Exploring data using R

Kamarul Imran Musa, Wan Nor Arifin

2017-07-06

Contents

Chapter 1

Introduction to R

This chapter introduces readers to the basics of working with data in R. We will start with installing R in your computer and getting familiar with RStudio interface. These will be followed by the basics of handling data in R.

1.1 R and RStudio

1.1.1 Installing R and RStudio

Install R base package: http://www.r-project.org/

Install RStudio: http://www.rstudio.com/

1.1.2 Getting familiar with the interface

Consists of 4 tabs:

1. Source
2. Console
3. Environment & History
4. Misc. Most important Plots, Packages & Help

1.1.3 R script

source tab

- important
- everything done here
- keep track what's going on
- not recommended to type in console

1.1.4 Working with packages

what is package/library

1.1.4.1 Installing packages

```
install.packages("package.name")
```

1.1.4.2 Loading libraries

```
library("package.name")
```

1.2 Working with Data

1.2.1 Setting working directory

general steps

- codes
- point-and-click

1.2.2 Data management

concerns reading data from data set, displaying data.

advanced, direct input in the code, esp. useful for tables.

1.2.2.1 Reading data set

Easiest is to read .csv file.

```
read.csv("file.name")
```

For SPSS file, need `foreign` package

```
library("foreign")
read.spss("file.name")
```

Can read data in table format from text file. From text file

```
read.table("file.name", header = TRUE)
```

1.2.2.2 Viewing data set

Easy, just type the name,

```
data
```

Nicer, using `View()`

```
View(data)
```

Important tasks

```
dim(data)
str(data)
names(data)
```

1.2.3 More about data management

- subsetting
- new variable
- recoding
- direct input for table -> how to get aggregate data into R => two ways

Chapter 2

Textual

In this chapter, we will go through a number of R functions for basic statistics. The focus will be on the results that are presented in form of numbers in text or tables (textual). We will mostly use the builtin functions (from R standard library). Extra packages will be introduced whenever necessary.

2.1 Basic descriptive statistics

In this part, we are going to use the functions as applied to a variable. For this purpose, we are going to use builtin datasets in R. You can view the available datasets by

```
data()
```

```
## Data sets in package ‘datasets’:

## AirPassengers                     Monthly Airline Passenger Numbers 1949-1960
## BJsales                           Sales Data with Leading Indicator
## BJsales.lead (BJsales)            Sales Data with Leading Indicator
## BOD                               Biochemical Oxygen Demand
## CO2                               Carbon Dioxide Uptake in Grass Plants
## ...
```

We can view any dataset description by appending “?” to the dataset name. For example,

```
?chickwts
```

We will start by using `chickwts` dataset that contains both numerical (`weight`) and categorical (`feed`) variables. We can view the first six observations,

```
head(chickwts)
```

```
##   weight      feed
## 1    179 horsebean
## 2    160 horsebean
## 3    136 horsebean
## 4    227 horsebean
## 5    217 horsebean
## 6    168 horsebean
```

the last six observations,

```
tail(chickwts)
```

```
##    weight   feed
## 66    352 casein
## 67    359 casein
## 68    216 casein
## 69    222 casein
## 70    283 casein
## 71    332 casein
```

and the dimension of the data (row and column).

```
dim(chickwts)
```

```
## [1] 71  2
```

Here we have 71 rows (71 subjects) and two columns (two variables).

Next, view the names of the variables,

```
names(chickwts)
```

```
## [1] "weight" "feed"
```

and view the details of the data,

```
str(chickwts)
```

```
## 'data.frame':    71 obs. of  2 variables:
##  $ weight: num  179 160 136 227 217 168 108 124 143 140 ...
##  $ feed  : Factor w/ 6 levels "casein","horsebean",..: 2 2 2 2 2 2 2 2 2 2 ...
```

which shows that `weight` is a numerical variable and `feed` is a factor, i.e. a categorical variable. `feed` consists of six categories or levels.

We can view the levels in `feed`,

```
levels(chickwts$feed)
```

```
## [1] "casein"    "horsebean" "linseed"   "meatmeal"  "soybean"   "sunflower"
```

2.1.1 Describing a numerical variable

A numberical variable is described by a number of descriptive statistics below.

To judge the central tendency of the `weight` variable, we obtain its mean,

```
mean(chickwts$weight)
```

```
## [1] 261.3099
```

and median,

```
median(chickwts$weight)
```

```
## [1] 258
```

To judge its spread and variability, we can view its minimum, maximum and range

```
min(chickwts$weight)
```

```
## [1] 108
```

```
max(chickwts$weight)
```

```
## [1] 423
```

```
range(chickwts$weight)
```

```
## [1] 108 423
```

and obtain its standard deviation (SD)

```
sd(chickwts$weight)
```

```
## [1] 78.0737
```

variance,

```
var(chickwts$weight)
```

```
## [1] 6095.503
```

quantile,

```
quantile(chickwts$weight)
```

```
##    0%   25%   50%   75%  100%
## 108.0 204.5 258.0 323.5 423.0
```

and interquartile range (IQR)

```
IQR(chickwts$weight)
```

```
## [1] 119
```

There are nine types of quantile algorithms in R (for `quantile` and `IQR`), the default being type 7. You may change this to type 6 (Minitab and SPSS),

```
quantile(chickwts$weight, type = 6)
```

```
##   0%  25%  50%  75% 100%
##  108  203  258  325  423
```

```
IQR(chickwts$weight, type = 6)
```

```
## [1] 122
```

In addition to SD and IQR, we can obtain its median absolute deviation (MAD),

```
mad(chickwts$weight)
```

```
## [1] 91.9212
```

It is actually simpler to obtain most these in a single command,

```
summary(chickwts$weight)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   108.0   204.5   258.0   261.3   323.5   423.0
```

even simpler, obtain all of the statistics using `describe` in the `psych` package

```
install.packages("psych")
```

```
library(psych)
describe(chickwts$weight)
```

```
##    vars  n   mean    sd median trimmed   mad min max range  skew kurtosis
## X1    1 71 261.31 78.07    258     261 91.92 108 423   315 -0.01    -0.97
##      se
## X1 9.27
```

2.1.2 Describing a categorical variable

A categorical variable is described by its count, proportion and percentage by categories.

We obtain the count of the `feed` variable,

```
summary(chickwts$feed)
```

```
##    casein horsebean   linseed  meatmeal   soybean sunflower
##        12        10        12        11        14        12
```

```
table(chickwts$feed)
```

```
##
##    casein horsebean   linseed  meatmeal   soybean sunflower
##        12        10        12        11        14        12
```

both `summary` and `table` give the same result.

`prop.table` gives the proportion of the result from the count.

```
prop.table(table(chickwts$feed))
```

```
##
##    casein horsebean   linseed  meatmeal   soybean sunflower
## 0.1690141 0.1408451 0.1690141 0.1549296 0.1971831 0.1690141
```

the result can be easily turned into percentage,

```
prop.table(table(chickwts$feed))*100
```

```
##
##    casein horsebean   linseed  meatmeal   soybean sunflower
##  16.90141  14.08451  16.90141  15.49296  19.71831  16.90141
```

To view the count and the percentage together, we can use `cbind`,

```
cbind(n = table(chickwts$feed), "%" = prop.table(table(chickwts$feed))*100)
```

```
##            n        %
## casein    12 16.90141
## horsebean 10 14.08451
## linseed   12 16.90141
## meatmeal  11 15.49296
## soybean   14 19.71831
## sunflower 12 16.90141
```

We need the quotation marks " " around the percentage sign %, because % also serves as a mathematical operator in R.

2.2 More on descriptive statistics

Just now, we viewed all the statistics as applied to a variable. In this part, we are going to view the statistics on a number of variables. This includes viewing a group of numerical variables or categorical variables, or a mixture of numerical and categorical variables. This is relevant in a sense that, most of the time, we want to view everything in one go (e.g. the statistics of all items in a questionnaire), compare the means of several groups and obtain cross-tabulation of categorical variables.

2.2.1 Describing numerical variables

Let us use `women` dataset,

```
head(women)
```

```
##   height weight
## 1     58    115
## 2     59    117
## 3     60    120
## 4     61    123
## 5     62    126
## 6     63    129
```

```
names(women)
```

```
## [1] "height" "weight"
```

```
str(women)
```

```
## 'data.frame':    15 obs. of  2 variables:
##  $ height: num  58 59 60 61 62 63 64 65 66 67 ...
##  $ weight: num  115 117 120 123 126 129 132 135 139 142 ...
```

which consists of `weight` and `height` numerical variables.

