COMP6235 -- Foundations of Data Science

This is a tutorial on some functions in R for COMP6235 Foundations of Data Science -- part II. This follows on from part I of the tutorial.

Exploring data

In this section I'll briefly mention some R function that can be used to explore data. As an example data set I'll use the file usedcars.csv . Let's first read the file into memory:

```
> usedcars <- read.csv ("usedcars.csv", stringsAsFactors = FALSE)
```

To explore the structure of the data, we can use the "str ()" function which will give us some information about various fields/vectors and data types in the data set.

```
> usedcars <- read.csv ("usedcars.csv", stringsAsFactors = FALSE)
> str (usedcars)
'data.frame':150 obs. of 6 variables:
$ year : int 2011 2011 2011 2011 2012 2010 2011 2010 2011 2010 ...
$ model : chr "SEL" "SEL" "SEL" "SEL" ...
$ price : int 21992 20995 19995 17809 17500 17495 17000 16995 16995
16995 ...
$ mileage : int 7413 10926 7351 11613 8367 25125 27393 21026 32655
36116 ...
$ color : chr "Yellow" "Gray" "Silver" "Gray" ...
$ transmission: chr "AUTO" "AUTO" "AUTO" "AUTO" ...
```

We can gather the following:
(1) 150 obs -> the data set contains 150 records (one would often say "n=150", i.e. we have 150 observations).
(2) There are 6 features in the data set (year, model, mileage, color, transmission)
(3) We see of which data type each feature is.
(4) The following entries are the first 4 data entries for each feature.

Exploring numeric variables

We can use the summary command to get a quick overview of some summary stats, e.g.:

```
> summary (usedcars$year)
Min. 1st Qu. Median Mean 3rd Qu. Max.
2000 2008 2009 2009 2010 2012
```

which gives some quick information about min/max/mean/median values and interquartile ranges (see my lecture for more information). The information provided can also be explored with separate commands, e.g.

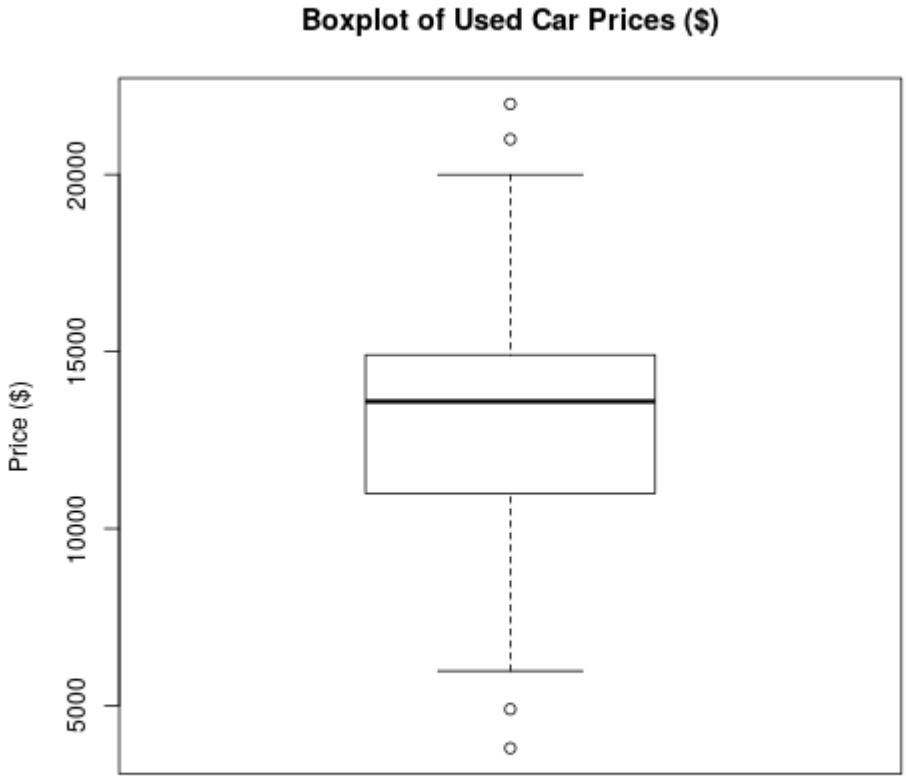
```
> mean (usedcars$year)
> median (usedcars$year)
```

Similarly, functions like "min (), max (), range (), IQR ()" etc. can be used. Explore some of them yourself.

