Prof. Dr. Steffen Wagner
Angewandte Statistik
Fachbereich II
Berliner Hochschule für Technik

Exercise 03: (Basic) Summary Statistics, Boxplots and Matrices

Statistical Computing – WiSe 2022/23

Preparations

a. Make sure you work inside your RStudio project. Details please see Exercise 01.

b. Install the R package `carDate` by running the following code

```
install.packages("carData")
```

Another possibility to install packages is via the *Packages* tab in the lower right pane of RStudio.

c. Open a new R script file (either `Shift + Ctrl + N` or via the menu), type the following comment/code in the first line and save the file.

```
#==========================================================================#
# Statistical Computing: exercise sheet 3                                  #
# Basic Statistics and Explanatory Data Analysis (EDA)                     #
#==========================================================================#
rm(list = ls(all.names = TRUE))
```

d. Ask your professor about the command `rm(list = ls(all.names = TRUE))` in the next lecture.

e. Work through the following examples as you did last week. In many of the code examples the output is not included, you should always read and understand the output. Please add comments (in your own words) to your code so that it will make sense to you when you return several weeks (or years) later.

f. When there is a question in the text e.g. *"What is the mean and standard deviation of the prestige Scores?"*, copy the question and the answer as a a comment to your R code. (Later on, we will use RMarkdown for this purpose.)

The Prestige data set and basic summary statistics

When analysing data you will usually start by carrying out some simple procedures to get a feeling for the data. Typically, we will produce some appropriate plots and calculate some useful summary statistics.

Loading the data

a. In this section we will be using the Canadian *Prestige* data set. To work with this data set add the following code to your script and run the commands.

```
# 01: load additional packages ---------------------------------------------
library(carData)

# 02: load the  data set ---------------------------------------------------
data(Prestige)
```

Remarks:

- `data(Prestige)` loads the data into the Global Environment.
- This works since you loaded the package `carData` before.

b. Now run the command `?Prestige` or `help(Prestige)` and read the documentation for the data set.

Summarising numeric variables

a. Add a new section to your R script via

```
# 03: Summarising numeric variables ----------------------------------------
```

b. First of all we should find out how many rows and columns our data set has via

```
dim(Prestige)
```

c. To find the names of the variables type

```
names(Prestige)
```

Notice that there is a variable called `prestige` in the data frame called `Prestige`.

d. A quick method of seeing what the data type for each variable is, is

```
str(Prestige)
```

The basic commands to summarise the numeric variables in the data set are straightforward.

`mean()`	arithmetic mean
`median()`	median
`sd()`	standard deviation
`var()`	variance
`quantile()`	quantile

You will learn about the standard deviation and variance[1] one of the next lectures, but they measure how spread out the data values are.

e. Calculate the following statistics for the variable `income`:

```
mean(Prestige$income)
median(Prestige$income)
sd(Prestige$income)
var(Prestige$income)
sqrt(var(Prestige$income))
```

The units of the mean, median and standard deviation are the same as the measured variable e.g. Canadian dollars. The units of the variance are squared units e.g. dollars^2. In terms of understanding the numbers, the standard deviation is much more helpful than the variance.

f. We may also want to compute the quantiles of a set of numeric data. The command is `quantile`. For example, to obtain the *lower* and *upper* quartiles of income:

```
quantile(Prestige$income, probs = c(0.25, 0.75))
```

g. Approximately how many `income` values would you expect to be less than the lower quartile found above? Check your answer using

```
sum(
  Prestige$income < quantile(Prestige$income, probs = 0.25)
)
```

[1] In most statistic software the *sample* variance and *sample* standard deviation are calculated which means that the formula uses $n-1$ in the denominator.

and

```
mean(
  Prestige$income < quantile(Prestige$income, probs = 0.25)
)
```

h. Discuss why the above expressions work although

```
Prestige$income < quantile(Prestige$income, probs = 0.25)
```

returns a logical vector?

