R Review Project Part 1

Review Questions

General Concepts

1. What is TCGA & why is it important?
   TCGA stands for "The Cancer Genome Atlas." The TCGA was organized by the National Cancer Institute (NCI) and the National Human Genome Research Institute (NHGRI) with the goal of creating a public database that displayed a wide range of genes across a vast patient sample. TCGA contains genomic data from over 20,000 samples of thirty three cancer types and presents clinical data, mutation data, and protein data (from the CPTAC). Because TCGA is a large public database that provided both clinical and multi-omic data, it served a crucial role in advancing our understanding of cancer.

2. What are some strengths & weaknesses of TCGA?
   Strengths of TCGA include having a large patient sample for various different cancer types and displaying numerous forms of data, such as clinical, mutation, and protein data. This allows researchers to collaborate with one another and carefully analyze patient and multi-omic data to make new developments in the field of cancer. On the other hand, however, TCGA does not show the progression of cancer in patients over time since they are collected at one point in time. TCGA also only focuses on common cancers and underrepresents non-white demographics.

Coding Skills

1. What commands are used to save a file to your GitHub repository?
   The GitHub commands used to save a file are as follows: use cd to go to your local repository, use git status to check which files have local changes that need to be uploaded to GitHub, use git add add a file, use git commit-m to commit changes, and use git push to push changes into GitHub repository.

2. What commands must be run in order to use a package in R?
   To use a package in R, you must first install the package. Use **install.packages("I_Am_Awesome")**, then **library(I_Am_Awesome)**.

3. What commands must be run in order to use a *Bioconductor* package in R?
   To use a *Bioconductor* package in R, you must first install the package. Use **if (!require("BiocManager", quietly = TRUE)) install.packages("BiocManager") BiocManager::install(version = "3.17")**, then **library(BiocManager)**.

4. What is Boolean Indexing? What are some applications of it?
   Boolean indexing is an effective R technique for data cleaning and subsetting/selection.

Boolean indexing involves creating a vector where selected data is true to filter and create a mask. You can use Boolean indexing to keep certain data, delete null data, subset data (ex: young/old, male/female), select certain data points based on a row or column, finding barcodes of certain patients, etc.

5. Draw a mock-up (just a few rows & columns) of a sample dataframe. Show an example of the following & explain what each line of code does.
    a. An ifelse() statement

```r
data <- data.frame(Name = c("Bartholomew", "Bob", "Charlie", "David"),
                   Age = c(2, 14, 39, 17))
#Using the data.frame function, a new data frame is created, called "data." The two columns are "Name" (which contains "Bartholomew", "Bob", "Charlie", "David") and "Age" (which contains 2, 14, 39, 17).

data$AgeCategory <- ifelse(data$Age >= 18, "Adult", "Minor")
#I am adding a new column called "AgeCategory" to categorize the Names as either "Adult" or "Minor" based on their age. The ifelse is a condition operation that identifies this. If the age is greater than or equal to 18, the name will be categorized as "Adult," and if the age is less than 18, the name will be categorized as "Minor."

print(data)
#Using the print function, the data.frame "data" is now displayed below. It will show the name, age, and age category.
```

Description: df [4 × 3]

Name <chr>	Age <dbl>	AgeCategory <chr>
Bartholomew	2	Minor
Bob	14	Minor
Charlie	39	Adult
David	17	Minor

4 rows

    b. Boolean Indexing

```r
data <- data.frame(Name = c("Bartholomew", "Bob", "Charlie", "David"),
                   Age = c(2, 14, 39, 17))
#Using the data.frame function, a new data frame is created, called "data." The two columns are "Name" (which contains "Bartholomew", "Bob", "Charlie", "David") and "Age" (which contains 2, 14, 39, 17).

data$AgeCategory <- ifelse(data$Age >= 18, "Adult", "Minor")
#I am adding a new column called "AgeCategory" to categorize the Names as either "Adult" or "Minor" based on their age. The ifelse is a condition operation that identifies this. If the age is greater than or equal to 18, the name will be categorized as "Adult," and if the age is less than 18, the name will be categorized as "Minor."

data <- data[data$AgeCategory == "Minor", ]
#This line of code uses Boolean indexing to filter out all of the adults from the data frame "data."

print(data)
#Using the print function, the data.frame "data" is now displayed below. It will show the name, age, and age category. Because adults were filtered out, the data will only show the names and ages of those that are minors.
```

Description: df [3 × 3]

	Name <chr>	Age <dbl>	AgeCategory <chr>
1	Bartholomew	2	Minor
2	Bob	14	Minor
4	David	17	Minor

3 rows