Introduction to R for social researchers

Alexandru Cernat

Practical 1

Initial set-up

Before we start let's do the prep work.

1. Open Rstudio
2. Set up a working directory so we have all our files in one place. Go to the menu on the top. Go to "Session" -> "Set Working Directory" -> "Choose Directory". Choose there a folder on your computer to store everything from the session.
3. Open a new script. In the menu go to "File" -> "New File" -> "R Script". **From now on write only in the script**. When you are happy with what you wrote you can select the code you want to run and press the "Run" button. Alternatively you can use the shortcut "ctrl + enter". Remember to save your script regularly. You can use this to reproduce your work and as a back up if something goes wrong. Also, remember to add comments so it's easier to understand what you did in the future.

Bellow you will have some tasks to do. Use the slides provided as inspiration for how to solve them. Sometimes I might show you some code that you should also copy and run on your computer before progressing. **If you get stuck ask for help!**

Creating objects

We will start with some basic tasks of creating objects in `R`.

Create a numeric vector called "num_vct" that includes the numbers: 5, 10, 150, 8. Also, print to see how it looks like. Also, divide the vector by 5 and see what you get.

```
num_vct <- c(5, 10, 150, 8)
num_vct
```

```
## [1]   5  10 150   8
```

```
num_vct/5
```

```
## [1]  1.0  2.0 30.0  1.6
```

Create a logical vector called "log_vct" that includes values: TRUE, FALSE, F, F and T. Print it. Calculate the mean of the vector. Why do you think you get that result for the mean?

```
log_vct <- c(TRUE, FALSE, F, F, T)

log_vct
```

```
## [1]  TRUE FALSE FALSE FALSE  TRUE
```

```
mean(log_vct)
```

```
## [1] 0.4
```

Create a string vector called "str_vct" that includes: "I am", "loving", "R".

```
str_vct <- c("I am", "loving", "R")

str_vct
```

```
## [1] "I am"   "loving" "R"
```

Run the code bellow to apply the function `paste()` to it. What do you think this function does?

```
paste(str_vct, collapse = " ")
```

```
## [1] "I am loving R"
```

Use the `typeof()` command on the three objects you created above. Do they give you the output you expect?

```
typeof(num_vct)
```

```
## [1] "double"
```

```
typeof(log_vct)
```

```
## [1] "logical"
```

```
typeof(str_vct)
```

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

Create a list called "list1" with the three objects you created (num_vct, log_vct, str_vct).

```
list1 <- list(num_vct, log_vct, str_vct)
list1
```

```
## [[1]]
## [1]   5  10 150   8
##
## [[2]]
## [1]  TRUE FALSE FALSE FALSE  TRUE
##
## [[3]]
## [1] "I am"   "loving" "R"
```

Give the names "a", "b", "c" and "d" to the elements of the numeric vector ("num_vct").

```
names(num_vct) <- c("a", "b", "c", "d")
num_vct
```

```
##   a   b   c   d
##   5  10 150   8
```

Create a factor with the values 1, 0, 0, 0, 1 where 0 represents "Did not vote" and 1 represents "Voted". Call it "fct_var" and do a table of it.

```
fct_var <- factor(c(1, 0, 0, 0, 1),
                  labels = c("Voted", "Did not vote"))

table(fct_var)
```

```
## fct_var
##        Voted Did not vote
##            3            2
```

Do a matrix with all the numbers between 1 and 12 that has 4 rows and save it as "matrix1". Print it.

```
matrix1 <- matrix(1:12, nrow = 4)
matrix1
```

```
##      [,1] [,2] [,3]
## [1,]    1    5    9
## [2,]    2    6   10
## [3,]    3    7   11
## [4,]    4    8   12
```

Try to do the matrix again but this time use the option "byrow = T". Save the result as "matrix2" and print it. What do you think is the effect of the option?

```
matrix2 <- matrix(1:12, nrow = 4, byrow = T)
matrix2
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
## [4,]   10   11   12
```

Create a dataset called "df" with three variables:

- age: 14, 80, 35, 65
- sex: "m", "f", "f", "m"
- vote: FALSE, TRUE, TRUE, FALSE

Print the result.

```
df <- data.frame(age = c(14, 80, 35, 65),
                 sex = c("m", "f", "f", "m"),
                 vote = c(FALSE, TRUE, TRUE, FALSE))

df
```

```
##   age sex  vote
## 1  14   m FALSE
## 2  80   f  TRUE
## 3  35   f  TRUE
## 4  65   m FALSE
```

