Problem Set #05 Statistics 506

Due: Nov 21, 10am on Canvas

Instructions

- **Review the attribution of sources discussions.**
- Submit the output of your Quarto document on Canvas. The output should include a link to your GitHub repository for this assignment.
- Unless otherwise explicitly stated, all problems should be solved in **R**.
- Your output file should include all code required to solve the problem. Code folding may be useful to make the output more readable.
- Use a consistent and readable code style for your code. Lack of a consistent and readable code style will negatively affect your grade.
- Some of these exercises may require you to use commands or techniques that were not covered in class or in the course notes. You can use the web as needed to identify appropriate approaches. Part of the purpose of these exercises is for you to learn to be resourceful and self sufficient. Questions are welcome at all times, but please make an attempt to locate relevant information yourself first.
- Be sure to properly document any functions you write using roxygen, and add comments as appropriate to make it clear what you are doing.
- If submitting an HTML file, please make sure to make it self-contained.

Problem 1 - Plotting

Use the "nnmaps" data set again. I'd recommend using **ggplot2** for these, but you can do them in base R or **plotly** if you'd prefer.

a. Produce a graph of the **mean** monthly temperature in **celsius**. Draw a scatter-plot where the x-axis is month and the y-axis is the average monthly temperature in celsius. Add a line connecting the points within each season and color the lines and points by season (same color for lines and points in the same season). Be sure both x-axis and

the legend are ordered appropriately, and that all labels/titles are clear and ready for publication (e.g. no variable names).

b. Produce a similar plot with four sets of lines: mean monthly temperature in celsius, mean montly O3, mean monthly PM10, and mean monthly dewpoint. Figure out a way to distinguish between the four lines. Again, make sure the plot is publication-ready (clean and clear labels). Use the plot to answer the following question:

   "Which of the four variables seems to have the least seasonal trend?"

Problem 2 - OOP Programming

Create a class to represent a polynomial expression (such as $7x^3 - 3x + 2$) called `poly`. Do this using S4.

a. For the `poly` class, define the following:

   - A constructor
   - A validator
   - A `show` method
   - Addition and substraction

b. Use your poly class to run the following code:

```
p1 <- make_poly("3x^2 + 2")
p2 <- make_poly("7x^3 - 2x^2 - x + 17")
p1
p2
p1 + p2
p1 - p2
```

   (Your constructor does not need to be called `make_poly` nor take in a string - see below. You should construct the same polynomial expression in your code.)

Note that there are a lot of choices to be made here. How are you going to store the class? A named list? A named vector? A string? Separate vectors for coefficient and power? A matrix? A `data.frame`? What are users going to pass into the constructor? A string (as in my example above?)? A named list? A named vector? Separate vectors? Etc.

Each decision has different impacts on the complexity of other bits. Make one choice to make your constructor easy and it may make arithmetic harder. Change the way `poly` are stored and suddenly printing is hard to work with. This will likely be an interative process - make a preliminary choice, then tweak it as you go.

You only need handle the case with one indeterminate, and you may assume that indeterminate is always `"x"` (unless you want to allow users to input a different indeterminate).

Very minor extra credit for ensuring that `show` reduces noice by a) reducing x^1 and x^0 to x and 1 as appropriate, and b) suppressing terms with coefficients of 0.

Problem 3 - data.table

Repeat Problem 1 from PS04 using `data.table`.