COS30018 Intelligent Systems Assignment B

Semester 2, 2023  
Assignemnt B Report

Robin Findlay-Marks (103603871) and the Project leader: Mukesh Malani

Self-Assessment Details

The following checklists provide an overview of my self-assessment for this unit.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Pass (P) | Credit (C) | Distinction (D) | High Distinction (Low HD) | (High HD) | |
| Self-Assessment (please tick) | ✔ |  |  |  |  |

*Self-assessment Statement*

|  |  |
| --- | --- |
|  | Included? (tick) |
| Learning Summary Report | ✓ |
| Complete Pass (“core”) task work, approved in Canvas | ✓ |

*Minimum Pass Checklist*

|  |  |
| --- | --- |
|  | Included? (tick) |
| Additional non-core task work (or equivalent) in a private repository and accessible to staff account. |  |
| Spike Extension Report (for spike extensions) in Canvas |  |
| Custom Project plan (for D and/or low HD), and/or High HD Research Plan document in Canvas (optional) |  |

*Credit Checklist, in addition to Pass Checklist*

# Introduction

This report summarises what I have done for the COS30018 Intelligent Systems Assignment B. For this semester I was given a basic stock prediction file as a starting point (v0.1). Over the following weeks I was given several tasks to incrementally improve the v0.1 code until it is done.

# Instructions to run the program

To run the program, first install the requirements listed in the requirements.txt file which are as follows:

sklearn

tensorflow

matplotlib

numpy

pandas

pandas-datareader

yahoo\_fin

yfinance

scikit-learn

mplfinance

statsmodels

cython

prophet

These can be install individually with the pip install command, or by running the pip install command on the requirements.txt file.

The next step to run the program is to open the Intellegent-Systems-Assignemnt-B folder in the command prompt.

Finally run the command to run the stock\_prediction.py file which is:

py stock\_prediction.py

To change the settings of the program, open the parameters.py, which has all the settings and parameters used in the program.

One important setting to note is the ‘MODE’ parameter. This parameter decides which of several modes to run the program in. They are; 1 – to use the basic RNN models with the data split into train and test by a specified date, 2 - to use the basic RNN models with the data split into train and test by a ratio, 3 – display candlestick chart of the past NDAYS of downloaded data, 4 – display boxplot chart of the past NDAYS of downloaded data, 5 – run the prophet prediction model and any other number, which uses the basic RNN models with the data split into train and test by a random date.

Some additional functionality instructions (assumes parameters are set to default for each instruction)

To test the Multivariate function, set the MODE parameter to 2 and MULTIVARIATE parameter to true

To test the Ensemble with ARIMA function, set the MODE parameter to 2 and ENSEMBLE parameter to true

To test the Ensemble with SARIMA function, set the MODE parameter to 2, ENSEMBLE parameter to true and SARIMA parameter to true

To test the Random Forest function, set the MODE parameter to 2, FOREST parameter to true

To test the hyperparameters, scroll down on the ADVANCED SETTINGS and then Hyperparameters section in parameters.py.

To change the RNN model used, change the hyperparameters’s RNN\_HYPERPARAM to 1 for LSTM, 2 for SimpleRNN, 3 for GRU

**Overall system architecture**

The overall program uses just 2 python files. The stock\_prediction.py file runs all the functions, while the parameters.py file has all the settings and parameters for the program.

The parameters.py file has various settings such as the ticker symbol, start and end date of the stock data range and how many days into the future to predict are present. Each setting should have a comment to explain what it does and I have already explained what is needed to run the different modes of the program in section of this document detailing how to run the program. As such I will not explain how parameters.py will work.

For the stock\_prediction.py file, it starts with the Main() function at the bottom of the file. This function takes the input from the MODE parameter and puts it into a switch to determine what functions and functionality to use.

Main function used:

A screen shot of a computer program

Description automatically generatedA screen shot of a computer code

Description automatically generated

**Implemented data processing techniques**

The program implements several data processing techniques. This starts with the checkFiles function, which can be seen below.

A screenshot of a computer program

Description automatically generated

This function checks if the waned data has already been downloaded. If it has been, it loads the data from the file, otherwise it downloads the data from the internet and then stores it as a file.

