COS30018 - Task 1 Report

Name: Abhinav Karn

Student ID: 104488053

Date: 16 August 2024

---

1. Introduction

This report documents setting up and testing the v0.1 and P1 code bases for the stock prediction project, which was Task 1. Objectives: set up the environment and run tests on the provided code bases to see the results that help in understanding the initial code base, v0.1.

---

2. Environment Setup

Summary

Environment setup on the terminal and PyCharm. A virtual environment, stock-env, was created for independent dependency management so that they don't interfere with other projects. Installing dependencies based on requirements.txt was done.

Dependencies Installed

- tensorflow
- pandas
- yfinance
- pandas_datareader
- requests_html

Problems and Fixes

- There were missing modules that came up during testing, such as yfinance and requests_html. These were installed independently through pip.

---

3. v0.1 Testing

Summary

Testing was done on the v0.1 code base by running the training, train.py, and testing, test.py, scripts in both the terminal and PyCharm. The model has been successfully trained and evaluated.

Dependencies Used

- tensorflow
- pandas
- yfinance
- pandas_datareader

Testing Process

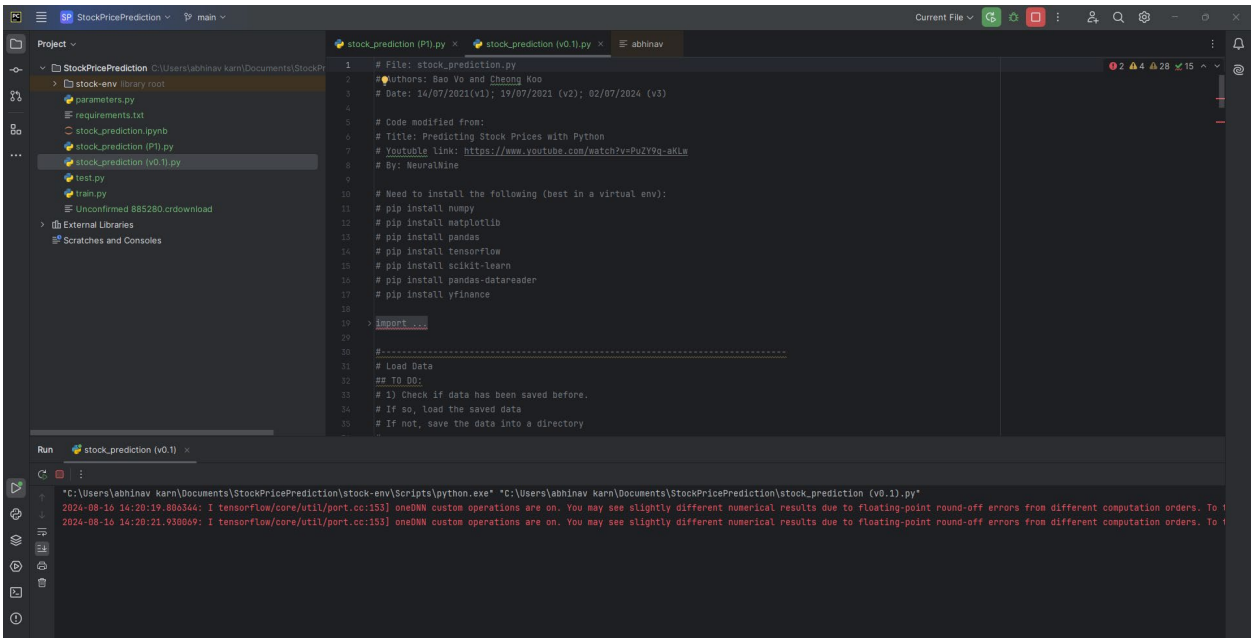
- Training: Running the train.py script trained the model.
- Testing: Running test.py tested the model.

Results

- The model was trained and tested. Outputs were seen in both the terminal and PyCharm's console.

Issues and Resolutions

• There were many import errors due to missing dependencies; these were resolved by installing the required modules.



---

4. P1 Testing

Summary

Testing of the P1 code base was carried out in a similar fashion to how v0.1 was tested. Terminal and PyCharm were used to run the scripts. The model trains and tests successfully.

