Workflow - Profiling Internet Users

Ghalib Saleem

Code Structure

The code is structured according to its functionality. All the functions related to data are bundled together in a folder, same goes for all the functions and classes.

```
InfoSecProject                         // Root Folder
├── datahandlers
│   ├── data_export.py
│   └── datamain.py
├── helper
│   ├── calc_functions.py
│   ├── helper_operations.py
│   ├── progress_bar.py
│   └── spearman_correlation.py
├── input
│   └── *.xlxs
├── models
│   ├── split_data.py
│   ├── split_item.py
│   ├── user_data.py
│   └── user_info.py
├── output
│   └── *.xlxs
├── saved_obj
│   └── *.obj
├── main.py                            // Starting Point
└──README.md
```

Note: - progress_bar.py is an opensource code snippet created by Aubrey Taylor for printing a progress bar in the console view. Link

Project GitHub Link

Input

The project takes two types of input:

saved_obj/*.obj

input/*.xlsx

The above mention input is required to run the script. It gives an error if not provided. When the program is executed for the very first time. It will read the excel files and save the object files with all the necessary information. So, in the next execution, it will read information from obj files instead of excel files and then perform all the steps.

Note: - To save the time from reading data from excel files, all the saved objects have been uploaded to GitHub along with all the code and excel files. Link

Pre-Requirement

All the libraries should be installed for the project execution and in this program the libraries used are

1. Xlrd
2. Scipy

Execution

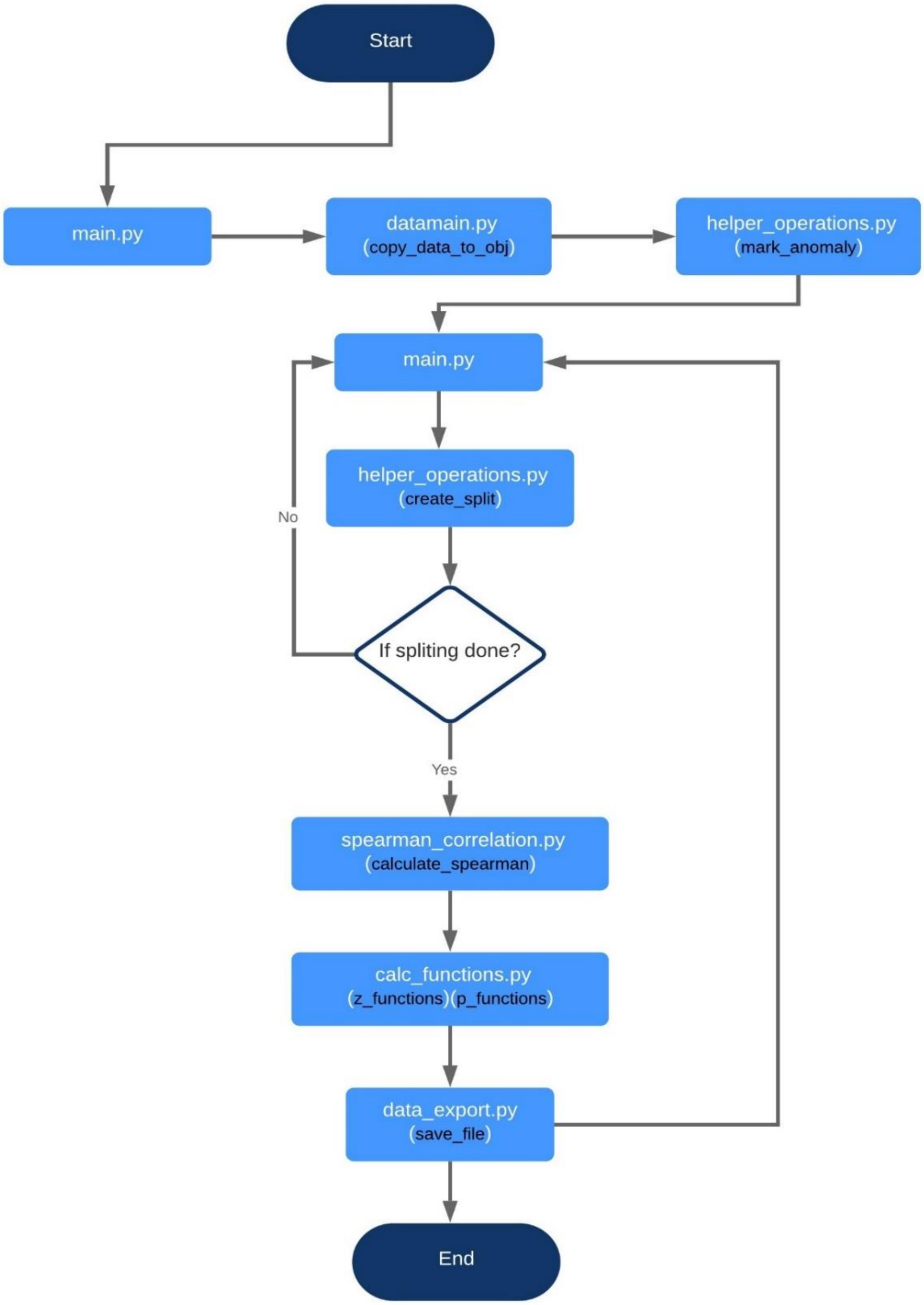
cd <PATH_TO_DOWNLOAD_FOLDER>/InfoSecProject
python3 main.py
When the script runs, it prints all the important steps done or working on it.
Then it will save the output in the “output” folder.

