Practical 2

DATAFRAMES IN R AND DATA IMPORTING

Objective:

- Understand how to create and manipulate data frames in R.
- Learn how to import data from various sources (e.g., CSV, Excel).
- Apply basic data analysis techniques.

Dataframes in R continuation:

Data frames

Similar to a vector, data frame is data structure that is used for storing data in R. It is a list of vectors which are of equal lengths but can be of different data types.

The vectors are presented as columns, each with a name and represent variables. The rows represent observations and can be named or unnamed.

While they can be constructed from scratch, data frames are typically produced from importing data sets e.g. csv or Excel format files.

```
y1 = c(1:5)
y2 =c("NEHA", "DWIJESH","SNEHA","AB","AK")
y3 = c(12,32,43,54,12)
data.frame(y1,y2,y3)
data.frame(airquality)
```

##

Categorical variables in R are stored into a factor.

factor(x = character(), levels, labels = levels, ordered = is.ordered(x))

R does not use the terms *nominal*, *ordinal*, and *interval/ratio* for types of variables.

In R, nominal variables can be coded as variables with *factor* or *character* classes.

Continuous variable/Interval/ratio data can be coded as variables
with *numeric* or *integer* classes. An *L* used with values to tell R to store the data as an integer class.

We can code ordinal data as either numeric or factor variables, depending on how we will be summarizing, plotting, and analyzing it.

sex <- factor(c("male", "female", "female", "male"))

levels(sex)

nlevels(sex)

Nominal Categorical Variable

Create a color vector

color_vector <- c('blue', 'red', 'green', 'white', 'black', 'yellow')

Convert the vector to factor

factor_color <- factor(color_vector)

factor_color

Ordinal Categorical Variable

Create Ordinal categorical vector

day_vector <- c('evening', 'morning', 'afternoon', 'midday', 'midnight', 'evening')

Convert `day_vector` to a factor with ordered level

factor_day <- factor(day_vector, order = TRUE, levels =c('morning', 'midday', 'afternoon', 'evening', 'midnight'))

Print the new variable

factor_day

Levels: morning < midday < afternoon < evening < midnight

Append the line to above code

Count the number of occurence of each level

summary(factor_day)

##Continuous Data

dataset <- mtcars

class(dataset$mpg)

PACKAGES IN R.

- dplyr- for manipulating data frames
- tidyr-for cleaning up information
- Stringr-for working with strings or text information
- Lubridate-for manipulating date information
- Httr- for working with website data
- Ggvis-(stands for grammar of graphics) this is for interactive visualizations
- Ggplot2-for creating graphics or data visualizations in R
- Shiny- for interactive applications which u can install on website
- Rio-(stands for R input and output) its for importing and exporting data
- Rmarkdown-allows to create interactive notebooks or documents for sharing information

One package to load them all

- Packman (stands for package manager)
- Command for installation of packages – install.packages("nameof the package")
- Eg.:run install.packages("pacman")
- This will make it available in the hardware but loading means actually making it accessible for use.
- Command for loading is library("nameof the package")
- Eg.: library("pacman")

Once pacman is installed, other packages can be installed using pacman.

- Or use "pacman::p_load"
- P_load function from pacman without actually loading pacman.
- Command: pacman::p_load (pacman, dplyr,Ggally, ggplot2, ggthemes, ggvis, httr,lubridate, plotly, rio, rmarkdown, shiny, stringr, tidyr)
- So once you have pacman, this command will help you to install ,make it available and load all these packages.

Base packages

- Packages which come it with r natively like the datasets packages
- You still have to load and unload packages manually.
- Use the command- library(datasets)

Clear packages

- With pacman, specific packages can be unloaded separately
- Command- p_unload(dplyr, tidyr)
- Or unload all using the command- p_unload(all) , this command unloads all the third party packages.
- For base packages use command- detach("package:datasets",unload=TRUE)
- To clear console, command- cat("\014")

Importing data from excel

Place the students.csv file in your working directory.

Syntax:

df <- read.table("<FileName>.txt", header = TRUE)

dat <- read.csv(

file = "data.csv",

header = TRUE,

sep = ",",

dec = "."

)

Example:

- data_csv <- read.csv("students.csv") print(data_csv)

OR

Install the readxl package using install.packages("readxl").

Use the following code to import the file:

library(readxl)

data_excel <- read_excel("scores.xlsx")

print(data_excel)

Exercises

1. Create a csv file for the following data

Source Cu

1 Site1 19.700

2 Site2 10.643

3 Site1 33.792

4 Site2 5.353

5 Site2 19.890

6 Site2 26.966

2. Create a csv file for the following data

City ProductA ProductB ProductC

1 Seattle 23 11 12

2 London 89 6 56

3 Tokyo 24 7 13

4 Berlin 36 34 44

5 Mumbai 3 78 14

3. The dataset swiss contains a standardized fertility measure and various socioeconomic indicators for each of 47 French-speaking provinces of Switzerland in about 1888. Import this dataset.
4. Write a R program to create an ordered factor from data of minimum 20 elements consisting of the names of months.
5. Create a Simple Data Frame:
   Construct a data frame containing the following information:

- Columns: Product, Price, Quantity
- Add details for 4 different products.

   Rename Columns in a Data Frame:
   Create a data frame with columns A, B, and C, and then rename them to Subject, Marks, and Grade.

6. Combine Two Data Frames:
   Create two data frames:

- Data Frame 1: Name, Department
- Data Frame 2: Name, Marks
  Combine them into a single data frame using merge() based on the Name column.

7. Check Data Frame Properties:
   Write a script to create a data frame and then display the number of rows, columns, and the structure (str()) of the data frame.
8. Create a Data Frame from Vectors:
   Using the following vectors:

Copy code
Names <- c("John", "Sara", "Ali", "Nina")
Ages <- c(20, 21, 22, 23)

Construct a data frame and print it.

9. Read and Display Data from CSV:
   Import the students.csv file, and write a script to display the first 5 rows using head() and the last 5 rows using tail().
10. Read Excel File and Select Columns:
    Import the scores.xlsx file. Extract and display only the Name and Marks columns.