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?

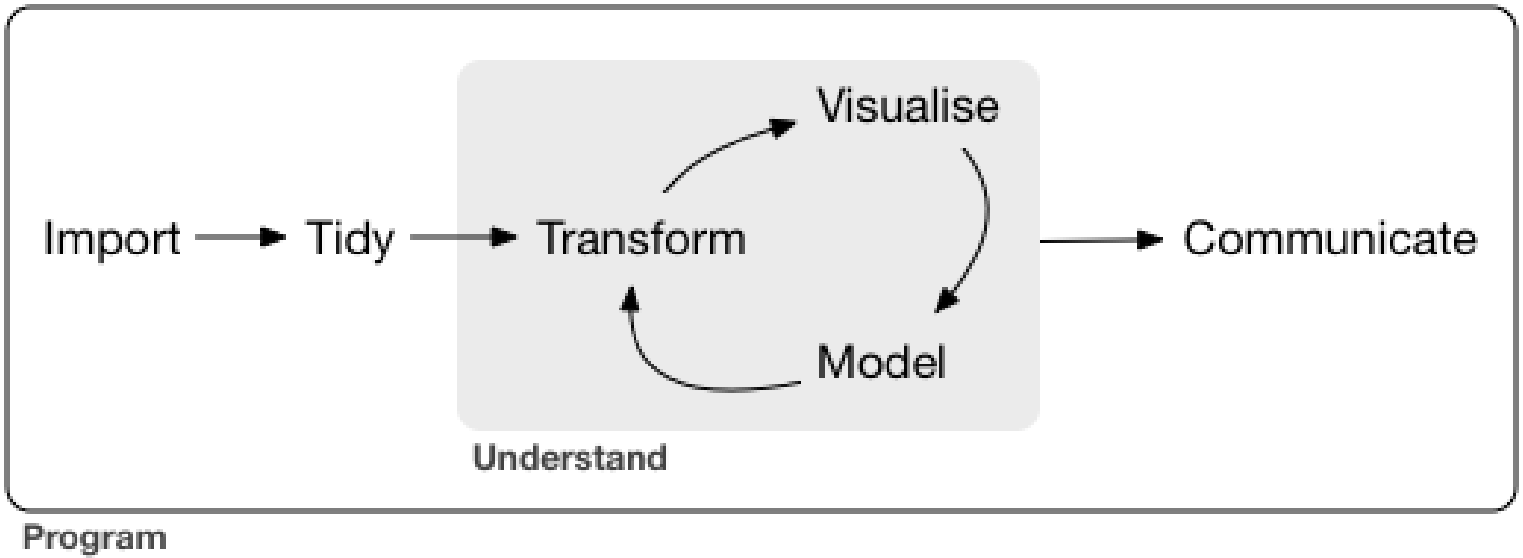
Please be sure you have the following packages installed:

- **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

Exercise I

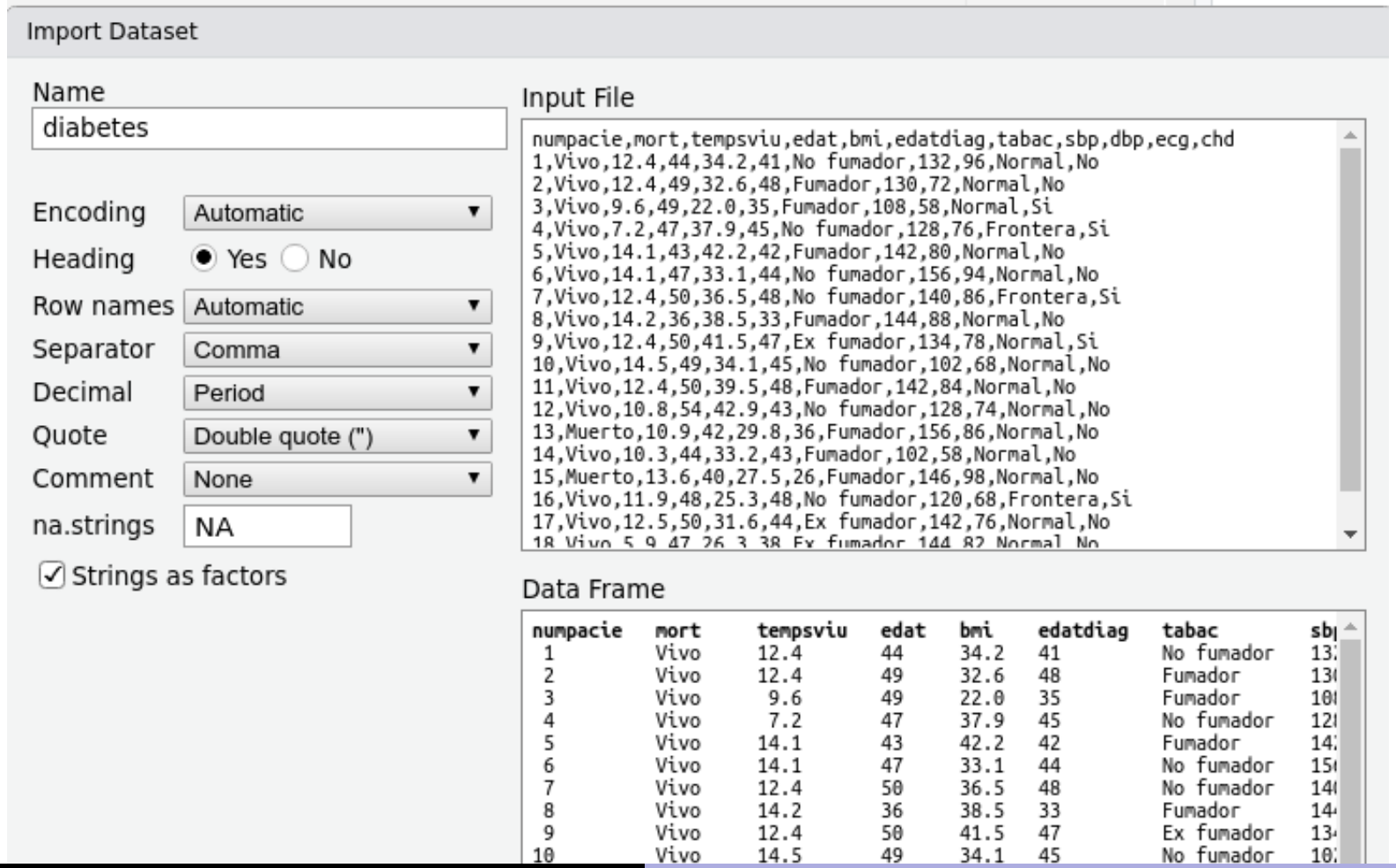
- Load the osteoporosis dataset
- Proceed similarly as to what we have done above and obtain information on
  - How many variables and observations
  - How are them

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.



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

Check the levels of a factor

Usually when humans fill the database. . . a plenty of errors could be found :(

- 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**

How to correct it?

We can use: `recode_factor`:

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

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

Return to the original version:

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

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

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

Use dplyr function `mutate_if` can do it easily:

```
diab <- diab %>% mutate_if(is.character, as.factor)

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     <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     <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    <fct> No fumador, Fumador, Fumador, No fumador, Fumador, No 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      <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

Descriptive Statistics: Numerical summaries

Numerical Summaries (I)

We can access individual variables within a data frame using the $ operator. Let's print out all the *edat* values in the data. Let's then see what are `unique` values of each. Then let's calculate the mean , `median` and `range` for the *edad* variable:

```
#display all the values
diab$edat
```

```
##   [1] 44 49 49 47 43 47 50 36 50 49 50 54 42 44 40 48 50 47 38 35 51 40 54 53 45
##  [26] 41 34 38 43 45 40 44 48 51 36 52 41 49 44 37 51 47 45 38 35 50 53 48 40 43
##  [51] 54 52 69 38 50 64 44 38 62 47 78 49 63 71 51 59 50 66 42 40 67 86 52 42 60
##  [76] 75 81 60 60 63 62 57 71 58 42 45 66 61 48 82 35 57 56 49 50 53 71 55 69 59
## [101] 47 75 80 57 52 48 57 58 51 33 52 52 64 31 69 59 38 49 49 68 40 36 60 74 61
## [126] 54 35 46 40 53 66 61 41 41 41 46 80 63 72 41 52 53 61 53 75 40 61 62 49
```

Numerical Summaries (II)

```
# Get the unique values of edat
diab$edat %>% unique()
```

```
##  [1] 44 49 47 43 50 36 54 42 40 48 38 35 51 53 45 41 34 52 37 69 64 62 78 63 71
## [26] 59 66 67 86 60 75 81 57 58 61 82 56 55 80 33 31 68 74 46 72
```

