Exercise Sheet 1 – Data Mining Wirtschaftsinformatik, HTW Berlin

Martin Spott

last revision 14.04.2020

The exercises are about linear regression. We use the data set `01_heights_weights_genders.csv` which you can download from Moodle. It contains the weight (in american pounds) and height (in inches) of 5000 women and 5000 men.

The exercises will help you revise basic concepts of R, do descriptive statistics with visualisations and build linear regression models. Download the data set `01_heights_weights_genders.csv` from Moodle. It contains the weight (in american pounds) and height (in inches) of people. R comes with many functions for data visualisation like `plot`, `hist` and `boxplot`, however I recommend to have a look at other graphing libraries that are more powerful, flexible and produce better looking graphs, in particular *ggplot2*. Install it once with `install.packages("ggplot2")` and load it with `library(ggplot2)` whenever you start a new R session.

A good starting point for using *ggplot2* is the Cookbook for R, specifically Section 8 on Graphs. A link to the more comprehensive book *R Graphics Cookbook* by Winston Chang can be found on the same page.

Other useful graphing packages include *lattice* and *plotly*.

Exercise 1.1

a) Import the data using the function `read.csv()` and assign it to a variable called `weight_df` (a data frame) using the following code fragment. You may have to specify the path of the file.

```
weight_df  <- read.csv("01_heights_weights_genders.csv")
```

b) Explore the data frame `weight_df` using functions like `str()`, `dim()`, `names()`, `head()` and `View()`. How many columns has the data, what are the column names, how many rows, what are the data types etc?

c) Scale the columns for *height* and *weight* to use metric measures *cm* and *kg*: use *1 inch = 2.54 cm* and *1 kg = 2.2 pound*. Hint: A column in a data frame can be addressed using e.g. `weight_df$Height`. Please remember that R distinguishes upper and lower case.

d) Explore the value ranges of the scaled columns *height* and *weight* using `summary()`, box plots and histograms. Distinguish between men and women. Hint: Subsets can be produced using `subset(weight_df, Gender == "Male")` or `weight_df[weight_df$Gender == "Male",]`. Alternatively, `Gender` can be used to define the colour in a plot.

e) Produce a scatter plot with *height* on the x-axis and *weight* on the y-axis. Add descriptive labels to all the axes and give it a title. Again distinguish between men and women using separate plots or use colour.

Exercise 1.2

a) Find out how to build a linear regression model using the function `lm()` (also see the example in b).

b) Build linear models of the data using *height* as input (independent variable) and *weight* as output (dependent variable) for men, women and all. Assign the result to the variables `weight_lm_m`, `weight_lm_f`, `weight_lm_all`. Example:

```
weight_lm_all <- lm(Weight ~ Height, data = weight_df)
```

c) Explore the data structures `weight_lm_x`, e.g. use `names(weight_lm_x)` to learn about the columns (with `x` being one of `m`, `f`, `all`). Refer to the help pages of `lm` to find out what they mean.

d) Add the regression lines to the scatter plots from Exercise 1 e). If you used the basic `plot` function in Exercise 1.1 e) give `abline(weight_lm_x, col="red")` a try and look at the help pages for more information. `ggplot2` offers quite convenient functions for linear regression lines as well.

e) Compare the three regression lines and interpret the differences.

Exercise 1.3

a) Use `summary(weight_lm_x)` to explore the three linear models:
   - their residuals
   - the coefficients of the model with standard error, t- and p-values of the statistical hypothesis test
   - residual standard error and R-squared (squared correlation) Interpret the p-values in terms of rejecting or not rejecting the null hypothesis of a parameter being zero.

b) Look at the confidence intervals of the linear models using `confint()`. Change the confidence levels from the default 0.95, see how the confidence intervals change and explain why (use the parameter `level = ...`).