The next step is to split the data into training and test datasets, which is run in one of three ways. The first, is to split the data by a specified date, the code can be seen below,

A screen shot of a computer program

Description automatically generated

the second splits the data by a ratio, the code can be seen below,

A screen shot of a computer program

Description automatically generatedA screen shot of a computer program

Description automatically generated

and the uses a random date to split it, by generating a random date between the start and end date, then inputting that into the split by date function, which can be seen below.

A screen shot of a computer code

Description automatically generated

Another data processing method that is present is to display the downloaded data as a candlestick chart (code below)

A screen shot of a computer code

Description automatically generated

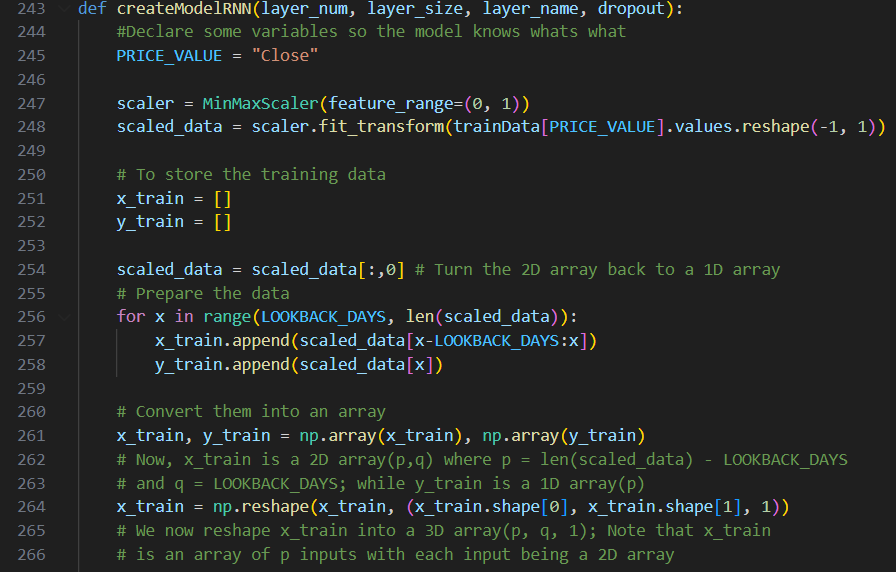
and a boxplot chart (code below).

A computer screen shot of colorful text

Description automatically generated

I believe that these are all the main data processing techniques used in the program.

**Implemented machine learning techniques**

The program implements several machine learning techniques. The first of these is a function to function with several inputs and return a deep learning model. The input parameters are the number of layers, the size of each layer and the RNN model subtype. The code for this function can be seen below. 

A screen shot of a computer program

Description automatically generated

The input variables that are used for this function are modified from the parameters.py, with various hyperparameter presets in the file. They can be viewed below.

A screenshot of a computer

Description automatically generated A screen shot of a computer program

Description automatically generated

The next machine learning technique used is the multistep prediction. This means that the program must be able to predict an amount of days into the future, determined by an input. For this I made the RNN model predict a day in the future, then add that to the model’s input data, which is then used to make a prediction another day into the future. This process repeats, until the input (PREDICTION\_DAYS) days into the future is predicted. The code for this part of the function is below.

A screen shot of a computer code

Description automatically generated

The next machine learning technique used is the multivariate prediction. This means that the program must use the data other than just the closing data as inputs to predict the closing price.

The code for this can be viewed below

A screen shot of a computer program

Description automatically generatedA screen shot of a computer program

Description automatically generatedA screen shot of a computer code

Description automatically generated

The final machine learning technique used that is not part of the extension is an ensemble model of the RNN model and a ARIMA or SARIMA model. An option to use ARIMA or SARIMA is present. The ensemble approach I took, was to get the values for each model and get the average of them to get the ensemble value.

The S/ARIMA code can be viewed below

**A screen shot of a computer program

Description automatically generated**

The ensembling of the data code can be viewed below

A screen shot of a computer program

Description automatically generated

**Details of extension work**

For the extension work I decided to add two new machine learning techniques. The first of these is using facebook’s prophet to make predictions. The code for this is fairly simple concise, being only one function, however due to it’s unique data input method, it has it’s own separate MODE from the rest of the code.

