Project 1

Vignettes

Vignettes are explanations of some concept, package, etc. with text, code, and output interweaved. We already know how to make them with R Markdown!

Project Objectives

This project involves creating a vignette (an HTML file with a table of contents) about reading in data in JSON format and exploring it (summaries and graphs - more details on this later). You will then create a blog post linking to your vignette.

Project Work

All project work should be done in a github repo. Ideally, you have connected RStudio with github and can work from the command line within RStudio. All major updates should be made through github so we can track your activity.

Vignette Content Details

You are going to create a vignette for reading and summarizing a JSON data set. You'll contact the National Hockey League (NHL) API. The components that must be present in each vignette include:

- Describe JSON data. What is it, where does it get used, and why is it a good way to store data? This should be detailed enough that someone that hasn't seen JSON data would have a good idea what they are dealing with. You should link to references where applicable.
- Discuss the possible packages/functions that are available for reading JSON data into R. (There are three major packages for JSON data.) Choose one and explain why you've chosen it.
- Write functions to contact the NHL records 'Franchise' API (the previous text is a link) and return well-formatted, parsed data:
  - You should have a function to return parsed data from the following calls:
    - /franchise (Returns id, firstSeasonId and lastSeasonId and name of every team in the history of the NHL)
    - /franchise-team-totals (Returns Total stats for every franchise (ex roadTies, roadWins, etc))
    - /site/api/franchise-season-records?cayenneExp=franchiseId=ID (Drill-down into season records for a specific franchise)
      - User should be able to specify the franchise
    - /franchise-goalie-records?cayenneExp=franchiseId=ID (Goalie records for the specified franchise)
      - User should be able to specify the franchise
    - /franchise-skater-records?cayenneExp=franchiseId=ID (Skater records, same interaction as goalie endpoint)
      - User should be able to specify the franchise
- Once you have the functions to query the data, you should perform a basic exploratory data analysis. Not all things reported need to show something interesting or meaningful (i.e. graphs that show no relationship are fine) but you should discuss each graph (if you don't know hockey, that is ok - simply discuss the graphs and summaries as best you can). A few requirements are below:

  - You should create a new variable at some point
  - You should create some contingency tables and numeric summaries by some of your categorical variables
  - You should create some plots (at least a side-by-side bar plot, side-by-side box plots, and scatter plots with coloring)
- Your code chunks should be shown in the final document unless they are set up chunks or other behind the scenes things that aren't important.

Blog Post and Repo Stuff

On your project repo you should go into the settings and enable github pages (feel free to select a theme too!). This will make it so your repo can be accessed like your blog (username.github.io/repo-name).

Knit your .Rmd file with the output type `output: rmarkdown::github_document` in the YAML header. This will create a .md file which will automatically be rendered by github.

Rename this as the README.md file for the repo. This should make the website for the repo correspond to the output of your .Rmd file (essentially).

Once you've completed your vignette you should write a brief blog post outlining your project and the findings you made. You should also reflect on the process you went through for this project. Discuss things like:

- what would you do differently?
- what was the most difficult part for you?
- what are your big take-aways from this project?

Rubric for Grading (total = 100 points)

Item	Points	Notes
General use of Good Programming Practices	10	All code should be indented, follow naming conventions, etc.
Use of headings, table of contents, chunk options, etc.	5	Worth either 0, 3, or 5
Data type (JSON) description	12	Worth either 0, 4, 8, or 12 points
Discussion of relevant R packages	8	Worth either 0, 2, 4, 6, or 10
Functions to query the API	30	Worth either 0, 5, 10, . . . , or 30 points
Creation of relevant new variable(s)	5	Worth either 0, 3, or 5
Calculation of relevant numeric & graphical summaries	25	Worth either 0, 5, . . . , or 25 points
Blog post and repo setup	5	Worth either 0, 2, 4, or 5 points

Notes on grading:

- For items worth say 0, 5, 10, or 15 points, we will generally move you down one level for each syntax, logical, or other error present in the code. The same holds true for missing a required item or lacking in a description.
- If your work was not completed and documented using your github repo you will lose 15 points on the project.