The variables can be easily viewed together by `summary`,

```
summary(women)
```

```
##      height          weight     
##  Min.   :58.0   Min.   :115.0  
##  1st Qu.:61.5   1st Qu.:124.5  
##  Median :65.0   Median :135.0  
##  Mean   :65.0   Mean   :136.7  
##  3rd Qu.:68.5   3rd Qu.:148.0  
##  Max.   :72.0   Max.   :164.0  
```

even better using `describe` (`psych`),

```
describe(women)
```

```
##        vars  n   mean    sd median trimmed   mad min max range skew
## height    1 15  65.00  4.47     65   65.00  5.93  58  72    14 0.00
## weight    2 15 136.73 15.50    135  136.31 17.79 115 164    49 0.23
##        kurtosis   se
## height    -1.44 1.15
## weight    -1.34 4.00
```

2.2.2 Describing categorical variables

Let us use `infert` dataset,

```
head(infert)
```

```
##   education age parity induced case spontaneous stratum pooled.stratum
## 1    0-5yrs  26      6       1    1           2       1              3
## 2    0-5yrs  42      1       1    1           0       2              1
## 3    0-5yrs  39      6       2    1           0       3              4
## 4    0-5yrs  34      4       2    1           0       4              2
## 5   6-11yrs  35      3       1    1           1       5             32
## 6   6-11yrs  36      4       2    1           1       6             36
```

```
names(infert)
```

```
## [1] "education"      "age"            "parity"         "induced"
## [5] "case"           "spontaneous"    "stratum"        "pooled.stratum"
```

```
str(infert)
```

```
## 'data.frame':    248 obs. of  8 variables:
##  $ education     : Factor w/ 3 levels "0-5yrs","6-11yrs",..: 1 1 1 1 2 2 2 2 2 2 ...
##  $ age           : num  26 42 39 34 35 36 23 32 21 28 ...
##  $ parity        : num  6 1 6 4 3 4 1 2 1 2 ...
##  $ induced       : num  1 1 2 2 1 2 0 0 0 0 ...
##  $ case          : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ spontaneous   : num  2 0 0 0 1 1 0 0 1 0 ...
##  $ stratum       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ pooled.stratum: num  3 1 4 2 32 36 6 22 5 19 ...
```

We notice that `induced`, `case` and `spontaneous` are not yet set as categorical variables, thus we need to `factor` the variables. We view the value labels in the dataset description,

```
?infert
```

We label the values in the variables according to the description as

```
infert$induced = factor(infert$induced, levels = 0:2, labels = c("0", "1", "2 or more"))
infert$case = factor(infert$case, levels = 0:1, labels = c("control", "case"))
infert$spontaneous = factor(infert$spontaneous, levels = 0:2, labels = c("0", "1", "2 or more"))
str(infert)
```

```
## 'data.frame':    248 obs. of  8 variables:
##  $ education     : Factor w/ 3 levels "0-5yrs","6-11yrs",..: 1 1 1 1 2 2 2 2 2 2 ...
##  $ age           : num  26 42 39 34 35 36 23 32 21 28 ...
##  $ parity        : num  6 1 6 4 3 4 1 2 1 2 ...
##  $ induced       : Factor w/ 3 levels "0","1","2 or more": 2 2 3 3 2 3 1 1 1 1 ...
##  $ case          : Factor w/ 2 levels "control","case": 2 2 2 2 2 2 2 2 2 2 ...
##  $ spontaneous   : Factor w/ 3 levels "0","1","2 or more": 3 1 1 1 2 2 1 1 2 1 ...
##  $ stratum       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ pooled.stratum: num  3 1 4 2 32 36 6 22 5 19 ...
```

and we now all these variables are turned into factors.

Again, the variables can be easily viewed together by `summary`,

```
summary(infert[c("education", "induced", "case", "spontaneous")])
```

```
##    education        induced          case        spontaneous
##  0-5yrs : 12   0        :143   control:165   0        :141  
##  6-11yrs:120   1        : 68   case   : 83   1        : 71  
##  12+ yrs:116   2 or more: 37                 2 or more: 36  
```

We do not use `table` here in form of `table(infert[c("education", "induced", "case", "spontaneous")])` because `table` used in this form will give us 3-way cross-tabulation instead of count per categories. Cross-tabulation of categorical variables will be covered later.

To obtain the proportion and percentage results, we have to use `lapply`,

```
lapply(infert[c("education", "induced", "case", "spontaneous")],
       function(x) summary(x)/length(x))
```

```
## $education
##    0-5yrs   6-11yrs   12+ yrs 
## 0.0483871 0.4838710 0.4677419 
## 
## $induced
##         0         1 2 or more 
## 0.5766129 0.2741935 0.1491935 
## 
## $case
##   control      case 
## 0.6653226 0.3346774 
## 
## $spontaneous
##         0         1 2 or more 
## 0.5685484 0.2862903 0.1451613
```

```
lapply(infert[c("education", "induced", "case", "spontaneous")],
       function(x) summary(x)/length(x)*100)
```

```
## $education
##   0-5yrs  6-11yrs  12+ yrs 
##  4.83871 48.38710 46.77419 
## 
## $induced
##         0         1 2 or more 
##  57.66129  27.41935  14.91935 
## 
## $case
##  control     case 
## 66.53226 33.46774 
## 
## $spontaneous
##         0         1 2 or more 
##  56.85484  28.62903  14.51613
```

because we need `lappy` to obtain the values for each of the variables. `lappy` goes through each variable and performs this particular part,

```
function(x) summary(x)/length(x)
```

`function(x)` is needed to specify some extra operations to any basic function in R, in our case `summary(x)` divided by `length(x)`, in which the summary results (the counts) are divided by the number of subjects (`length(x)` gives us the "length" of our dataset).

Now, since we already learned about `lapply`, we may also obtain the same results by using `summary` (within lapply), table and prop.table.

```
lapply(infert[c("education", "induced", "case", "spontaneous")], summary)
```

```
## $education
##  0-5yrs 6-11yrs 12+ yrs
##      12     120     116
##
## $induced
##         0         1 2 or more
##       143        68        37
##
## $case
## control    case
##     165      83
##
## $spontaneous
##         0         1 2 or more
##       141        71        36
```

```
lapply(infert[c("education", "induced", "case", "spontaneous")], table)
```

```
## $education
##
##  0-5yrs 6-11yrs 12+ yrs
##      12     120     116
##
## $induced
##
##         0         1 2 or more
##       143        68        37
##
## $case
##
## control    case
##     165      83
##
## $spontaneous
##
##         0         1 2 or more
##       141        71        36
```

```
lapply(infert[c("education", "induced", "case", "spontaneous")],
       function(x) prop.table(table(x)))
```

```
## $education
## x
##    0-5yrs   6-11yrs   12+ yrs
## 0.0483871 0.4838710 0.4677419
##
## $induced
## x
##         0         1 2 or more
## 0.5766129 0.2741935 0.1491935
##
## $case
## x
##   control      case
## 0.6653226 0.3346774
##
## $spontaneous
## x
##         0         1 2 or more
## 0.5685484 0.2862903 0.1451613
```

```
lapply(infert[c("education", "induced", "case", "spontaneous")],
       function(x) prop.table(table(x))*100)
```

```
## $education
## x
##   0-5yrs  6-11yrs  12+ yrs
##  4.83871 48.38710 46.77419
##
## $induced
## x
##         0         1 2 or more
##  57.66129  27.41935  14.91935
##
## $case
## x
##  control     case
## 66.53226 33.46774
##
## $spontaneous
## x
##         0         1 2 or more
##  56.85484  28.62903  14.51613
```

Notice here, whenever we do not need to specify extra operations on a basic function, e.g. `summary` and `table`, all we need to write after the comma in `lapply` is the basic function without `function(x)` and `(x)`.

2.2.3 Describing the variables together

In the preceeding sections, we intentionally went through the descriptive statistics of a variable, followed by a number of variables of the same type. This will give you the basics in dealing with the variables. Most commonly, the variables are described by groups or in form cross-tabulated counts/percentages.

