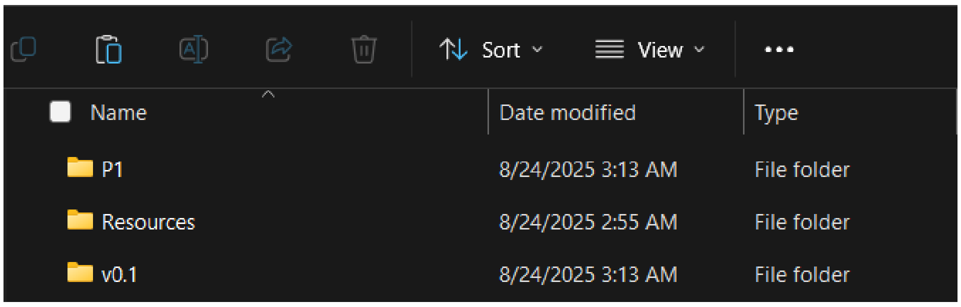
Task 1 - Setup

Kelvin Dang - 104776732 - Swinburne University of Technology

Setting up the environment

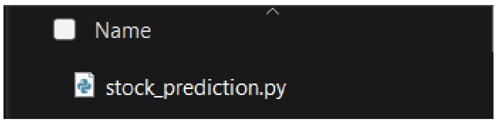
Firstly, I downloaded the codebases of the current assignment project (**v0.1**) and the reference project (**P1**) to my local machine.



Name	Date modified	Type
P1	8/24/2025 3:13 AM	File folder
Resources	8/24/2025 2:55 AM	File folder
v0.1	8/24/2025 3:13 AM	File folder

Figure 1: The codebases of **v0.1** and **P1**.

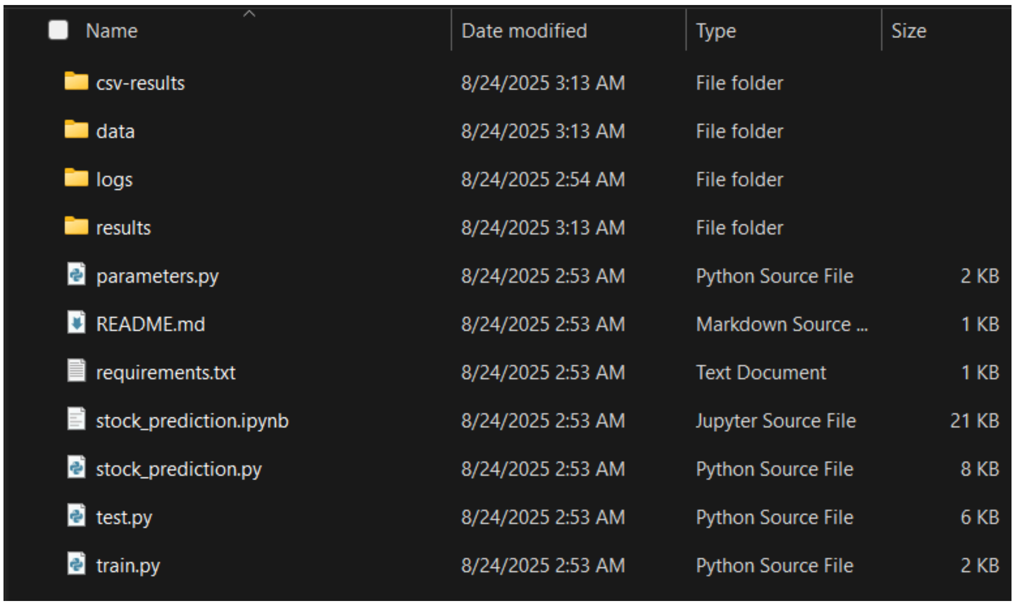
The **v0.1** codebase was a single Python file named `stock-prediction.py`.



Name
stock_prediction.py

Figure 2: The current project codebase (**v0.1**).

