Exploratory Analysis with R

Alex Sanchez, Miriam Mota, Ricardo Gonzalo and Santiago Perez-Hoyos

Statistics and Bioinformatics Unit. Vall d'Hebron Institut de Recerca

Outline: Exploratory Analysis with R

- Descriptive Statistics
  - Numerical summaries
  - Graphical exploration

Based on this Course: [*BIMS 8382, University of Virginia School of Medicine (USA)*] (https://bioconnector.github.io/workshops/index.html).

What packages we will use today?

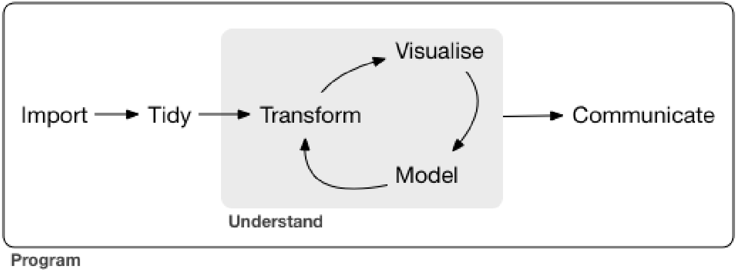
Please be sure you have the following packages installed:

- **dplyr** - subletting, sorting, transforming variables, grouping
- **ggplot2** - system for creating graphics
- **readxl** - reading .xls files

```
# install.packages("dplyr", dependencies = TRUE)
# install.packages("ggplot2", dependencies = TRUE)
# install.packages("readxl", dependencies = TRUE)

library(dplyr)
library(ggplot2)
library(readxl)
```

The Data Science Approach in R



Section 1

Getting started

Getting started (I)

1. Load the dataset *diabetes*:

```
diab <- read_excel("datasets/diabetes_mod.xls")
```

2. Check if we have loaded it correctly:

```
diab[1:4, 1:8]
```

```
## # A tibble: 4 x 8
##   numpacie mort  tempsviu  edat   bmi edatdiag tabac        sbp
##      <dbl> <chr>    <dbl> <dbl> <dbl>    <dbl> <chr>      <dbl>
## 1        1 Vivo      12.4    44  34.2       41 No fumador   132
## 2        2 Vivo      12.4    49  32.6       48 Fumador      130
## 3        3 Vivo       9.6    49  22         35 Fumador      108
## 4        4 Vivo       7.2    47  37.9       45 No fumador   128
```

Getting started (II): functions to check a dataframe:

- Content
  - head(name of dataframe): shows the first few rows - tail(): shows the last few rows
- Size
  - dim(): returns the number of rows and the number of columns - nrow(): returns the number of rows - ncol(): returns the number of columns
- Summary
  - colnames() or names(): returns the column names - glimpse(): returns a glimpse of your data: structure, class, length and content of each column

Getting started (III)

```
head(diab)
```

```
## # A tibble: 6 x 11
##   numpacie mort  tempsviu  edat   bmi edatdiag tabac      sbp   dbp ecg    chd
##      <dbl> <chr>    <dbl> <dbl> <dbl>    <dbl> <chr>    <dbl> <dbl> <chr>  <chr>
## 1        1 Vivo      12.4    44  34.2       41 No fuma~   132    96 Normal No
## 2        2 Vivo      12.4    49  32.6       48 Fumador    130    72 Normal No
## 3        3 Vivo       9.6    49  22         35 Fumador    108    58 Normal Si
## 4        4 Vivo       7.2    47  37.9       45 No fuma~   128    76 Front~ Si
## 5        5 Vivo      14.1    43  42.2       42 Fumador    142    80 Normal No
## 6        6 Vivo      14.1    47  33.1       44 No fuma~   156    94 Normal No
```

Getting started (IV)

```
dim(diab)
```

```
## [1] 149  11
```

```
nrow(diab)
```

```
## [1] 149
```

```
colnames(diab)
```

```
##  [1] "numpacie" "mort"     "tempsviu" "edat"     "bmi"      "edatdia
##  [7] "tabac"    "sbp"      "dbp"      "ecg"      "chd"
```