2.2.3.1 By groups

To obtain all the descriptive statistics by group, we can use `by` with the relevant functions. We start with numerical variables

```
by(infert[c("age", "parity")], infert$case, summary)
```

```
## infert$case: control
##       age            parity
##  Min.   :21.00   Min.   :1.000
##  1st Qu.:28.00   1st Qu.:1.000
##  Median :31.00   Median :2.000
##  Mean   :31.49   Mean   :2.085
##  3rd Qu.:35.00   3rd Qu.:3.000  
##  Max.   :44.00   Max.   :6.000  
## ---------------------------------------------------------
## infert$case: case
##       age            parity     
##  Min.   :21.00   Min.   :1.000  
##  1st Qu.:28.00   1st Qu.:1.000  
##  Median :31.00   Median :2.000  
##  Mean   :31.53   Mean   :2.108  
##  3rd Qu.:35.50   3rd Qu.:3.000  
##  Max.   :44.00   Max.   :6.000
```

```
by(infert[c("age", "parity")], infert$case, describe)
```

```
## infert$case: control
##        vars   n  mean   sd median trimmed  mad min max range skew kurtosis
## age       1 165 31.49 5.25     31   31.34 5.93  21  44    23 0.23    -0.72
## parity    2 165  2.08 1.24      2    1.88 1.48   1   6     5 1.32     1.42
##          se
## age    0.41
## parity 0.10
## ---------------------------------------------------------
## infert$case: case
##        vars  n  mean   sd median trimmed  mad min max range skew kurtosis
## age       1 83 31.53 5.28     31   31.39 5.93  21  44    23 0.21    -0.77
## parity    2 83  2.11 1.28      2    1.90 1.48   1   6     5 1.32     1.34
##          se
## age    0.58
## parity 0.14
```

We can also use `describeBy`, which is an the extension of `describe` in the `psych` package.

```
describeBy(infert[c("age", "parity")], group = infert$case)
```

```
## 
##  Descriptive statistics by group 
## group: control
##        vars   n  mean   sd median trimmed  mad min max range skew kurtosis
## age       1 165 31.49 5.25     31   31.34 5.93  21  44    23 0.23    -0.72
## parity    2 165  2.08 1.24      2    1.88 1.48   1   6     5 1.32     1.42
##          se
## age    0.41
## parity 0.10
## ---------------------------------------------------------
## group: case
##        vars  n  mean   sd median trimmed  mad min max range skew kurtosis
## age       1 83 31.53 5.28     31   31.39 5.93  21  44    23 0.21    -0.77
## parity    2 83  2.11 1.28      2    1.90 1.48   1   6     5 1.32     1.34
##          se
## age    0.58
## parity 0.14
```

which gives us an identical result.

For categorical variables, using `summary`

```
by(infert[c("education", "induced", "spontaneous")], infert$case, summary)
```

```
## infert$case: control
##    education       induced      spontaneous
##  0-5yrs : 8   0         :96   0         :113
##  6-11yrs:80   1         :45   1         : 40
##  12+ yrs:77   2 or more:24   2 or more: 12
## --------------------------------------------------------
## infert$case: case
##    education       induced      spontaneous
##  0-5yrs : 4   0         :47   0         :28
##  6-11yrs:40   1         :23   1         :31
##  12+ yrs:39   2 or more:13   2 or more:24
```

```
by(infert[c("education", "induced", "spontaneous")], infert$case,
   function(x) lapply(x, function(x) summary(x)/length(x)))
```

```
## infert$case: control
## $education
##     0-5yrs    6-11yrs    12+ yrs
## 0.04848485 0.48484848 0.46666667
##
## $induced
##         0         1 2 or more
## 0.5818182 0.2727273 0.1454545
##
## $spontaneous
##          0          1  2 or more
## 0.68484848 0.24242424 0.07272727
##
## --------------------------------------------------------
## infert$case: case
## $education
##     0-5yrs    6-11yrs    12+ yrs
## 0.04819277 0.48192771 0.46987952
##
## $induced
##         0         1 2 or more
## 0.5662651 0.2771084 0.1566265
##
## $spontaneous
##         0         1 2 or more
## 0.3373494 0.3734940 0.2891566
```

```
by(infert[c("education", "induced", "spontaneous")], infert$case,
   function(x) lapply(x, function(x) summary(x)/length(x)*100))
```

```
## infert$case: control
## $education
##    0-5yrs   6-11yrs   12+ yrs
##  4.848485 48.484848 46.666667
##
## $induced
##         0         1 2 or more
##  58.18182  27.27273  14.54545
```

```
##
## $spontaneous
##         0          1 2 or more
## 68.484848 24.242424  7.272727
##
## --------------------------------------------------------
## infert$case: case
## $education
##    0-5yrs   6-11yrs   12+ yrs
##  4.819277 48.192771 46.987952
##
## $induced
##         0          1 2 or more
##  56.62651  27.71084  15.66265
##
## $spontaneous
##         0          1 2 or more
##  33.73494  37.34940  28.91566
```

or by using `table`

```
by(infert[c("education", "induced", "spontaneous")], infert$case,
   function(x) lapply(x, table))
```

```
## infert$case: control
## $education
##
##  0-5yrs 6-11yrs 12+ yrs
##       8      80      77
##
## $induced
##
##         0          1 2 or more
##        96        45        24
##
## $spontaneous
##
##         0          1 2 or more
##       113        40        12
##
## --------------------------------------------------------
## infert$case: case
## $education
##
##  0-5yrs 6-11yrs 12+ yrs
##       4      40      39
##
## $induced
##
##         0          1 2 or more
##        47        23        13
##
## $spontaneous
##
##         0          1 2 or more
##        28        31        24
```

```r
by(infert[c("education", "induced", "spontaneous")], infert$case,
   function(x) lapply(x, function(x) prop.table(table(x))))
```

```
## infert$case: control
## $education
## x
##     0-5yrs    6-11yrs    12+ yrs
## 0.04848485 0.48484848 0.46666667
##
## $induced
## x
##         0         1 2 or more
## 0.5818182 0.2727273 0.1454545
##
## $spontaneous
## x
##          0          1  2 or more
## 0.68484848 0.24242424 0.07272727
##
## --------------------------------------------------------
## infert$case: case
## $education
## x
##     0-5yrs    6-11yrs    12+ yrs
## 0.04819277 0.48192771 0.46987952
##
## $induced
## x
##         0         1 2 or more
## 0.5662651 0.2771084 0.1566265
##
## $spontaneous
## x
##         0         1 2 or more
## 0.3373494 0.3734940 0.2891566
```

```r
by(infert[c("education", "induced", "spontaneous")], infert$case,
   function(x) lapply(x, function(x) prop.table(table(x))*100))
```

```
## infert$case: control
## $education
## x
##    0-5yrs   6-11yrs   12+ yrs
##  4.848485 48.484848 46.666667
##
## $induced
## x
##         0         1 2 or more
##  58.18182  27.27273  14.54545
##
## $spontaneous
## x
##         0         1 2 or more
## 68.484848 24.242424  7.272727
##
## --------------------------------------------------------
## infert$case: case
## $education
## x
##    0-5yrs   6-11yrs   12+ yrs
##  4.819277 48.192771 46.987952
##
## $induced
## x
##         0         1 2 or more
##  56.62651  27.71084  15.66265
##
## $spontaneous
## x
##         0         1 2 or more
##  33.73494  37.34940  28.91566
```

Please note that simply replacing `table` for `summary` as in `by(infert[c("education", "induced", "spontaneous")], infert$case, table)` will not work as intended. `education` will be nested in `induced`, which is nested in `spontaneous`, listed by `case` instead. And yes, to obtain the proportions and percentages, it gets slightly more complicated as we have to specify `function` twice in `by`.

2.2.3.2 Simple cross-tabulation

As long as the categorical variables are already `factor`ed properly, there should not be a problem to obtain the cross-tabulation tables. For example between `education` and `case`,

```
table(infert$education, infert$case)
```

```
##
##            control case
##   0-5yrs        8    4
##   6-11yrs      80   40
##   12+ yrs      77   39
```

We may also include row and column headers, just like `cbind`,

```
table(education = infert$education, case = infert$case)
```

```
##          case
## education control case
##   0-5yrs        8    4
##   6-11yrs      80   40
##   12+ yrs      77   39
```

Since we are familiar with the powerful `lappy`, we can use it to get cross-tabulation of all of the factors with `case` status,

```
lapply(infert[c("education", "induced", "spontaneous")], function(x) table(x, infert$case))
```

```
## $education
##
## x         control case
##   0-5yrs        8    4
##   6-11yrs      80   40
##   12+ yrs      77   39
##
## $induced
##
## x           control case
##   0              96   47
##   1              45   23
##   2 or more      24   13
##
## $spontaneous
##
## x           control case
##   0             113   28
##   1              40   31
##   2 or more      12   24
```

We may also view subgroup counts (nesting). Here, the cross-tabulation of `education` and `case` is nested within `induced`

```r
table(infert$education, infert$case, infert$induced)
```

```
## , ,  = 0
##
##
##           control case
##   0-5yrs        4    0
##   6-11yrs      57   21
##   12+ yrs      35   26
##
## , ,  = 1
##
##
##           control case
##   0-5yrs        0    2
##   6-11yrs      16   11
##   12+ yrs      29   10
##
## , ,  = 2 or more
##
##
##           control case
##   0-5yrs        4    2
##   6-11yrs       7    8
##   12+ yrs      13    3
```

which will look nicer if we apply `by`

```r
by(infert[c("education", "case")], infert$induced, table)
```

```
## infert$induced: 0
##          case
## education control case
##   0-5yrs        4    0
##   6-11yrs      57   21
##   12+ yrs      35   26
## --------------------------------------------------------
## infert$induced: 1
##          case
## education control case
##   0-5yrs        0    2
##   6-11yrs      16   11
##   12+ yrs      29   10
## ---------------------------------------------------------
## infert$induced: 2 or more
##          case
## education control case
##   0-5yrs        4    2
##   6-11yrs       7    8
##   12+ yrs      13    3
```

2.3 Summary

In this chapter, we learned about how to handle numerical and categorical variables and obtain the basic and relevant statistics. In the next chapter, we are going to learn about how to explore the variables in visually in form of the relevant graphs and plots.

