E401/M518: Problem Set 4a

Programming in R

Fall 2023

Due: December 5 2022 (for extra credit only)

Please work on the following questions and hand in your solutions in groups of at most 2 students. You are asked to answer all questions, but I will only select 2 questions randomly to grade.

Part 1: R questions

Question 1: Functions in R

1. Practice turning the following code snippets into functions. Think about what each function does. What would you call it? How many arguments does it need? Can you rewrite it to be more expressive or less duplicative?

```
mean(is.na(x))
x / sum(x, na.rm = TRUE)
sd(x, na.rm = TRUE) / mean(x, na.rm = TRUE)
```

2. Write `both_na()`, a function that takes two vectors of the same length and returns a vector with all positions that have an `NA` in both vectors.

3. What do the following functions do? Why are they useful even though they are so short?

```
is_directory <- function(x) file.info(x)$isdir
is_readable <- function(x) file.access(x, 4) == 0
```

4. Read the source code for each of the following three functions, puzzle out what they do, and then brainstorm better names.

```
f1 <- function(string, prefix) {
  substr(string, 1, nchar(prefix)) == prefix
}
f2 <- function(x) {
  if (length(x) <= 1) return(NULL)
```

```
  x[-length(x)]
}
f3 <- function(x, y) {
  rep(y, length.out = length(x))
}
```

5. Implement a `fizzbuzz` function. It takes a single number as input. If the number is divisible by three, it returns "fizz". If it's divisible by five it returns "buzz". If it's divisible by three and five, it returns "fizzbuzz". Otherwise, it returns the number. Make sure you first write working code before you create the function.

6. How could you use `cut()` to simplify this set of nested if-else statements? How would you change the call to `cut()` if I'd used `<` instead of `<=`? What is the other chief advantage of `cut()` for this problem? (Hint: What happens if you have many values in `temp`?)

```
if (temp <= 0) {
  "freezing"
} else if (temp <= 10) {
  "cold"
} else if (temp <= 20) {
  "cool"
} else if (temp <= 30) {
  "warm"
} else {
  "hot"
}
```

7. What does this `switch()` call do? What happens if `x` is "e"? Experiment, then carefully read the documentation.

```
# Test the switch function.
x <- "e"
switch_out <- switch(x,
  a = ,
  b = "ab",
  c = ,
  d = "cd"
)
switch_out
```

Question 2: Vectors and lists in R

1. Carefully read the documentation of `is.vector()`. What does it actually test for? Why does `is.atomic()` not agree with the definition of atomic vectors above?

2. Create functions that take a vector as input and return:

    1. The last value. Should you use `[` or `[[`?
    2. The elements at even numbered positions.
    3. Every element except the last value.
    4. Only even numbers.
3. Why is `x[-which(x > 0)]` not the same as `x[x <= 0]`?
4. What happens when you subset with a positive integer that's bigger than the length of the vector? What happens when you subset with a name that doesn't exist?
5. Draw the following lists as nested sets:
    1. `list(a, b, list(c, d), list(e, f))`
    2. `list(list(list(list(list(list(a)))))`

Question 3: Iteration in R

1. Write for loops to do the following tasks. Think about the output, sequence, and body **before** you start writing the loop.
    1. Compute the mean of every column in `mtcars`.
    2. Determine the type of each column in `nycflights13::flights`.
    3. Compute the number of unique values in each column of `iris`. The iris-data that we discussed during our Python tutorial is also one of the standard data sets in R and can be loaded using `data("iris)`.
    4. Generate 10 random normals for each of $\mu = -10$, 0, 10, and 100.
2. Eliminate the for loop in each of the following examples by taking advantage of an existing function that works with vectors:

```
out <- ""
for (x in letters) {
out <- stringr::str_c(out, x)
}

x <- sample(100)
sd <- 0
for (i in seq_along(x)) {
sd <- sd + (x[i] - mean(x)) ^ 2
}
sd <- sqrt(sd / (length(x) - 1))

x <- runif(100)
out <- vector("numeric", length(x))
out[1] <- x[1]
for (i in 2:length(x)) {
```

```
out[i] <- out[i - 1] + x[i]
}
```

3. It's common to see for loops that don't preallocate the output and instead increase the length of a vector at each step:

```
output <- vector("integer", 0)
for (i in seq_along(x)) {
  output <- c(output, lengths(x[[i]]))
}
output
```

   How does this affect performance? Design and execute an experiment.

Part 2: Your project

Continue working on your project. This week, think about how you want to structure and write your report and how you would like to document your analyses. Experiment with RMarkdown to see whether it suits your workflow well. Set up a Markdown document that outlines the structure of your final project report: Add some meta data for the title page, create some sections, write a paragraph of text summarizing your research question, create some graphs visualizing your data and include them in your document. There's no need for handing in this document. Instead, in **at most** one or two paragraphs, describe your RMarkdown experience. Do you think you will use it for writing your project report? What do you like about it? What features are you missing?

Since I usually get a lot of questions about how your project writeup should look like: It obviously depends heavily on which specific topic you are writing about. A good general guideline is the structure of a typical empirical paper in Economics, which consists of the following sections:

1. Abstract: This is not necessary for your project, but it can be helpful to organize your thoughts. An abstract typically consists of less than 100 words that summarizes the essence of your paper.
2. Introduction: Outline your research question and why it is interesting, summarize the data and the main methods and your main results. This should not be more than 2 pages of your report. In an actual paper you would also include a literature review that explains how your analysis relates to related work that other people have done (this is *not* necessary for your course project).
3. Data description: Describe where your data is coming from, what it contains, how it was collected, what the good and bad features are. Then, provide some descriptive statistics of the most important variables that you're using in your analysis. If your exploratory data analysis is relatively brief you can also do it in this section (a few tables, graphs, etc.)
4. Exploratory data analysis: In an actual paper this would probably not be a separate section. Since EDA was a relatively big part of our course, it might make sense to

spend a bit more time on this and write this extra section with several (informative!) graphs that visualize the main features of the data, and some tables that illustrate the variation you have in your data.

5. Econometric methods and model description: Describe the econometric methods you use for analyzing your data. Argue why it makes sense to use a particular method (or sequence of methods) in your application. Specify precisely which model(s) you are estimating, i.e., what is the dependent variable, what are the regressors, how do you choose tuning parameters, and so on. You shouldn't write a book and explain every detail of the method (although you may want to write up more details if you think it helps you to understand the method better), but your description should be detailed enough so that any of your classmates is able to reproduce your analysis.
6. Discussion of your results: Present your table(s) of results with regression outputs etc. and interpret the numbers. Highlight which are the most important and most surprising findings. Obviously, for your course project, there might not be a lot of interesting surprises, but that's totally fine.
7. Robustness checks: This is another section that you often see in empirical papers. Its main purpose is to show that your main results are not sensitive to the details of how you specified the model. Some things you would typically do are: add some more regressors, use a different transformation of regressors or dependent variable, use a slightly different method (e.g., LASSO instead of Ridge, probit instead of logit, add various layers of fixed effects to your panel regression, etc.). Again, for your course project I don't expect much here and it's probably not worth to write a separate section on this.
8. Conclusion: In about one page, summarize your main findings and takeaway message. If your analysis gave you some inspiration on new questions that should be analyzed then you can also use the conclusion to add an "outlook" paragraph that might be helpful guidance for other people that are interested in working on the same topic.