Introduction to R

Session 3 – Data Wrangling (with dplyr and tidyr)

Statistical Consulting Centre

consulting@stat.auckland.ac.nz
The Department of Statistics
The University of Auckland

19 July, 2017

factor

What is a factor?

A variable which takes either qualitative values, ordinal values or a discrete set of quantitative values. The values of a factor are called its levels*.*

Examples of factors:

- `Gender` with 2 *qualitative* levels: `Male` and `Female`.
- `Education` with 6 *ordinal* levels: `None` < `"Primary compl` < `Incpl secondary` < `Secondary compl` < `Incpl university` < `University degree`.
- `Income` has 9 *quantitative* levels when the mid-values of the income ranges are used: 5000, 12500, 17500, 22500, 27500, 35000, 45000, 60000 and 85000.

factor

- R stores two *additional* pieces of information for each factor: (1) the unique set of levels and (2) an integer value, assigned by R, for each unique level.
- The integer values are assigned to factor levels so that they have an order associated with them.
- By default, the unique levels are assigned the values 1, 2,. . . , according to ascending alphabetical order. This is not always appropriate!

```
typeof(Patient.df$Sex)
```

```
## [1] "character"
```

```
levels(Patient.df$Sex)
```

```
## NULL
```

```
Patient.df$bmi.group <- factor(Patient.df$bmi.group)
levels(Patient.df$bmi.group)
```

Which other variables should also be factors?

```r
str(Patient.df)
```

```
## 'data.frame':    17030 obs. of  11 variables:
##  $ Patient.ID : int  3 4 9 10 11 19 34 44 45 48 ...
##  $ Age        : int  21 32 48 35 48 44 42 24 67 56 ...
##  $ Sex        : chr  "Male" "Female" "Female" "Male" ...
##  $ Race       : int  1 1 1 1 1 2 2 1 2 1 ...
##  $ Weight     : num  180 NA 150 204 155 ...
##  $ Height     : num  70.4 63.9 61.8 69.8 NA 70.2 62.6 64.4
##  $ Smoke      : int  NA NA 2 NA 2 1 1 1 NA 2 ...
##  $ Smoke.group: chr  NA NA "No" NA ...
##  $ Race.group : chr  "Caucasian" "Caucasian" "Caucasian" "C
##  $ bmi        : num  25.5 NA 27.6 29.4 NA ...
##  $ bmi.group  : Factor w/ 3 levels "normal","overweight",..
```

Which other variables should also be factors?

```
Patient.df$Sex <- factor(Patient.df$Sex)
Patient.df$Race.group <- factor(Patient.df$Race.group)
Patient.df$Smoke.group <- factor(Patient.df$Smoke.group)
```

Converting numbers to factors

```
test <- factor(c(0, 1, 2))
test
```

```
## [1] 0 1 2
## Levels: 0 1 2
```

- then convert back to numbers?

And convert back to numbers

- need to convert it to character, using as.character() first, then convert back to numbers, using as.numeric().

```
test
```

```
## [1] 0 1 2
## Levels: 0 1 2
```

```
as.numeric(test)
```

```
## [1] 1 2 3
```

```
as.numeric(as.character(test))
```

```
## [1] 0 1 2
```

Your turn

- Binning ages into age groups
- Sometimes we are interested in examining responses by age group.
- Now, assign each of the 17030 patients to one of three age groups: "Under 35", "36 to 60" and "Over 61".
- Convert `age.group` to a `factor` with levels in ascending order.

Your turn

Assign each of the 17030 patients to one of three age groups: “Under 35”, “36 to 60” and “Over 61”.

```
Patient.df$age.group <- ifelse(Patient.df$Age <= 35,
    "Under 35", ifelse(Patient.df$Age <= 60, "36 to 60",
        "Over 61"))
table(Patient.df$age.group)
```

```
##
## 36 to 60  Over 61 Under 35
##     5969     5476     5585
```

Your turn

Convert `age.group` to a `factor` with levels in ascending order.

```r
Patient.df$age.group <- factor(Patient.df$age.group,
    levels = c("Under 35", "36 to 60", "Over 61"))
table(Patient.df$age.group)
```

```
##
## Under 35 36 to 60  Over 61
##     5585     5969     5476
```

Relational data

- In general, it is rare in data analysis involves only a single table of data.
- Examples:
  - Patient information and blood test measurements
  - Experimental design and measurments from the high-thoughput biological instrument

Cholesterol.df

Serum Cholesterol level, mg/100ml, measured on:

