Final Project Guidelines

Important dates

1. Project approval due 2/14 (turn in with PS03)
2. Project update due 3/28 (turn in with PS07)
3. Presentation during 5/1 class
4. Project due 5/10 at 5pm EST

Overview

The final project in this class is an opportunity to work on an independent coding problem that benefits the students' research or training. This course builds toward a strong programming foundation in R, but with how rapidly the landscape of statistical computing is changing, it is important students can implement their skills to accomplish new coding tasks that arise.

It is critical to keep in mind this is a **coding** project. The work I expect you to complete for this project is therefore much different than any final project or paper you've done in other classes. The emphasis of this project is the coding element. Your project can involve statistical modeling, but the work I will pay most if not all attention to will be the comprehension and mastery of the coding elements leading up to running the statistical model (or in making sense of the output).

Project ideas

The project has a great deal of flexibility so students can pursue projects that will benefit them. Some common themes might be:

- Data collection
  - Ex: scraping data and organizing into a dataset
- Data cleaning
  - Ex: You've already scraped data, but it is far to messy to use
- Writing new software
  - Ex: blockclustr
- Writing simulations for a methods paper
  - Ex: catSurv
- Writing an RShiny App
  - Ex: visualizing data or model, here kmeans clustering
- Using new software
  - Note: students pursing this need to pay extra attention to the emphasis on the coding portion of this project (rather than the statistical portion)

Idea help

If you are interested, I also have personal projects that students can work on for this class:

1. Cleaning, increasing functionality, testing, and writing vignettes for my online conversation app named Chatter
2. Integrating ChatGPT into Chatter
3. Writing extremely efficient code for simulations involving complex combinatorics. (This project is for an algorithm that finds an optimal solution for constructing groups for group-based experiments)

Expectations

Communication and scope of project

The amount of effort a coding project might take is hard to predict. Some might be much quicker than you imagined. Perhaps someone else has already done the work for you! Or, some might be much more difficult than you imagined.

When at the beginning stages of thinking through a project idea, I expect you'll do sufficient research to make sure someone else hasn't already provided a solution for your problem or task. Projects for this class should not replicate work. Instead, talk to me about ways in which you can advanced the idea further.

In addition to thinking about if the project is not ambitious enough, you'll also have to think about if the project is too ambitious. I am less worried about this! I'd rather you have a lofty goal and make progress toward it in the class, even if you don't reach the finish line.

The **first project checkpoint** (Project approval due 2/14 (turn in with PS03)) will ask you to talk about your idea, to look around to make sure it hasn't been done, and to consider how much you think you can accomplish in the semester. We will communicate to make sure we are on the same page.

The **second project checkpoint** (Project update due 3/28 (turn in with PS07)) should make significant progress toward the goals we agreed to. We will check in and agree to any changes in the goals (either to add more to the project or reduce the scope a bit).

Formatting

The programming field has a strong emphasis on creating public goods. Because I imagine students in this class will have overlapping interests, you might find yourself wanting to use the code generated by a classmate in the future. Therefore, your projects themselves will be publicly visible to your classmates (and I encourage you to make them publicly visible to all when the class concludes).

To do so, you will host your code on Github. You will either create a folder for your project within your fork of the course repository (you/ProgrammingSpring2023/FinalProject) or you will create a new repository for the project (you/FinalProject) that is publicly visible.

Code

The final project will have efficient, clean, and organized code.

- Efficient
  - repetitive tasks should be functionalized
  - computationally intensive code should consider speed
- Clean
  - Code should be readable:
    - Commented
    - Not have too many characters per line (80 is a common maximum)
    - Properly indentend
- Organized
  - Object names should be short, consistent, and meaningful
  - File names, directories, etc. should also be short, consistent, and meaningful
  - Project repository (or folder in course repository) will have a README file that explains all the contents and any additional information needed to replicate the author's steps