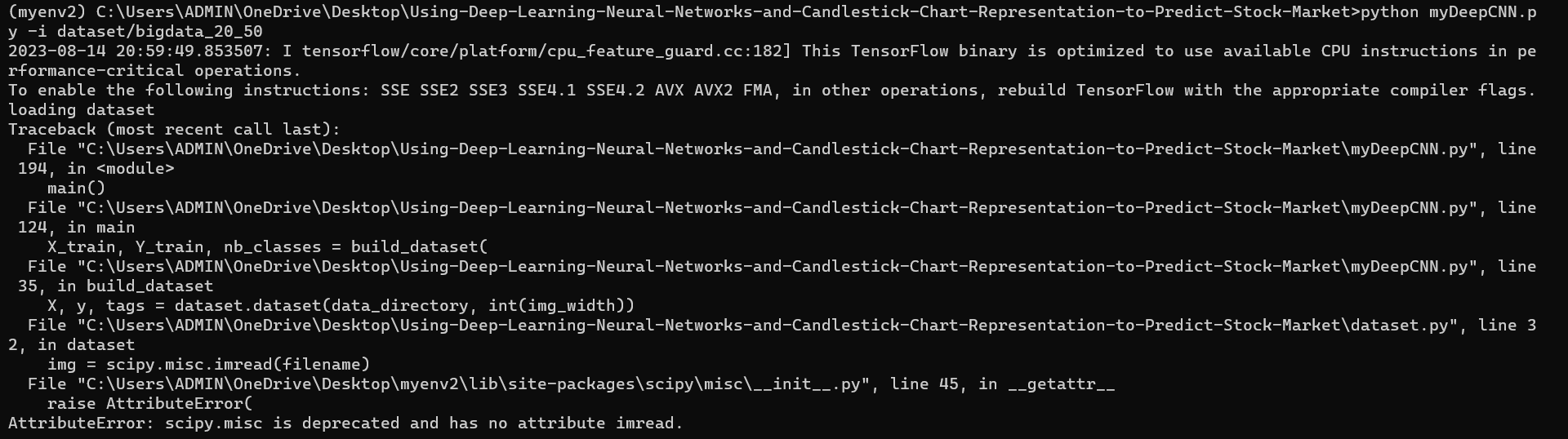
* Attribute Error:



*module “keras.backend” has no attribute “image\_dim\_ordering”*

*Fix:* In line 119, I have change the code of the file “myDeepCNN.py”, from “K.image\_dim\_ordering()” to “K.image\_data\_format()”.

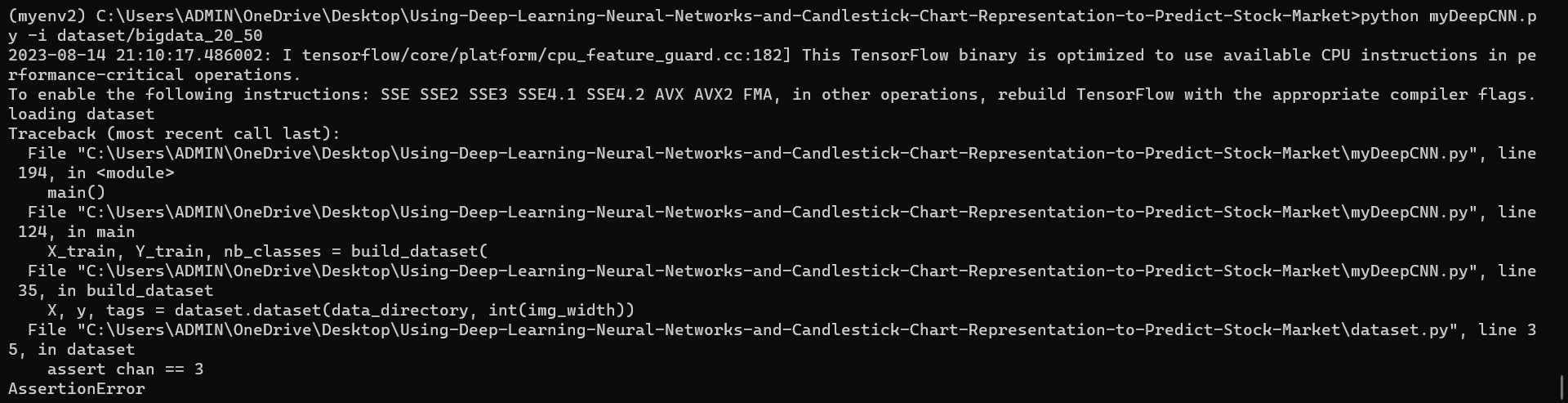
* Attribute Error:



*“scipy.misc” is deprecated and has no attribute “imread”*

*Fix:* Import the library “imageio” using “pip”, then in the file “myDeepCNN.py”, line 5, change the code from “import scipy.misc” to “import image.io”, and line 33, from “scipy.misc.imread” to “imageio.imread”.