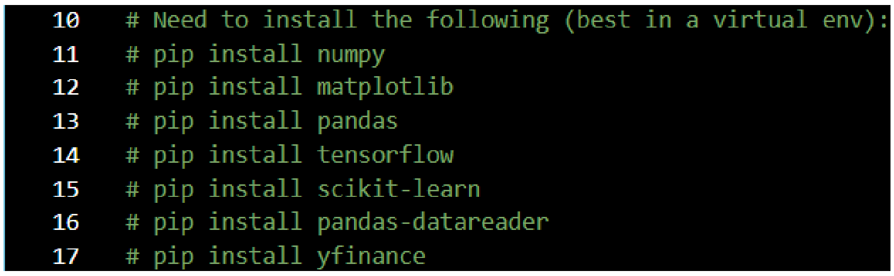
On the other hand, the **P1** codebase was successfully downloaded, without any missing source files.



Name	Date modified	Type	Size
csv-results	8/24/2025 3:13 AM	File folder	
data	8/24/2025 3:13 AM	File folder	
logs	8/24/2025 2:54 AM	File folder	
results	8/24/2025 3:13 AM	File folder	
parameters.py	8/24/2025 2:53 AM	Python Source File	2 KB
README.md	8/24/2025 2:53 AM	Markdown Source ...	1 KB
requirements.txt	8/24/2025 2:53 AM	Text Document	1 KB
stock_prediction.ipynb	8/24/2025 2:53 AM	Jupyter Source File	21 KB
stock_prediction.py	8/24/2025 2:53 AM	Python Source File	8 KB
test.py	8/24/2025 2:53 AM	Python Source File	6 KB
train.py	8/24/2025 2:53 AM	Python Source File	2 KB

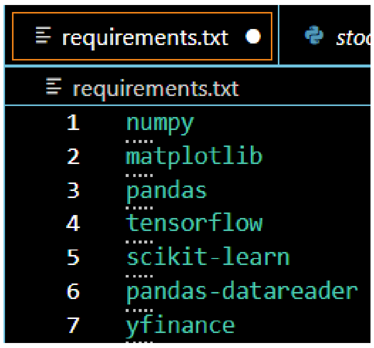
Figure 3: The **P1** project codebase.

Then, I tried to respectively create requirements files for **v0.1** and **P1**, having watched the provided tutorial video of **v0.1**, read **P1**'s GitHub page, and the given page with more details about requirements files. For **v0.1**, I also followed the instructions from the comments in the code.



```
# Need to install the following (best in a virtual env):
# pip install numpy
# pip install matplotlib
# pip install pandas
# pip install tensorflow
# pip install scikit-learn
# pip install pandas-datareader
# pip install yfinance
```

Figure 4: The instructions on setting up v0.1's environment.

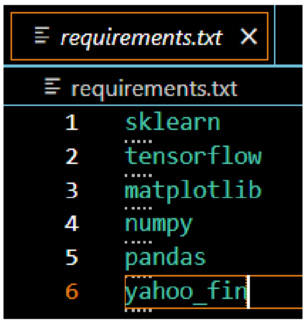


```
numpy
matplotlib
pandas
tensorflow
scikit-learn
pandas-datareader
yfinance
```

Figure 5: v0.1's requirements file.

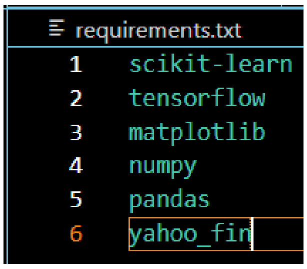
Meanwhile, **P1**'s source code had provided me with its own requirements file, so there is no need for me to create another.

However, I still had to modify that file, changing `sklearn` to `scikit-learn` in line 1, due to the deprecation of the `sklearn` PyPI package.



```
1  sklearn
2  tensorflow
3  matplotlib
4  numpy
5  pandas
6  yahoo_fin
```

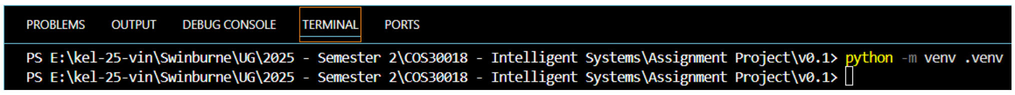
Figure 6: **P1**'s requirements file.



```
1  scikit-learn
2  tensorflow
3  matplotlib
4  numpy
5  pandas
6  yahoo_fin
```

Figure 7: **P1**'s requirements file after being modified.