Getting started (IV)

```
glimpse(diab)
```

```
## Rows: 149
## Columns: 11
## $ numpacie <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18~
## $ mort     <chr> "Vivo", "Vivo", "Vivo", "Vivo", "Vivo", "Vivo", "Vivo", "Vivo~
## $ tempsviu <dbl> 12.4, 12.4, 9.6, 7.2, 14.1, 14.1, 12.4, 14.2, 12.4, 14.5, 12.~
## $ edat     <dbl> 44, 49, 49, 47, 43, 47, 50, 36, 50, 49, 50, 54, 42, 44, 40, 4~
## $ bmi      <dbl> 34.2, 32.6, 22.0, 37.9, 42.2, 33.1, 36.5, 38.5, 41.5, 34.1, 3~
## $ edatdiag <dbl> 41, 48, 35, 45, 42, 44, 48, NA, 47, 45, 48, 43, 36, 43, 26, 4~
## $ tabac    <chr> "No fumador", "Fumador", "Fumador", "No fumador", "Fumador", ~
## $ sbp      <dbl> 132, 130, 108, 128, 142, 156, 140, 144, 134, 102, 142, 128, 1~
## $ dbp      <dbl> 96, 72, 58, 76, 80, 94, 86, 88, 78, 68, 84, 74, 86, 58, 98, 6~
## $ ecg      <chr> "Normal", "Normal", "Normal", "Frontera", "Normal", "Normal",~
## $ chd      <chr> "No", "No", "Si", "Si", "No", "No", "Si", "No", "Si", "No", "~
```

Variables and data types

- Data managed in R . . .
  - is stored as *variables*
- Variables can be of distinct types
  - Numerical
    - numeric (13.7)
    - int (3)
  - Character
    - “R is cute”
  - Factors
    - A,B,C,D
    - WT, Mut
  - Logical
    - TRUE/FALSE

Exercise I

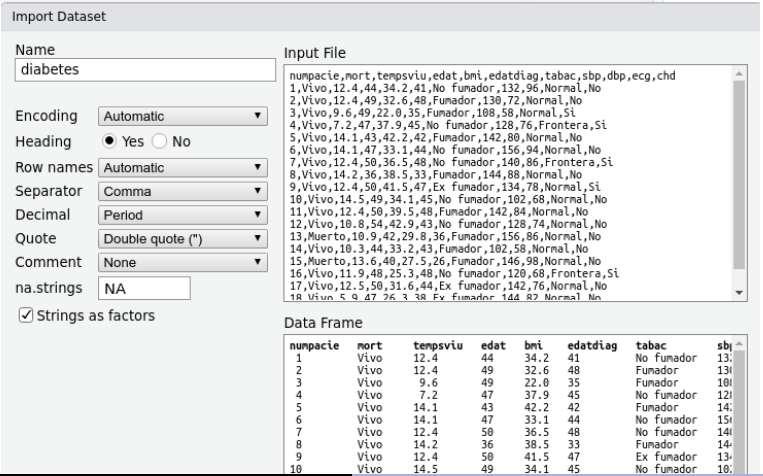
- Load the osteoporosis dataset
- Proceed similarly as to what we have done above and obtain information on
  - How many variables and observations
  - How are them

More about factors

- Each data type is what it seems to be, but factors require more explanation.
- Factors are intended to describe categories such as “sex”, “blood group”, but also “risk” or “stage”.
- Factors are useful to describe groups without having to use numeric codes.
- Factors may be created while reading the file or later using the `factor` and `as.factor` commands.

Create factor while reading

- Import the diabetes dataset from the `diabetes.csv` file using the Rstudio dialog.