- Day1
- Day5
- Day10

Reading data into R

```r
setwd("your working directory")
Patient.df <- read.csv("CholesterolNA.csv")
head(Cholesterol.df)
```

```
##   Patient.ID Day1 Day5 Day10
## 1          3  268  276   281
## 2          4  160  170   170
## 3          9  236  245   252
## 4         10  225  231   235
## 5         11  260  256   257
## 6         19  187  194   195
```

names(), dim() and str()

```r
# Names of the variables
names(Cholesterol.df)
```

```
## [1] "Patient.ID" "Day1"       "Day5"
## [4] "Day10"
```

```r
dim(Cholesterol.df)
```

```
## [1] 17030     4
```

```r
str(Cholesterol.df)
```

```
## 'data.frame':    17030 obs. of  4 variables:
##  $ Patient.ID: int  3 4 9 10 11 19 34 44 45 48 ...
##  $ Day1      : int  268 160 236 225 260 187 216 137 NA 156
##  $ Day5      : int  276 170 245 231 256 194 212 135 NA 157
##  $ Day10     : int  281 170 252 235 257 195 222 136 NA 159
```

Combining data-frame by columns (`cbind()`)

```
combined.df <- cbind(Patient.df, Cholesterol.df[, -1])
```

- Thus, the function for combining the data-frame by rows is `rbind()`.
- Make sure the dimensions are correct for two combining data-frames.
- Also, you need to make sure each row in `Patient.df` matches each row in `Cholesterol.df`.

Reading data into R (Again)

```r
setwd("your working directory")
Patient.df <- read.csv("Cholesterol.csv")
head(Cholesterol.df)
```

```
##   Patient.ID Day1 Day5 Day10
## 1          3  268  276   281
## 2          4  160  170   170
## 3          9  236  245   252
## 4         10  225  231   235
## 5         11  260  256   257
## 6         19  187  194   195
```

names(), dim() and str() (Again)

```
# Names of the variables
names(Cholesterol.df)
```

```
## [1] "Patient.ID" "Day1"       "Day5"
## [4] "Day10"
```

```
dim(Cholesterol.df)
```

```
## [1] 16062     4
```

```
str(Cholesterol.df)
```

```
## 'data.frame':    16062 obs. of  4 variables:
##  $ Patient.ID: int  3 4 9 10 11 19 34 44 48 49 ...
##  $ Day1      : int  268 160 236 225 260 187 216 137 156 179
##  $ Day5      : int  276 170 245 231 256 194 212 135 157 183
##  $ Day10     : int  281 170 252 235 257 195 222 136 159 183
```

Combining data-frame by columns (cbind())

```
combined.df <- cbind(Patient.df, Cholesterol.df)
```

```
## Error in data.frame(..., check.names = FALSE): arguments in
```

Better way

- Combining the based on the `Patient.ID` in both `Patient.df` and `Cholesterol.df`.
- First thing is to make sure the `Patient.ID` are unique in both data-frames.

Better way

- Combining the based on the Patient.ID in both Patient.df and Cholesterol.df.
- First thing is to make sure the Patient.ID are unique in both data-frames.

```
sum(table(Patient.df$Patient.ID) > 1)
```

```
## [1] 0
```

```
sum(table(Cholesterol.df$Patient.ID) > 1)
```

```
## [1] 0
```

dplyr R package

- dplyr R package provides some useful functions that correspond to the most data manipulation tasks.

```
library(dplyr)
```

- Mutating joins allow you to combine variables from multiple tables, there are four common types:
  - left_join()
  - right_join()
  - full_join()
  - inner_join()

dplyr R package

```
x <- data.frame(key = c(1,2,3), val.x = c("x1","x2","x3"))
y <- data.frame(key = c(1,2,4), val.y = c("y1","y2","y4"))
```