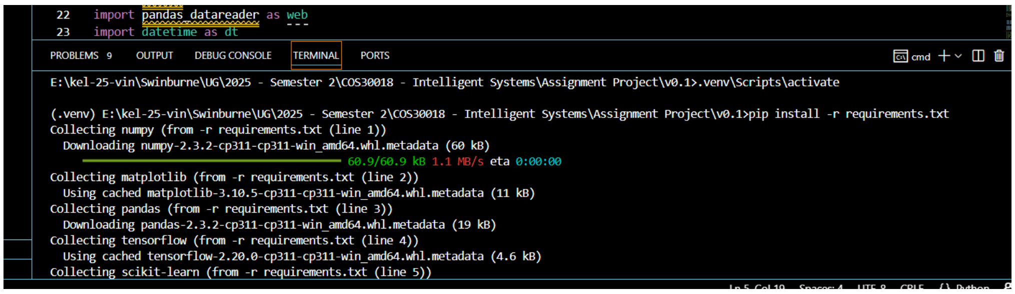
To begin with, I setup a virtual environment for the **v0.1** project by running the command `python -m venv .venv`.



```
PS E:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\v0.1> python -m venv .venv
PS E:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\v0.1>
```

Figure 8: Creating a virtual environment for v0.1.

Then, I activate it, and install all packages specified in the requirements file.



```
import pandas_datareader as web
import datetime as dt
```

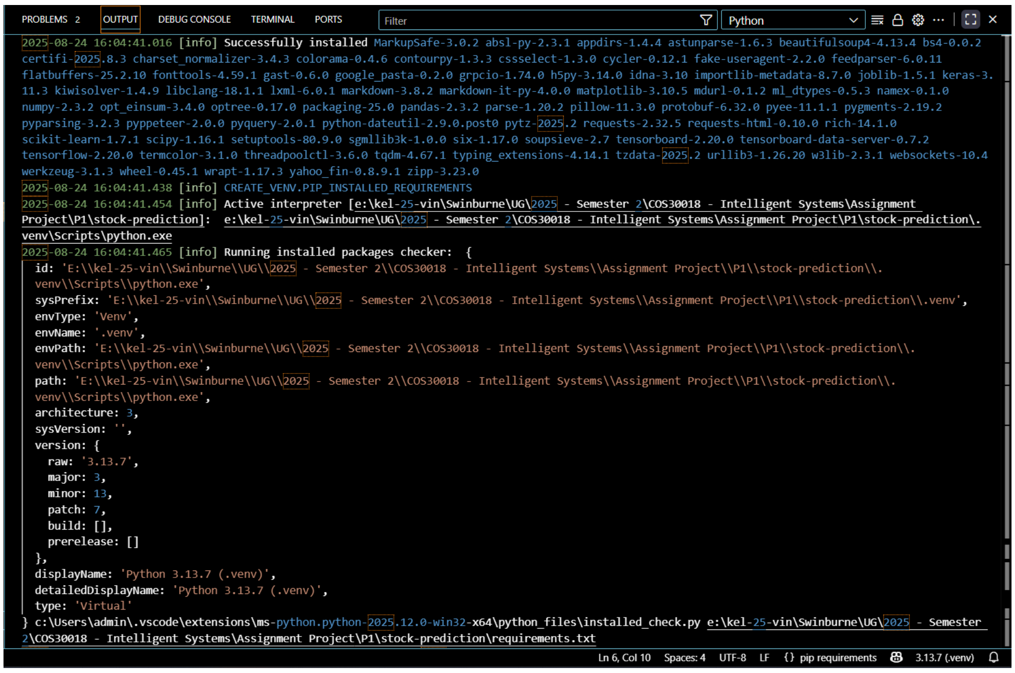
PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
E:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\v0.1>.venv\Scripts\activate

