Exercise: Representing Data with R

Preparation

Install R and RStudio

Download and install R and RStudio on your computer.

- Download R
- Download RStudio

Create a new project in RStudio and add a new R-script file.

Watch the videos

For this exercise, first watch the following videos from Harvard's course *CS50's Introduction to Programming with R*:

- Representing Data
- Vectors
- Data Frames
- R Console

After watching the videos, see if you can solve the following exercises to practice your skills in representing data with R.

1. Check Your Understanding

Answer the following questions based on the videos:

1. What is an Integrated Development Environment (IDE), and why is RStudio specifically beneficial when working with R compared to more general IDEs like Visual Studio Code?
2. What are the benefits of using factors in R, and how do factors differ from vectors or data frames? Provide an example where using a factor might be more useful than a vector.

3. What is the significance of the working directory in RStudio, and how can you change it using R code? Why is it important to set the correct working directory when working with data files?

4. Describe vector arithmetic in R. How does R handle operations when you add two vectors element-wise? Provide an example of vector addition.

5. R provides several special values like `NA`, `NaN`, `Inf`, and `NULL` to handle specific cases in data processing. Explain the differences between these special values, and give examples of when each would appear in data analysis. How do these values impact functions like `sum()` or `mean()`, and what strategies can be employed to handle them effectively in computations?

2. Practice Problems

Vectors

You are tasked with analyzing a set of weight data for five individuals, both before and after a diet. The goal is to calculate summary statistics and understand the differences in weight before and after the diet.

a) Create the Initial Weight Vector

- Define a numeric vector called `weight` containing the following values: 91, 75.5, 61, 88.5, 120, representing the weights (in kg) of five individuals before starting their diet.

Hint: Use the `c()` function to create the vector.

b) Determine the Storage Mode

- Use an appropriate function to determine the storage mode of the `weight` vector and verify it is numeric.

c) Summarize the Initial Weights

- Calculate the following summary statistics for the `weight` vector:
  - Mean
  - Range
  - Standard Deviation
  - Maximum and Minimum values

d) Define the Post-Diet Weight Vector

- Create a new vector `weight_after_diet` with the following values: 89.5, 75, 56, 96.5, 115, representing the weights of the same individuals after completing their diet.

e) Calculate Weight Loss

- Calculate the weight loss for each individual by subtracting `weight_after_diet` from `weight` and store the result in a vector called `weight_loss`.

f) Summarize the Weight Loss

- Calculate the following statistics for the `weight_loss` vector:
  - Mean weight loss
  - Median weight loss
- Briefly discuss the difference between the mean and median values for the weight loss.

Data Frames

1. Using the `data.frame` function, create a data frame called `foods` with two columns: `item` and `calories`. The `item` column should contain the values: "Apple", "Banana", "Carrot", "Donut". The `calories` column should contain the respective values: 95, 105, 25, 250. Then, using both the `$` notation and bracket notation, access and print the `calories` column.
2. Using the `foods` data frame from the previous exercise, retrieve the second row (Banana) using bracket notation. Additionally, retrieve the number of calories for the "Carrot" (third row, second column) using bracket notation, and print it.
3. Add a new column to the `foods` data frame called `type`, where the values are set as follows: "Fruit" for Apple and Banana, "Vegetable" for Carrot, and "Snack" for Donut. Then, modify the row names of the data frame to be the food items. Finally, access the row corresponding to "Donut" using its row name and print the entire row.
4. Using the `foods` data frame from the previous exercises, add a new column called `low_calorie`. This column should contain `TRUE` if the food item has fewer than 100 calories and `FALSE` otherwise. Use a boolean expression to determine the values for this column. Finally, print the updated data frame.