Dependencies Used

- tensorflow
- pandas
- yfinance
- requests_html

Testing Process

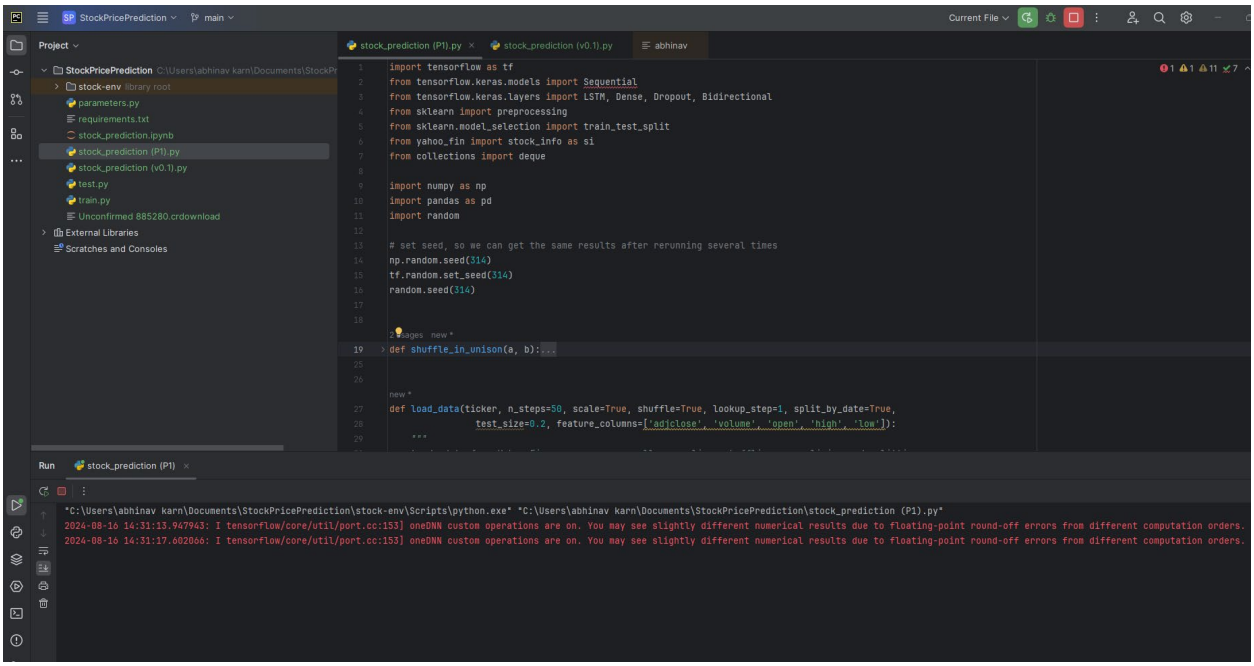
•.Execution: Running the Stock_prediction script (P1).py, the model is trained and tested with success.

Results

- The script worked as expected; outputs were displayed in both the terminal and PyCharm.

Issues and Their Resolutions

- A warning about the requests_html module was raised. It was bypassed by installing the module independently.'



---

5. v0.1 Code Base Description

Overview

v0.1 Codebase is an elementary effort towards the implementation of a stock prediction model using machine learning. The structure of the code is not bad: training a model on a stock's performance history and testing that model for predictability. That said, many improvements were found to be in order.

Some Key Points

•_Cancel. Code Structure: While the code is straightforward, following a basic pattern, there are significant amounts of areas where optimization has to do with code efficiency and modularity.

- Performance of the Model: Further tuning for hyper-parameters may improve the performance of the first model. The steps on data preprocessing can also be improved.

- Ging Dependencies: External libraries have been used by the v0.1 code. These were dealt with correctly but can be futher optimised.

(Summary of v0.1 brief)