Add a new row to the dataset with an individual that is 21, male, and did not vote. Save the new dataset as "df2". Print the result.

```
df2 <- rbind(df, c(21, "m", FALSE))
df2
```

```
##   age sex  vote
## 1  14   m FALSE
## 2  80   f  TRUE
## 3  35   f  TRUE
## 4  65   m FALSE
## 5  21   m FALSE
```

Add a column to "df2" called "work" that takes values: FALSE, FALSE, TRUE, TRUE, TRUE. Save the data as "df3" and print it.

```
df3 <- cbind(df2, work = c(FALSE, FALSE, TRUE, TRUE, TRUE))

df3
```

```
##   age sex  vote  work
## 1  14   m FALSE FALSE
## 2  80   f  TRUE FALSE
## 3  35   f  TRUE  TRUE
## 4  65   m FALSE  TRUE
## 5  21   m FALSE  TRUE
```

Selecting elements

Now that we created different types of objects let's practice selection.

Select the second element of "str_vct".

Select the third element of "log_vct".

Select the first and third element of "num_vct" (in one command).

```
str_vct[2]
```

```
## [1] "loving"
```

```
log_vct[3]
```

```
## [1] FALSE
```

```
num_vct[c(1, 3)]
```

```
##   a   c
##   5 150
```

Select the second element of "num_vct" using its name ("b").

```
num_vct["b"]
```

```
##  b
## 10
```

For the "matrix1" object you created before try to (separately and without saving the result):

- select the first row
- select the second and third column
- select the element in the second row and third column
- exclude rows 1 and 3

```
matrix1[1, ]
```

```
## [1] 1 5 9
```

```
matrix1[, c(2, 3)]
```

```
##      [,1] [,2]
## [1,]    5    9
## [2,]    6   10
## [3,]    7   11
## [4,]    8   12
```

```
matrix1[2, 3]
```

```
## [1] 10
```

```
matrix1[-c(1, 3), ]
```

```
##      [,1] [,2] [,3]
## [1,]    2    6   10
## [2,]    4    8   12
```

For the "df3" object you created before try to (separately and without saving the result):

- select the "vote" variable. Try doing it once using the name and once using the column number
- exclude the second and third rows as well as the forth variable (in one command)
- select the rows with age below 65

```
df3$vote
```

```
## [1] "FALSE" "TRUE"  "TRUE"  "FALSE" "FALSE"
```

```
df3[, 3]
```

```
## [1] "FALSE" "TRUE"  "TRUE"  "FALSE" "FALSE"
```

```
df3[-c(2, 3), -4]
```

```
##   age sex  vote
## 1  14   m FALSE
## 4  65   m FALSE
## 5  21   m FALSE
```

```
df3[df3$age < 65, ]
```

```
##   age sex  vote  work
## 1  14   m FALSE FALSE
## 3  35   f  TRUE  TRUE
## 5  21   m FALSE  TRUE
```

Practical 2

Let's practice a few more things that you have learned.

Try to order the "df3" data you created in the first practical by age. Try to do it both going from the smallest age to the highest one and the other way around.

```
df3[order(df3$age), ]
```

```
##   age sex  vote  work
## 1  14   m FALSE FALSE
## 5  21   m FALSE  TRUE
## 3  35   f  TRUE  TRUE
## 4  65   m FALSE  TRUE
## 2  80   f  TRUE FALSE
```

```
df3[order(df3$age, decreasing = T), ]
```

```
##   age sex  vote  work
## 2  80   f  TRUE FALSE
## 4  65   m FALSE  TRUE
## 3  35   f  TRUE  TRUE
## 5  21   m FALSE  TRUE
## 1  14   m FALSE FALSE
```

Next, try to remove the "sex" variable from the "df2" dataset.

```
df2
```

```
##   age sex  vote
## 1  14   m FALSE
## 2  80   f  TRUE
## 3  35   f  TRUE
## 4  65   m FALSE
## 5  21   m FALSE
```

```
# option 1, select the other variables
df2[, c(1, 3)]
```

```
##   age  vote
## 1  14 FALSE
## 2  80  TRUE
## 3  35  TRUE
## 4  65 FALSE
## 5  21 FALSE
```

```
# option 2, delete the variable
df2$sex <- NULL

# see result
df2
```

```
##   age  vote
## 1  14 FALSE
## 2  80  TRUE
## 3  35  TRUE
## 4  65 FALSE
## 5  21 FALSE
```

Using "df3" replace the values on the work variable with FALSE if age is larger than 64.

```
# the condition we want
df3$age > 64
```

```
## [1] FALSE  TRUE FALSE  TRUE FALSE
```

```r
# apply the condition to selection
df3$work[df3$age > 64]
```

```
## [1] FALSE  TRUE
```

```r
# also do the replacing
df3$work[df3$age > 64] <- F
```

