Neuro-Data analysis with advanced python tools

Project guidelines

The purpose of the projects in the course is to apply advanced Python skills to analyze a neuroscience-related dataset. You will explore the data, identify a research question, and answer it using tools covered in the course, while packaging your work as a reproducible and well-documented Python project

The projects are highly recommended to be submitted in pairs. Can also be submitted individually.

Where to start?

- **Dataset Selection:** Choose a publicly available dataset related to neuroscience. Suggested sources such as **OpenNeuro:** https://openneuro.org/ or Kaggle.
- **Define a Research Question**, for example, How does sleep quality correlate with reaction time in a Stroop effect?
- **Data Processing**: explore the data, see how it is distributed, identify outliers, explore relationships between variables
- **Data Analysis**: try to answer your research question. Don't forget visualization!!
- **Results & Conclusion**: Summarize your findings and discuss the results.

Remember: you can use ANY python package. No restrictions.

What to submit?

Project proposal: 1-2 paragraphs. A very short description in which you define the problem, the dataset, the scope of the project and possible limitations. **Due till: Dec 30**

Working code and result: **Due till: Jan 23, 2024**

Including:

Codebase: A well-structured Python project with clear documentation. Include a README file explaining the dataset and the research question. Your code should handle common errors and provide unit test to ensure the correctness of your functions.

Report: 1–2-page summary of the findings. Include a brief overview of the data, your research question, key insights from your analysis, visualizations to support your conclusions.

Code review: **Jan 26-Feb 2, 2025**

Final Submission: Feb 5, 2025

What are the grading criteria? 15% technical complexity: the complexity of data, architecture, amount and complexity of code. 40% correctness: correct understanding of the project goals and implementation, as described in the paper. 10% Results and discussion: project vs. paper results, if different what can explain the difference 10% Testing and

Documentation: sufficient test cases and clear documentation. 10% Error Handling: ensure the code does not fail in unexpected situations. Proper use of try-except, assertion and so on. 5% Efficiency: efficient use of time/space, minimal redundant computations & code 5% Structure and Readability: Clean, well-structured, readable code with proper naming and comments, show proper use of classes, functions and so on. 5% Use of Proper tools and

Good Luck!