Final Project Description

David Gerard

© 2023, David Gerard, do not post online.

Description

For the final project, you will work in teams of 3 people to implement a large, version-controlled data science project. You have your choice of one of three projects:

1. **Web Data Shiny App**: Extract data from the web (either via an API using `{httr2}` or through web scraping using `{rvest}`) and design a Shiny app to visualize and interact with this data.
2. **API R Package**: Create an R package that connects to an API using the `{httr2}` package.
3. **Methodology R Package**: Develop an R package which implements a basic statistical or data science method not currently present in R.

Note: Collaborate with your team to decide upon your preferred project format and subject matter.

Grades

The grades are out of 23 for undergraduates (413), and out of 25 for graduates (613).

- 1 pt: Team and Project Selection
- 2 pts: Progress Report
- 5 pts: GitHub Repo
- 10 pts: Shiny App or R Package
- 5 pts: Presentation
- 2 pts: Graduate Student Project

Important Dates:

- 11/02/2023: Project Selection Due.
- 11/30/2023: Progress Report Due.
- 12/13/2023: Presentations.

Form 1: Shiny App

- Construct a Shiny App that draws upon data sourced from the web.
- Data should be fetched either via an API (with `{httr2}`) or through web scraping (with `{rvest}`). Direct CSV/XLSX/etc downloads are not permitted.
- Here is a list of public APIs that you can look at: https://github.com/public-apis/public-apis
- Evaluation criteria includes app sophistication, analysis complexity, and the enhancement of data representation through reactive elements.

Essential Features:

- A polished and intuitive layout.
- Engaging interactive graphics.
- Exclusion of in-app intensive data cleaning; utilize pre-cleaned datasets sourced from your analysis scripts, unless your app has real-time updates from the API.
- Efficient use of reactive elements for improved data portrayal.
- Examples for what reactive elements improve:
  - Highly improves:
    - Change a plot based on the time the units were collected, demonstrating an evolving relationship between variables.
    - Plot data according to the different levels of a categorical variable to better explore multidimensional relationships.
  - Moderately improves:
    - Selecting variables to plot.
    - Changing bin width of a histogram to look at different scales of the distribution.
    - Demonstrating how variable transformations result in linear relationships.
  - Does not improve:
    - Delayed evaluation where the evaluation would be instantaneous anyway.

Examples:

- Replicate renowned data visualization milestones found here.
- Provide up-to-date info on WMATA trains, along with a map of train positions.
- Provide up-to-date statistics for games from the Canadian Football League. If you've taken machine learning, you can train a basic model (like random forest) and provide win predictions based on a small number of features (e.g., current score, time remaining, and past team performance).

Form 2: API R Package

- Employ the {httr2} package to design an R package interfacing with a chosen API.
- Here is a list of public APIs that you can look at: https://github.com/public-apis/public-apis
- Restriction: Do not select an API that already has an associated R package (unless it's developed using `{httr}` instead of `{httr2}`).
- You should choose an API that is fairly complex, with multiple endpoints/queries.
- Required features:
  - User-friendly data retrieval in tidy data frames,
  - Full documentation.
  - Ensure R functions created are able to generate HTTP queries through `{httr2}`.
  - Stick to APIs demanding either an API key or OAuth.
  - Incorporate best practices for API packages as detailed in the `{httr2}` vignette.
  - Exhibit vast functionality within the package, exploring different facets of the API.
  - Maintain impeccable documentation, rigorous unit test coverage (> 70), and a clean coding style. Ensure it passes checks via continuous integration.
- Example good APIs, but there are tons:
  - Best Buy
  - Canadian Football League
    - There is a GitHub package that does this I know about but they use `{httr}`, not `{httr2}`. Either way, don't look at that repo.
  - Chicago Transit Authority
    - Again, there is a GitHub package that does this I know about. But again, they use `{httr}`, not `{httr2}`. Please don't look at that repo.

Form 3: Statistical/Data Method R Package

Build an R package that implements a basic statistical procedure that is currently not available. You will be graded on the following elements:

- Your package is sophisticated, containing multiple, connected functions. Please work with me to get your project pre-approved.
- Your package correctly implements the statistical/data procedure.
- Your package is user-friendly.
- Your package is well covered by unit tests
- Your package passes all checks via `R CMD check` in Continuous Integration on GitHub.
- Your package is well documented.
- Your package should have high unit test coverage and be written in a good style.

Possible Projects

Below is a list of possible project, but I will be very open to hearing your suggestions if you would like to go this route.

$n = 1$ inference

- Wall, Boen, and Tweedie (2001) provide an approach to calculate a finite confidence interval for the mean of a normal distribution when $n = 1$. Specifically, if $x \sim N(\mu, \sigma^2)$, then the confidence interval is of the form:

  $$\mu \in x \pm \xi \|x\|,$$

  for some $\xi > 0$. This work extends the approaches from Blachman and Machol (1987).
- For this project, I would like you to create an R package for computing confidence intervals for means when the sample size is 1. The details I have in mind are below.

Possible functionality:

1. Wall, Boen, and Tweedie (2001) list a couple ways to derive ξ (through either numerically solving an equation, or through closed-form approximations). Implement all of these methods.
2. Your function should allow the user to choose the coverage probability.