(.venv) E:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\v0.1>pip install -r requirements.txt
Collecting numpy (from -r requirements.txt (line 1))
  Downloading numpy-2.3.2-cp311-cp311-win_amd64.whl.metadata (60 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 60.9/60.9 kB 1.1 MB/s eta 0:00:00
Collecting matplotlib (from -r requirements.txt (line 2))
  Using cached matplotlib-3.10.5-cp311-cp311-win_amd64.whl.metadata (11 kB)
Collecting pandas (from -r requirements.txt (line 3))
  Downloading pandas-2.3.2-cp311-cp311-win_amd64.whl.metadata (19 kB)
Collecting tensorflow (from -r requirements.txt (line 4))
  Using cached tensorflow-2.20.0-cp311-cp311-win_amd64.whl.metadata (4.6 kB)
Collecting scikit-learn (from -r requirements.txt (line 5))
```

Figure 9: Activating the virtual environment, and installing packages.

Then, for the **P1** project, I followed the same steps mentioned above to create and activate a virtual environment, as well as to install the required packages. The results look a little bit different to that of **v0.1**, due to the fact that I had used VSCode's Command Palette, instead of Command Prompt.



PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS Filter Python

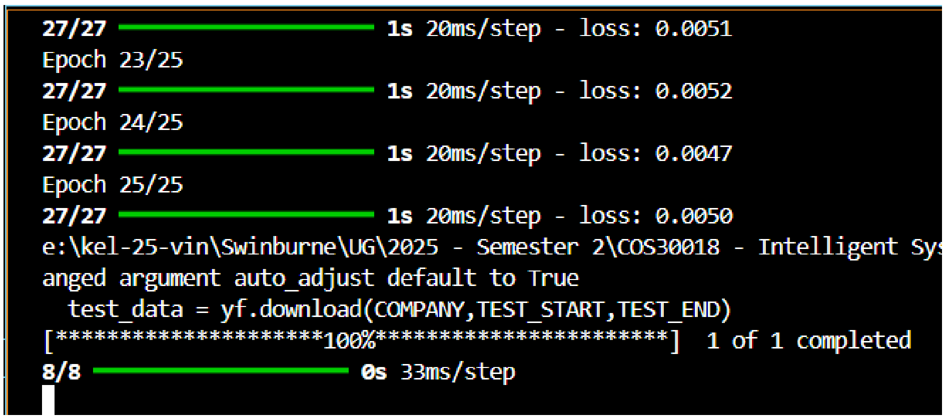
```
2025-08-24 16:04:41.016 [info] Successfully installed MarkupSafe-3.0.2 absl-py-2.3.1 appdirs-1.4.4 astunparse-1.6.3 beautifulsoup4-4.13.4 bs4-0.0.2
certifi-2025.8.3 charset_normalizer-3.4.3 colorama-0.4.6 contourpy-1.3.3 cssselect-1.3.0 cycler-0.12.1 fake-useragent-2.2.0 feedparser-6.0.11
flatbuffers-25.2.10 fonttools-4.59.1 gast-0.6.0 google_pasta-0.2.0 grpcio-1.74.0 h5py-3.14.0 idna-3.10 importlib-metadata-8.7.0 joblib-1.5.1 keras-3.
11.3 kiwisolver-1.4.9 libclang-18.1.1 lxml-6.0.1 markdown-3.8.2 markdown-it-py-4.0.0 matplotlib-3.10.5 mdurl-0.1.2 ml_dtypes-0.5.3 namex-0.1.0
numpy-2.3.2 opt_einsum-3.4.0 optree-0.17.0 packaging-25.0 pandas-2.3.2 parse-1.20.2 pillow-11.3.0 protobuf-6.32.0 pyee-11.1.1 pygments-2.19.2
pyparsing-3.2.3 pyppeteer-2.0.0 pyquery-2.0.1 python-dateutil-2.9.0.post0 pytz-2025.2 requests-2.32.5 requests-html-0.10.0 rich-14.1.0