Exercise:

a. What is the mean and standard deviation of the `prestige` Scores?

```
mean(Prestige$prestige)
```

```
## [1] 46.83333
```

b. What is the median of the census codes? Does this statistic even make sense?

```
# the mean of the census code can technically be calculated since the code is
# stored based on an integer code.
median(Prestige$census)
```

```
## [1] 5135
```

```
# Thus this integer code represents a nominal variable, there is no meaningful
# interpretation for the obtained number

# To avoid such potential sources for errors, the variable `census` should be
# converted explicitly to a factor variable (representation in R for nominal/ordinal
# data) via
Prestige$census <- factor(Prestige$census)

# Now the median can´t be converted any more
median(Prestige$census)
```

```
## Error in median.default(Prestige$census): benötige numerische Daten
```

Excursus: missing values

These applied functions all have arguments that control the treatment of missing values (e.g. `NA`). These arguments are there for convenience and readability they're not really necessary. For example, `mean(x, na.rm=TRUE)` is just a convenient and more readable way of expressing `mean(x[!is.na(x)])`.

a. Try this yourself. Create a vector `x` with the following values

```
x <- c(1:5, NA, 10:15, 100)
```

b. Investigate the above command by inspecting the outcome of the following expressions. Comment the R code with your findings.

```
is.na(x)
!is.na(x)
x[!is.na(x)]
```

c. Now run the following code and discuss the results

```
mean(x)
mean(x, na.rm = TRUE)
mean(x[!is.na(x)])
```

Summarising categorical variables

The `type` and `census` variables in our data frame `Prestige` represent categorical/nominal variables (i.e. those that take values in a discrete set of non-ordered categories). In this particular case, `census`is represented by numbers, even though the levels do not correspond to amounts. The variable can be analysed as if it is numeric even if this is not sensible (see that last exercise).

a. The way to summarise categorical variables is in a frequency table, i.e. count the number of times each value occurs:

```
table(Prestige$type)
```

b. To find the modal value of a factor variable the quickest method is to obtain the frequency table and read of which is the largest value. What is the mode of `type`?

c. The R command to find the mode is to obtain which position in the table has the highest frequency:

```
which.max(table(Prestige$type))

# you can also subset the table with that position
table(Prestige$type)[which.max(table(Prestige$type))]
```

d. Add up the counts for `bc`, `prof` and `wc` in the frequency table. How many rows does the data frame have? Notice that there is a difference between these two numbers. This is because there are some `NA`s in `type`. If we want to include these in the table use:

```
table(Prestige$type, useNA = "always")  # you could use also "ifany" instead

# Automatically calculate the sum of the counts per category:
addmargins(
  table(Prestige$type, useNA = "always")
)
```

e. If you want the proportions then we can use the function `prop.table`. Note that `prop.table` expects an object of class `table` as input, which means that most of the time we use the slightly odd expression `prop.table(table(...))`, e.g..

```
prop.table(
  table(Prestige$type,useNA="always")
)
```

The `summary` command

a. One of the most useful functions in R is `summary()`. Try this with

```
summary(Prestige$income)
```

Since *income* is a numeric variable, you get a standard 6-number summary of the data (minimum, lower quartile, median, mean, upper quartile and maximum).

b. Such a summary would be unsuitable for a categorical variable, so:

```
summary(Prestige$type)
```

Notice that the `summary` command automatically reports missing values.

c. The `Prestige` data frame has row names containing the types of job, so

```
row.names(Prestige)[is.na(Prestige$type)]
```

returns the 4 jobs, for which the value of `type` is missing.

The `summary` command can be applied to almost any R object. It is an example of a generic function this means that it does different things depending on the class of the object. At present we're only interested in simple summaries of data objects. Another important use of the command is to summarise the results of fitting a regression model. More on this later in the course.

d. What does summary do if you give it a data frame as an argument? Try it and find out.

Boxplots

A boxplot (a.k.a. *box-and-whisker plot*) is used to compare a numeric variable across several groups in a data frame. The exact format of a box plot differs between packages; in R the box is defined by the lower quartile, median and upper quartile, while the "whiskers" extend to the data point that is no more than 1.5 times the interquartile range from the box. Any observation outside the whiskers is defined as an *extreme value* (or a box plot outlier) and is shown as a point.

a. Run

```
boxplot(prestige ~ type,
        data = Prestige,
        xlab = "job type",
        col = "lightblue")
```

and discuss the result.
Notice the strange use of the tilde sign `~` in `prestige~type`. This is called an *model formula*. Model formulae are common in model fitting, and the details are not important now; you will come across them later in the course. For now, read the formula as `prestige` *depends on* '`type using the data frame`Prestige'.

b. Notice that the alphabetical ordering of the three values in the variable `type` is not ideal. This is because `type` is an ordinal variable and a more sensible ordering of these values would be `bc`, `wc` and `prof`. To override the default ordering, we redefine the factor variable with the levels specified as an argument.

```
Prestige$type <- factor(Prestige$type, levels = c("bc", "wc", "prof"))
boxplot(prestige ~ type,
        data = Prestige,
        xlab = "job type",
        col = "lightblue",
        varwidth = TRUE)
```

c. Check the help for `boxplot`[2] and find out what the argument `varwidth = TRUE` does to the visualization.