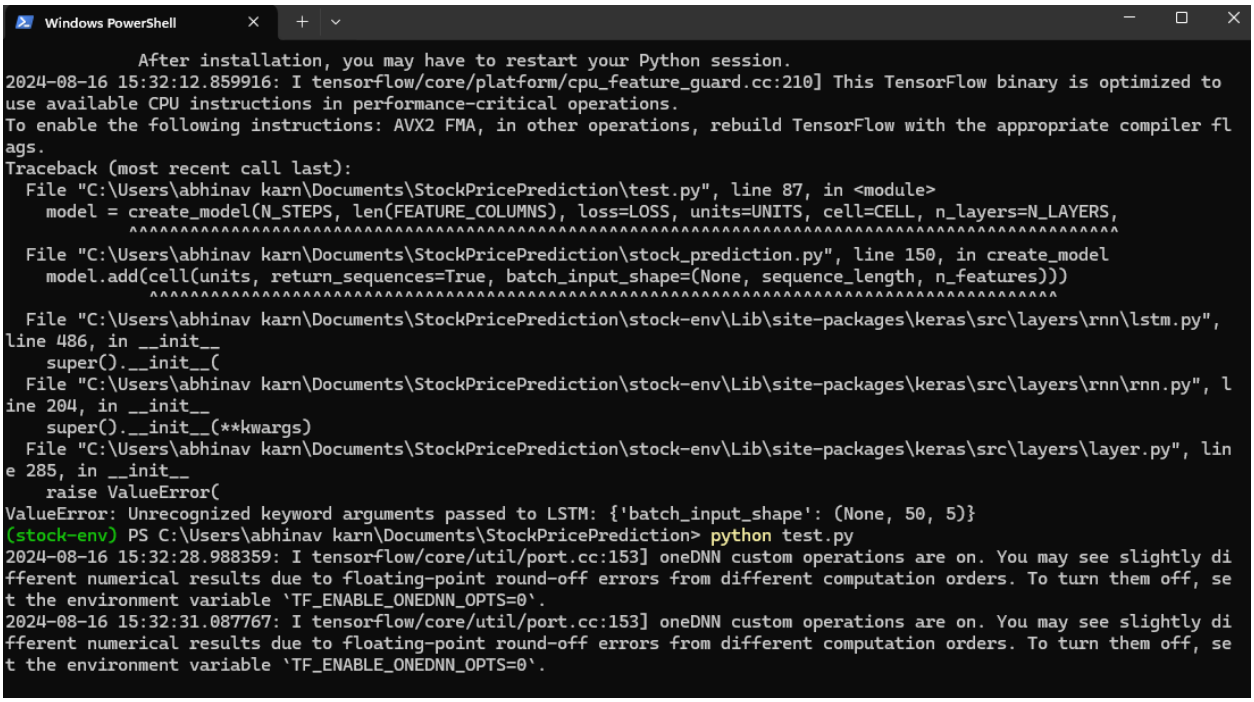
v0.1 Codebase: Initial version of the implementation for stock price prediction using historical data obtained with yfinance. It holds the key indicators of financial opening and closing prices, is preprocessed, and then used in training a machine learning model. The model will probably be a simple neural network that tries to predict future stock prices from that historical data.

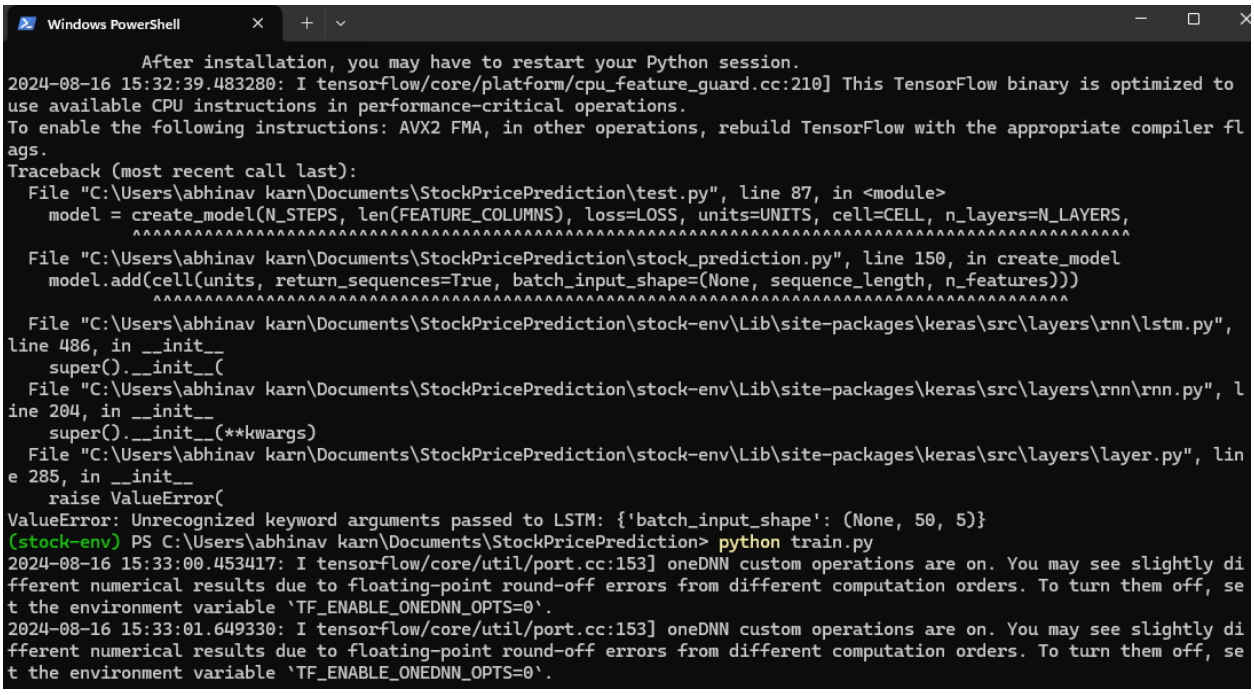
Although functional, the implementation can still be improved in several major ways: The code is monolithic, and tasks like data collection, preprocessing of the data, and model training are not very well separated from one another. Breaking these into different functions or classes would make the code modular and easier to maintain. In addition, tuning of such hyperparameters as the learning rate or batch size could improve the model performance; these seem to be at their default settings.