```
diab$edat %>% unique() %>% length()
```

```
## [1] 45
```

Numerical Summaries (III)

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

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

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

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

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

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

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

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

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

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

Numerical Summaries (IV)

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
```

Numerical Summaries (V)

A general summary of all variables:

```
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   :46.01
##               3rd Qu.:13.90   3rd Qu.:60.00   3rd Qu.:35.20   3rd Qu.:53.25
##               Max.   :16.90   Max.   :86.00   Max.   :59.70   Max.   :81.00
##                                                               NA's   :5
##         tabac          sbp              dbp              ecg       chd
##  Ex fumador:41   Min.   : 98.0   Min.   : 58.00   Anormal : 11   No:99
##  Fumador   :51   1st Qu.:124.5   1st Qu.: 74.00   Frontera: 27   Si:50
##  No fumador:57   Median :138.0   Median : 80.00   Normal  :111
##                  Mean   :139.3   Mean   : 90.04
##                  3rd Qu.:152.0   3rd Qu.: 88.00
##                  Max.   :222.0   Max.   :862.00
##                  NA's   :3
```

Numerical Summaries (VI)

What happens if we have missing data in our dataset?

```
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
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
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] 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
```

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

Descriptive Statistics: Graphical summaries

Exploratory Data Analysis (EDA)

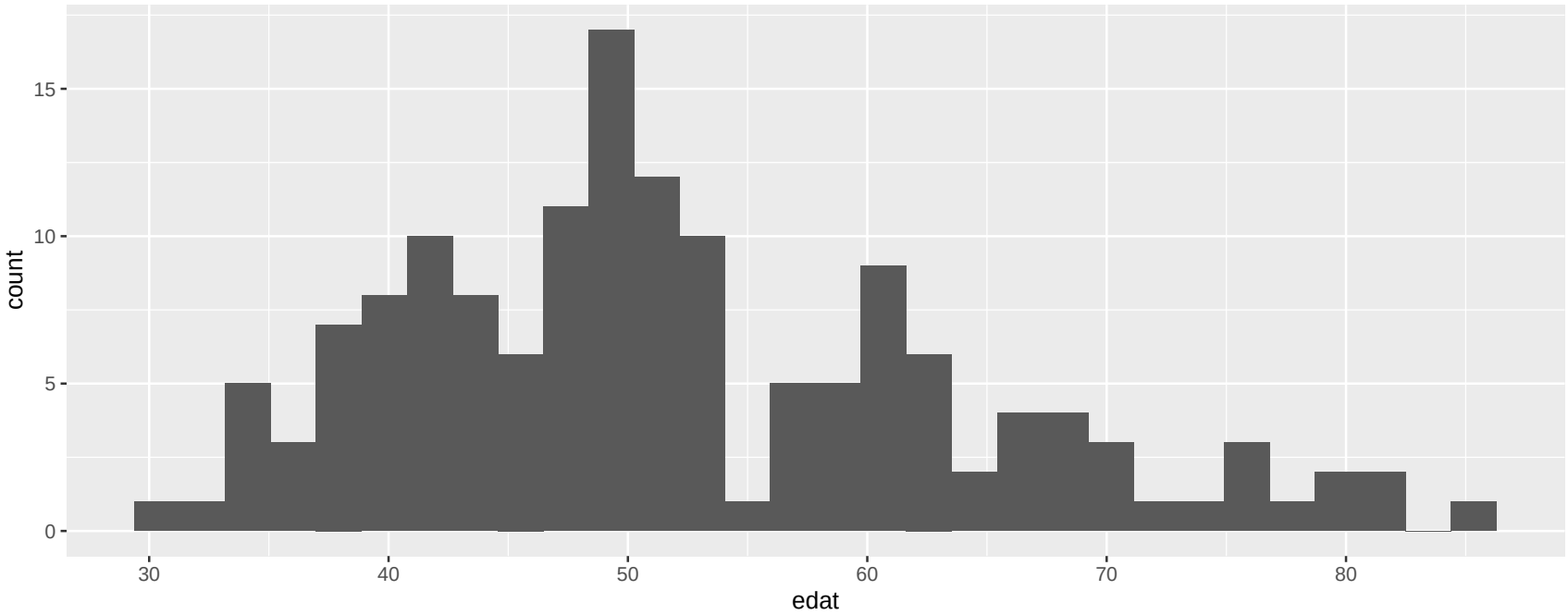
We could dedicate half of the course only to EDA. Here we will only see the most common approaches to visualize data:

- `Histograms`
- `Scatterplots`
- `Boxplots`

Histograms

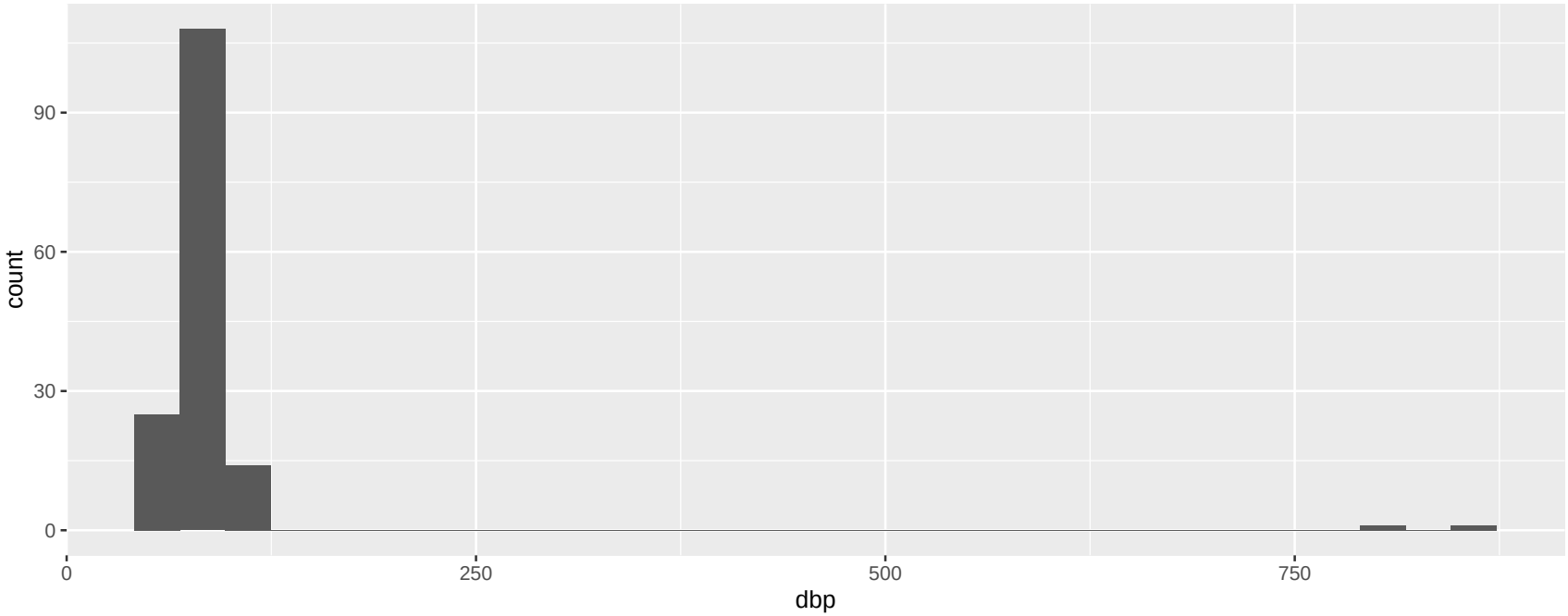
We will use histograms to plot the frequencies of each level of variables. This is the way to see the data distribution of particulars variables.

```
ggplot(diab, aes(edat)) +
  geom_histogram(bins = 30)
```



Histograms (II)