Visualizing numeric variables

A typical way of visualizing the distribution of a numerical variable is a boxplot which gives information about the distribution of this variable in the form of indicating Q1, Q2 (the median), Q3 (for the drawn box in the middle) and the "whiskers" to either indicate min/max or 1.5 times IQR below Q1/above Q3. Values outside of this range are considered outliers and drawn as dots. For example:

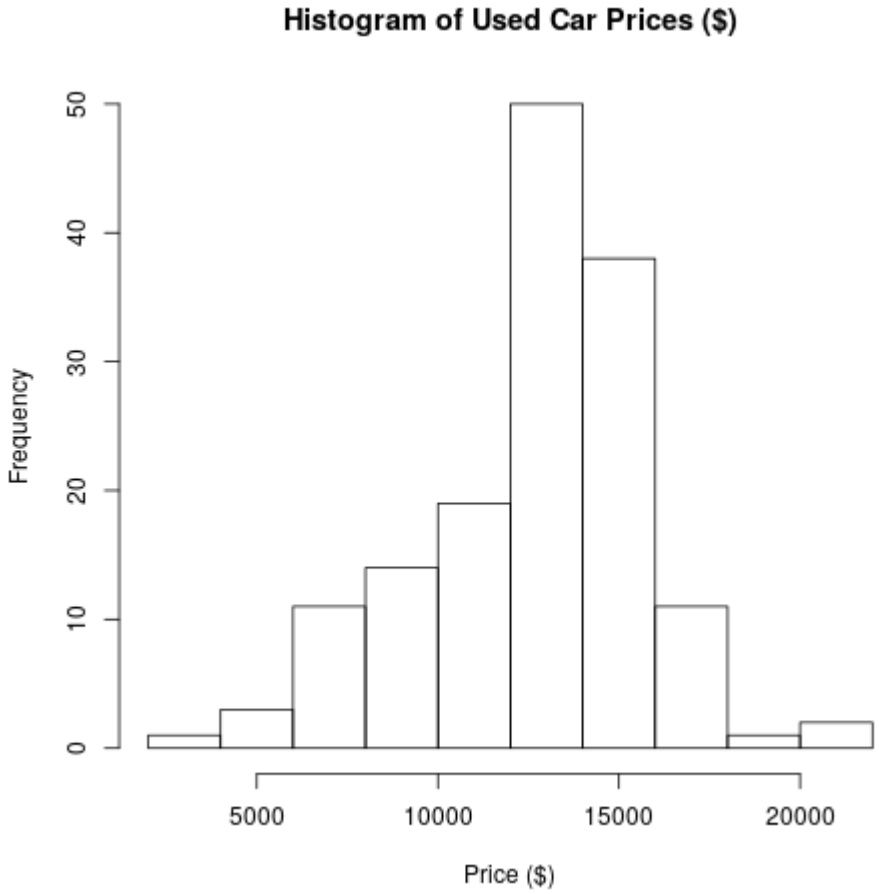
```
> boxplot(usedcars$price, mean="Boxplot of Used Car Prices", ylab="Price ($)")
```



Explore "?boxplot" to see how you can further customize the appearance of the plot.

Another way of visualizing data is via histograms -- see my Lecture on the topic. In R we can generate histograms via the "hist ()" function, e.g.

> hist (usedcars$price, main="Histogram of Used Car Prices", xlab="Price ($)")