Warning: Box plots are often great for investigating or summarising data, but at other times they can be very misleading. Things to be wary of are:

- when there are very many observations in each box, very few observations in each box or the number of observations for each group are quite different. (Now you know what the `varwidth` argument is good for!)
- Sometimes one or two extreme outliers values can compress the boxes into a small part of the vertical axis, making visual interpretation difficult.

Tydying up

a. Make sure your script file has sensible comments. At least one comment for each main section.
b. Try to tidy up your code a bit. If you entered a command just trying something out, but which has little to do with that section then delete the non-essential commands.
c. Save your script file. Use another folder inside your RStudio project folder named *scripts*
d. Leave RStudio and **do not** save the workspace!

[2]e.g. via `help(boxplot)` or pressing F1 when the cursor is on the word `boxplot` in the script window.

Written exercises

This section has two exercises as homework, for you to consolidate what you have learnt in the lecture.

Linear transformation of a mean and median

In Europe temperatures are usually measured in Celsius. In the USA temperatures are usually measured in Fahrenheit.

To convert a temperature from Celsius T_C into Fahrenheit T_F use the linear transformation formula

$$T_F = 1.8 \cdot T_C + 32.$$

Suppose you have data for the maximum temperatures in Berlin between March and May, and have already calculated the mean to be 15.2°C and the median to be 14.5°C.

Calculate the mean median temperatures for the data in Fahrenheit. Why don´t you need the original data to calculate these.

Solution:

$$\text{mean } \overline{T}_F = 1.8 \cdot 15.2 + 32 = 59.36$$
$$\text{median } T_{F,0.5} = 1.8 \cdot 14.5 + 32 = 58.1$$

Box plot by hand

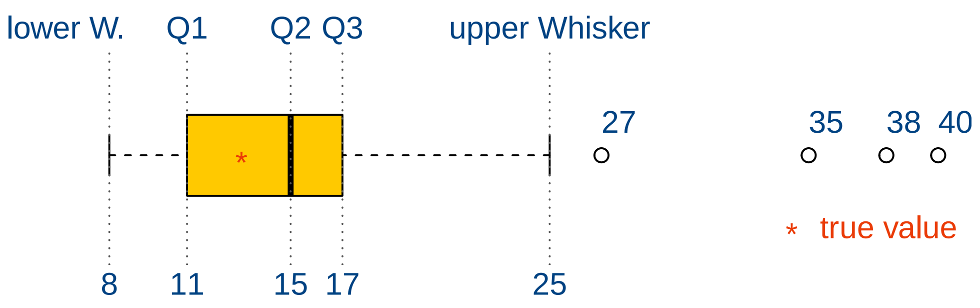
Shortly after introducing the metric system into Australia in the 1960s each of 44 Students was asked to estimate, to the nearest meter, the width of the lecture hall in which they were sitting. The true width was 13.1m. The data were collected by Professor T. Lewis (A Handbook of Small Data Sets by David J. Hand, Fergus Daly, K. McConway, D. Lunn, E. Ostrowski, 1994)

The data are:

8,	9,	10,	10,	10,	10,	10,	10,	11,	11
11,	11,	12,	12,	13,	13,	13,	14,	14,	14
15,	15,	15,	15,	15,	15,	15,	15,	16,	16
16,	17,	17,	17,	17,	18,	18,	20,	22,	25
27,	35,	38,	40.						

a. Obtain the Median, Q_1, Q_3 and the IQR.
b. Find the whisker values. Remember that the whisker values are values contained in the data set.
c. Are there box plot-outliers? If so, determine the values.
d. Draw the box plot by hand.

Boxpot of estimated lecture hall widths


lower W.
Q1
Q2
Q3
upper Whisker
* true value