8 More on data frames

(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
- If you do data analysis in R, you're going to be using data frames

Data frame

```
# Creating three atomic vectors
age <- c(11, 9, 8, 10, 5)
name <- c("Raju", "Raj", "Raba", "Rahul", "Rimi")
sex <- c("boy", "boy", "girl", "boy", "girl")
loc <- c(1, 2, 1, 1, 2)

# Creating data frame
df <- data.frame(age, name, gender=sex, loc)

# Convert categorical variables to factor
df$gender <- factor(df$gender)
df$loc <- factor(df$loc, labels=c("Urban", "Rural"))

# Print df
df
```

```
  age  name gender   loc
1  11  Raju    boy Urban
2   9   Raj    boy Rural
3   8  Raba   girl Urban
4  10 Rahul    boy Urban
5   5  Rimi   girl Rural
```

Some useful functions

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

```
[1] "age"    "name"   "gender" "loc"
```

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

```
[1] 5 4
```

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

```
'data.frame':   5 obs. of  4 variables:
 $ age   : num  11 9 8 10 5
 $ name  : chr  "Raju" "Raj" "Raba" "Rahul" ...
 $ gender: Factor w/ 2 levels "boy","girl": 1 1 2 1 2
 $ loc   : Factor w/ 2 levels "Urban","Rural": 1 2 1 1 2
```

Some useful functions

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

```
     age           name              gender      loc    
Min.   : 5.0   Length:5           boy :3    Urban:3  
1st Qu.: 8.0   Class :character   girl:2    Rural:2  
Median : 9.0   Mode  :character                      
Mean   : 8.6                                         
3rd Qu.:10.0                                         
Max.   :11.0                                         
```

```r
# 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 
```

```r
# 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?

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

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

```r
order(age)
```

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

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

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

```r
# Original data
df
```

```
  age  name gender   loc
1  11  Raju    boy Urban
2   9   Raj    boy Rural
3   8  Raba   girl Urban
4  10 Rahul    boy Urban
5   5  Rimi   girl Rural
```

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

```
  age  name gender   loc
5   5  Rimi   girl Rural
3   8  Raba   girl Urban
2   9   Raj    boy Rural
4  10 Rahul    boy Urban
1  11  Raju    boy Urban
```

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

```r
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

```r
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>
```

```r
# 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$place <- c("UK", "BN", "PK", "IN", "BN")
df
```

```
  age  name gender   loc place
1  11  Raju    boy Urban    UK
2   9   Raj    boy Rural    BN
3   8  Raba   girl Urban    PK
4  10 Rahul    boy Urban    IN
5   5  Rimi   girl Rural    BN
```

```
# removing loc
df$loc <- NULL
df
```

```
  age  name gender place
1  11  Raju    boy    UK
2   9   Raj    boy    BN
3   8  Raba   girl    PK
4  10 Rahul    boy    IN
5   5  Rimi   girl    BN
```

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

```r
# Full data
df
```

```
  age  name gender place
1  11  Raju    boy    UK
2   9   Raj    boy    BN
3   8  Raba   girl    PK
4  10 Rahul    boy    IN
5   5  Rimi   girl    BN
```

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

```
  age  name gender place
1  11  Raju    boy    UK
2   9   Raj    boy    BN
4  10 Rahul    boy    IN
```

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

```
[1] 10
```

Exercise 8.1

The data `mtcars` comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles. Load the data by running `data(mtcars)`

- 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`
- Rename the variable `hp` to `horsepower`
- Order the dataset in ascending order of the variable `mpg` (miles per gallon)
- Convert the variable `cyl` (number of cylinders) to factor type variable
- Create a subset of the `mtcars` dataset where `mpg` is less than 30, retaining only the first five variables. Save the resulting dataset as `mtcars_subset`.

Frequency table

A **frequency table** (known as frequency distribution) is a tabular format of summarizing data where frequency corresponding each data point is presented

Frequency table of ungrouped data is often not so useful.

```
table(mtcars$mpg)
```

```
10.4 13.3 14.3 14.7   15 15.2 15.5 15.8 16.4 17.3 17.8 18.1 18.7 19.2 19.7   21
   2    1    1    1    1    2    1    1    1    1    1    1    1    2    1    2
21.4 21.5 22.8 24.4   26 27.3 30.4 32.4 33.9
   2    1    2    1    1    1    2    1    1
```

Therefore, dividing the values into groups or **class intervals** is useful.

```
mtcars$mpg_cat <- cut(mtcars$mpg, breaks = c(10, 20, 30, 40),
                      labels= c("low", "med", "high"), right = T)
table(mtcars$mpg_cat)
```

```
 low  med high
  18   10    4
```

Frequency table

We can extend this further by producing a frequency for each combination of `mpg_cat` and `cyl`

```r
table(mtcars$mpg_cat, mtcars$cyl)
```

```
        4  6  8
  low   0  4 14
  med   7  3  0
  high  4  0  0
```

Exercise 8.1 (continued)

Using the `mtcars` data:

- Find the mean, median, mode, range, standard deviation, and IQR of the variable Miles/(US) gallon (`mpg`)
- Find the frequency table of Number of cylinders (`cyl`)
- Find the 2-way contingency table of Number of cylinders (`cyl`) and Number of forward gears (`gear`)