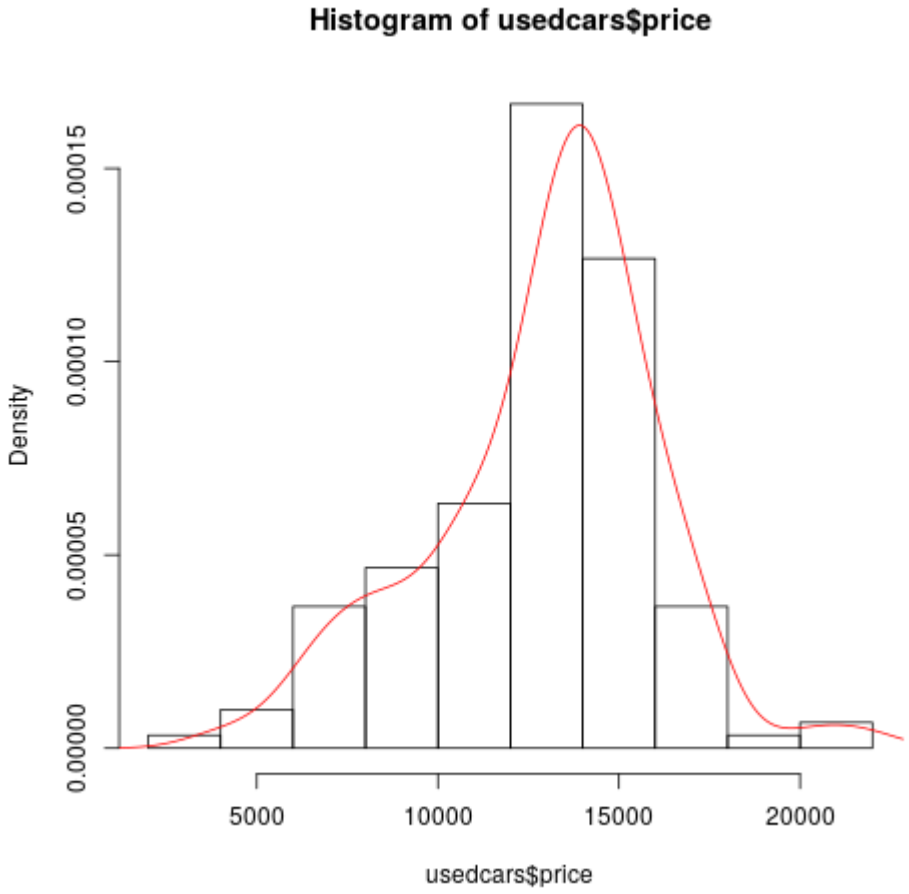


What about a comparison of the histogram to a kernel density estimate of the usedcars$price distribution? For this purpose, we'd first normalize the distribution:

> hist (usedcars$price, main="Histogram of Used Car Prices", xlab="Price ($)", probability=T)

and then add another plot using the lines command:

> lines (density (usedcars$price), col="red")



R also provides a number of functions to calculate the spread of distributions, e.g. "var (), sd (), or IQR ()".

Generating plots in pdf/png format

You can also use R to produce figures in pdf (or other formats) for reports/papers etc. The way to do this is to first set a graphics device, e.g.

> pdf ("boxpot.pdf")

where you set the output for the next plot command. You then simply repeat the plot command

> boxplot(usedcars$price, mean="Boxplot of Used Car Prices", ylab="Price ($)")

R will have generated a file "boxpot.pdf" but may not yet have written the

contents of this file (which happens when buffers are flushed). To enforce writing contents to the file, you can use the command "graphics.off ()". Graphics devices are available for a number of other formats, e.g. "png (filename)", etc. Explore how to customize them yourself.

IMPORTANT: in reports/papers etc. avoid inserting screenshots. When you use screenshots your files become huge, graphics don't scale very well (so always use vector graphics whenever possible) and the output does not look very professional.

Exploring categorical variables

Categorical data are often examined using tables -- these are basically frequency counts for the occurence of the categories. For instance, to examine the "color" field of the usedcars dataset, we could use:

> table (usedcars$color)
which produces the output:

Black Blue Gold Gray Green Red Silver White Yellow
35 17 1 16 5 25 32 16 3

i.e. we have 35 black cars, 17 blue cars, etc. We could also get proportions by using the prop.table () command and applying it to a table, i.e.:

> mytable <- table (usedcars$color)
> prop.table (mytable)
will now output proportions of cars belonging to the respective category

Black Blue Gold Gray Green Red Silver White Yellow
0.233333333 0.113333333 0.006666667 0.106666667 0.033333333
0.166666667 0.213333333 0.106666667 0.020000000