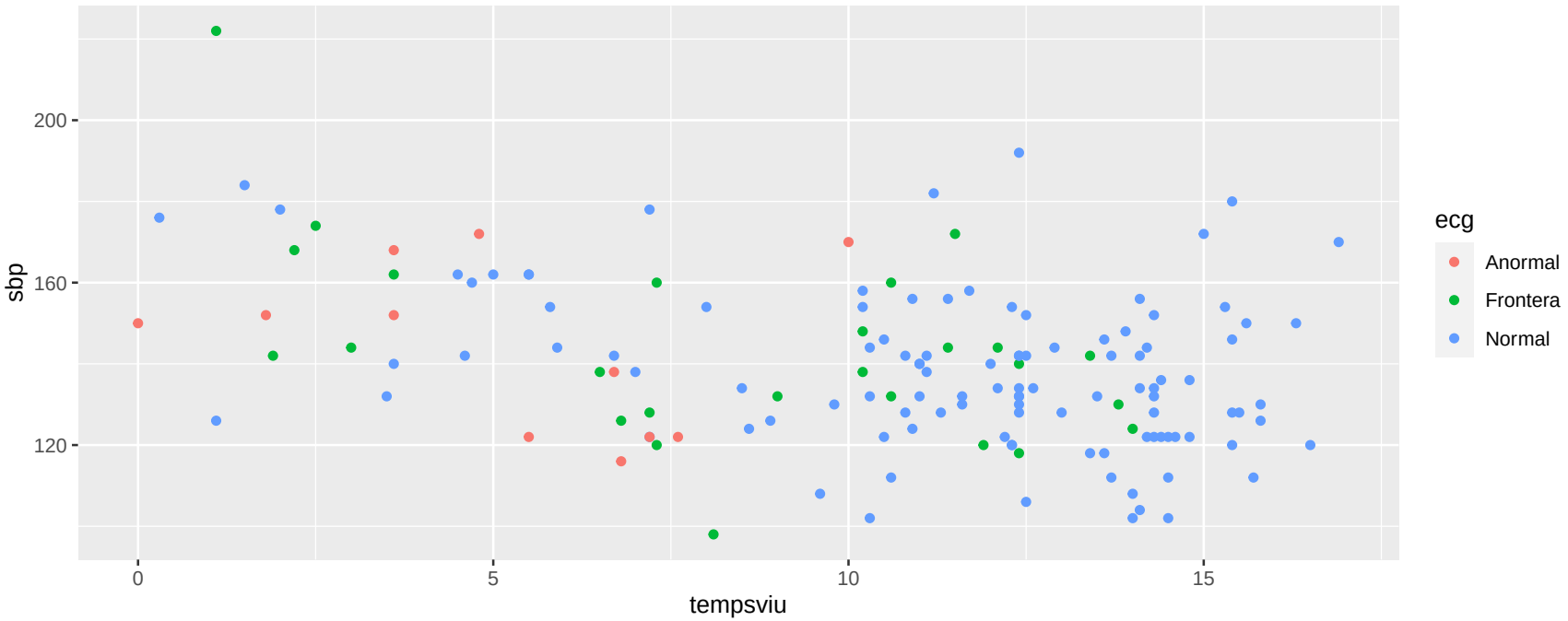
```
ggplot(diab, aes(dbp)) +
  geom_histogram(bins = 30)
```



Scatterplots. Two Continuous variables

This is the graphical way to check the relation between two variables:

```
ggplot(diab, aes(tempsviu, sbp, col = ecg)) +
  geom_point()
```