Import Dataset

Name: diabetes

Encoding: Automatic ▾

Heading: ● Yes ○ No

Row names: Automatic ▾

Separator: Comma ▾

Decimal: Period ▾

Quote: Double quote (") ▾

Comment: None ▾

na.strings: NA

☑ Strings as factors

Input File

```
numpacie,mort,tempsviu,edat,bmi,edatdiag,tabac,sbp,dbp,ecg,chd
1,Vivo,12.4,44,34.2,41,No fumador,132,96,Normal,No
2,Vivo,12.4,49,32.6,48,Fumador,130,72,Normal,No
3,Vivo,9.6,49,22.0,35,Fumador,108,58,Normal,Si
4,Vivo,7.2,47,37.9,45,No fumador,128,76,Frontera,Si
5,Vivo,14.1,43,42.2,42,Fumador,142,80,Normal,No
6,Vivo,14.1,47,33.1,44,No fumador,156,94,Normal,No
7,Vivo,12.4,50,36.5,48,No fumador,140,86,Frontera,Si
8,Vivo,14.2,36,38.5,33,Fumador,144,88,Normal,No
9,Vivo,12.4,50,41.5,47,Ex fumador,134,78,Normal,Si
10,Vivo,14.5,49,34.1,45,No fumador,102,68,Normal,No
11,Vivo,12.4,50,39.5,48,Fumador,142,84,Normal,No
12,Vivo,10.8,54,42.9,43,No fumador,128,74,Normal,No
13,Muerto,10.9,42,29.8,36,Fumador,156,86,Normal,No
14,Vivo,10.3,44,33.2,43,Fumador,102,58,Normal,No
15,Muerto,13.6,40,27.5,26,Fumador,146,98,Normal,No
16,Vivo,11.9,48,25.3,48,No fumador,120,68,Frontera,Si
17,Vivo,12.5,50,31.6,44,Ex fumador,142,76,Normal,No
18,Vivo,5.9,47,26.3,38,Ex fumador,144,82,Normal,No
```

Data Frame

```
numpacie  mort    tempsviu  edat  bmi   edatdiag  tabac       sb
 1        Vivo    12.4      44    34.2  41        No fumador  13
 2        Vivo    12.4      49    32.6  48        Fumador     13
 3        Vivo     9.6      49    22.0  35        Fumador     10
 4        Vivo     7.2      47    37.9  45        No fumador  12
 5        Vivo    14.1      43    42.2  42        Fumador     14
 6        Vivo    14.1      47    33.1  44        No fumador  15
 7        Vivo    12.4      50    36.5  48        No fumador  14
 8        Vivo    14.2      36    38.5  33        Fumador     14
 9        Vivo    12.4      50    41.5  47        Ex fumador  13
10        Vivo    14.5      49    34.1  45        No fumador  10
```

Check variable type

```
diabetes <- read.csv("datasets/diabetes.csv", stringsAsFactors=TRUE)
class(diabetes$mort)
```

```
## [1] "factor"
```

```
sapply(diabetes, class)
```

```
##  numpacie      mort  tempsviu      edat       bmi  edatdiag      taba
## "integer"  "factor" "numeric" "integer" "numeric" "integer"  "factor
##       dbp       ecg       chd
## "integer"  "factor"  "factor"
```

Repeat

- Re-read the file from excel or without setting the “stringsAsFactors” to TRUE

Creating factors directly

- Use factor or as.factor

```
diabetes <- read.csv("datasets/diabetes.csv", stringsAsFactors=FALSE)
class(diabetes$mort)
```

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

```
diabetes$mort <- as.factor(diabetes$mort)
class(diabetes$mort)
```

```
## [1] "factor"
```

```
levels(diabetes$mort)
```

```
## [1] "Muerto" "Vivo"
```

Warning! by default alphabetic order is used when creating factor levels.

```
vitalStatus <- factor(diabetes$mort, levels=c("Vivo", "Muerto"))
class(vitalStatus)
```

Change the levels of a factor

- When humans fill the database… many errors can happen :(
  - An answer like “SI”, could be entered like:

    `"SI", "Si", "si", "SI ", "SÍ", .....`
- All this possible answers **will be differents levels for the same variable**
- Tis may be solved using `recode_factor`:

```
diab$mort <- recode_factor(diab$mort, "Muerto" = "muerto")
levels(diab$mort)
```

```
## [1] "muerto" "Vivo"
```

