Problem Set #01 Statistics 506

Due: Sept 12, 10am on Canvas

Instructions

- **Review the attribution of sources discussions.**
- Submit the output of your Quarto document on Canvas, along with 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.

Problem 1 - Wine data

From https://archive.ics.uci.edu/dataset/109/wine, download the data set about wine. It contains two files of interest - "wine.data" with the actual rectangular data set, and "wine.names" with some information about the data. (Both files are plain-text - you can open then in any text editor, including directly in RStudio.)

a. Import the data into a `data.frame` in R. Use the information in the "wine.names" file to give appropriate column names. (Note: Downloading and unzipping the file can take place outside of your submitted document, but importing the file should be in the submission.)

b. The data contains information on three different classes of wine. Ensure that the number of wines within each class is correct as reported in "wine.names".

c. Use the data to answer the following questions:

   1. The wine with the highest alcohol content belongs to which class?
   2. The wine with the lowest alcohol content belongs to which class?
   3. German beers have, on average, 114 mg/l of magnesium. How many of these wines have higher levels of magnesium than that? (You may assume that the magnesium reported in the data is in the same units.)
   4. Within each class, how many wines have higher levels of magnesium than average German beer?

d. Create a table identifying the average value of each variable, providing one row for the overall average, and one row per class with class averages. (This table does not need to be "fancy" but should clearly identify what each value represents.)

e. Carry out a series of t-tests to examine whether the level of Ash differs across the three classes. Present the R output and interpret the results. (You may use an existing R function to carry out the t-test, or for **minor extra credit**, manually write your own calculation of the t-test p-values.)

Problem 2 - Perfect Powers

Perfect squares are integers which are squares of integers. E.g. 16 is a perfect square since $4^2 = 16$ with 4 as it's root with power 2; 20 is not a perfect square since $\sqrt{20} \approx 4.47$, which is not an integer. Perfect cubes are similar for third power, e.g. 216 is a perfect cube since $6^3 = 216$, 6 is the root with power 3 for 216. More generally, an integer n is a "perfect power" if there exists integers r (called the root) and p (called the power) such that $r^p = n$. (Update 9/12: For $p \geq 2$.)

a. Write a function "`isPerfectPower`" to identify whether a given integer is a perfect power for a given power. Do not use any existing functions to check this; do so with arithmetic.

   - Input: Two integers - the number to check, and specified power.
   - Output: A `list` of length 2; the first entry should be a logical, the second should be the root of the input or a reasonable value if the input is not a perfect power.

   E.g.

```
> isPerfectPower(27, power = 3)
$isPerfect
[1] TRUE
```

```
$root
[1] 3
```

b. Demonstrate your function works. Do so by writing another function "`findRootPower`" which calls your first function, using a loop to identify both the root and power. Your function should identify the *lowest* power for which the input is perfect (e.g. 64 is both 4^4 and 8^2, you should return 8^2 since $2 < 4$).

- Input: An integer.
- Output: The root and the power as a valid string equation with an appropriate message if the input is not a perfect power.

E.g.

```
> findRootPower(125)
[1] "125 = 5^3"
```

Use this new function to identify whether each of the following is a perfect power, and if so what its root and power is.

1. 27
2. 13060694016
3. 7776
4. 170859375
5. 58247422
6. 94143178827

Hints:

- Remember our discussion about floating power double numeric precision.
- For part b., make sure to check high enough powers, but don't let it run to absurdly high powers.
- Functions can `return` at any point, not just at the bottom.

Problem 3 - ChatGPT

Let's lean into ChatGPT and other LLM (Large Language Models).

a. Put the following prompt into ChatGPT (or your favorite other LLM) and copy its output (both the code it produces as well as any text it generates) into your submission. (If a non-ChatGPT LLM produces nonsense results, try ChatGPT. If that is still producing nonsense, let me know. Blockquotes might be useful for displaying this inside your Quarto document.)

Produce two R functions: The first should take in a vector of suits and a vector of ranks representing a 5-card hand, and return the name of the hand (in 5 card stud poker). The second should simulate dealing a round of cards in a game of poker (5 card stud). The user should be able to specify the number of players. Show the user all the hands, and use the first function to display the name of each of the hands.

b. See if the code runs without modification in R. If so, demonstrate it with a range of inputs. If not, fix it and explain what you fixed. (If you identify that the code is *massively* broken, try re-generating the response.) At this point we only care if the code runs without errors for a variety on inputs; we'll make sure it's correct below.

c. **Without asking ChatGPT or another LLM**, explain line-by-line what the code is doing. Don't forget about `help()` if it uses functions you aren't familiar with. The easiest way to display this would be to throughly comment the code. I'm looking for roughly one comment for each line of code. (Roughly because some lines of code are trivial, e.g. `x <- 4`; whereas others may require several lines of comments to explain.)

d. Determine whether the code produces accurate results. Explain how you made this determination. Check at least the following:

   - Are the inputs and outputs as described above?
   - Are the hands valid (e.g. real cards, no duplicates, right number)?
   - Are the names of the hands correct? (See here if you're not familiar with poker hands.)
   - Does it ensure no duplicates in cards across hands? What happens if you ask for more than 10 hands to be dealt (as there are only 52 cards in a standard deck)?

   Include all tests you carry out. (Set a random seed to make the results deterministic for the discussion.) If you find it is producing inaccurate results explain the following for each problem:

   1. What the error is.
   2. What is causing the error,
   3. Also, attempt to fix it. You will be graded on your attempt to fix the bug, not whether it's ultimately successful.