scikit-learn-1.7.1 scipy-1.16.1 setuptools-80.9.0 sgmllib3k-1.0.0 six-1.17.0 soupsieve-2.7 tensorboard-2.20.0 tensorboard-data-server-0.7.2
tensorflow-2.20.0 termcolor-3.1.0 threadpoolctl-3.6.0 tqdm-4.67.1 typing_extensions-4.14.1 tzdata-2025.2 urllib3-1.26.20 w3lib-2.3.1 websockets-10.4
werkzeug-3.1.3 wheel-0.45.1 wrapt-1.17.3 yahoo_fin-0.8.9.1 zipp-3.23.0
2025-08-24 16:04:41.438 [info] CREATE_VENV.PIP_INSTALLED_REQUIREMENTS
2025-08-24 16:04:41.454 [info] Active interpreter [e:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment
Project\P1\stock-prediction]:  e:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\P1\stock-prediction\.
venv\Scripts\python.exe
2025-08-24 16:04:41.465 [info] Running installed packages checker:  {
  id: 'E:\\kel-25-vin\\Swinburne\\UG\\2025 - Semester 2\\COS30018 - Intelligent Systems\\Assignment Project\\P1\\stock-prediction\\.
  venv\\Scripts\\python.exe',
  sysPrefix: 'E:\\kel-25-vin\\Swinburne\\UG\\2025 - Semester 2\\COS30018 - Intelligent Systems\\Assignment Project\\P1\\stock-prediction\\.venv',
  envType: 'Venv',
  envName: '.venv',
  envPath: 'E:\\kel-25-vin\\Swinburne\\UG\\2025 - Semester 2\\COS30018 - Intelligent Systems\\Assignment Project\\P1\\stock-prediction\\.
  venv\\Scripts\\python.exe',
  path: 'E:\\kel-25-vin\\Swinburne\\UG\\2025 - Semester 2\\COS30018 - Intelligent Systems\\Assignment Project\\P1\\stock-prediction\\.
  venv\\Scripts\\python.exe',
  architecture: 3,
  sysVersion: '',
  version: {
    raw: '3.13.7',
    major: 3,
    minor: 13,
    patch: 7,
    build: [],
    prerelease: []
  },
  displayName: 'Python 3.13.7 (.venv)',
  detailedDisplayName: 'Python 3.13.7 (.venv)',
  type: 'Virtual'
} c:\Users\admin\.vscode\extensions\ms-python.python-2025.12.0-win32-x64\python_files\installed_check.py e:\kel-25-vin\Swinburne\UG\2025 - Semester
2\COS30018 - Intelligent Systems\Assignment Project\P1\stock-prediction\requirements.txt
```

Ln 6, Col 10 Spaces: 4 UTF-8 LF {} pip requirements 3.13.7 (.venv)

Figure 10: Environment setup completed for P1.

Testing the provided codebases

After having all required packages installed, I ran the **v0.1** program using the Command Prompt, and it was successfully executed.