Describe data

You are next going to apply what you have learned to some real data. We will start out by using the "swiss" data that is in the memory of `R` by default. This has some statistics on Swiss regions at about 1888. You can print it like any object. You can also find out more about it using the `?swiss`:

```r
# print the data
swiss
```

```
##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
## Moutier           85.8        36.5          12         7    33.77
## Neuveville        76.9        43.5          17        15     5.16
## Porrentruy        76.1        35.3           9         7    90.57
## Broye             83.8        70.2          16         7    92.85
## Glane             92.4        67.8          14         8    97.16
## Gruyere           82.4        53.3          12         7    97.67
## Sarine            82.9        45.2          16        13    91.38
## Veveyse           87.1        64.5          14         6    98.61
## Aigle             64.1        62.0          21        12     8.52
## Aubonne           66.9        67.5          14         7     2.27
## Avenches          68.9        60.7          19        12     4.43
## Cossonay          61.7        69.3          22         5     2.82
## Echallens         68.3        72.6          18         2    24.20
## Grandson          71.7        34.0          17         8     3.30
## Lausanne          55.7        19.4          26        28    12.11
## La Vallee         54.3        15.2          31        20     2.15
## Lavaux            65.1        73.0          19         9     2.84
## Morges            65.5        59.8          22        10     5.23
## Moudon            65.0        55.1          14         3     4.52
## Nyone             56.6        50.9          22        12    15.14
## Orbe              57.4        54.1          20         6     4.20
## Oron              72.5        71.2          12         1     2.40
## Payerne           74.2        58.1          14         8     5.23
## Paysd'enhaut      72.0        63.5           6         3     2.56
## Rolle             60.5        60.8          16        10     7.72
## Vevey             58.3        26.8          25        19    18.46
## Yverdon           65.4        49.5          15         8     6.10
## Conthey           75.5        85.9           3         2    99.71
## Entremont         69.3        84.9           7         6    99.68
## Herens            77.3        89.7           5         2   100.00
## Martigwy          70.5        78.2          12         6    98.96
```

```
## Monthey           79.4        64.9          7         3    98.22
## St Maurice        65.0        75.9          9         9    99.06
## Sierre            92.2        84.6          3         3    99.46
## Sion              79.3        63.1         13        13    96.83
## Boudry            70.4        38.4         26        12     5.62
## La Chauxdfnd      65.7         7.7         29        11    13.79
## Le Locle          72.7        16.7         22        13    11.22
## Neuchatel         64.4        17.6         35        32    16.92
## Val de Ruz        77.6        37.6         15         7     4.97
## ValdeTravers      67.6        18.7         25         7     8.65
## V. De Geneve      35.0         1.2         37        53    42.34
## Rive Droite       44.7        46.6         16        29    50.43
## Rive Gauche       42.8        27.7         22        29    58.33
##              Infant.Mortality
## Courtelary               22.2
## Delemont                 22.2
## Franches-Mnt             20.2
## Moutier                  20.3
## Neuveville               20.6
## Porrentruy               26.6
## Broye                    23.6
## Glane                    24.9
## Gruyere                  21.0
## Sarine                   24.4
## Veveyse                  24.5
## Aigle                    16.5
## Aubonne                  19.1
## Avenches                 22.7
## Cossonay                 18.7
## Echallens                21.2
## Grandson                 20.0
## Lausanne                 20.2
## La Vallee                10.8
## Lavaux                   20.0
## Morges                   18.0
## Moudon                   22.4
## Nyone                    16.7
## Orbe                     15.3
## Oron                     21.0
## Payerne                  23.8
## Paysd'enhaut             18.0
## Rolle                    16.3
## Vevey                    20.9
## Yverdon                  22.5
## Conthey                  15.1
## Entremont                19.8
## Herens                   18.3
## Martigwy                 19.4
## Monthey                  20.2
## St Maurice               17.8
## Sierre                   16.3
## Sion                     18.1
## Boudry                   20.3
## La Chauxdfnd             20.5
```

```
## Le Locle                 18.9
## Neuchatel                23.0
## Val de Ruz               20.0
## ValdeTravers             19.5
## V. De Geneve             18.0
## Rive Droite              18.2
## Rive Gauche              19.3
```

```
# find out more about the data
?swiss
```

