Assignment 2019

Programming for Data Analysis

Due: last commit on or before November 11th

This document contains the instructions for Assignment 2019 for Programming for Data Analysis. Please be advised that all students are bound by the Quality Assurance Framework [4] at GMIT which includes the Code of Student Conduct and the Policy on Plagiarism. The onus is on the student to ensure they do not, even inadvertently, break the rules. A clean and comprehensive git history (see below) is the best way to demonstrate to the examiner that your submission is your own work. It is, however, expected that you draw on works that are not your own to build your submission and you should systematically reference those works to enhance your submission.

Problem statement

The following assignment concerns the `numpy.random` package in Python [2]. You are required to create a Jupyter [5] notebook explaining the use of the package, including detailed explanations of at least five of the distributions provided for in the package. There are four distinct tasks to be carried out in your Jupyter notebook.

1. Explain the overall purpose of the package.
2. Explain the use of the "Simple random data" and "Permutations" functions.
3. Explain the use and purpose of at least five "Distributions" functions.
4. Explain the use of seeds in generating pseudorandom numbers.

Submission

You must use the version control software git [1] to track your work and you will submit your assignment by providing a URL to your git repository. It is suggested you use GitHub [3] for this purpose and that you consider making your repository publicly available so that prospective employers may view it. However, should you wish to, you may restrict general public access to your repository so long as you give permission to the lecturer to view it. Furthermore, any git repository URL to which you provide access to the lecturer will suffice – you don't have to use GitHub. You must submit the URL

of your git repository using the link on the course Moodle page before the deadline. You can do this at any time, as the last commit before the deadline will be used as your submission for this assignment.

Any submission that does not have a full and incremental git history with informative commit messages over the course of the assignment timeline will be accorded a proportionate mark. It is expected that your repository will have at least tens of commits, with each commit relating to a reasonably small unit of work. In the last week of term, or at any other time, you may be asked by the lecturer to explain the contents of your git repository. While it is encouraged that students will engage in peer learning, any unreferenced documentation and software that is contained in your submission must have been written by you. You can show this by having a long incremental commit history and by being able to explain your code.

Minimum standard

The minimum standard for this assignment is a git repository containing a README, a gitignore file and a Jupyter notebook. The README need only contain an explanation of what is contained in the repository and how to run the Jupyter notebook. Your notebook should contain the main body of work and should list all references used in completing the assignment.

A good submission will be clearly organised and contain concise explanations of the particularities of the dataset. The analysis contained within the notebook will be well conceived, interesting, and well researched. Note that part of this assignment is about the use of Jupyter notebooks and so you should make use of all the functionality available in the software including images, links, code and plots. You may use any Python libraries that you wish, whether they have been discussed in class or not.

Marking scheme

This assignment will be worth 50% of your mark for this module. The following marking scheme will be used to mark the assignment out of 100%. Students should note, however, that in certain circumstances the examiner's overall impression of the assignment may influence marks in each individual component.

25%	**Research**	Investigation of the package as demonstrated by references, background information, and approach.
25%	**Development**	Clear, well-written, and efficient code with appropriate comments.
25%	**Consistency**	Good planning and pragmatic attitude to work as evidenced by commit history.
25%	**Documentation**	Concise descriptions and plots of theoretical and practical aspects of problems.

Advice for students

- Your git commit history should be extensive. A reasonable unit of work for a single commit is a small function, or a handful of comments, or a small change that fixes a bug. If you are well organised you will find it easier to determine the size of a reasonable commit, and it will show in your git history.
- Using information, code and data from outside sources is sometimes acceptable — so long as it is licensed to permit this, you clearly reference the source, and the overall assignment is substantially your own work. Using a source that does not meet these three conditions could jeopardise your mark.
- You must be able to explain your assignment during and after its completion. Bear this in mind when you are writing your README. If you had trouble understanding something in the first place, you will likely have trouble explaining it a couple of weeks later. Write a short explanation of it in your README, so that you can jog your memory later.
- Everyone is susceptible to procrastination and disorganisation. You are expected to be aware of this and take reasonable measures to avoid them. The best way to do this is to draw up an initial straight-forward assignment plan and keep it updated. You can show the examiner that you have done this in several ways. The easiest is to summarise the assignment plan in your README. Another way is to use a to-do list like GitHub Issues.
- Students have problems with assignments from time to time. Some of these are unavoidable, such as external factors relating to family issues or illness. In such cases allowances can sometimes be made. Other problems are preventable, such as missing the submission deadline because you are having internet connectivity issues five minutes before it. Students should be able to show that up until an issue arose they had completed a reasonable and proportionate amount of work and took reasonable steps to avoid preventable issues.

References

[1] Software Freedom Conservancy. Git.
`https://git-scm.com/`.

[2] NumPy developers. Numpy.
`http://www.numpy.org/`.

[3] Inc. GitHub. Github.
`https://github.com/`.

[4] GMIT. Quality assurance framework.
`https://www.gmit.ie/general/quality-assurance-framework`.

[5] Project Jupyter. Project jupyter home.
http://jupyter.org/.