Note 7.Factor & Table

Introduction to Statistical Programming

Chanmin Kim

Department of Statistics
Sungkyunkwan University

2022 Spring

Factor & Level

- Factor:
  - Vector with additional information (level).
  - Level: Unoverlapped values in a vector (nominal value).
  - `factor()`: Convert a vector into a factor.

```
> x <- c(5,10,5,8,10)
> y <- factor(x); y
[1] 5  10 5  8  10
Levels: 5 8 10

> str(y)
 Factor w/ 3 levels "5","8","10": 1 3 1 2 3
> unclass(y)
[1] 1 3 1 2 3
attr(,"levels")
[1] "5"  "8"  "10"
```

Factor & Level

- `length(`*factor*`)`: # of elements.
- `length(levels(`*factor*`))`: # of levels.
- New levels for the future possibility can be specified in `factor()`.

```
> length(y)
[1] 5
> levels(y)
[1] "5"  "8"  "10"
> length(levels(y))
[1] 3
> z <- factor(x,levels=c(5,8,10,15)); z
[1] 5  10 5  8  10
Levels: 5 8 10 15
> z[5] <- 15; z
[1] 5  10 5  8  15
Levels: 5 8 10 15
> z[5] <- 12
Warning message:
In ‘[<-.factor‘(‘*tmp*‘, 5, value = 12) :
  invalid factor level, NA generated
```

Functions for Factors

- `tapply(x,f,g)`: Apply function *g* to vector *x* by each level in factor *f*.
  - The length of vector (*x*) should be the same as the length of factor (*f*).
  - It returns a vector or matrix object.

```
> score <- c(92,84,80,67,80,75,60,93,77,84)
> dept <- c('a','b','b','c','a','c','a','b','c','a')
> tapply(score,dept,mean)
       a        b        c
79.00000 85.66667 73.00000

> d <- data.frame(gender=c('M','F','M','F','F','M'),
+                 age=c(35,27,33,28,37,29),
+                 salary=c(5200,3500,4700,3100,4600,4800))
> d$age30 <- ifelse(d$age > 30, 1, 0)
> tapply(d$salary,list(d$gender,d$age30),mean)
     0    1
F 3300 4600
M 4800 4950
```

Functions for Factors

- `split(x, f)`:
  - Split data frame or vector *x* into groups by each level in a factor *f*.
  - It returns a list of groups.

```
> d <- data.frame(gender=c('M','F','M','F','F','M'),
+                 age=c(35,27,33,28,37,29),
+                 salary=c(5200,3500,4700,3100,4600,4800))
> d$age30 <- ifelse(d$age > 30, 1, 0); d
  gender age salary age30
1      M  35   5200     1
2      F  27   3500     0
3      M  33   4700     1
4      F  28   3100     0
5      F  37   4600     1
6      M  29   4800     0
```

Functions for Factors

```
> split(d,d$gender)
$F
  gender age salary age30
2      F  27   3500     0
4      F  28   3100     0
5      F  37   4600     1

$M
  gender age salary age30
1      M  35   5200     1
3      M  33   4700     1
6      M  29   4800     0
```

FUNCTIONS FOR FACTORS

```
> split(d,list(d$gender,d$age30))
$F.0
  gender age salary age30
2      F  27   3500     0
4      F  28   3100     0

$M.0
  gender age salary age30
6      M  29   4800     0

$F.1
  gender age salary age30
5      F  37   4600     1

$M.1
  gender age salary age30
1      M  35   5200     1
3      M  33   4700     1
```

Functions for Factors

- by(*x*, *f*, *g*):
  - Apply function *g* to data frame or matrix *x* by each level in factor *f*.
  - c.f., `tapply()`: *x* should be a vector.

```
> by(iris[,1:2],iris$Species,cor)
iris$Species: setosa
             Sepal.Length Sepal.Width
Sepal.Length    1.0000000   0.7425467
Sepal.Width     0.7425467   1.0000000
------------------------------------------------------------
iris$Species: versicolor
             Sepal.Length Sepal.Width
Sepal.Length    1.0000000   0.5259107
Sepal.Width     0.5259107   1.0000000
------------------------------------------------------------
iris$Species: virginica
             Sepal.Length Sepal.Width
Sepal.Length    1.0000000   0.4572278
Sepal.Width     0.4572278   1.0000000
```

Table

- `table(x)`: Contingency (or frequency) table for factor or list of factors *x*.

```
> x <- c(5,12,15,12,5,5,15,12,15,5)
> table(x)
x
 5 12 15
 4  3  3

> x <- list(vote=c('Y','Y','N','Y','N'),age20=c(0,1,1,0,0))
> table(x)
    age20
vote 0 1
   N 1 1
   Y 2 1
```

TABLE

```
> x <- data.frame(vote=c('Y','Y','N','Y','N'),age20=c(0,1,1,0,0),
+                 party=c('D','R','D','D','R'))
> table(x)
, , party = D

    age20
vote 0 1
   N 0 1
   Y 2 0

, , party = R

    age20
vote 0 1
   N 1 0
   Y 0 1
```

Table Operations

- Table objects work like matrices or data frames.

```
> x <- list(vote=c('Y','Y','N','Y','N'),age20=c(0,1,1,0,0))
> tb <- table(x)
> class(tb)
[1] "table"
> tb[1,]
0 1
1 1

> tb/sum(tb)
    age20
vote   0   1
   N 0.2 0.2
   Y 0.4 0.2

> apply(tb,2,sum)
0 1
3 2
```

Table Operations

```
> addmargins(tb)
     age20
vote  0 1 Sum
  N   1 1   2
  Y   2 1   3
  Sum 3 2   5

> dimnames(tb)
$vote
[1] "N" "Y"

$age20
[1] 0 1

> dimnames(tb)$age20 = c('N','Y'); tb
    age20
vote N Y
   N 1 1
   Y 2 1
```

AGGREGATE() & CUT()

- aggregate(): Call tapply() once for each variable in a group.
- cut(*x*, *i*, *labels=F*): Assign elements of the vector *x* into an interval (elements) of the vector *i*.

```
> aggregate(iris[,-5],list(iris$Species),median)
     Group.1 Sepal.Length Sepal.Width Petal.Length Petal.Width
1     setosa          5.0         3.4         1.50         0.2
2 versicolor          5.9         2.8         4.35         1.3
3  virginica          6.5         3.0         5.55         2.0

> x <- c(2,5,10,15,19,3,17,11,8,6)
> y <- seq(0,20,2)
> cut(x,y,labels=F)
 [1]  1  3  5  8 10  2  9  6  4  3
```