Chapter 3

Grammar of variables

3.1 Prepare folder and data

3.2 Set the working directory

This can be done in 2 ways:

1. Using codes
2. Using point and click

To use point and click, use the down arrow button next to *More* . Then click ‘Set as working directory’

3.3 Read Data

```
library(foreign)
data_qol<-read.dta('qol.dta',convert.factors = T)
str(data_qol)
```

```
## 'data.frame':    365 obs. of  13 variables:
##  $ id       : num  308 335 94 329 350 22 171 274 332 147 ...
##  $ sex      : Factor w/ 2 levels "female","male": 1 2 1 1 1 2 1 1 2 2 ...
##  $ age      : num  55 41 50 47 67 57 60 54 60 45 ...
##  $ tahundx  : num  14 4 5 10 13 4 4 15 13 3 ...
##  $ tx       : Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
##  $ group    : Factor w/ 2 levels "\"group A\"",..: 2 2 1 2 2 1 1 1 2 1 ...
##  $ complica : Factor w/ 2 levels "no","yes": 2 1 1 2 1 2 1 1 2 1 ...
##  $ hba1c    : num  8.1 8 7.5 9.4 11.7 8.1 7.5 9.2 NA NA ...
##  $ fbs      : num  6.9 4.8 8 3.6 12.5 8.5 NA NA NA NA ...
##  $ rbs      : num  16.7 7.4 13.2 7.4 NA 7.8 9.4 7.8 NA 12.4 ...
##  $ tg_total : num  0.92 1.66 0.74 0.94 3.01 1.3 NA 1.9 NA NA ...
##  $ choleste : num  7.09 2.91 5.94 3.27 7.1 3.54 NA 5.7 NA NA ...
##  $ ADDQSCORE: num  0 -0.222 -0.333 -0.36 -0.44 ...
##  - attr(*, "datalabel")= chr ""
##  - attr(*, "time.stamp")= chr ""
##  - attr(*, "formats")= chr  "%10.0g" "%10.0g" "%10.0g" "%10.0g" ...
##  - attr(*, "types")= int  255 255 255 255 255 255 255 255 255 255 ...
##  - attr(*, "val.labels")= chr  "" "sex" "" "" ...
##  - attr(*, "var.labels")= chr  "id_no" "sex" "" "" ...
##  - attr(*, "version")= int 8
##  - attr(*, "label.table")=List of 4
##   ..$ sex     : Named int  0 1
##   .. ..- attr(*, "names")= chr  "female" "male"
##   ..$ tx      : Named int  1 2 3 4
##   .. ..- attr(*, "names")= chr  "diet only" "OHA and diet only" "insulin and diet only" "all"
##   ..$ group   : Named int  1 2
##   .. ..- attr(*, "names")= chr  "\"group A\"" "\"group B\""
##   ..$ complica: Named int  0 1
##   .. ..- attr(*, "names")= chr  "no" "yes"
```

3.4 Browse data

1. First few rows
2. Last few rows

```
head(data_qol)
```

```
##    id    sex age tahundx                    tx     group complica hba1c
## 1 308 female  55      14 insulin and diet only "group B"      yes   8.1
## 2 335   male  41       4                   all "group B"       no   8.0
## 3  94 female  50       5     OHA and diet only "group A"       no   7.5
## 4 329 female  47      10                   all "group B"      yes   9.4
## 5 350 female  67      13                   all "group B"       no  11.7
## 6  22   male  57       4     OHA and diet only "group A"      yes   8.1
##    fbs  rbs tg_total choleste  ADDQSCORE
## 1  6.9 16.7     0.92     7.09  0.0000000
## 2  4.8  7.4     1.66     2.91 -0.2222222
## 3  8.0 13.2     0.74     5.94 -0.3333333
## 4  3.6  7.4     0.94     3.27 -0.3600000
## 5 12.5   NA     3.01     7.10 -0.4400000
## 6  8.5  7.8     1.30     3.54 -0.5000000
```

```
tail(data_qol)
```

```
##      id  sex age tahundx                tx     group complica hba1c  fbs
## 360  14 male  45      10 OHA and diet only "group A"       no   9.6 12.6
## 361 170 male  57       4 OHA and diet only "group A"       no    NA   NA
## 362 214 male  48       5 OHA and diet only "group A"       no    NA   NA
## 363 174 male  45       2 OHA and diet only "group A"       no   8.5   NA
## 364 130 male  64      16 OHA and diet only "group A"       no   6.1  3.8
## 365 219 male  46       2         diet only "group A"       no   5.9   NA
##      rbs tg_total choleste ADDQSCORE
## 360   NA       NA       NA -8.833333
## 361  9.4       NA       NA -8.833333
## 362 10.7       NA       NA -9.000000
## 363  9.6       NA       NA -9.000000
## 364  7.9       NA       NA -9.000000
## 365  6.3       NA       NA -9.000000
```

3.5 Select columns

Let us create a new dataframe with only id, sex and hba1c as the variables

```
data_qol2<-subset(data_qol, select = c('sex', 'age', 'hba1c'))
str(data_qol2)
```

```
## 'data.frame':    365 obs. of  3 variables:
##  $ sex  : Factor w/ 2 levels "female","male": 1 2 1 1 1 2 1 1 2 2 ...
##  $ age  : num  55 41 50 47 67 57 60 54 60 45 ...
##  $ hba1c: num  8.1 8 7.5 9.4 11.7 8.1 7.5 9.2 NA NA ...
```

alternatively, we can use other subsetting functions

```
data_qol3<-data_qol[,c('sex','age','hba1c')]
str(data_qol3)
```

```
## 'data.frame':    365 obs. of  3 variables:
##  $ sex  : Factor w/ 2 levels "female","male": 1 2 1 1 1 2 1 1 2 2 ...
##  $ age  : num  55 41 50 47 67 57 60 54 60 45 ...
##  $ hba1c: num  8.1 8 7.5 9.4 11.7 8.1 7.5 9.2 NA NA ...
```

3.6 Select rows

```
data_qol4<-subset(data_qol, age > 30)
str(data_qol4)
```

```
## 'data.frame':    363 obs. of  13 variables:
##  $ id       : num  308 335 94 329 350 22 171 274 332 147 ...
##  $ sex      : Factor w/ 2 levels "female","male": 1 2 1 1 1 2 1 1 2 2 ...
##  $ age      : num  55 41 50 47 67 57 60 54 60 45 ...
##  $ tahundx  : num  14 4 5 10 13 4 4 15 13 3 ...
##  $ tx       : Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
##  $ group    : Factor w/ 2 levels "\"group A\"",..: 2 2 1 2 2 1 1 1 2 1 ...
##  $ complica : Factor w/ 2 levels "no","yes": 2 1 1 2 1 2 1 1 2 1 ...
##  $ hba1c    : num  8.1 8 7.5 9.4 11.7 8.1 7.5 9.2 NA NA ...
##  $ fbs      : num  6.9 4.8 8 3.6 12.5 8.5 NA NA NA NA ...
##  $ rbs      : num  16.7 7.4 13.2 7.4 NA 7.8 9.4 7.8 NA 12.4 ...
##  $ tg_total : num  0.92 1.66 0.74 0.94 3.01 1.3 NA 1.9 NA NA ...
##  $ choleste : num  7.09 2.91 5.94 3.27 7.1 3.54 NA 5.7 NA NA ...
##  $ ADDQSCORE: num  0 -0.222 -0.333 -0.36 -0.44 ...
```

```
summary(data_qol4$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   32.00   47.00   53.00   52.91   59.00   80.00
```