Changing *characters (chr)* to *factors (Factor)*

An alternative way to turn characters into factors is the `mutate_if` function:

```
library(dplyr)
diab <- diabetes %>% mutate_if(is.character, as.factor)
glimpse(diab)
```

```
## Rows: 149
## Columns: 11
## $ numpacie <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18~
## $ mort     <fct> Vivo, Vivo, Vivo, Vivo, Vivo, Vivo, Vivo, Vivo, Vivo, Vivo, V~
## $ tempsviu <dbl> 12.4, 12.4, 9.6, 7.2, 14.1, 14.1, 12.4, 14.2, 12.4, 14.5, 12.~
## $ edat     <int> 44, 49, 49, 47, 43, 47, 50, 36, 50, 49, 50, 54, 42, 44, 40, 4~
## $ bmi      <dbl> 34.2, 32.6, 22.0, 37.9, 42.2, 33.1, 36.5, 38.5, 41.5, 34.1, 3~
## $ edatdiag <int> 41, 48, 35, 45, 42, 44, 48, 33, 47, 45, 48, 43, 36, 43, 26, 4~
## $ tabac    <fct> No fumador, Fumador, Fumador, No fumador, Fumador, No fumador~
## $ sbp      <int> 132, 130, 108, 128, 142, 156, 140, 144, 134, 102, 142, 128, 1~
## $ dbp      <int> 96, 72, 58, 76, 80, 94, 86, 88, 78, 68, 84, 74, 86, 58, 98, 6~
## $ ecg      <fct> Normal, Normal, Normal, Frontera, Normal, Normal, Frontera, N~
## $ chd      <fct> No, No, Si, Si, No, No, Si, No, Si, No, No, No, No, No, No, S~
```

Section 2

Exploratory Data Analysis: Numerical summaries

Numerical Summaries (I)

- There are many functions to provide numerical summaries

```r
#Mean, median and rang
mean(diab$edat)
```

```
## [1] 52.16779
```

```r
median(diab$edat)
```

```
## [1] 50
```

```r
sd(diab$edat)
```

```
## [1] 11.77285
```

```r
var(diab$edat)
```

```
## [1] 138.6
```

```r
range(diab$edat)
```

```
## [1] 31 86
```

Numerical Summaries (II)

A general summary of all variables is provided by distinct functions

```
summary(diab[, 2:11])
```

```
##      mort        tempsviu          edat            bmi           edatdiag
##  Muerto: 25   Min.   : 0.00   Min.   :31.00   Min.   :18.20   Min.   :26.00
##  Vivo  :124   1st Qu.: 7.30   1st Qu.:43.00   1st Qu.:26.60   1st Qu.:38.00
##               Median :11.60   Median :50.00   Median :31.20   Median :45.00
##               Mean   :10.52   Mean   :52.17   Mean   :31.78   Mean   :45.99
##               3rd Qu.:13.90   3rd Qu.:60.00   3rd Qu.:35.20   3rd Qu.:53.00
##               Max.   :16.90   Max.   :86.00   Max.   :59.70   Max.   :81.00
##         tabac         sbp             dbp               ecg      chd
##  Ex fumador:41   Min.   : 98.0   Min.   : 58.00   Anormal : 11   No:99
##  Fumador   :51   1st Qu.:124.0   1st Qu.: 74.00   Frontera: 27   Si:50
##  No fumador:57   Median :138.0   Median : 80.00   Normal  :111
##                  Mean   :139.1   Mean   : 90.04
##                  3rd Qu.:152.0   3rd Qu.: 88.00
##                  Max.   :222.0   Max.   :862.00
```

Improving the `summary` function

- There are many packages to do descriptive statistics.
- See Dabbling with data
- Give a try, for instance to the `skimr` or `summarytools` packages.

More complete descriptions (I)

```
library(summarytools)
```

```
## Registered S3 method overwritten by 'pryr':
##   method      from
##   print.bytes Rcpp
```

```
## For best results, restart R session and update pander using devtools
```

```
dfSummary(diabetes)
```

```
## Data Frame Summary
## diabetes
## Dimensions: 149 x 11
## Duplicates: 1
##
## --------------------------------------------------------------------
## No   Variable      Stats / Values               Freqs (% of Valid)
## ---- ------------- ---------------------------- ---------------------
## 1    numpacie      Mean (sd) : 75 (43.2)        148 distinct values
##      [integer]     min < med < max:
##                    1 < 75 < 149
```