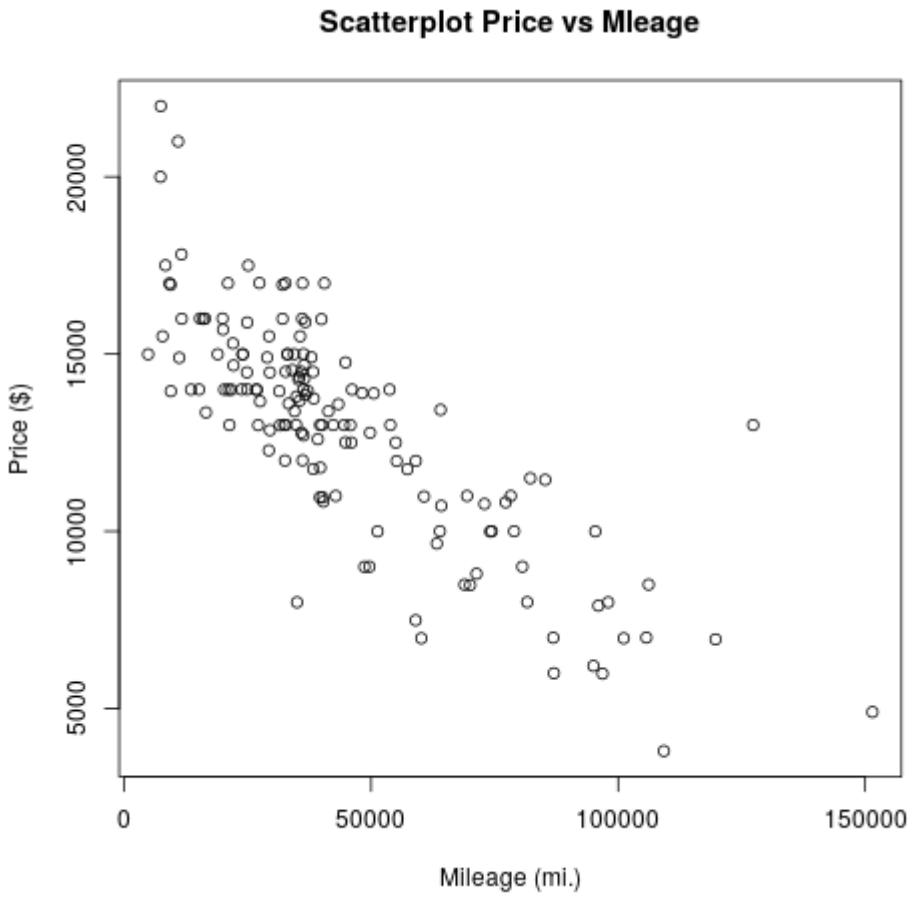
Relationships between variables

So far we have been examining one variable at a time, i.e. we have only analyzed univariate statistics. Often we are also interested in questions regarding the relationship between variables. In case of two variables, we'd have a look at bivariate relationships, and so on.

Scatterplots

are diagrams that visualize bivariate relationships, we basically plot one feature of the data set vs. another feature. For instance, we might be interested in possible relationships between price and mileage in the usedcars data set. Typically, when plotting such relationships, one will assign the y-variable to the dependent variable, e.g.:

> plot (x=usedcars$mileage, y=usedcars$price, main="Scatterplot Price vs Mleage", xlab="Mileage (mi.)", ylab="Price ($)")