alternatively, we can use other subsetting functions

```
data_qol5<-data_qol[data_qol$age>30,]
str(data_qol5)
```

```
## 'data.frame':    363 obs. of  13 variables:
##  $ id       : num  308 335 94 329 350 22 171 274 332 147 ...
##  $ sex      : Factor w/ 2 levels "female","male": 1 2 1 1 1 2 1 1 2 2 ...
##  $ age      : num  55 41 50 47 67 57 60 54 60 45 ...
##  $ tahundx  : num  14 4 5 10 13 4 4 15 13 3 ...
##  $ tx       : Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
##  $ group    : Factor w/ 2 levels "\"group A\"",..: 2 2 1 2 2 1 1 1 2 1 ...
##  $ complica : Factor w/ 2 levels "no","yes": 2 1 1 2 1 2 1 1 2 1 ...
##  $ hba1c    : num  8.1 8 7.5 9.4 11.7 8.1 7.5 9.2 NA NA ...
##  $ fbs      : num  6.9 4.8 8 3.6 12.5 8.5 NA NA NA NA ...
##  $ rbs      : num  16.7 7.4 13.2 7.4 NA 7.8 9.4 7.8 NA 12.4 ...
##  $ tg_total : num  0.92 1.66 0.74 0.94 3.01 1.3 NA 1.9 NA NA ...
##  $ choleste : num  7.09 2.91 5.94 3.27 7.1 3.54 NA 5.7 NA NA ...
##  $ ADDQSCORE: num  0 -0.222 -0.333 -0.36 -0.44 ...
##  - attr(*, "datalabel")= chr ""
##  - attr(*, "time.stamp")= chr ""
##  - attr(*, "formats")= chr  "%10.0g" "%10.0g" "%10.0g" "%10.0g" ...
##  - attr(*, "types")= int  255 255 255 255 255 255 255 255 255 255 ...
##  - attr(*, "val.labels")= chr  "" "sex" "" "" ...
##  - attr(*, "var.labels")= chr  "id_no" "sex" "" "" ...
##  - attr(*, "version")= int 8
##  - attr(*, "label.table")=List of 4
##   ..$ sex     : Named int  0 1
##   .. ..- attr(*, "names")= chr  "female" "male"
##   ..$ tx      : Named int  1 2 3 4
##   .. ..- attr(*, "names")= chr  "diet only" "OHA and diet only" "insulin and diet only" "all"
##   ..$ group   : Named int  1 2
##   .. ..- attr(*, "names")= chr  "\"group A\"" "\"group B\""
##   ..$ complica: Named int  0 1
##   .. ..- attr(*, "names")= chr  "no" "yes"
```

```
summary(data_qol5$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   32.00   47.00   53.00   52.91   59.00   80.00
```

3.7 Select rows and columns together

```
data_qol6<-subset(data_qol,age>30 & sex=='male', select = c(id, sex, age, group))
str(data_qol6)
```

```
## 'data.frame':    211 obs. of  4 variables:
##  $ id   : num  335 22 332 147 247 185 331 323 314 305 ...
##  $ sex  : Factor w/ 2 levels "female","male": 2 2 2 2 2 2 2 2 2 2 ...
##  $ age  : num  41 57 60 45 59 48 58 69 65 73 ...
##  $ group: Factor w/ 2 levels "\"group A\"",..: 2 1 2 1 1 1 2 2 2 2 ...
```

```
table(data_qol6$sex)
```

```
##
## female   male
##      0    211
```

3.8 Generate a new variable

```
data_qol$age_cat<-data_qol$age
View(data_qol)
```

3.9 Categorize into new variables

3.9.1 From a numerical variable

```
data_qol$age_cat<-cut(data_qol$age_cat,
                      breaks=c(min(data_qol$age),40,60,Inf),
                      labels=c('min-39','40-59','60-above'))
min(data_qol$age)
```

```
## [1] 21
```

```
table(data_qol$age_cat)
```

```
## 
##   min-39    40-59 60-above 
##       32      259       73
```

```
str(data_qol$age_cat)
```

```
##  Factor w/ 3 levels "min-39","40-59",..: 2 2 2 2 3 2 2 2 2 2 ...
```

3.9.2 From a categorical variable

```
table(data_qol$tx)
```

```
## 
##             diet only     OHA and diet only insulin and diet only 
##                    10                   238                    26 
##                   all 
##                    91
```

```
str(data_qol$tx)
```

```
##  Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
```

Create a variable with ‘Diet only’ vs ‘Diet+Drug’. This is a little bit complicated

```
data_qol$tx2<-data_qol$tx
str(data_qol$tx2)
```

```
##  Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
```

```
str(data_qol$tx)
```

```
##  Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
```

```
table(data_qol$tx2)
```

```
## 
##             diet only     OHA and diet only insulin and diet only
##                     10                   238                    26
##                    all
##                     91
```

```
library(plyr)
data_qol$tx2<-revalue(data_qol$tx,c('diet only'='diet', 'OHA and diet only'='med',
                                    'insulin and diet only'='med', 'all'='med'))
table(data_qol$tx2)
```

```
##
## diet  med
##   10  355
```

3.10 Dealing with missing data

```
data_qol$tx3<-data_qol$tx
str(data_qol$tx3)
```

```
##  Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
```

```
str(data_qol$tx)
```

```
##  Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
```

```
table(data_qol$tx3)
```

```
##
##              diet only     OHA and diet only insulin and diet only
##                     10                   238                    26
##                    all
##                     91
```

3.10.1 Replace values with 'NA'

```
data_qol$tx3<-revalue(data_qol$tx,c('diet only'=NA))
table(data_qol$tx3)
```

```
##
##     OHA and diet only insulin and diet only                   all
##                   238                    26                    91
```

```
str(data_qol$tx3)
```

```
##  Factor w/ 3 levels "OHA and diet only",..: 2 3 1 3 3 1 1 1 3 1 ...
```

3.11 Additional package

3.12 Package 'dplyr'

'dplyr' package is a very useful package that encourage users to use proper verb when manipulating variables (columns) and observations (rows)

It has 9 useful functions 1. filter() 2. arrange() 3. select() 4. distinct() 5. mutate() and transmute() 6. summarise() 7. sample_n() and sample_frac()

Package 'dplyr' is very useful when it is combined with another function that is 'group_by'

'

3.13 Prepare folder and data

3.14 Set the working directory

This can be done in 2 ways:

1. Using codes
2. Using point and click

To use point and click, use the down arrow button next to *More* . Then click 'Set as working directory'

3.15 List the files inside the working directory

All files will be displayed when you click 'Files'.

Or you can use this code,

```
list.files()
```

```
##  [1] "01-intro.knit.md"
##  [2] "01-intro.md"
##  [3] "01-intro.Rmd"
##  [4] "01-intro.utf8.md"
##  [5] "02-text.html"
##  [6] "02-text.md"
##  [7] "02-text.pdf"
##  [8] "02-text.Rmd"
##  [9] "03-Grammar_of_Var.knit.md"
## [10] "03-Grammar_of_Var.Rmd"
## [11] "04-eda_graphs_files"
## [12] "04-eda_graphs_files (Case Conflict)"
## [13] "04-EDA_Graphs.knit.md"
## [14] "04-EDA_Graphs.md"
## [15] "04-EDA_Graphs.Rmd"
## [16] "04-EDA_Graphs.utf8.md"
## [17] "05-graphical.Rmd"
## [18] "06-report.Rmd"
## [19] "07-summary.Rmd"
## [20] "08-references.Rmd"
## [21] "_book"
## [22] "book chapters.Rproj"
## [23] "_bookdown_files"
## [24] "cholest.csv"
## [25] "cholest.dta"
## [26] "cholest.sav"
## [27] "cholest.xlsx"
## [28] "eye.csv"
## [29] "eye.dta"
## [30] "eye.sav"
## [31] "eye.xlsx"
## [32] "index.Rmd"
## [33] "_main.docx"
## [34] "_main.log"
## [35] "_main.Rmd"
## [36] "metab1.csv"
## [37] "metab1.dta"
## [38] "qol.csv"
## [39] "qol.dta"
## [40] "qol.sav"
## [41] "qol.xlsx"
## [42] "r codes_unused"
## [43] "site"
## [44] "Template_R_bookdown"
```

3.16 Reading dataset from SPSS file (.sav)

Dataset in SPSS format will end with .sav. To read SPSS data into R we use 'foreign' library.