The code for this can be viewed below A screen shot of a computer program

Description automatically generated

The other machine learning technique I added was random forest. This was more complicated having a main function, and three sub functions. The code for this can be viewed below.

A computer screen shot of text

Description automatically generatedA screen shot of a computer program

Description automatically generated

A computer screen shot of a program code

Description automatically generatedA computer screen shot of text

Description automatically generated

Due to the small page limit for this document, please refer to the task 7 report for more info on how these techniques were implemented

**Demonstration of working system**

RNN model predictions with LSTM, then SimpleRNN, then GRU

**A graph with green lines and numbers

Description automatically generatedA graph showing price and price

Description automatically generated with medium confidenceA graph with green lines and numbers

Description automatically generated**

First image is 60 days in the future prediction (multistep), then second is Multivariate prediction

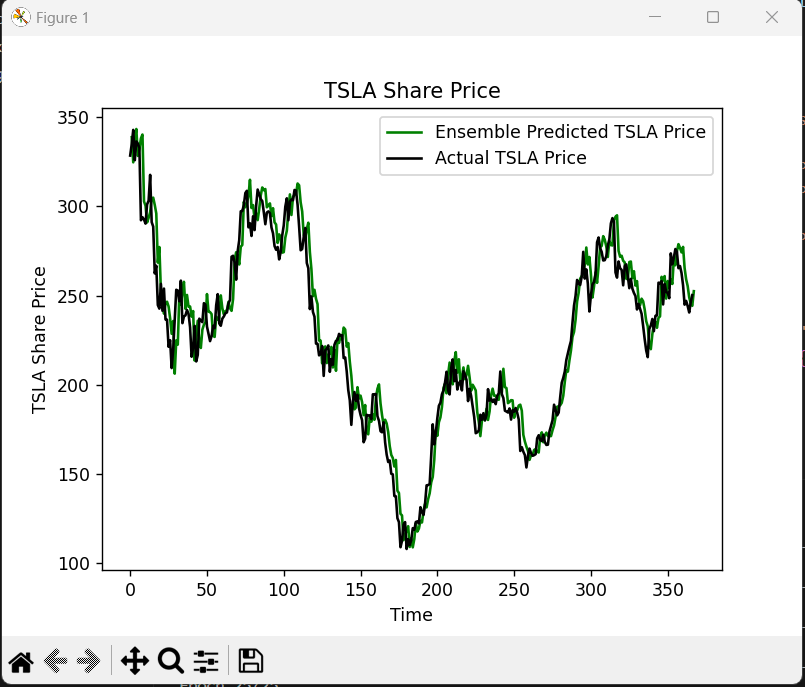
**A graph with green lines and white text

Description automatically generatedA graph with red and green lines

Description automatically generated**

First image is Arima ensemble predictions, then Sarima ensemble predictions

A graph showing the price of a stock market

Description automatically generated

First image is candlestick chart, then boxplot chart

A screenshot of a graph

Description automatically generated A screenshot of a computer screen

Description automatically generated

First image is random forest predictions, then the prophet predictions

A graph showing the price of a stock market

Description automatically generated A screenshot of a graph

Description automatically generated

**Reflection on work done**

Overall I am happy with the work I did for this assignment. I felt like I learnt a lot about the basics for how to make and run several different machine learning models and what each model is useful for. If in the future I use machine learning models I will have already working code that I made that I can use as a launching pad to get the work done.

I would have liked to be able to get the prophet predictions working so they make more than vaguely similar predictions to the actual data. I am happy with how the random forest code runs, although it clearly needs some tweaking to get that working as well as some of the other models.

My only worry is that I completed this report in the way that was expected of me. I am unsure if I was meant to include this many screenshots of the work I did or if it was meant to be filled with explanations of how the code works. I concluded that it should be screenshots and a summary since reports 1-7 contained more detailed explanations of the work done. Please refer to these reports if you wish for further explanations.

**Summary/Conclusion**

For this assignment B, I spent a lot of hours working to get all the features of the program working and I believe I succeeded. In addition to this, I also added some extra machine learning functionality in the form of the simple extension.

URL to github:

https://github.com/R0binicus/Intellegent-Systems-Assignemnt-B