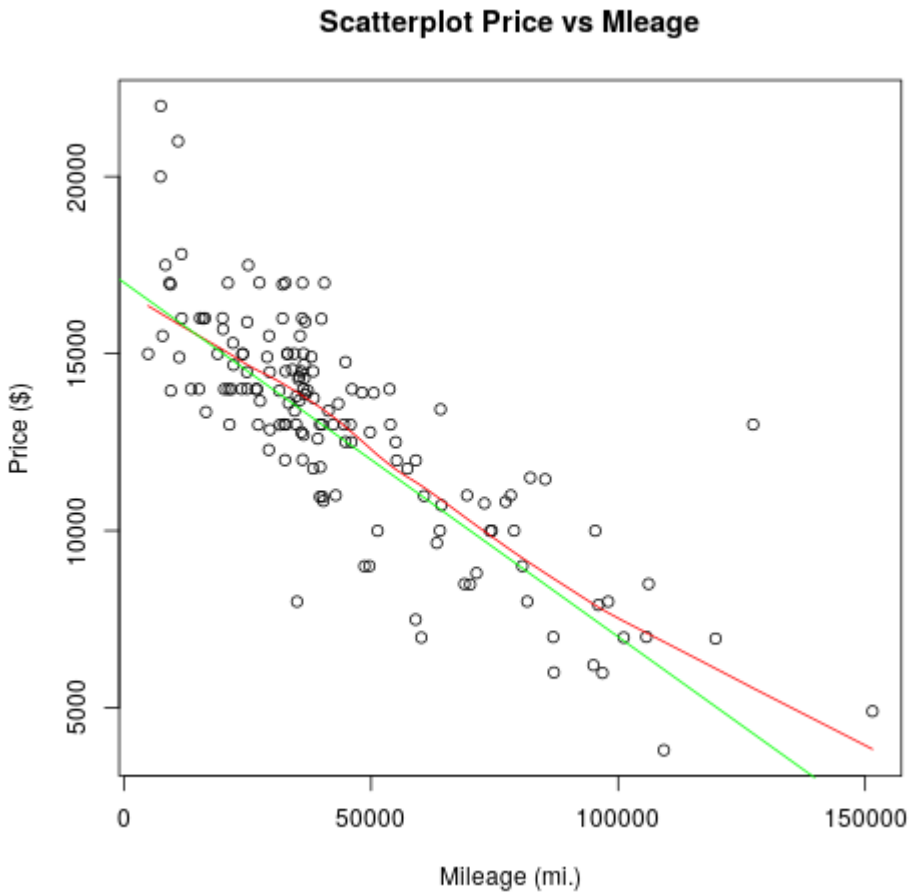
Note, that alternatively, we could have used the "~" sign in the plot function to plot y vs. x, i.e.
> plot (usedcars$price~usedcars$mileage, main="Scatterplot Price vs Mleage", xlab="Mileage (mi.)", ylab="Price ($)", type="p")

would have produced the same plot with a bit less typing. I also specified that data points should be plotted as points by giving the option type="p" -- one could have chosen type="l" for lines, type="b" for both are some others (used ?plot to explore).
The plot seems to indicate a negative relationship between mileage and price

of car, not very surprising. Suppose I want to indicate this relationship in the figure? I could, e.g., do this by adding a straight line:
> abline (17000, -.1, col="green")

which will add a green line with intercept $17000 and slope -.1$/mile to the graph. This is probably not the best indicator of the relationship between price and mileage. Alternatively, we could also use the "lines" command to add all sorts of different lines to the graph. For instance, we could combine this with the "lowess" command (technically: lowess produces a locally weighted regression line) to generate a curved line that indicates the relationship:

> lines (lowess (usedcars$price ~ usedcars$mileage), col="red")



Using the lines command, or the similar points command, it is possible to add data from additional variables or even from different data sets to the same graph. If you are planning to do this, you might need to use the xlim and ylim options to the plot command to set an appropriate x or y range for the initial plot. For example, plot

(usedcars$price~usedcars$mileage,ylim=range(0:5000)) would force the y-axis to a range of zero to 5000.

Let us come back to the used cars data set and have a look at relationships between categorical variables. For instance, we might be interested in questions like whether the type of transmission (manual or automatic) has an effect on price. To test this, we could simply compare mean values of prices of cars with automatic or manual transmission. One way to approach this question is to first select cars with a specific type of transmission and than investigate summary statistics, e.g. we could use:

```
> summary(usedcars[usedcars$transmission=="MANUAL",])
> summary(usedcars[usedcars$transmission=="AUTOMATIC",])
```