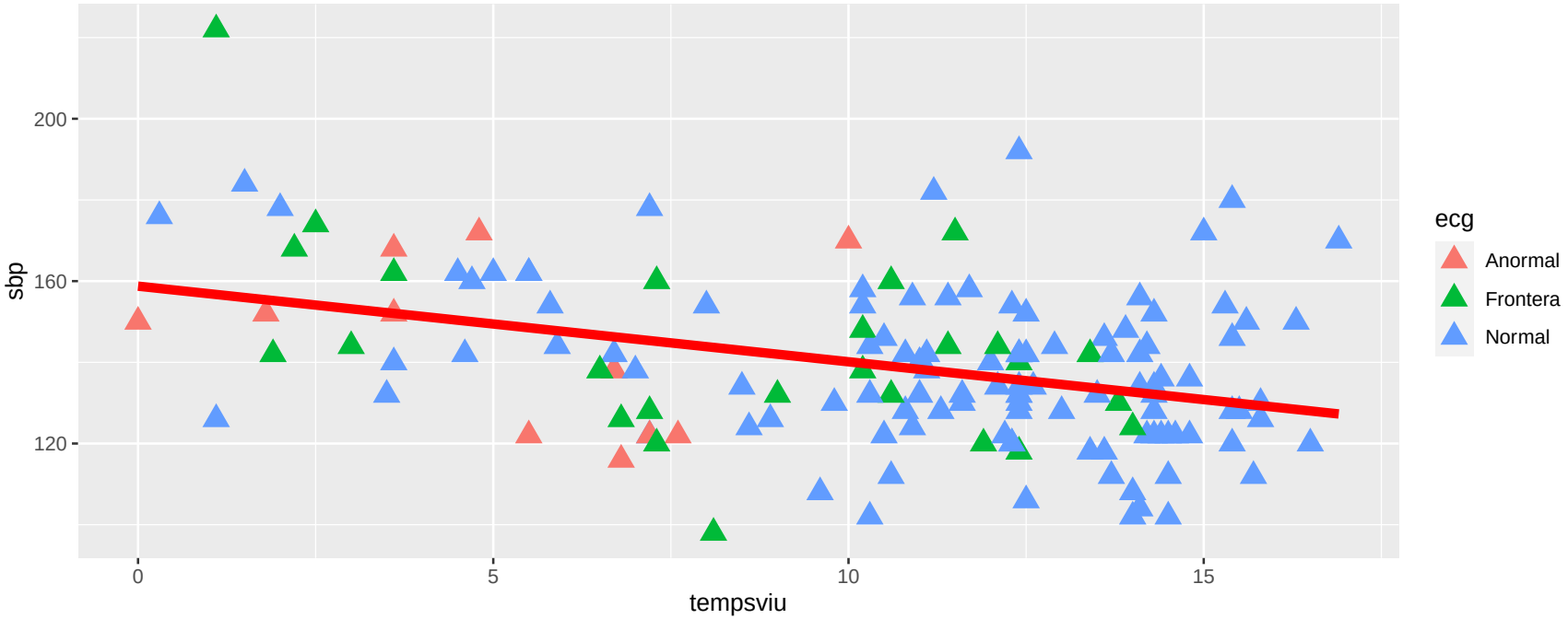


Scatterplots (II)

```
ggplot(diab, aes(tempsviu, sbp, col = ecg)) +
  geom_point(size = 4, pch = 17) +
  geom_smooth(lwd=2, se=FALSE, method="lm", col="red")
```