Create a object to represent the SPSS data that we will read into R.

```
library(foreign)
dataSPSS<-read.spss('qol.sav', to.data.frame = TRUE)
```

3.17 Describing data

Let us examine the data

```
str(dataSPSS)
```

```
## 'data.frame':    365 obs. of  13 variables:
##  $ id       : num  308 335 94 329 350 22 171 274 332 147 ...
##  $ sex      : Factor w/ 2 levels "female","male": 1 2 1 1 1 2 1 1 2 2 ...
##  $ age      : num  55 41 50 47 67 57 60 54 60 45 ...
##  $ tahundx  : num  14 4 5 10 13 4 4 15 13 3 ...
##  $ tx       : Factor w/ 4 levels "diet only","OHA and diet only",..: 3 4 2 4 4 2 2 2 4 2 ...
##  $ group    : Factor w/ 2 levels "\"group A\"",..: 2 2 1 2 2 1 1 1 2 1 ...
##  $ complica : Factor w/ 2 levels "no","yes": 2 1 1 2 1 2 1 1 2 1 ...
##  $ hba1c    : num  8.1 8 7.5 9.4 11.7 8.1 7.5 9.2 NA NA ...
##  $ fbs      : num  6.9 4.8 8 3.6 12.5 8.5 NA NA NA NA ...
##  $ rbs      : num  16.7 7.4 13.2 7.4 NA 7.8 9.4 7.8 NA 12.4 ...
##  $ tg_total : num  0.92 1.66 0.74 0.94 3.01 1.3 NA 1.9 NA NA ...
##  $ choleste : num  7.09 2.91 5.94 3.27 7.1 3.54 NA 5.7 NA NA ...
##  $ ADDQSCORE: num  0 -0.222 -0.333 -0.36 -0.44 ...
##  - attr(*, "variable.labels")= Named chr  "id_no" "sex" "" "" ...
##   ..- attr(*, "names")= chr  "id" "sex" "age" "tahundx" ...
##  - attr(*, "codepage")= int 65001
```

Now, let us summarize our data

```
summary(dataSPSS)
```

```
##        id            sex           age           tahundx
##  Min.   :  1.0   female:153   Min.   :21.00   Min.   : 1.000
##  1st Qu.:126.0   male  :212   1st Qu.:47.00   1st Qu.: 4.000
##  Median :227.0                Median :53.00   Median : 7.000
##  Mean   :221.5                Mean   :52.75   Mean   : 8.795
##  3rd Qu.:325.0                3rd Qu.:59.00   3rd Qu.:12.000
##  Max.   :416.0                Max.   :80.00   Max.   :38.000
##
##                       tx            group     complica      hba1c
##  diet only            : 10   "group A":248   no :225   Min.   : 4.100
##  OHA and diet only    :238   "group B":117   yes:140   1st Qu.: 7.500
##  insulin and diet only: 26                             Median : 9.050
##  all                  : 91                             Mean   : 9.301
##                                                        3rd Qu.:10.775
##                                                        Max.   :19.900
##                                                        NA's   :111
##       fbs              rbs            tg_total        choleste
##  Min.   : 2.700   Min.   : 3.900   Min.   :0.380   Min.   : 2.020
##  1st Qu.: 5.700   1st Qu.: 7.925   1st Qu.:1.125   1st Qu.: 4.308
##  Median : 8.000   Median :11.300   Median :1.570   Median : 5.210
##  Mean   : 9.003   Mean   :12.045   Mean   :2.002   Mean   : 5.437
##  3rd Qu.:11.900   3rd Qu.:15.000   3rd Qu.:2.385   3rd Qu.: 6.423
##  Max.   :29.200   Max.   :31.500   Max.   :8.020   Max.   :13.100
##  NA's   :178      NA's   :83       NA's   :191     NA's   :181
##    ADDQSCORE
##  Min.   :-9.000
##  1st Qu.:-5.590
##  Median :-3.944
##  Mean   :-4.179
##  3rd Qu.:-2.556
##  Max.   : 0.000
##
```

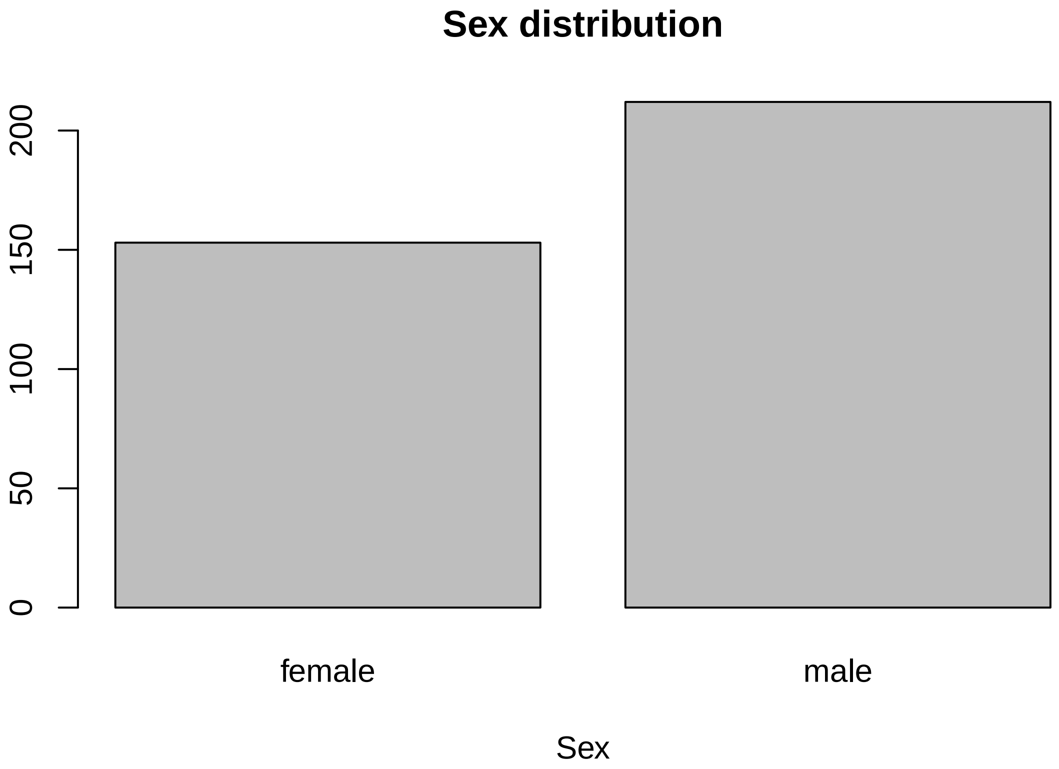
3.18 Graphing or Plotting data

You must ask yourselves these: 1. Which variable do you want to plot? 2. What is the type of that variable? Factor? Numerical? 3. Are you going to plot another variable together?

3.19 One variable: A categorical or factor variable

We can create a simple barchart

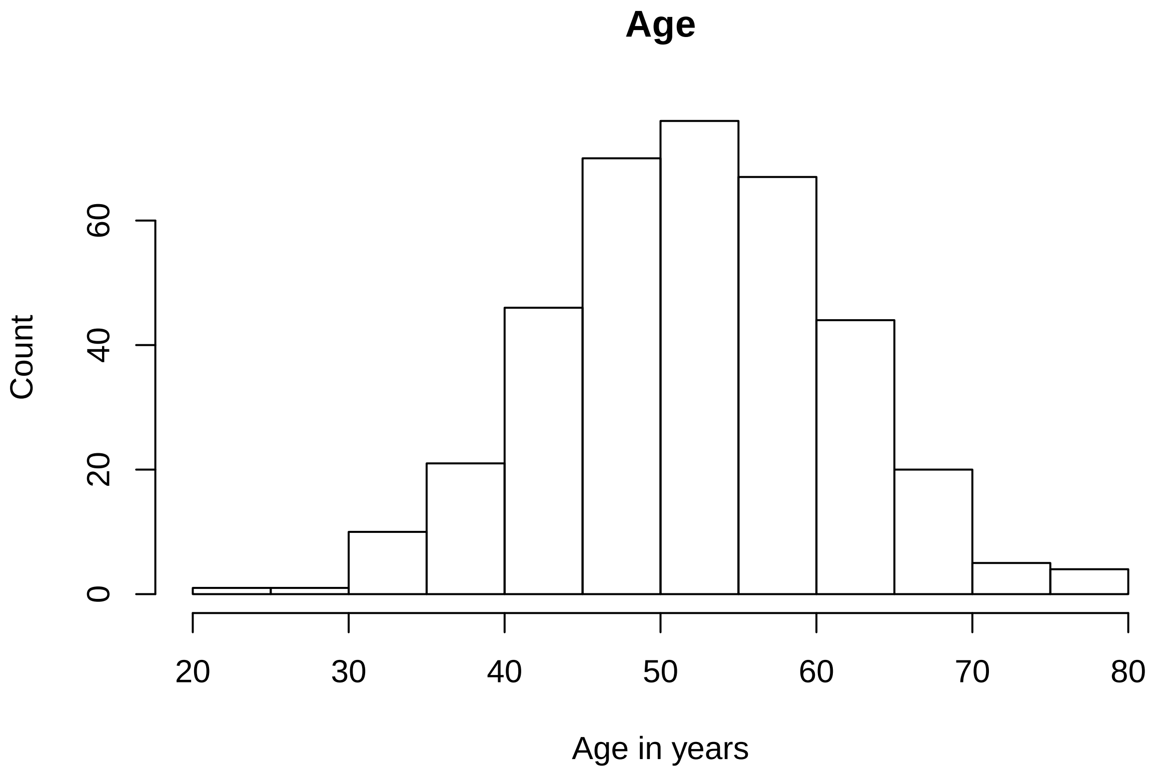
```
dist.sex<-table(dataSPSS$sex)
barplot(dist.sex,
        main='Sex distribution',
        xlab='Sex')
```