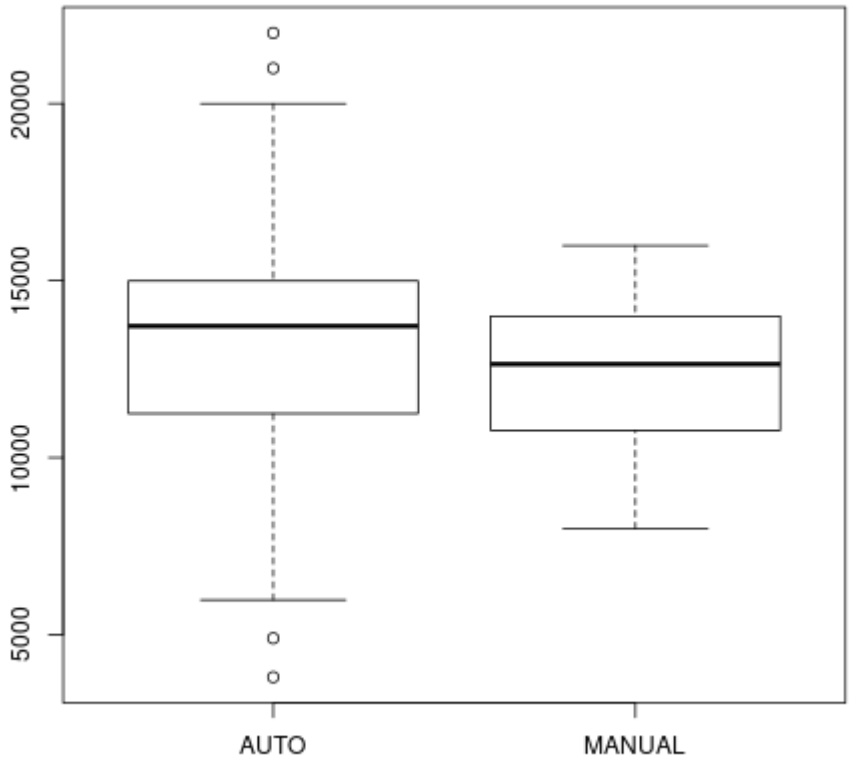
and would find values of $12200 and $13093, respectively. This can also be done much quicker using the tapply command:

```
> tapply (usedcars$price, usedcars$transmission, summary)
```

which tells R to split the field usedcars$price of the usedcars data frame by the variable transmission and then apply the command summary to the result.

We might now wonder if this allows us to conclude that cars with manual transmission are on average cheaper than cars with automatic transmission? We will discuss this type of problem in more detail in the lectures. For the moment, we would hypothesize that the distribution of car prices around the mean values would surely affect our verdict. To have a look at this, we should produce suitable boxplots that visualize both distributions in the same graph. We can do this via

```
>boxplot (usedcars$price ~ usedcars$transmission)
```



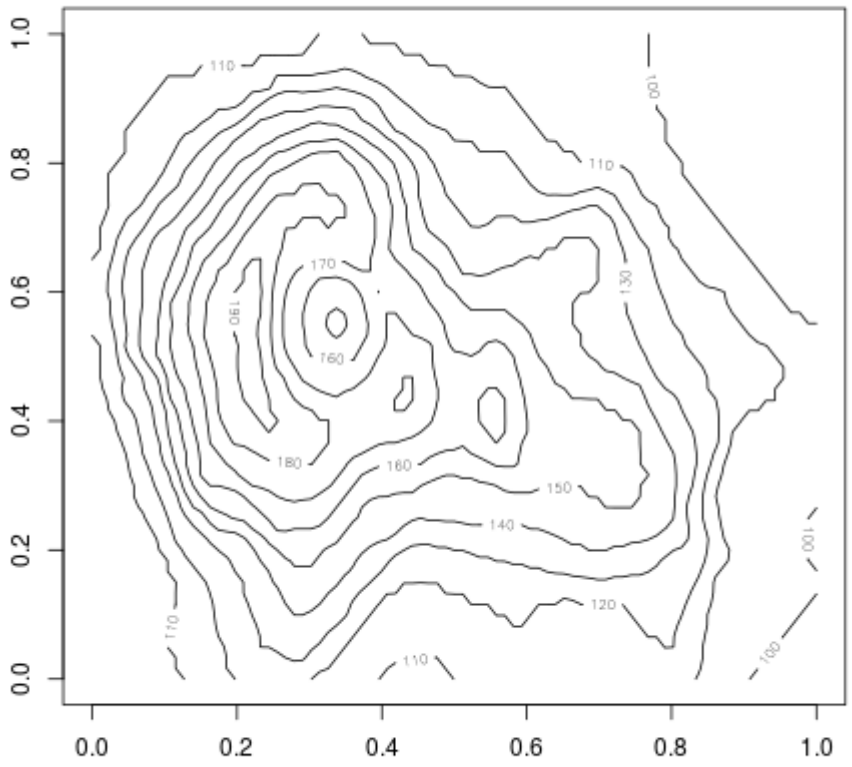
We see that there is quite a bit of overlap between both distributions, i.e. there are many cars with manual transmission that are more expensive than cars with automatic transmission. We might have to be a bit more careful with our statistics here -- which factors would we have to consider?