```
27/27 ━━━━━━━━━━━━━━━━━━━━ 1s 20ms/step - loss: 0.0051
Epoch 23/25
27/27 ━━━━━━━━━━━━━━━━━━━━ 1s 20ms/step - loss: 0.0052
Epoch 24/25
27/27 ━━━━━━━━━━━━━━━━━━━━ 1s 20ms/step - loss: 0.0047
Epoch 25/25
27/27 ━━━━━━━━━━━━━━━━━━━━ 1s 20ms/step - loss: 0.0050
e:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Sys
anged argument auto_adjust default to True
  test_data = yf.download(COMPANY,TEST_START,TEST_END)
[*********************100%***********************]  1 of 1 completed
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step
```

Figure 11: The **v0.1** model being trained.

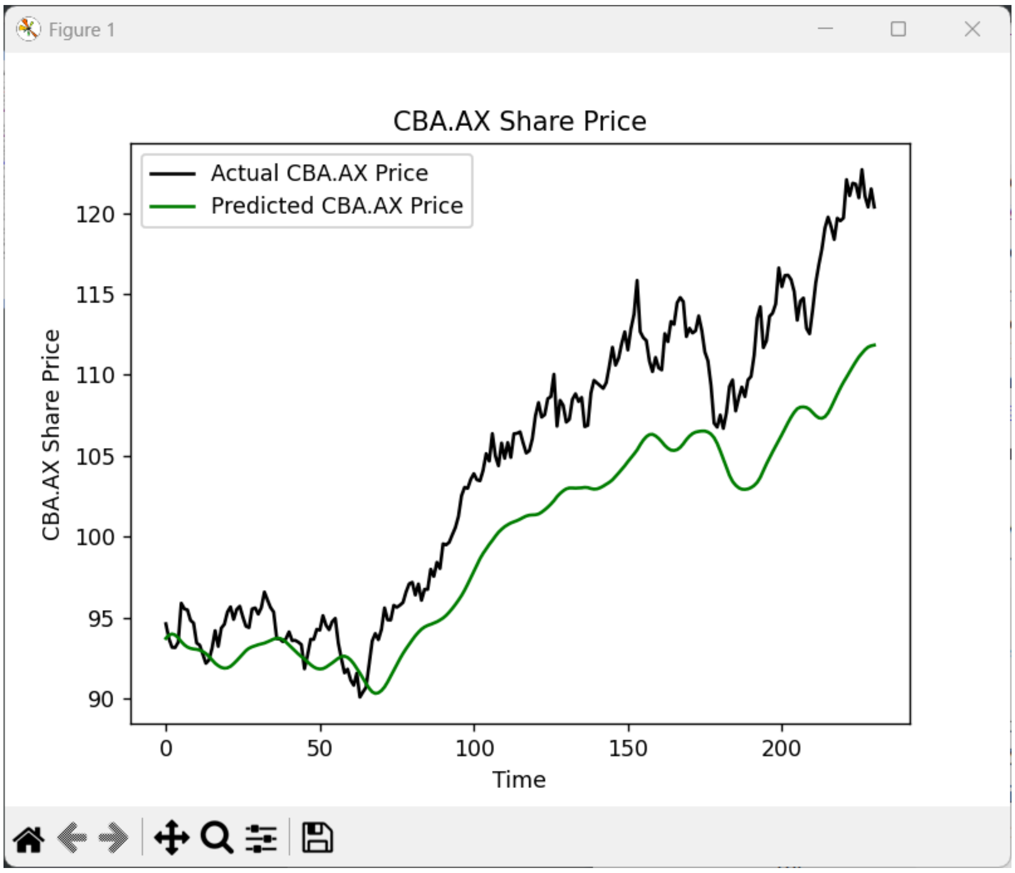
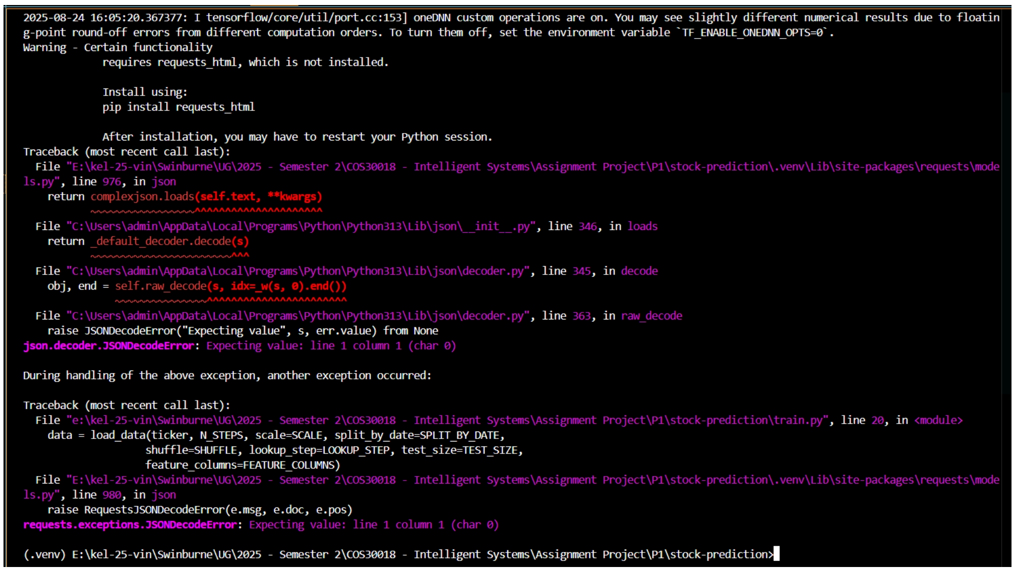


Figure 12: The performance of the **v0.1** model.

Then, for the **P1** project, I followed the same steps mentioned above to create and activate a virtual environment, install packages, and train the model as well.

However, after configuring the virtual environment and installing packages using the requirements file, I encountered errors from the HTTP API request, as the figure below shows.



