chapter 7

Descriptive Statistics
Frequency & Contingency Tables

Instructor: Huang, Jia-Ping

Contents

- Descriptive statistics
- Tables

Descriptive statistics

- The `summary()` function

```
> myvars <- c("mpg", "hp", "wt")
> summary(mtcars[myvars])
      mpg              hp              wt
 Min.   :10.4   Min.   : 52.0   Min.   :1.51
 1st Qu.:15.4   1st Qu.: 96.5   1st Qu.:2.58
 Median :19.2   Median :123.0   Median :3.33
 Mean   :20.1   Mean   :146.7   Mean   :3.22
 3rd Qu.:22.8   3rd Qu.:180.0   3rd Qu.:3.61
 Max.   :33.9   Max.   :335.0   Max.   :5.42
```

Descriptive statistics

- The `summary()` does not provide enough information to understand a sample of data.
- Use a combination of `sapply()` and functions like `mean()`, `sd()`, `var()`, `min()`, `max()`, `median()`, `length()`, `range()`, `quantile()`, etc., to produce the statistics you need.

sapply()

- `sapply(x, FUN, options)`

- `apply()` applies a function over the margins of an array
- `sapply()` applies a function over a list or vector.

```
> mydata <- data.frame(x = rnorm(20, 2, 1),
y = rnorm(20, 3, 2))
> apply(mydata, 2, sd)
        x         y
0.9729847 1.6987539
> sapply(mydata, sd)
        x         y
0.9729847 1.6987539
```

Listing 7.2 Descriptive statistics via `sapply()`

```
> mystats <- function(x, na.omit=FALSE){
                if (na.omit)
                    x <- x[!is.na(x)]
                m <- mean(x)
                n <- length(x)
                s <- sd(x)
                skew <- sum((x-m)^3/s^3)/n
                kurt <- sum((x-m)^4/s^4)/n - 3
                return(c(n=n, mean=m, stdev=s, skew=skew, kurtosis=kurt))
              }

> myvars <- c("mpg", "hp", "wt")
> sapply(mtcars[myvars], mystats)
             mpg      hp       wt
n         32.000   32.000  32.0000
mean      20.091  146.688   3.2172
stdev      6.027   68.563   0.9785
skew       0.611    0.726   0.4231
kurtosis  -0.373   -0.136  -0.0227
```

Other useful functions

- `describe()` in the `Hmics` package
- `stat.desc()` in the `pastecs` package
- `describe()` in the `psych` package

The aggregate() function

- We can divide our data set into groups and produce descriptive statistics for each group.
- This can be done by using the aggregate() function.

```
> myvars <- c("mpg", "hp", "wt")

> aggregate(mtcars[myvars], by=list(am=mtcars$am), mean)
  am   mpg    hp    wt
1  0 17.147 160.26 3.769
2  1 24.392 126.85 2.411

> aggregate(mtcars[myvars], by=list(am=mtcars$am), sd)
  am   mpg    hp    wt
1  0 3.8330 53.908 0.7774
2  1 6.1665 84.062 0.6169
```

The by() function

- `aggregate()` only allows you to use single-value functions such as `mean()`.
- With `by()`, you can return several statistics at once.

```
by(data, INDICES, FUN)
```

```
> dstats <- function(x)sapply(x, mystats)
> myvars <- c("mpg", "hp", "wt")
> by(mtcars[myvars], mtcars$am, dstats)

  mtcars$am: 0
               mpg        hp        wt
  n         19.000    19.0000   19.000
  mean      17.147   160.2632    3.769
  stdev      3.834    53.9082    0.777
  skew       0.014    -0.0142    0.976
  kurtosis  -0.803    -1.2097    0.142
  ------------------------------------
  mtcars$am: 1
               mpg        hp        wt
  n         13.0000    13.000   13.000
  mean      24.3923   126.846    2.411
  stdev      6.1665    84.062    0.617
  skew       0.0526     1.360    0.210
  kurtosis  -1.4554     0.563   -1.174
```

Tables

- R provides several methods for creating frequency and contingency tables.

Table 7.1 Functions for creating and manipulating contingency tables

Function	Description
`table(var1, var2, ..., varN)`	Creates an *N*-way contingency table from *N* categorical variables (factors)
`xtabs(formula, data)`	Creates an *N*-way contingency table based on a formula and a matrix or data frame
`prop.table(table, margins)`	Expresses table entries as fractions of the marginal table defined by the `margins`
`margin.table(table, margins)`	Computes the sum of table entries for a marginal table defined by the `margins`
`addmargins(table, margins)`	Puts summary `margins` (sums by default) on a table
`ftable(table)`	Creates a compact, "flat" contingency table

One-way tables

- table()

```
> library(vcd)
> mytable <- with(Arthritis, table(Improved))
> mytable
Improved
  None   Some Marked
    42     14     28
```

- prop.table()

```
> prop.table(mytable)
Improved
  None   Some Marked
 0.500  0.167  0.333
```

Two-way tables

```
mytable <- table(A, B)

mytable <- xtabs(~ A + B, data=mydata)
```

```
> mytable <- xtabs(~ Treatment+Improved, data=Arthritis)
> mytable
          Improved
Treatment  None  Some  Marked
  Placebo    29     7       7
  Treated    13     7      21
```

Take row/column sums

```
> margin.table(mytable, 1)
Treatment
Placebo Treated
      3      41
```

The index 1 refers to the first variable in the table() statement.

```
> prop.table(mytable, 1)
           Improved
Treatment    None    Some    Marked
  Placebo   0.674   0.163    0.163
  Treated   0.317   0.171    0.512
```

```
> margin.table(mytable, 2)
Improved
   None    Some  Marked
     42      14      28
> prop.table(mytable, 2)
           Improved
Treatment   None   Some   Marked
  Placebo  0.690  0.500   0.250
  Treated  0.310  0.500   0.750
```

```
> prop.table(mytable)
           Improved
Treatment    None    Some   Marked
  Placebo  0.3452  0.0833  0.0833
  Treated  0.1548  0.0833  0.2500
```

Proportions of the overall sum

Add marginal sums

```
> addmargins(mytable)
                Improved
Treatment    None     Some    Marked      Sum
  Placebo     29        7         7        43
  Treated     13        7        21        41
  Sum         42       14        28        84
> addmargins(prop.table(mytable))
                Improved
Treatment    None     Some    Marked      Sum
  Placebo   0.3452   0.0833   0.0833   0.5119
  Treated   0.1548   0.0833   0.2500   0.4881
  Sum       0.5000   0.1667   0.3333   1.0000
```

Summary

- Descriptive statistics is the first step of statistical analysis.
- Creating frequency/contingency tables is a useful way of data visualization.
- Important functions: `summary()`, `sapply()`, `aggregate()`, `by()`, `table()`, `prop.table()`