Heat maps and contour plots

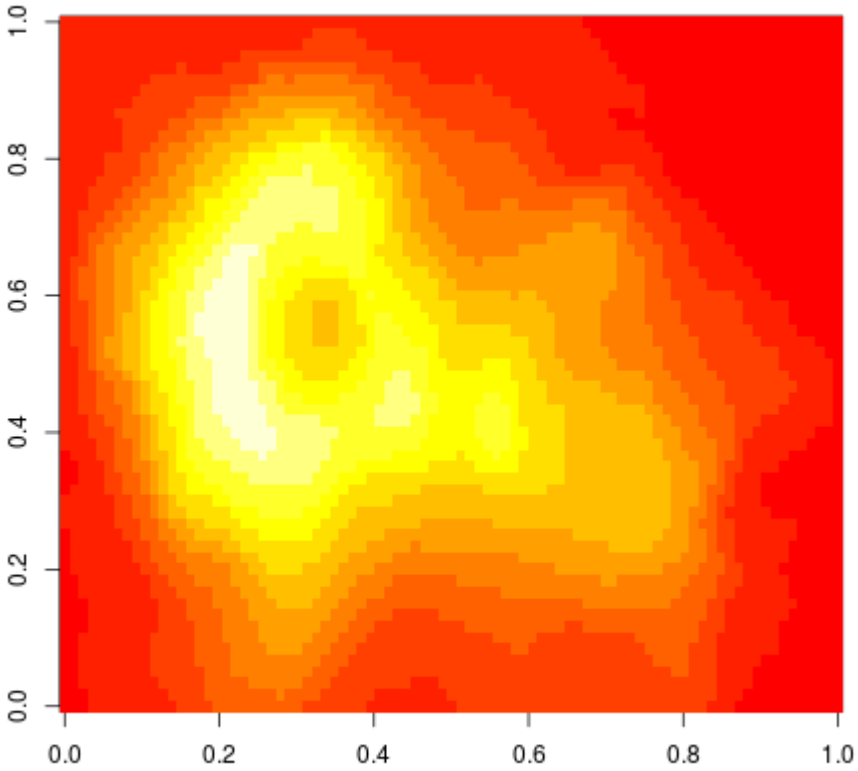
Sometimes you want to look at how one variable is related to two others simultaneously. For example, in a demographic data set you might want to explore income as a function of both IQ and education. Contour plots, heatmaps, and 3D surface plots are the appropriate tools for this, and they are all possible in R.
These types of plots expect the data to be in a slightly different format to what we have seen so far. Instead of a simple rectangular file, we expact the data to be in the form of a vector of x values, a vector of y-values, and a matrix of the z values we are going to plot. If you have your own data set which is arranged in rows and columns already, you can have R read it in as a matrix using the following:
> matrixData <- as.matrix(read.table ("file.dat"))

For now we will just use one of R's built-in data sets, the "volcano" data set, which is already in the correct format. We can load it using:
> data (volcano)