Grouped summeries

If we want to group the descriptive summaries by other variables we can use `group_by` function:

```
diab %>%
  group_by(tabac, ecg) %>%
  summarize(mean(edat))
```

```
## `summarise()` has grouped output by 'tabac'. You can override using the `.groups` argument.
```

```
## # A tibble: 9 x 3
## # Groups:   tabac [3]
##   tabac      ecg      `mean(edat)`
##   <fct>      <fct>           <dbl>
## 1 Ex fumador Anormal          68.5
## 2 Ex fumador Frontera         59.8
## 3 Ex fumador Normal           51.1
## 4 Fumador    Anormal          58
## 5 Fumador    Frontera         44.8
## 6 Fumador    Normal           44.7
## 7 No fumador Anormal          66.5
## 8 No fumador Frontera         53.8
## 9 No fumador Normal           56.0
```

Handling missing data

- What happens if we have missing data in our dataset?
- The file `diabetes_mod.xls` contains some missings

```
diabetes_mod <- read_excel("datasets/diabetes_mod.xls")
diab <- diabetes_mod %>% mutate_if(is.character, as.factor)
mean(diab$sbp)
```

```
## [1] NA
```

NA indicates *missing data* in the variable

Let's look the `sbp` variable:

```
diab$sbp
```

```
##   [1] 132 130 108 128 142 156 140 144 134 102 142 128 156 102 146 120 142 144
##  [19]  NA 134 130 122 132 150 134 142 124 102 134 118 192 122 122 112 142 152
##  [37] 112 118 152 136 134 130 108 126 132 144 126 128  NA 128 142 132 148 170
##  [55] 140 138 112 140 138 130 178 158 168 146 128 132 154 154 122 144 178 162
##  [73] 142 120 124 174 142 160 122 162 132 116 152 144  98 138 138 184 158 176
##  [91] 118 172 182 144 142 154 122 222 150 142 128 122 162 172 132 112 138 128
## [109] 132 120 140 140 172 136 152 126 104 142 128 122 122 122 122 168 162  NA
## [127] 126 180 132 150 106 154 122 120 120 144 134 148 170 160 154 124 130 156
## [145] 162 132 120 160 146
```

Numerical Summaries (VII)

How to work with *missing data*:

```
?mean
```

```
## Help on topic 'mean' was found in the following packages:
##
##   Package               Library
##   base                  /usr/lib/R/library
##   rapportools           /home/alex/R/x86_64-pc-linux-gnu-library/4.1
##
##
## Using the first match ...
```

```
mean(diab$sbp, na.rm = TRUE)
```

```
## [1] 139.2603
```

```
is.na(diab$sbp)
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
```

Numerical Summaries (VIII)

How to work with *missing data*:

```
sum(is.na(diab$sbp))
```

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

```
sum(is.na(diab$dbp))
```

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

```
diab_noNAS <-na.omit(diab)
dim(diab)
```

```
## [1] 149  11
```

```
dim(diab_noNAS)
```

```
## [1] 141  11
```

See also: Remove Rows with NA in R Data Frame

EXERCISE

1. With the `diab` dataset

- Show only the rows from 35 to 98 and columns 5, 7, and from 9 to 11
- Change the level of the variable *tabac*, from **No Fumador** to **No_Fumador**
- Display the unique values for the variable *bmi*. Count how many exist.
- Display the mean of *edatdiag*, grouped by *ecg*

Section 3

Exploratory Data Analysis (EDA): Graphical summaries

Exploratory Data Analysis (EDA): Graphical summaries

- We could dedicate one whole course to Data Visualization (at least see our “Statistical Pill on Data Visualization”)
- Here we will only see the most common approaches to visualize data:
  - Histograms
  - Barplots
  - Piecharts
  - Boxplots
  - Scatterplots

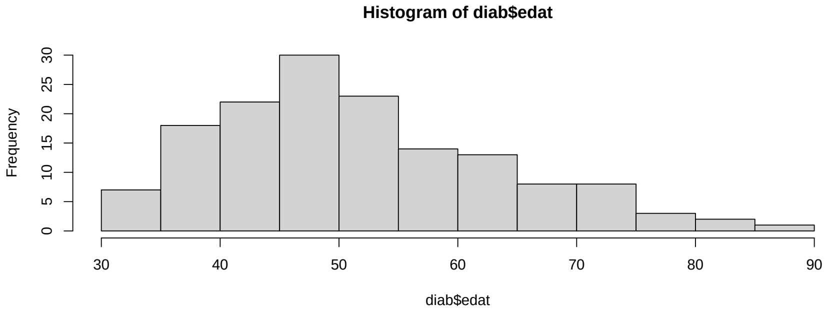
R graphics engines

- R is very powerful and flexible at doing graphics.
- This comes at a price: Complex graphics (that we do not show here) may require sonme extra effort.
- Much work has been done to simplify this
    - There exist graphical tools that allow for the interactive construction of plots.
    - There exist new approaches to plotting that try to be more intuitive than “traditional” ones.
- `ggplot` is one of such approaches.