Use the commands you have learned before to explore the "swiss" data (dim, head, tail, View, summary). *The data already exists in the memory of* `R` *so you don't need to import it.*

```
dim(swiss)
```

```
## [1] 47  6
```

```
head(swiss)
```

```
##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
## Moutier           85.8        36.5          12         7    33.77
## Neuveville        76.9        43.5          17        15     5.16
## Porrentruy        76.1        35.3           9         7    90.57
##              Infant.Mortality
## Courtelary               22.2
## Delemont                 22.2
## Franches-Mnt             20.2
## Moutier                  20.3
## Neuveville               20.6
## Porrentruy               26.6
```

```
tail(swiss)
```

```
##              Fertility Agriculture Examination Education Catholic
## Neuchatel         64.4        17.6          35        32    16.92
## Val de Ruz        77.6        37.6          15         7     4.97
## ValdeTravers      67.6        18.7          25         7     8.65
## V. De Geneve      35.0         1.2          37        53    42.34
## Rive Droite       44.7        46.6          16        29    50.43
## Rive Gauche       42.8        27.7          22        29    58.33
##              Infant.Mortality
## Neuchatel                23.0
## Val de Ruz               20.0
## ValdeTravers             19.5
## V. De Geneve             18.0
## Rive Droite              18.2
## Rive Gauche              19.3
```

```
View(swiss)
summary(swiss)
```

```
##    Fertility      Agriculture     Examination      Education
##  Min.   :35.00   Min.   : 1.20   Min.   : 3.00   Min.   : 1.00
##  1st Qu.:64.70   1st Qu.:35.90   1st Qu.:12.00   1st Qu.: 6.00
##  Median :70.40   Median :54.10   Median :16.00   Median : 8.00
##  Mean   :70.14   Mean   :50.66   Mean   :16.49   Mean   :10.98
##  3rd Qu.:78.45   3rd Qu.:67.65   3rd Qu.:22.00   3rd Qu.:12.00
##  Max.   :92.50   Max.   :89.70   Max.   :37.00   Max.   :53.00
##     Catholic       Infant.Mortality
##  Min.   :  2.150   Min.   :10.80
##  1st Qu.:  5.195   1st Qu.:18.15
##  Median : 15.140   Median :20.00
##  Mean   : 41.144   Mean   :19.94
##  3rd Qu.: 93.125   3rd Qu.:21.70
##  Max.   :100.000   Max.   :26.60
```

Calculate the mean and variation of the "Fertility" and "Infant.Mortality" variables.

```
mean(swiss$Fertility)
```

```
## [1] 70.14255
```

```
var(swiss$Fertility)
```

```
## [1] 156.0425
```

```
mean(swiss$Infant.Mortality)
```

```
## [1] 19.94255
```

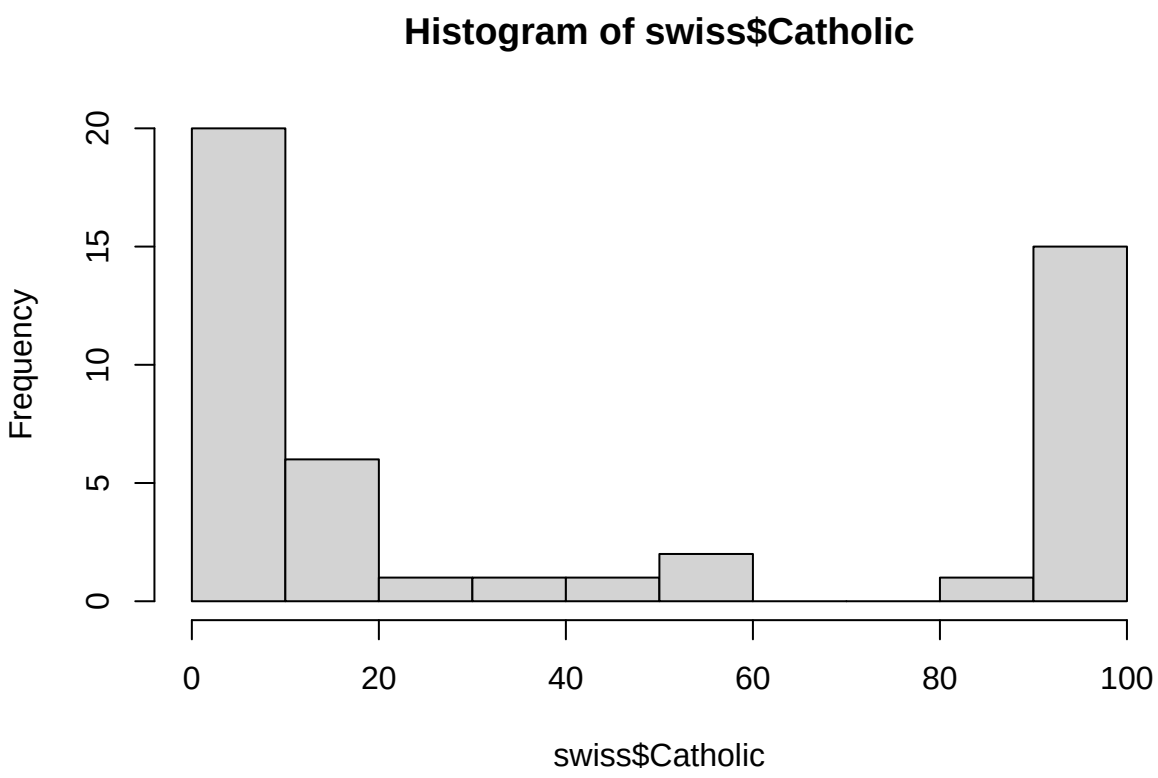
```
var(swiss$Infant.Mortality)
```

```
## [1] 8.483802
```