* Assertion Error:

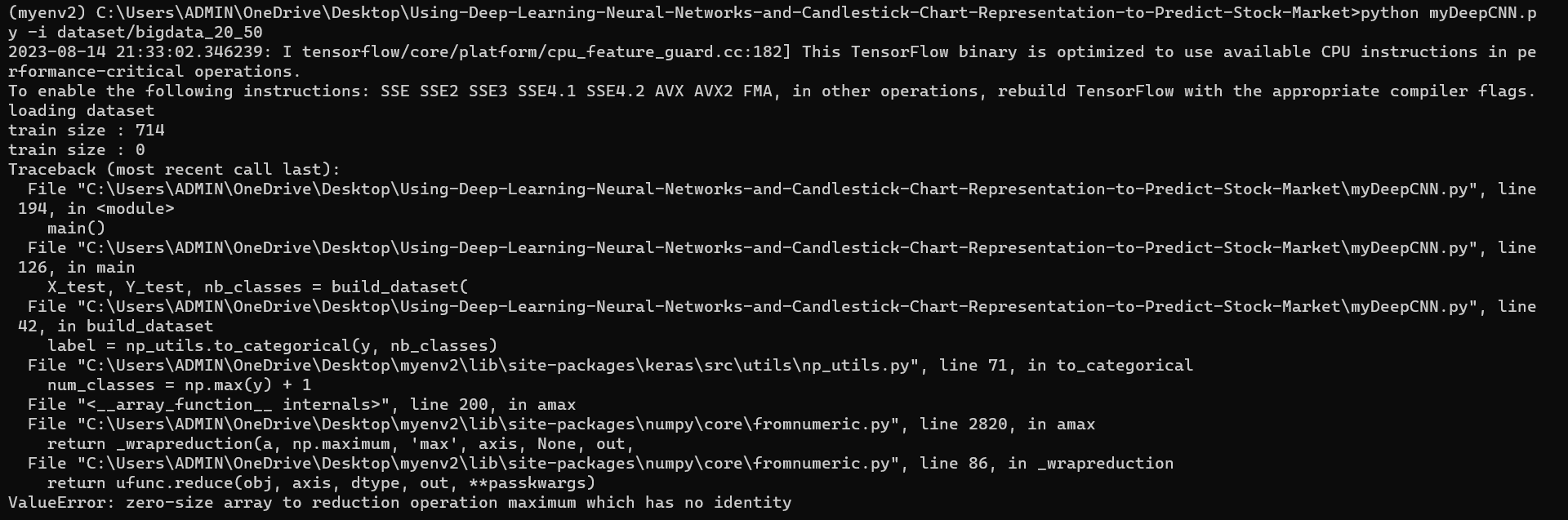


*“scipy.misc” is deprecated and has no attribute “imread”*

*Fix:* In the file “dataset.py”, delete the line 35 to ignore issues like Grayscale Images or Incorrect Image Formats.

ơ

* Value Error **(Unsolved)**



*Zero-size array to reduction operation maximum which has no identity*

**SUMMARY**

The provided code of **v0.1** aims to build a kind of prediction model using LSTM (Long Short-Term Memory) neural networks, predicting the stock price of a given company, which is Tesla (TSLA), using historical data. These are the summary of its main functionality:

* *Importing necessary libraries*: These include “numpy” (for numerical operations), “pandas” (for manipulating data), “pandas-datareader” (for retrieving financial data), “sklearn” (“scikit-learn”) (for preprocessing data), “matplotlib” (for visualization), “tensorflow” (for building neural networks), and “yfinance” (for downloading stock data)
* *Load data:* Historical stock price for the company Tesla is downloaded from Yahoo Finance using “yfinance” library. The data is loaded within a specific date range between 1/1/2015 and 1/1/2020, and only the "Close" prices are considered.
* *Prepare data****:*** The loaded “Close” stock price data is normalized using “MinMaxScaler”, a class of “sklearn” library, to transfer them into the range of 0 to 1. Then, it constructs training data samples using a sliding window approach. Each training sample consists of the closing prices of the past.
* *Build the Model:* A sequential LSTM model is created using the Keras, a subpackage of the library “tensorflow”. The model architecture includes multiple LSTM layers for capturing temporal patterns in the data. Dropout layers are added to mitigate overfitting. The model is set up to predict the next day's closing price.
* *Training the Model:* The model is compiled with the Adam optimizer and the mean squared error loss function. After that, it's trained on the constructed training data (“x\_train” and “y\_train”) using a specified number of epochs and batch size.
* *Test the Model:* The model's performance is evaluated on a separate test dataset, which is downloaded in a similar manner as the training data. The script constructs inputs for the model from the historical data, makes predictions, and scales the predictions back to their original range.
* *Plot Predictions:* The script uses “matplotlib” to plot the actual and predicted stock prices on the same graph. The x-axis represents time, and the y-axis represents stock prices. Actual prices are plotted in black, while predicted prices are plotted in green.
* *Predict Next Day:* The script attempts to predict the stock price for the next day using the most recent “PREDICTION\_DAYS” data points. It scales the data, feeds it into the trained model, and then inverts the scaling to obtain the final prediction.