Histograms

- We will use histograms to plot the frequencies of each range of values in continuous variables.
- These plots provide an approximation to the distribution of the variables being represented.

```
hist(diab$edat)
```

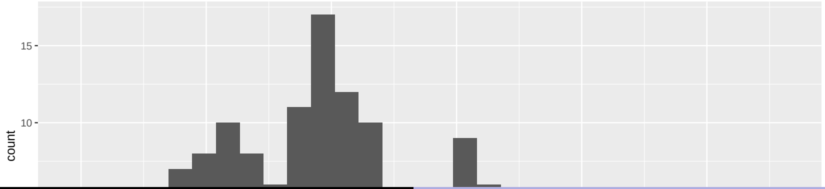


A histogram in ggplot

- Every ggplot has, at least, three components
  - The data used
  - The variables that go in each axis (the "aesthetics")
  - The type of plot
- Plots are built progressively "adding layers"

```r
library(ggplot2)
ggplot(data=diab, aes(x=edat))+
  geom_histogram()
```

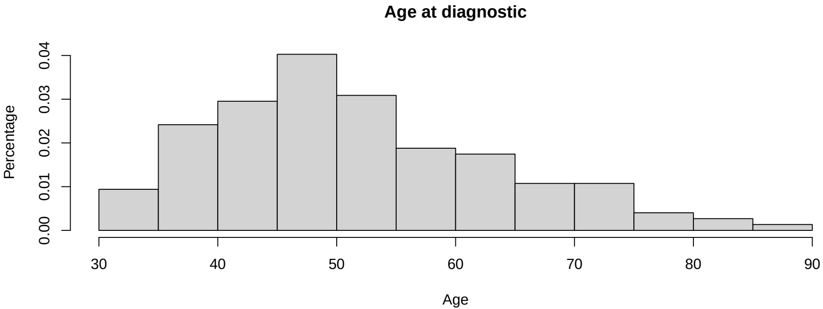
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Improving plots with graphic parameters

- Plots can be improved using *graphical parameters*
- Some parameters are the same in all graphs. Others are specific of one or other graph.

```
hist(diab$edat, main="Age at diagnostic", probability = TRUE, xlab="Age
```

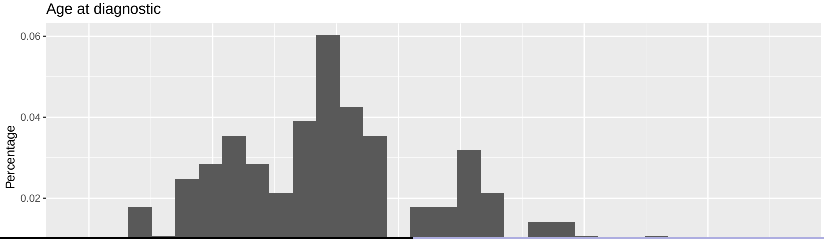


Improving plots in ggplots

- Plots can be improved using *graphical parameters*
- Some parameters are the same in all graphs. Others are specific of one or other graph.