Steps

1. Excel reading is done by "**xlrd**" library. After reading the data from excel files all the data has been filtered according to project requirements like considering data from Monday to Friday between 8 AM to 5 PM and two weeks data only.
2. While filtering the data if User data does not start from Monday and it has more than two weeks data then, in this case, the first week is ignored and if data is not enough then data is considered as it is. This is done so that all the user starts from Monday and ends on Friday.
3. Filtered data has been saved as an obj file using python pickle.
4. All the files which do not have any value after all the above filters are marked as an anomaly.
5. Now the data splitting is done according to the window size for all Users. All the windows which do not have a packet have zero ratios.
6. If the splitting is not successful or does not have any value go back to step 5 for the next windows.
7. Calculate the Spearman Correlation for using "**Scipy**" for all Users Combinations. In the case of previously defined anomaly set correlation to None and in case the user has no data in the whole week set correlation to None.
8. Calculate Z-Value using the given formula for all users.
9. Calculate the P-Value using the given formula for all users.
   Note: - Adjust any value if needed
10. Save the P-Value and Distinguishability in a CSV file in the output folder.

Note: - Step 1,2,3 and 5 are done using parallel programming. So that program can utilize the CPU up to their full potential and speed the heavy-duty process.

Control Flow



Output

The Output folder contains two types of files
P-Value Files
Distinguishability Files
P-Value Files: - There are 3 P-Value Files in the output folder each for 10, 227, 300 seconds window.
Distinguishability Files: - There are 3 Distinguishability files which contain the conclusion of comparison between all the files. All the decision is based on the P-Value of the same comparison.

Sample Output: -

```
************* Start of the program **************
Start: Object reading from file system
Obj read Progress: |██████████████████████████████████████████████████| 100.0%
Completed
End: Object reading from file system
Check: All 54 User is loaded properly
-----------------------------------------------------------------------------------
--------------
Start: Splitting
Splitting Progress: |██████████████████████████████████████████████████| 100.0%
Completed
End: Splitting
End of Create Split for time interval 10
Done processing for Interval : 10
Start: Splitting
Splitting Progress: |██████████████████████████████████████████████████| 100.0%
Completed
End: Splitting
End of Create Split for time interval 227
Done processing for Interval : 227
Start: Splitting
Splitting Progress: |██████████████████████████████████████████████████| 100.0%
Completed
End: Splitting
End of Create Split for time interval 300
Done processing for Interval : 300
************* End of the program **************
```