Data-frame x

key	val.x
1	x1
2	x2
3	x3

Data-frame y

key	val.y
1	y1
2	y2
4	y4

left_join()

```r
left_join(x, y, by = "key")
```

Data-frame x

key	val.x
1	x1
2	x2
3	x3

Data-frame y

key	val.y
1	y1
2	y2
4	y4

left join

key	val.x	val.y
1	x1	y1
2	x2	y2
3	x3	NA

right_join()

```r
right_join(x, y, by = "key")
```

Data-frame x

key	val.x
1	x1
2	x2
3	x3

Data-frame y

key	val.y
1	y1
2	y2
4	y4

right join

key	val.x	val.y
1	x1	y1
2	x2	y2
4	NA	y4

full_join()

```r
full_join(x, y, by = "key")
```

Data-frame x

key	val.x
1	x1
2	x2
3	x3

Data-frame y

key	val.y
1	y1
2	y2
4	y4

full join

key	val.x	val.y
1	x1	y1
2	x2	y2
3	x3	NA
4	NA	y4

inner_join()

```r
inner_join(x, y, by = "key")
```

Data-frame x

key	val.x
1	x1
2	x2
3	x3

Data-frame y

key	val.y
1	y1
2	y2
4	y4

inner join

key	val.x	val.y
1	x1	y1
2	x2	y2

dplyr R package

These four types of mutating join are differ in their behaviour when a match is not found.

- `left_join()`
- `right_join()`
- `full_join()`
- `inner_join()`

Which one should we use for this `Patient.df` and `Cholesterol.df` data-frames?

Combining two tables

```
combined.df <- left_join(Patient.df, Cholesterol.df)
```

- Variable names:

```
names(combined.df)
```

```
##  [1] "Patient.ID"  "Age"         "Sex"
##  [4] "Race"        "Weight"      "Height"
##  [7] "Smoke"       "Smoke.group" "Race.group"
## [10] "bmi"         "bmi.group"   "age.group"
## [13] "Day1"        "Day5"        "Day10"
```

Combining two tables

```
str(combined.df)
```

```
## 'data.frame':    17030 obs. of  15 variables:
##  $ Patient.ID : int  3 4 9 10 11 19 34 44 45 48 ...
##  $ Age        : int  21 32 48 35 48 44 42 24 67 56 ...
##  $ Sex        : Factor w/ 2 levels "Female","Male": 2 1 1 2
##  $ Race       : int  1 1 1 1 1 2 2 1 2 1 ...
##  $ Weight     : num  180 NA 150 204 155 ...
##  $ Height     : num  70.4 63.9 61.8 69.8 NA 70.2 62.6 64.4
##  $ Smoke      : int  NA NA 2 NA 2 1 1 1 NA 2 ...
##  $ Smoke.group: Factor w/ 2 levels "No","Yes": NA NA 1 NA 1
##  $ Race.group : Factor w/ 3 levels "African","Caucasian",..
##  $ bmi        : num  25.5 NA 27.6 29.4 NA ...
##  $ bmi.group  : Factor w/ 3 levels "normal","overweight",..
##  $ age.group  : Factor w/ 3 levels "Under 35","36 to 60",..
##  $ Day1       : int  268 160 236 225 260 187 216 137 NA 156
##  $ Day5       : int  276 170 245 231 256 194 212 135 NA 157
```

Categorical Variables

```
table(combined.df$age.group)
```

```
##
## Under 35 36 to 60  Over 61
##     5585      5969      5476
```

```
table(combined.df$Sex)
```

```
##
## Female   Male
##   9077   7953
```

. . .

`for` loop to get column summary statistics

```r
for (i in c("age.group", "Sex", "Smoke.group", "Race.group",
    "bmi.group")) {
    print(table(combined.df[, i]))
}
```

for loop to get column summary statistics

```
##
## Under 35 36 to 60  Over 61
##     5585     5969     5476
##
## Female   Male
##   9077   7953
##
##   No  Yes
## 4255 4371
##
##   African Caucasian     Other
##      4860     11612       553
##
##     normal overweight      obese
##       6916       5866       4185
```

Smart way: (apply) to get column summary statistics

```
apply(combined.df[, c("age.group", "Sex", "Smoke.group",
    "Race.group", "bmi.group")], 2, table)
```

```
## $age.group
##
## 36 to 60  Over 61 Under 35
##     5969     5476     5585
##
## $Sex
##
## Female   Male
##   9077   7953
##
## $Smoke.group
##
##   No  Yes
## 4255 4371
```

apply

```
apply(X, MARGIN, FUN, ...)
```

- X: A data frame.
- MARGIN: 1 indicates rows, 2 indicates columns.
- FUN: function, what do you want R to do with the rows or columns of the data frame
- ...: optional arguments to FUN.

Translation: Do something (FUN) to every row (or column) (MARGIN) of a data frame (X).

Continous Variables

```r
mean(combined.df$Age)
```

```
## [1] 48.7919
```

Your turn

Your turn

```r
apply(combined.df[, c("Age", "Height", "Weight", "bmi",
    "Day1")], 2, mean)
```

```
##     Age  Height  Weight     bmi    Day1
## 48.7919      NA      NA      NA      NA
```