Modify variables

Next we are going to create some new variables by recording the existing ones. First, let's look at the "Catholic" variable using a histogram:

```
hist(swiss$Catholic)
```

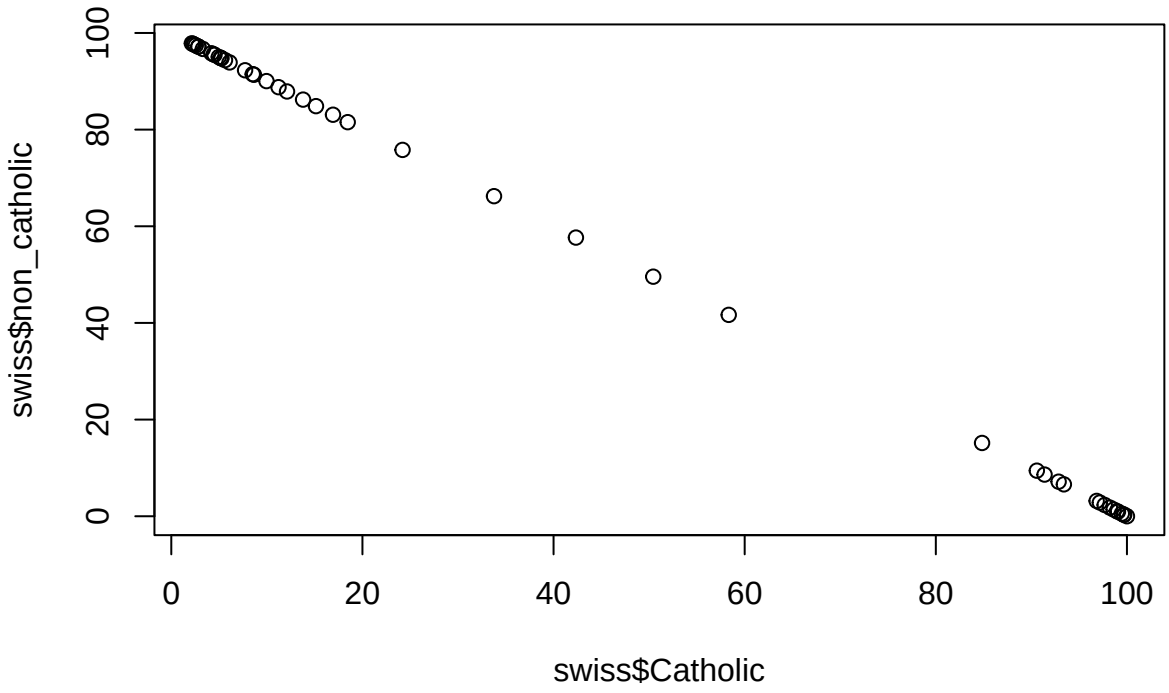


Imagine you wanted to calculate the percentage of non-catholics in each region. Make such a variable and add it to the dataset as "non_catholic". *Tip: you can do 100 - the "Catholic" variable.*

```
swiss$non_catholic <- 100 - swiss$Catholic
```

To check the recode we can make a plot of the two variables:

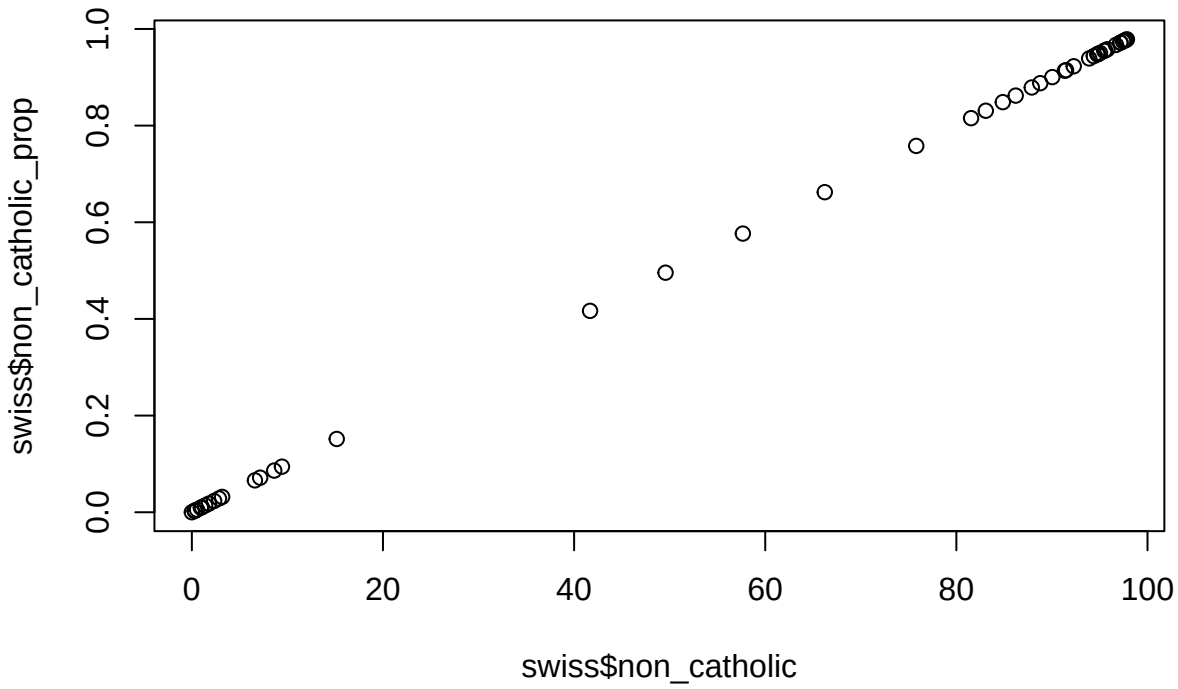
```
plot(swiss$Catholic, swiss$non_catholic)
```



How do you interpret the plot? Do you think the recoding worked as expected?

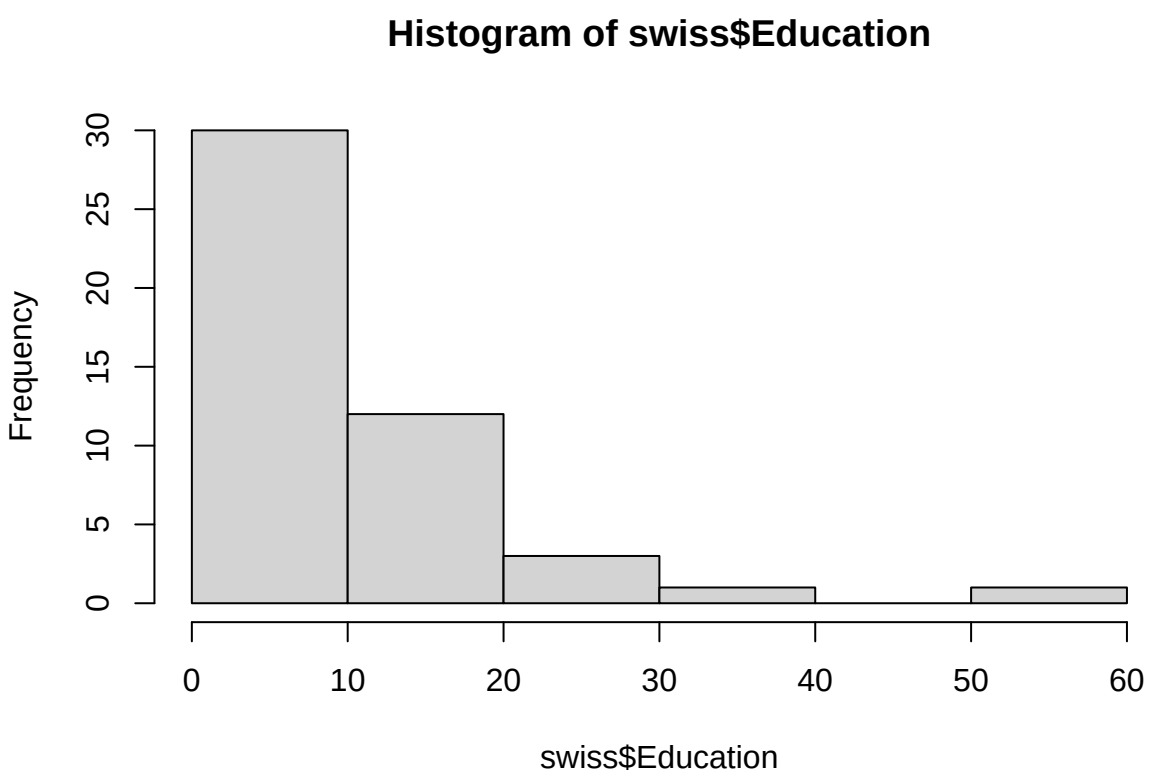
Imagine that we want to calculate proportions instead of percentages. Make a new variable, "non_catholic_prop", by dividing the previous variable by 100. Then plot "non_catholic" and "non_catholic_prop" (like we did above) to check the result.

```
swiss$non_catholic_prop <- swiss$non_catholic/100
plot(swiss$non_catholic, swiss$non_catholic_prop)
```