Scatterplots (II)

```
## `geom_smooth()` using formula 'y ~ x'
```

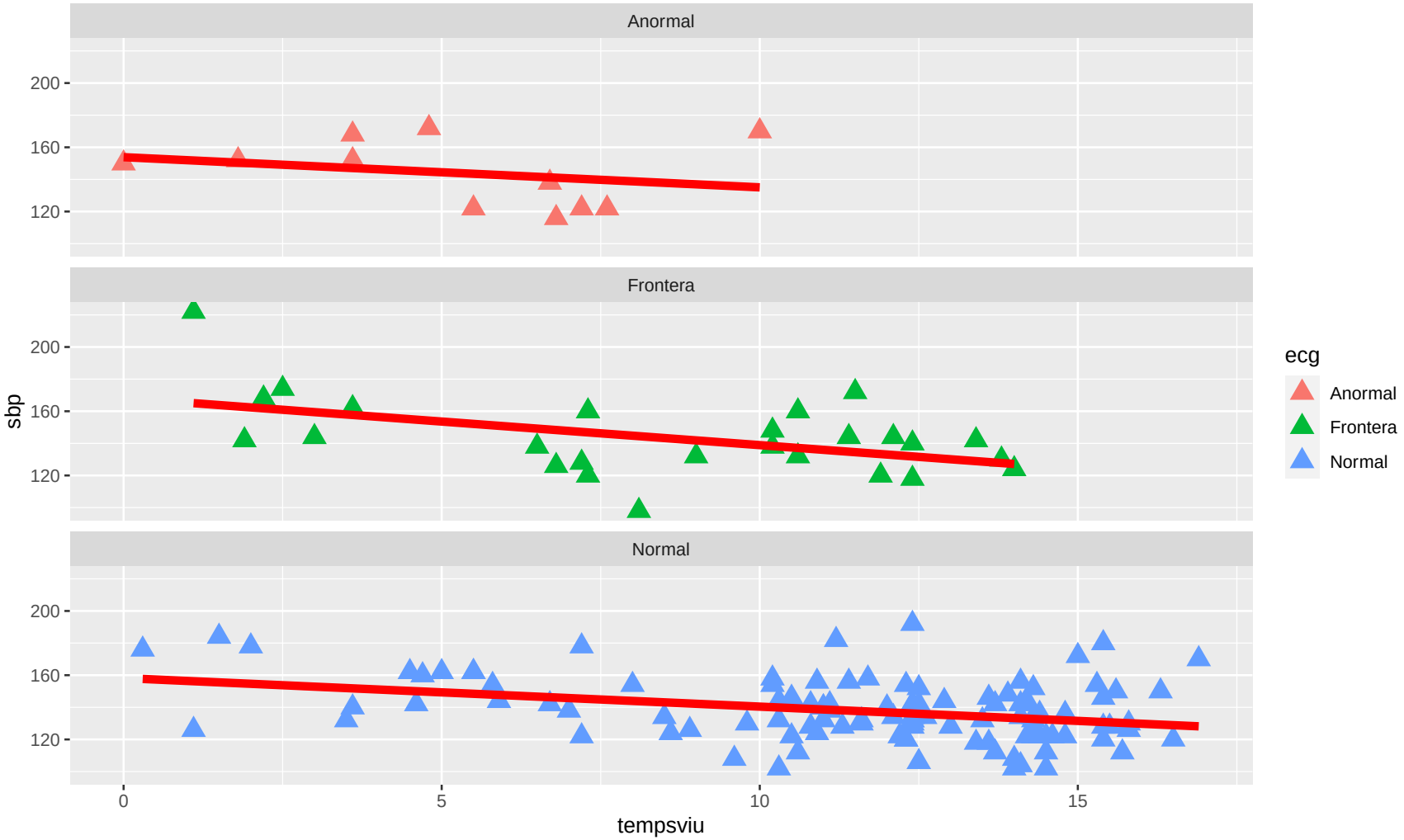


Faceting

```
ggplot(diab, aes(tempsviu, sbp, col = ecg)) +
  geom_point(size = 4, pch = 17) +
  geom_smooth(lwd = 2, se=FALSE, method="lm", col="red") +
  facet_wrap(~ ecg, ncol = 1)
```

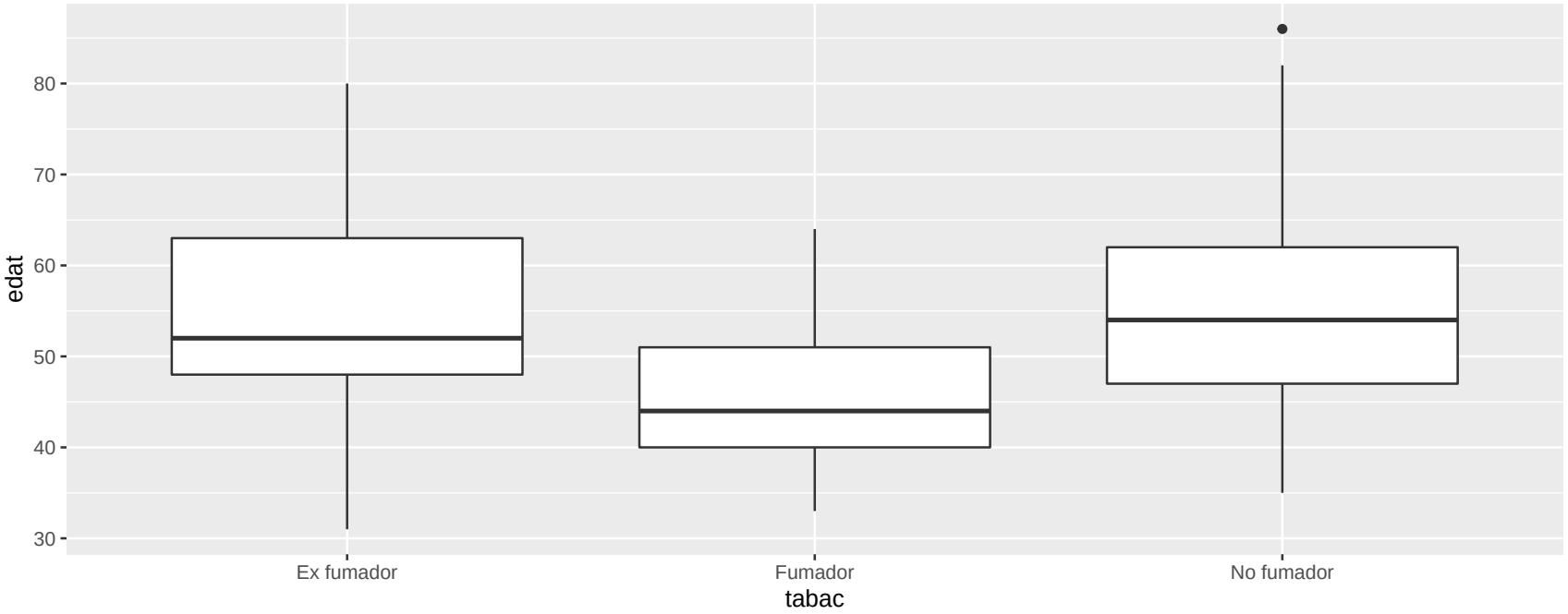
Faceting

```
## `geom_smooth()` using formula 'y ~ x'
```