3. You can arbitrarily shift x by some constant a and obtain $x - a \sim N(\mu - a, \sigma^2)$. So another confidence interval would be:

$$\mu \in x - a \pm \xi \|x - a\|$$

   So allow the user to choose a.
4. Use the confidence interval to obtain a p-value against some null.
5. Wall, Boen, and Tweedie (2001) also describe confidence intervals of a similar form when $n > 1$, but are sub-optimal when $n > 2$. Implement these approaches and have a vignette comparing it to the standard t-intervals.
6. There exist extensions to confidence intervals for variances when $n = 1$ (Portnoy and DasGupta 2022). Implement it!

Tests for Normality

Implement all of the tests in D'agostino, Belanger, and D'Agostino Jr (1990)

- Calculate the sample skew and kurtosis
- Calculate the Fisher-g estimators of skew and kurtosis
- Compute the test for skewness
- Compute the test for kurtosis
- Compute the Omnibus Test
- Create your own function to run a normal probability plot, which also displays the estimated skew and kurtosis.
- There are nine other tests for normality in Shapiro, Wilk, and Chen (1968). Implement them!

Early Data Package

- Make a dataset based package containing the data from the earliest known data visualizations (Michael Friendly and Ulargui 2010).
- See https://www.datavis.ca/milestones/ for a more comprehensive data base on this.
- You can parse the data then make functions to re-create the oldest data visualizations.
- The data could be available to the user in the form of a package.

(2 pts) Progress Report

The progress report is a 1 page report (double spaced, 12 pt font, 1 in margins) describing what you have done so far, and what you still need to do.

For shiny apps, I expect all data to have been downloaded and cleaned into a tidy CSV file (or multiple CSV files). Demonstrate that you have done this. If, on the other hand, your app makes real-time updates based on an API, then demonstrate that you can send and receive simple queries and have authentication established.

For API R packages, I expect you to be able to send and receive simple queries, and to have authentication already established in the package.

For analysis R packages, I expect you to have one or two basic functions created implementing some of the approaches in the topic you have listed.

(5 pts) GitHub Repo

- Your repo should have a consistent commit history with clear commit messages.
- Your repo should show a clear division of labor. I can see who wrote what lines of code in your repo, so spreading out the work will be a part of your grade. If I see that one individual has not done anything, this will affect their grade.
- Your repo should show that you have worked on the project regularly since it was assigned. Avoid waiting until the last minute to begin your project. Regular commits indicate continuous effort. Your grade in this section depends on not waiting until the last moment.

(5 pts) Presentation

- Show off your work to your fellow students!
- Prepare a 20 minute presentation demonstrating your R package or your shiny app.
- You don't need to make any slides here. But they can help to explain statistical methods or what the data are.
- Mostly, I just want you to demo the R package under various use-cases, or demo the shiny app under various use cases.
- You should consider this to be like a job interview where you are trying to impress me with your previous data science work.

- You have seen me demonstrate new packages and methods all semester, so you should be pretty comfortable with the format I'm looking for.

(2 pts) Graduate Students

If your group contains graduate students (defined as being signed up for 613), then there will be a little extra work for them. This is to differentiate the class between 413 and 613.

- If you are building an R package (project forms 2 and 3), read my notes on S3 methods and implement S3 methods for `plot()` and `summary()` based on new classes in your package.
  - E.g. implement a plot method based on the output of an API call.
  - E.g. implement a plot method based on the output of a statistical procedure.
- If you are building a Shiny App, you must include one statistical procedure in your app and explain the results of your statistical analysis during your presentation.
  - If you have taken Regression or machine learning, I expect a complicated analysis, such as multiple linear regression, ANOVA, or some machine learning prediction algorithm.
  - If you have only had basic or intermediate statistics, it's OK to only do simple linear regression. See my notes here. But you should really take regression and machine learning if you want a job in Data Science.

References

Blachman, N., and R. Machol. 1987. "Confidence Intervals Based on One or More Observations." *IEEE Transactions on Information Theory* 33 (3): 373–82. https://doi.org/10.1109/TIT.1987.1057306.

D'agostino, Ralph B, Albert Belanger, and Ralph B D'Agostino Jr. 1990. "A Suggestion for Using Powerful and Informative Tests of Normality." *The American Statistician* 44 (4): 316–21. https://doi.org/10.1080/00031305.1990.10475751.

Michael Friendly, Pedro Valero-Mora, and Joaquín Ibáñez Ulargui. 2010. "The First (Known) Statistical Graph: Michael Florent van Langren and the 'Secret' of Longitude." *The American Statistician* 64 (2): 174–84. https://doi.org/10.1198/tast.2010.09154.

Portnoy, Stephen, and Anirban DasGupta. 2022. "Valid Confidence Intervals for μ, σ When There Is Only One Observation Available." *arXiv*. https://doi.org/10.48550/arXiv.2202.03556.

Shapiro, Samuel S, Martin B Wilk, and Hwei J Chen. 1968. "A Comparative Study of Various Tests for Normality." *Journal of the American Statistical Association* 63 (324): 1343–72.

Wall, Melanie M., James Boen, and Richard Tweedie. 2001. "An Effective Confidence Interval for the Mean with Samples of Size One and Two." *The American Statistician* 55 (2): 102–5. https://doi.org/10.2307/2685995.