```
2025-08-24 16:05:20.367377: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floatin
g-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
Warning - Certain functionality
             requires requests_html, which is not installed.

             Install using:
             pip install requests_html

             After installation, you may have to restart your Python session.
Traceback (most recent call last):
  File "E:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\P1\stock-prediction\.venv\Lib\site-packages\requests\mode
ls.py", line 976, in json
    return complexjson.loads(self.text, **kwargs)
           ~~~~~~~~~~~~~~~~~^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\admin\AppData\Local\Programs\Python\Python313\Lib\json\__init__.py", line 346, in loads
    return _default_decoder.decode(s)
           ~~~~~~~~~~~~~~~~~~~~~~~^^^
  File "C:\Users\admin\AppData\Local\Programs\Python\Python313\Lib\json\decoder.py", line 345, in decode
    obj, end = self.raw_decode(s, idx=_w(s, 0).end())
               ~~~~~~~~~~~~~~~^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\admin\AppData\Local\Programs\Python\Python313\Lib\json\decoder.py", line 363, in raw_decode
    raise JSONDecodeError("Expecting value", s, err.value) from None
json.decoder.JSONDecodeError: Expecting value: line 1 column 1 (char 0)

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "e:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\P1\stock-prediction\train.py", line 20, in <module>
    data = load_data(ticker, N_STEPS, scale=SCALE, split_by_date=SPLIT_BY_DATE,
                    shuffle=SHUFFLE, lookup_step=LOOKUP_STEP, test_size=TEST_SIZE,
                    feature_columns=FEATURE_COLUMNS)
  File "E:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\P1\stock-prediction\.venv\Lib\site-packages\requests\mode
ls.py", line 980, in json
    raise RequestsJSONDecodeError(e.msg, e.doc, e.pos)
requests.exceptions.JSONDecodeError: Expecting value: line 1 column 1 (char 0)

