R for Data Science

Rutendo F. Sigauke

2024-04-18

Overview

In this worksheet we will go over essential R tools for Data Science. This document will go over the following topics:

1. We will revisit **R data structures**, and how they can be used for Data Science.
2. We will explore useful **R commands**.
3. We will write **R functions** and execute them.
4. We will review **R loops**.
5. We will talk about communicating our data with an **R notebook** (Note: This document was generated in an R Notebook.).

Resources

- For more details of using R for Data Science, see: https://r4ds.hadley.nz/
- For more on general R usage: https://intro2r.com/ and http://venus.ifca.unican.es/Rintro/index.html#
- For advanced R data structures: http://adv-r.had.co.nz/Data-structures.html
- Deeper dive into R markdown: https://bookdown.org/yihui/rmarkdown/
- R markdown and R notebooks cheat sheets: https://rstudio.github.io/cheatsheets/html/rmarkdown.html
- Tidyverse style guide: https://style.tidyverse.org/index.html

R data structures

There are various data structures in R. This includes *vectors*, *factors*, *matrices*, *arrays*, *data frames*, *lists*, and *functions*.

- *Vectors*: are one-dimensional arrays used to store collection data of the same mode.
- *Factors*: are vectors of categorical variables designed to group the components of another vector with the same size.
- *Matrices*: are two-dimensional arrays that can store collections of data of the same mode. Values in a matrix are accessed by two integer indices `mat[row,column]`.
- *Arrays*: are similar to matrices but they can be multi-dimensional.
- *Lists*: are ordered collection of objects and the elements can be of different types.
- *Data Frames*: are a generalization of matrices where different columns can store different mode data (e.g. characters, numeric).
- *Functions*: are objects created by the to automate specific operations.

More details on each of the data structures can be found here: http://venus.ifca.unican.es/Rintro/dataStruct.html

Vectors

1. Generate a vector of several numeric values.
2. Check the type of data class.
3. Perform a mathematical operation on the vector.
4. Create a sequence of integers.
5. Repeat value the value 42 eight times.
6. Create a random sequence of vectors (**CHALLENGE:** explain the `sample` function)

Factors

1. Create a vector of factors.

Matrices

1. Create a 3 rows and 4 columns matrix with values 1 to 12.
2. Create a second matrix with values of your choice.
3. Apply mathematics operations **between** the two matrices. See more examples here: https://www.geeksforgeeks.org/operations-on-matrices-in-r/

Arrays

1. Create an array with 3 dimensions.

Data frames

1. Create a data frame with three columns with `id` (character), `score` (numeric), and `grade` (character).
2. Check the structure of the data frame.
3. Change the grade column to factor and view the factor.
4. Add column with rank of grades as a factor (e.g. "first","second"...) and view the factors. (**CHALLENGE:** reorder the factor so that the order makes sense (i.e. increasing))
5. Check the class of each column in the data frame

Lists

1. Create a list with three characters.
2. Create a list with a data.frame, list, and matrix. You can select from the data structures created above.
3. View contents of the new list one item at a time.
4. Check names of the entries in the list.
5. Rename each of the names in the list.

Functions

Simple function

1. Write a simple function that takes in a name and prints "Hello World, my name is **my_name** !"

Loops in R

Loops are helpful for performing repetative tasks. There are three types of loops `for`, `while` and `repeat`. Loops are essential for programming across many languages.

See https://intro2r.com/loops.html and https://www.geeksforgeeks.org/loops-in-r-for-while-repeat/ for more details and examples.

`for` loops

i. Write code to display numbers from 1 to 10 using `for` loop in R.

ii. Write a `for` loop to iterate across a list. (**CHALLENGE:** explain the `seq_along` function)

iii. Loop across a 3x3 matrix and print each cell value.

`while` loop

i. Write code to display numbers from 1 to 10 using a `while` loop.

`repeat` loop

i. Write code to display numbers from 1 to 10 using a `repeat` loop.

Alternatives to loops

There are other options other than loops that fall in the `apply` family (`apply()`, `lapply()`, `tapply()`, `sapply()`, `vapply()`, and `mapply()`). See this chapter in *An Introduction to R* https://intro2r.com/loops.html#if-not-loops-then-what for more details.

Complex functions with loops

2. Create a function `hello_friend` that says "Hello **example_name**!".
3. Create a vector with 4 names and loop through to run the function `hello_friend`.
4. Write and run a function for the clustering algorithm (from Apr 8th). (**HINT:** We can use a `for` loop across all the k values.)
5. Clean up the code for clustering, and run the updated code!
6. Write functions to save analysis files (table with clusters and figures as png).

HINT 1: To save tables we can use `write.table()`, and you can specify output formats (e.g. csv or tsv).
HINT 2: The `ggplot2` package has a function to save the figures called `ggsave()`.

Miscellaneous

1. The package `cowplot` allows for easy plotting along with `ggplot2`. Examples : https://cran.r-project.org/web/packages/cowplot/vignettes/introduction.html and https://wilkelab.org/cowplot/articles/index.html
2. We can create interactive plots with `plotly` and `ggfortify`. The `plotly` website https://plotly.com/r/basic-charts/ has a few examples.
3. `shiny` offers more versatile interactive applications for communicating data in R. The `shiny` website https://www.rstudio.com/products/shiny/ has a number of example apps and details on hosting an app locally through RStudio.