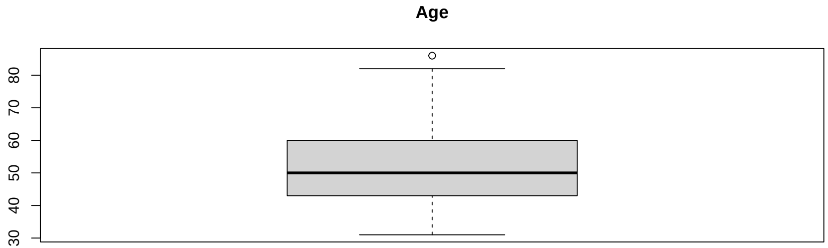
```
ggplot(data=diab, aes(x=edat))+
  geom_histogram(aes(y=..density..))+
  ggtitle("Age at diagnostic")+
  xlab("Age") + ylab("Percentage")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



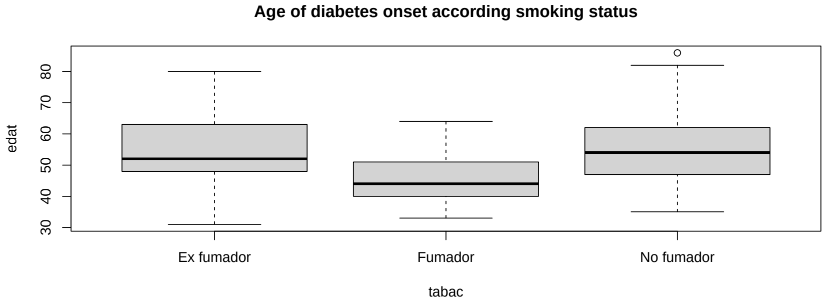
Boxplot. A one-dimensional histogram

```
boxplot (diab$edat, main ="Age")
```



Boxplot: Decomposing plots by groups

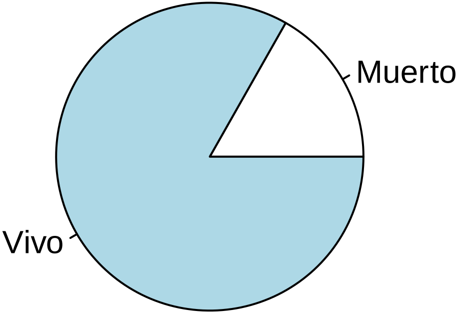
```
boxplot(edat~tabac, data=diab, main="Age of diabetes onset according sm
```



Plots for categorical variables

- Some simple principles
  - Use pie charts only with categorical variables in nominal scale
  - Use barplots for any categorical variable
  - Never use 3D-plots

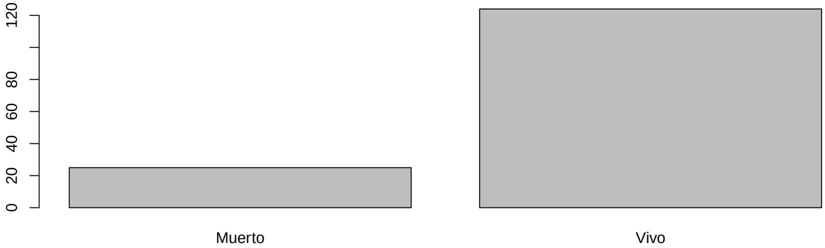
```
t <- table(diab$mort)
pie(t)
```



Barplots

- Similar to pie charts but, implicitly, suggest ordering

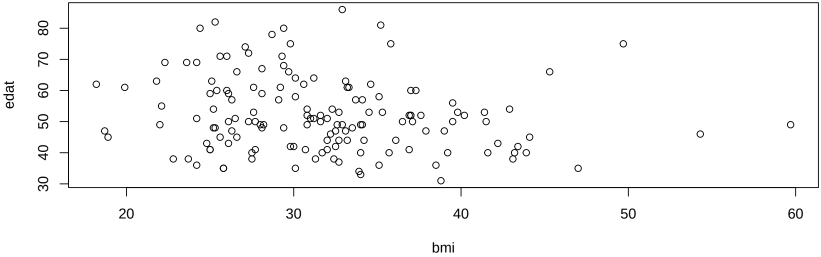
```
barplot(t)
```



Scatterplots

- For two variables, simply use `plot`

```
plot(edat~bmi, data=diab)
```



EXERCISE

- With the `diab` dataset
  - Use the best graphic type to plot the relation between *sbp* and *dbp*
  - Show graphically the relation between *edat* and *ecg*
  - Plot the *sbp* frequencies

EXERCISE

- Using the *osteoporosis.csv* dataset
- Load the dataset and check if it is correctly loaded
- Calculate the mean and standard deviation of imc grouped by clasific
- Plot the distribution of edat