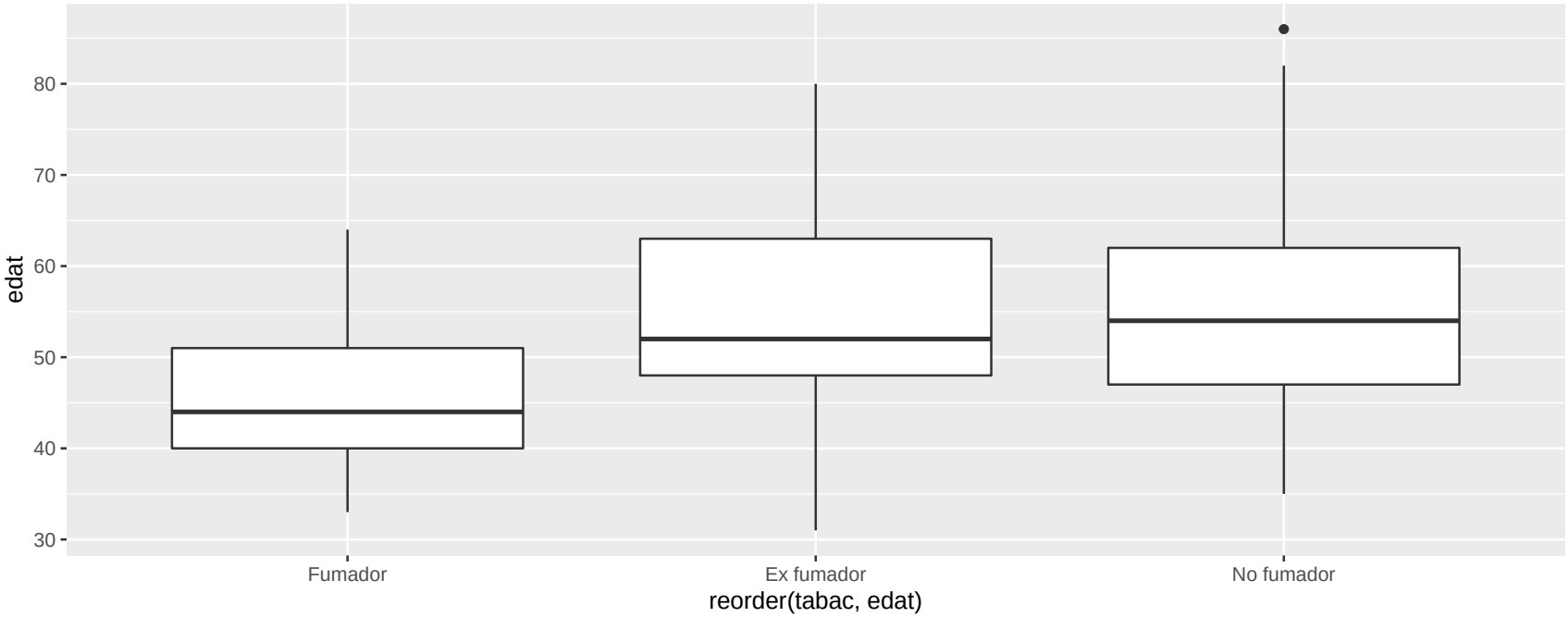
Boxplot. Continuous versus categorical

```
ggplot(diab, aes(tabac, edat)) +
  geom_boxplot()
```



Boxplot (II)

```
ggplot(diab, aes(x= reorder(tabac, edat), y = edat)) +
  geom_boxplot()
```



EXERCISE

2. 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
   - Improve the first graphic (add linear regression, avoid strange data in dbp, . . .)

EXERCISE

4. 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
   - Plot the relationship between talla and peso