(.venv) E:\kel-25-vin\Swinburne\UG\2025 - Semester 2\COS30018 - Intelligent Systems\Assignment Project\P1\stock-prediction>
```

Figure 13: API request error while I was trying to run **P1**.

I tried several ways of debugging after a short period of personal research.

- First, I installed `requests_html` as the error log suggested, but nothing changed.
- Then, I found an online question on Reddit mentioning the exact same problem I encountered. I tried to debug as the comments instructed, but nothing changed either.
    - As the writer of the question said, the code had been working for a long time before suddenly giving errors. I think the problem is down to the API. The link to this question is included in the **References** section.

Due to this problem, I will use the testing results published on the provided tutorial page of **P1**. The link to this page will also be included in the **References** section.

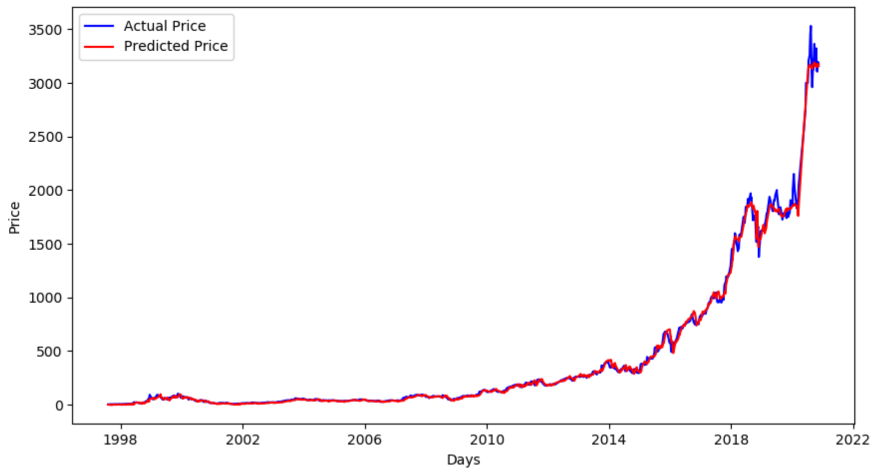
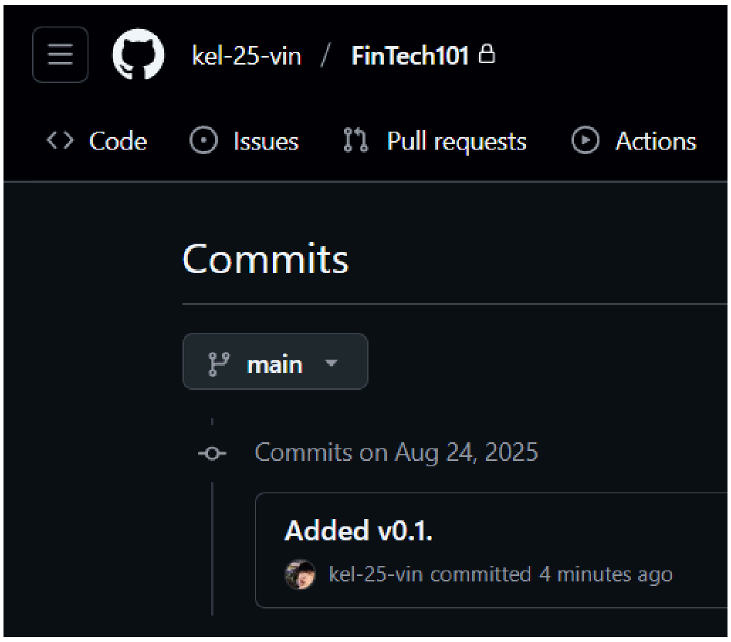


Figure 14: P1's testing results, collected by its authors.

Project setup

I set up a GitHub repository for my project with the name of **FinTech101**, as the project document mentioned, and committed **v0.1**. As GitHub Wikis is no longer available for free users, I will personally use this repository for version control.



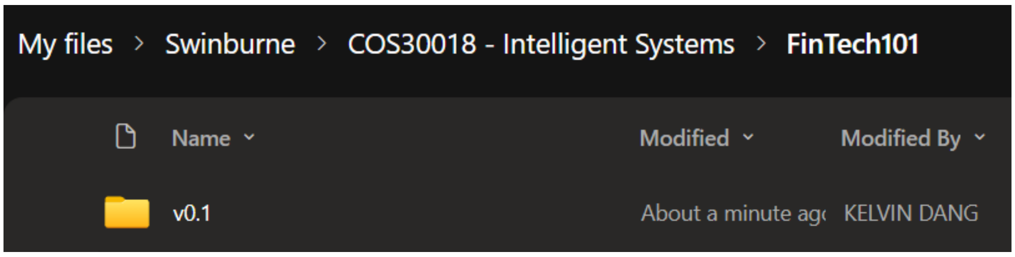
Commits

Commits on Aug 24, 2025

Added v0.1.
kel-25-vin committed 4 minutes ago

Figure 15: My first commit to FinTech101.

As the Convenor's announcement suggested, I uploaded **v0.1** to the destination folder I created for my project to share with my tutor. This will be the folder I choose to upload task reports as well.



My files > Swinburne > COS30018 - Intelligent Systems > FinTech101

Name	Modified	Modified By
v0.1	About a minute ago	KELVIN DANG

Figure 16: My project, uploaded to OneDrive.

Performance comparisons

As attached above, **Figure 12** and **Figure 14** show the testing performance of **v0.1** and **P1**, respectively.

In the training and testing process, **v0.1** chose Mean Squared Error (MSE) as the loss function, while **P1** chose Huber loss. These are all absolute loss functions, and the models were trained using this with the aim of lowering the absolute loss value. However, the values of the stocks that they predicted (CBA.AX for **v0.1**, and AMZN for **P1**) are very distant from each other, as we all can easily see. Therefore, I suggest comparing their relative (percentage) loss, which is loss in percent, rather than in absolute values.

In this way, **v0.1** usually missed the actual price by about 10-13%, while **P1**'s prediction curve almost exactly matched the actual price curve. As a result, we can conclude that **P1** outperformed **v0.1** in this stock price prediction problem.

References

- The online question thread I found, which mentioned the exact same problem I have encountered while testing the codebase of **P1**.
    - https://www.reddit.com/r/learnpython/comments/vby891/python_http_api_request_error_expecting_value
- **P1**'s tutorial page.

    - https://thepythoncode.com/article/stock-price-prediction-in-python-using-tensorflow-2-and-keras