3.20 One variable: A numerical variable

histogram

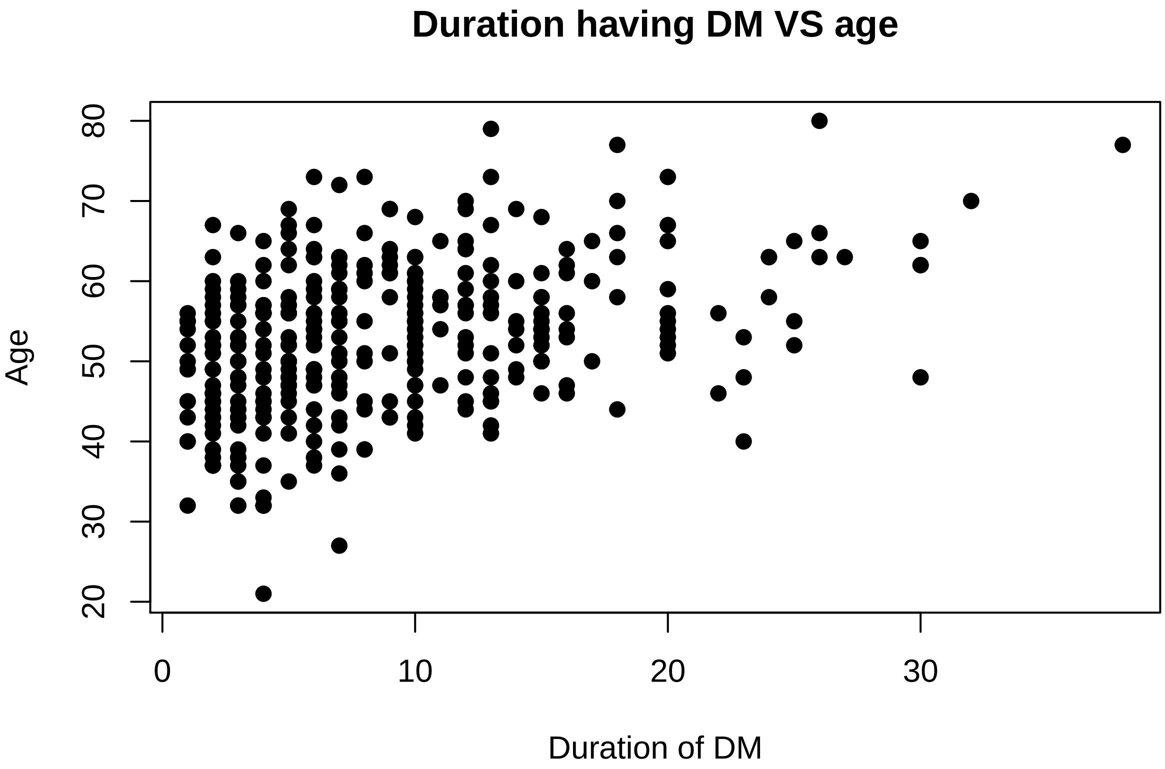
```
hist(dataSPSS$age, main = 'Age',
     xlab='Age in years',
     ylab='Count')
```



3.21 Two variables : A numerical with another numerical variable

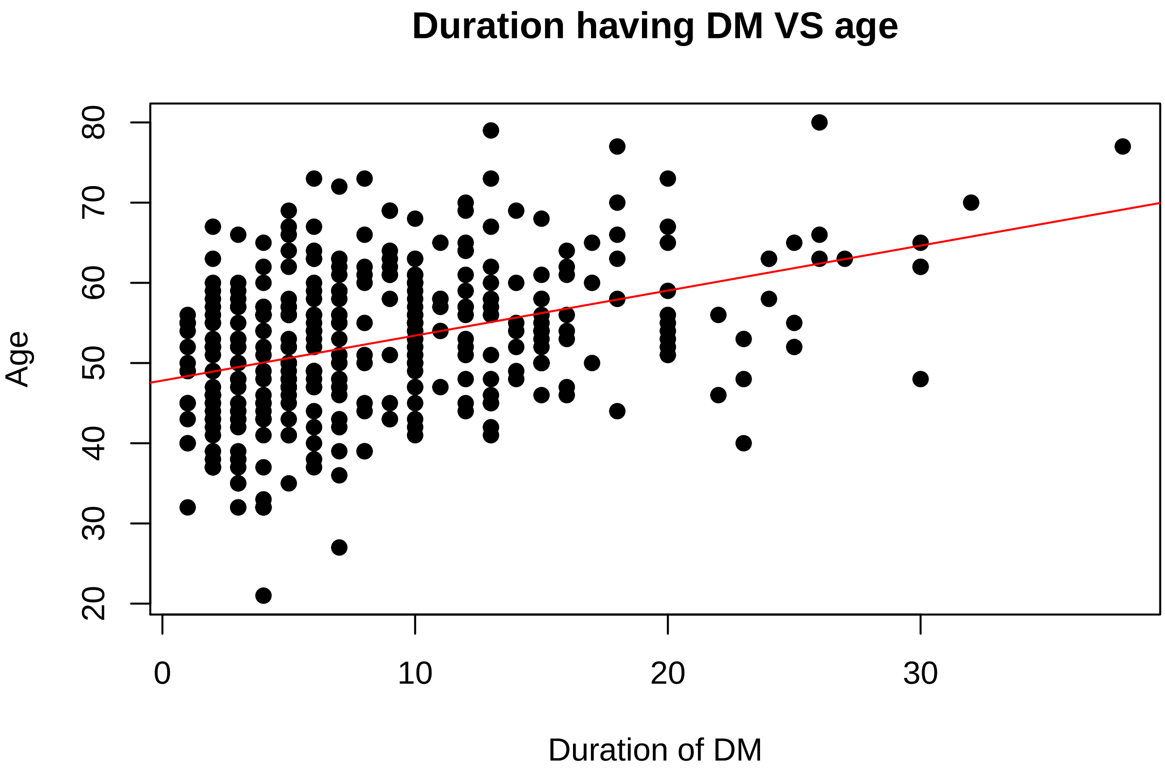
We will use *scatterplot* to plot

```
plot(dataSPSS$tahundx, dataSPSS$age,
     main = 'Duration having DM VS age',
     xlab = 'Duration of DM', ylab = 'Age',
     pch = 19)
```