Do a histogram of the "Education" variable.

```
# check distribution of education
hist(swiss$Education)
```



Imagine we want to dichotomize the education variable. We want to make a variable with two categories: "1" if "Education" is bellow 12 and "2" if it is above 12. Use the `ifelse()` command to make such a variable and call it "ed_level".

```
swiss$ed_level <- ifelse(swiss$Education < 12, 1, 2)
```

Do a table of the new variable as well as a cross-table between the original variable and the new one.

```
table(swiss$ed_level)
```

```
##
##  1  2
## 31 16
```

```
table(swiss$Education, swiss$ed_level)
```

```
##
##      1 2
##   1  1 0
##   2  3 0
##   3  4 0
##   5  2 0
##   6  4 0
##   7  7 0
##   8  4 0
```

```
##   9  3 0
##   10 2 0
##   11 1 0
##   12 0 5
##   13 0 3
##   15 0 1
##   19 0 1
##   20 0 1
##   28 0 1
##   29 0 2
##   32 0 1
##   53 0 1
```

To make the new variable easier to understand let's make it a factor. Create a new factor variable "ed_level_fct" which takes "ed_level" and adds the labels "Low" and "High".

```
swiss$ed_level_fct <- factor(swiss$ed_level, labels = c("Low", "High"))
```

Do a cross-table between "ed_level" and "ed_level_fct". Was the coding correct?

```
table(swiss$ed_level, swiss$ed_level_fct)
```

```
##
##     Low High
##   1  31    0
##   2   0   16
```

Bonus track

If there is time you can play with the European Social Survey. Download the "ess9_raw.csv" in your working directory. Then, use the command you learned in the presentation to import it.

```
ess9_full <- read.csv("ess9_raw.csv")
```

Make a smaller dataset in which you only keep the following variables: "idno", "cntry", "gndr", "eduyrs", "ppltrst", "vote", "happy".

```
vars_int <- c("idno", "cntry", "gndr", "eduyrs",
              "ppltrst", "vote", "happy")

ess9 <- ess9_full[vars_int]
```

Explore the data using the commands you learned above (head, tail, dim, view and summary).

```
dim(ess9)
```

```
## [1] 36015     7
```

```
summary(ess9)
```

```
##       idno          cntry                gndr          eduyrs
##  Min.   :    1   Length:36015       Min.   :1.000   Min.   : 0.00
##  1st Qu.:12281   Class :character   1st Qu.:1.000   1st Qu.:11.00
##  Median :24378   Mode  :character   Median :2.000   Median :12.00
##  Mean   :24375                      Mean   :1.528   Mean   :13.01
##  3rd Qu.:36471                      3rd Qu.:2.000   3rd Qu.:16.00
##  Max.   :48642                      Max.   :2.000   Max.   :51.00
##                                                     NA's   :505
##     ppltrst            vote           happy
##  Min.   : 0.000   Min.   :1.000   Min.   : 0.000
##  1st Qu.: 3.000   1st Qu.:1.000   1st Qu.: 7.000
##  Median : 5.000   Median :1.000   Median : 8.000
##  Mean   : 5.094   Mean   :1.368   Mean   : 7.377
##  3rd Qu.: 7.000   3rd Qu.:2.000   3rd Qu.: 9.000
##  Max.   :10.000   Max.   :3.000   Max.   :10.000
##  NA's   :109      NA's   :412     NA's   :148
```

Recode the sex and vote variables like we did in the slides and try to find out if there are differences between man and women in the voting likelihood.

```
# recode gender
ess9$gndr_fct <- factor(ess9$gndr,
                        labels = c("Male", "Female"))

# code to 0 all values except 1
ess9$vote2 <- ifelse(ess9$vote == 1, "Yes", "No")

# code as missing if value is 3
ess9$vote2 <- ifelse(ess9$vote == 3, NA, ess9$vote2)

# make tables
tab1 <- table(ess9$gndr_fct, ess9$vote2)
tab1
```

```
##
##              No   Yes
##   Male    3278 12050
##   Female  3930 13397
```