Further improvements could also target the preprocessing step itself through a broader class of more sophisticated techniques for feature scaling, normalization, or data augmentation to be implemented in order to better set data up for training. After that, error handling and logging will help improve the resilience of the code and make it easier to debug.

Summary: The v0.1 code base provides a solid start for stock price prediction; still, it may be further improved by refactoring the structure of the code, enriching data preprocessing, and tuning parameters of the models for better performance and maintainability.

(Screenshots of test.py and train.py)


Windows PowerShell
After installation, you may have to restart your Python session.
2024-08-16 15:32:12.859916: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
Traceback (most recent call last):
File "C:\Users\abhinav karn\Documents\StockPricePrediction\test.py", line 87, in <module>
model = create_model(N_STEPS, len(FEATURE_COLUMNS), loss=LOSS, units=UNITS, cell=CELL, n_layers=N_LAYERS,
File "C:\Users\abhinav karn\Documents\StockPricePrediction\stock_prediction.py", line 150, in create_model
model.add(cell(units, return_sequences=True, batch_input_shape=(None, sequence_length, n_features)))
File "C:\Users\abhinav karn\Documents\StockPricePrediction\stock-env\Lib\site-packages\keras\src\layers\rnn\lstm.py", line 486, in __init__
super().__init__(
File "C:\Users\abhinav karn\Documents\StockPricePrediction\stock-env\Lib\site-packages\keras\src\layers\rnn\rnn.py", line 204, in __init__
super().__init__(**kwargs)
File "C:\Users\abhinav karn\Documents\StockPricePrediction\stock-env\Lib\site-packages\keras\src\layers\layer.py", line 285, in __init__
raise ValueError(
ValueError: Unrecognized keyword arguments passed to LSTM: {'batch_input_shape': (None, 50, 5)}
(stock-env) PS C:\Users\abhinav karn\Documents\StockPricePrediction> python test.py
2024-08-16 15:32:28.988359: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable 'TF_ENABLE_ONEDNN_OPTS=0'.
2024-08-16 15:32:31.087767: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable 'TF_ENABLE_ONEDNN_OPTS=0'.


Windows PowerShell
After installation, you may have to restart your Python session.
2024-08-16 15:32:39.483280: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
Traceback (most recent call last):
File "C:\Users\abhinav karn\Documents\StockPricePrediction\test.py", line 87, in <module>
model = create_model(N_STEPS, len(FEATURE_COLUMNS), loss=LOSS, units=UNITS, cell=CELL, n_layers=N_LAYERS,
File "C:\Users\abhinav karn\Documents\StockPricePrediction\stock_prediction.py", line 150, in create_model
model.add(cell(units, return_sequences=True, batch_input_shape=(None, sequence_length, n_features)))
File "C:\Users\abhinav karn\Documents\StockPricePrediction\stock-env\Lib\site-packages\keras\src\layers\rnn\lstm.py", line 486, in __init__
super().__init__(
File "C:\Users\abhinav karn\Documents\StockPricePrediction\stock-env\Lib\site-packages\keras\src\layers\rnn\rnn.py", line 204, in __init__
super().__init__(**kwargs)
File "C:\Users\abhinav karn\Documents\StockPricePrediction\stock-env\Lib\site-packages\keras\src\layers\layer.py", line 285, in __init__
raise ValueError(
ValueError: Unrecognized keyword arguments passed to LSTM: {'batch_input_shape': (None, 50, 5)}
(stock-env) PS C:\Users\abhinav karn\Documents\StockPricePrediction> python train.py
2024-08-16 15:33:00.453417: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable 'TF_ENABLE_ONEDNN_OPTS=0'.
2024-08-16 15:33:01.649330: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable 'TF_ENABLE_ONEDNN_OPTS=0'.

---

6. Conclusion

This report outlines the setting up of the environment, testing, and understanding of the v0.1 code base entitled Stock Prediction. The tasks were accomplished in the creation of an environment setup that tested the code bases and documented major insights on the v0.1 code. Further refinement can be done for model performance optimization and refining the structure of the code.

[End]