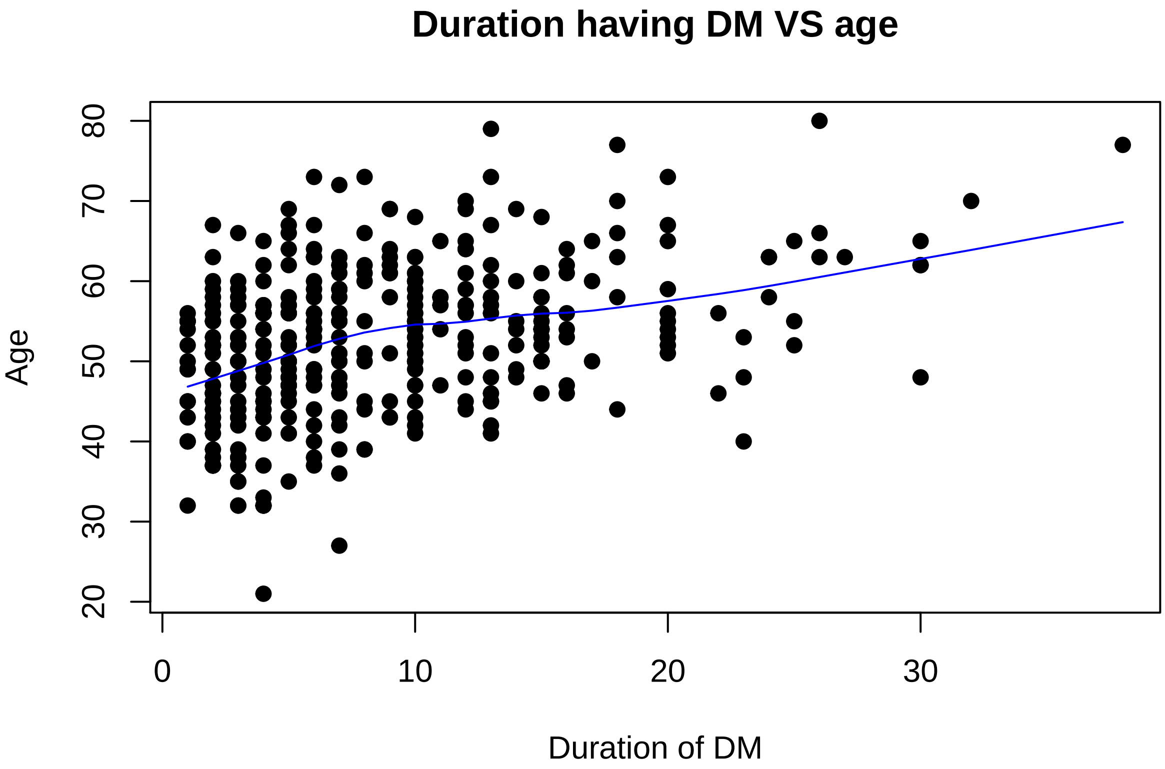
Let us make a fit line

```
plot(dataSPSS$tahundx, dataSPSS$age,
     main = 'Duration having DM VS age',
     xlab = 'Duration of DM', ylab = 'Age',
     pch = 19)
abline(lm(dataSPSS$age~dataSPSS$tahundx), col = 'red')
```



and a lowess

```r
plot(dataSPSS$tahundx, dataSPSS$age,
     main = 'Duration having DM VS age',
     xlab = 'Duration of DM', ylab = 'Age',
     pch = 19)
lines(lowess(dataSPSS$tahundx,dataSPSS$age), col = 'blue')
```



3.22 Two variables : A categorical variable with a categorical variable

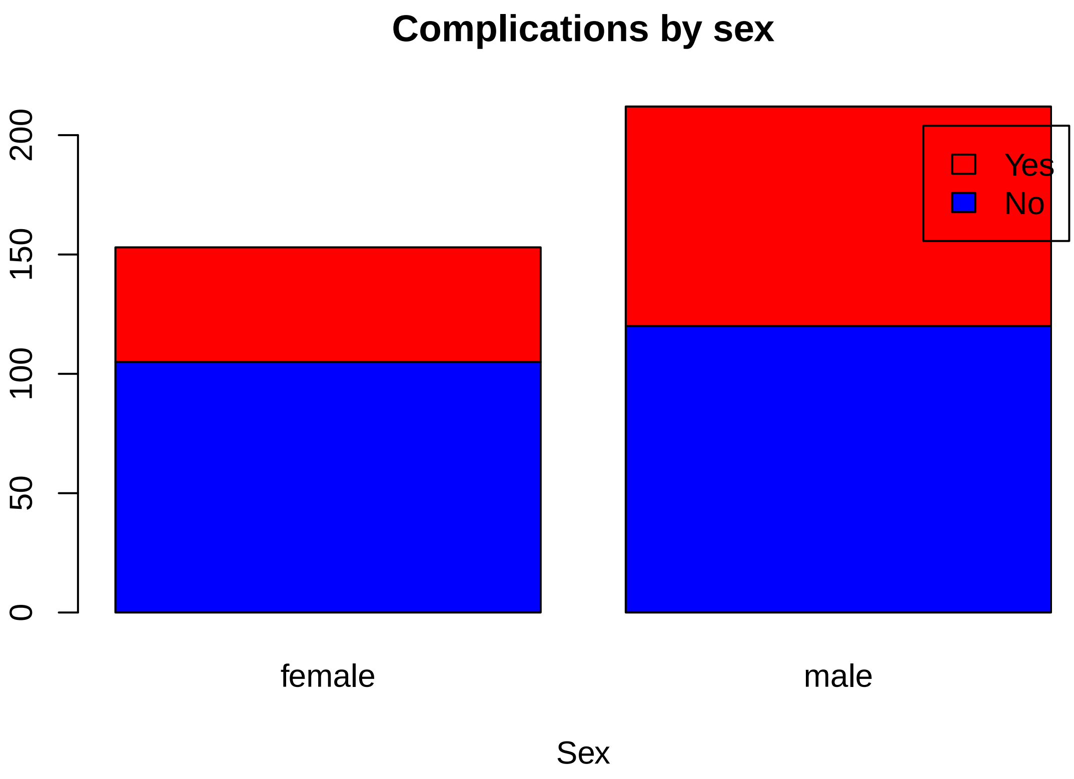
Now, we will plot 2 categorical variables simultenously.

First, we will use stacked barchart

```
compl.sex<-table(dataSPSS$complica,dataSPSS$sex)
compl.sex
```

```
##
##        female male
##   no      105  120
##   yes      48   92
```

```
barplot(compl.sex,
        main='Complications by sex',
        xlab='Sex',
        col=c('blue','red'),
        legend=c('No','Yes'))
```

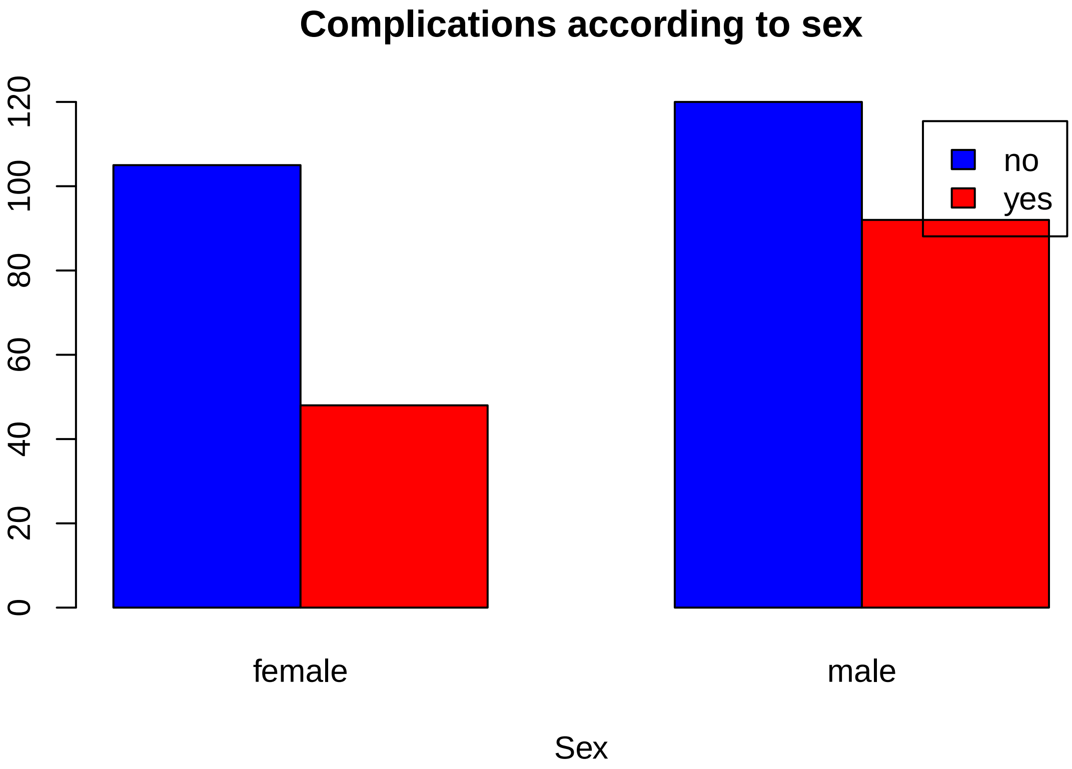


Next, we will use grouped barchart

```
compl.sex
```

```
##
##        female male
##   no      105  120
##   yes      48   92
```

```
barplot(compl.sex,
        main = 'Complications according to sex',
        xlab = 'Sex',
        col = c('blue','red'),
        legend = c('no','yes'),
        beside = TRUE)
```


Complications according to sex
0
20
40
60
80
100
120
no
yes
female
male
Sex

Chapter 4

Graphical

Test GIT Test GIT 2 - commit

Chapter 5

Reporting results

Chapter 6

Final Words

We have finished a nice book.