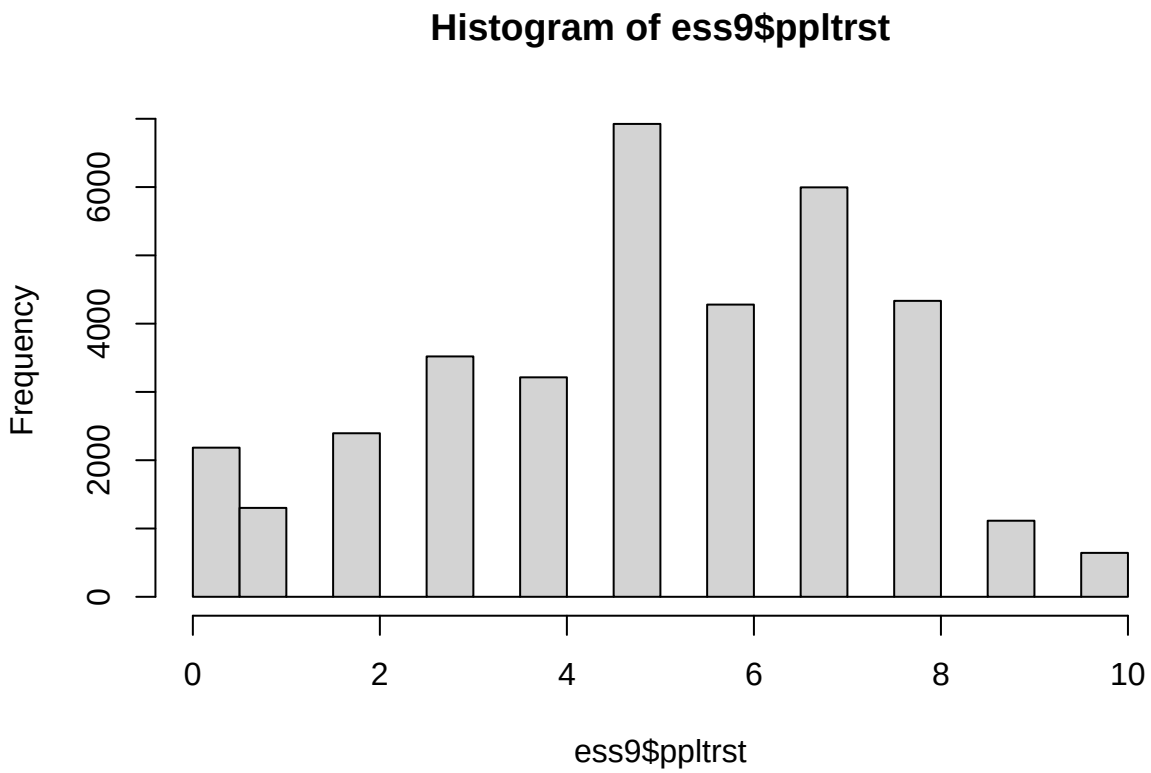
```
tab2 <- prop.table(tab1, 1)

round(tab2, 2)
```

```
##
##             No  Yes
##   Male   0.21 0.79
##   Female 0.23 0.77
```

Do a histogram of the "ppltrst" (How much you trust people, larger value = more trust).

```
hist(ess9$ppltrst)
```



Calculate the average trust in the entire data. Then calculate average trust in Great Britain ("GB" in the "cntry" variable) and in Russia ("RS" in the "cntry" variable). Are there differences between countries in the amount of trust in other people?

```
mean(ess9$ppltrst, na.rm = T)
```

```
## [1] 5.094051
```

```
mean(ess9$ppltrst[ess9$cntry == "GB"], na.rm = T)
```

```
## [1] 5.17508
```

```
mean(ess9$ppltrst[ess9$cntry == "RS"], na.rm = T)
```

```
## [1] 3.745704
```

Do the same for happiness ("happy" variable).

```
mean(ess9$happy, na.rm = T)
```

```
## [1] 7.377394
```

```r
mean(ess9$happy[ess9$cntry == "GB"], na.rm = T)
```

```
## [1] 7.570259
```

```r
mean(ess9$happy[ess9$cntry == "RS"], na.rm = T)
```

```
## [1] 6.76308
```

Save the final dataset you used as a csv file called "ess9_small.csv".

```r
write.csv(ess9, "ess9_small.csv")
```