8 More on data frame

(AST230) R for Data Science

Md Rasel Biswas

Data frame

- A two-dimensional array with two or more atomic vectors of the *same length* is known as a *data frame*
- Most useful storage structure for data analysis
- Columns are variables, and rows are observations
- R's equivalent to spreadsheet
- R function `data.frame()` is used to create a new data frame, where atomic vectors can be used as inputs

Data frame

```r
# Creating two atomic vectors
cage <- c(11, 9, 8, 10, 5)
cgender <- c("boy", "boy", "girl", "boy", "girl")
# Creating data frame
df <- data.frame(age = cage, gender = cgender)
# Print df
df
```

```
  age gender
1  11    boy
2   9    boy
3   8   girl
4  10    boy
5   5   girl
```

Some useful functions

```
# Variable names of the data frame
names(df)
```

```
[1] "age"    "gender"
```

```
# Dimension of the data frame
dim(df)
```

```
[1] 5 2
```

```
# Details of a df
str(df)
```

```
'data.frame':   5 obs. of  2 variables:
 $ age   : num  11 9 8 10 5
 $ gender: chr  "boy" "boy" "girl" "boy" ...
```

```
is.data.frame(df)
```

```
[1] TRUE
```

Some useful functions

```
# Summary of the data frame
summary(df)
```

```
     age          gender
Min.   : 5.0   Length:5
1st Qu.: 8.0   Class :character
Median : 9.0   Mode  :character
Mean   : 8.6
3rd Qu.:10.0
Max.   :11.0
```

```
# Summary of a specific variable
summary(df$age)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    5.0     8.0     9.0     8.6    10.0    11.0
```

```
# Frequency table of a variable
table(df$gender)
```

```
boy girl
  3    2
```

Ordering data frames

We want to reorder the observations of the data `df` by the variable `age`.

Recall: `order()` is used to order an atomic vector by its value. Remember the following example?

```
age <- c(11, 9, 8, 10, 5)
sort(age)
```

```
[1]  5  8  9 10 11
```

```
order(age)
```

```
[1] 5 3 2 4 1
```

```
age[order(age)] #equivalent to sort()
```

```
[1]  5  8  9 10 11
```

```
# Original data
df
```

```
  age gender
1  11    boy
2   9    boy
3   8   girl
4  10    boy
5   5   girl
```

```
# Ordering the data by `age`
df[order(df$age), ]
```

```
  age gender
5   5   girl
3   8   girl
2   9    boy
4  10    boy
1  11    boy
```

Handling missing data

- The `NA` (Not Applicable) character is used as a placeholder of missing observation in R
- Most of the R functions have an argument `na.rm`, which takes a logical value to exclude the missing value from the calculation

```
mean(c(1:10, NA, 14:16),
     na.rm = TRUE)
```

```
[1] 7.692308
```

- `na.omit()` is used to exclude all rows of a data frame that include a missing observation

```
xmd <- data.frame(
  x = c(NA, 11:14),
  y = c(rep("boy", 4), NA))
xmd # Data with missing values
```

```
   x    y
1 NA  boy
2 11  boy
3 12  boy
4 13  boy
5 14 <NA>
```

```
# Data after omitting missing values
na.omit(xmd)
```

```
   x   y
2 11 boy
3 12 boy
4 13 boy
```

Adding new column or rows

Adding a new variable using `$`

```
df$loc <- c("UK", "BN", "CZ", "CZ", "UK")
df
```

```
  age gender loc
1  11    boy  UK
2   9    boy  BN
3   8   girl  CZ
4  10    boy  CZ
5   5   girl  UK
```

```
# convert `gender` to a factor
df$gender_fac <- factor(df$gender)
df
```

```
  age gender loc gender_fac
1  11    boy  UK        boy
2   9    boy  BN        boy
3   8   girl  CZ       girl
4  10    boy  CZ        boy
5   5   girl  UK       girl
```

Adding new column or rows

```
# rbind for rows
df1 <- data.frame(id = 1:4, height = c(120, 150, 132, 122),
                          weight = c(44, 56, 49, 45))
df1
```

```
  id height weight
1  1    120     44
2  2    150     56
3  3    132     49
4  4    122     45
```

```
df2 <- data.frame(id = 5:6, height = c(119, 110),
                          weight = c(39, 35))
df2
```

```
  id height weight
1  5    119     39
2  6    110     35
```

```
rbind(df1, df2)
```

```
  id height weight
1  1    120     44
2  2    150     56
3  3    132     49
4  4    122     45
5  5    119     39
6  6    110     35
```

Adding new column or rows

```
# cbind for columns
df1
```

```
  id height weight
1  1    120     44
2  2    150     56
3  3    132     49
4  4    122     45
```

```
df3 <- data.frame(location = c("UK", "CZ", "CZ", "UK"))
df3
```

```
  location
1       UK
2       CZ
3       CZ
4       UK
```

```
cbind(df1, df3)
```

```
  id height weight location
1  1    120     44       UK
2  2    150     56       CZ
3  3    132     49       CZ
4  4    122     45       UK
```

Analyse a subset of data

- We have already discussed subsetting data frames

```
# Full data
df
```

```
  age gender loc gender_fac
1  11    boy  UK        boy
2   9    boy  BN        boy
3   8   girl  CZ       girl
4  10    boy  CZ        boy
5   5   girl  UK       girl
```

```
# A subset of boy's data
df_boy <- df[df$gender == "boy", ]
df_boy
```

```
  age gender loc gender_fac
1  11    boy  UK        boy
2   9    boy  BN        boy
4  10    boy  CZ        boy
```

```
# Mean age of boys
mean(df_boy$age)
```

```
[1] 10
```

Exercise 8

- Load the `mtcars` data, which is available in R
- Obtain the variable list of the data frame `mtcars`
- How many observations and variables do the `mtcars` data have?
- Check the types of the variables of `mtcars`
- Extract the first and the last variables from the `mtcars` data set
- Order the dataset in ascending order of the variable `mpg` (miles per gallon)
- Convert the variable `cyl` (number of cylinders) to factor variable
- Obtain a subset of `mtcars` for which `mpg` is less than 30 and save the dataset naming `mtcars30`
- Find the mean, median, mode, standard deviation, and IQR of `mpg` variable of the `mtcars30` dataset
- Find the frequency table of `cyl` of `mtcars` data