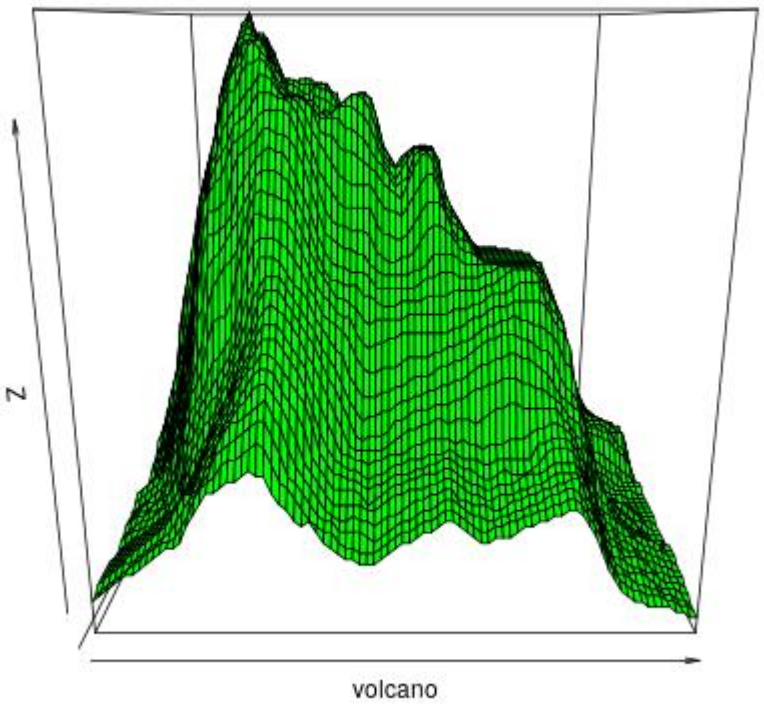
For a contour plot use:
>contour (volcano)



For a heatmap use:
>image (volcano)



For a 3D plot use:
>persp (volcano, col="green")



where I have set the color option to green in the last one. All of these commands come with a plethora of options, explore at your own convenience using the help function. If you use latex as a text processing system, pdf files should import best. For word-based processors, png-files should be most convenient to import. For more information and to explore other options, try help(Devices), help(postscript), help(pdf), help(png), etc.

Quantifying relationships between variables

Correlation coefficients are a basic statistical tools for describing relationships between variables -- we will talk about the various types of correlation coefficients in more detail in the lectures later. For now, a correlation coefficient is a number that ranges between -1 and +1, and describes the strength and direction of the linear relationship between two variables.

- A positive value correlation means a linear relationship in the positive direction, i.e. as the first variable gets bigger, the second gets bigger as well. Think of the relationship between height and weight -- we expect that, on average, somebody of large height also has larger

weight, but this is not always the case. The result would be a positive correlation coefficient between zero and one.

- A correlation of zero, or close to zero, indicates the absence of a (linear) relationship. This is another way of saying knowledge of variable one tells us nothing about variable two. E.g. the correlation between peoples IQ and their house numbers is surely close to zero. (Something to be aware of is that you can have close to zero correlation even though there might be a strong non-linear correlation in your data!)
- A negative correlation indicates a linear relationship for which one variable decreases as the other increases. An example of a moderate negative correlation might be the relationship between the age of a driver and the speed of vehicle on the motorway.

Although there are no hard-and-fast rules about what counts as a high or low correlation coefficient, a very approximate guide to these numbers might be as follows:

- 0...0.3: Weak relationship; may be an artefact of the data set and in fact there is no significant relationship at all
- 0.3-0.6: Moderate relationship; you might be on to something, or you might not
- 0.6-0.9: Strong relationship; you can be confident that the two variables are connected in some way.
- 0.9-1.0: Very strong relationship, these two variables are virtually measuring the same thing.

We will talk about the significance of correlations in a bit more detail in a later lecture.
Let's come back to our used car example. Our previous plots indicated the presence of a negative relationship between price and mileage, i.e. as mileage gets larger, we typically find lower prices. We can quantify this with a correlation coefficient using the R-command "cor". However, cor requires a numeric data frame as input, so I first need to transform the usedcars data:

```
> df <- data.frame (usedcars$mileage, usedcars$price)
> cor (df)

usedcars.mileage usedcars.price
usedcars.mileage 1.0000000 -0.8061494
usedcars.price -0.8061494 1.0000000
```

gives a matrix of correlations between all columns of df. Here we note the presence of a strong negative relationship. However, when plotting the relationship we noticed that it is not exactly a linear relationship, we'll see

how to quantify this in a bit. By default cor uses a Pearson correlation coefficient, if relationships are monotonic, but not exactly linear, we might try spearman's rho or kendall's correlation coefficient instead.

When exploring the links between variables in a data set that has many variables, the matrix of correlations can be an excellent place to start. To get some experience of this, let us use one of R's inbuilt data sets, describing the relationship between various social measures across different regions of Switzerland. Type:

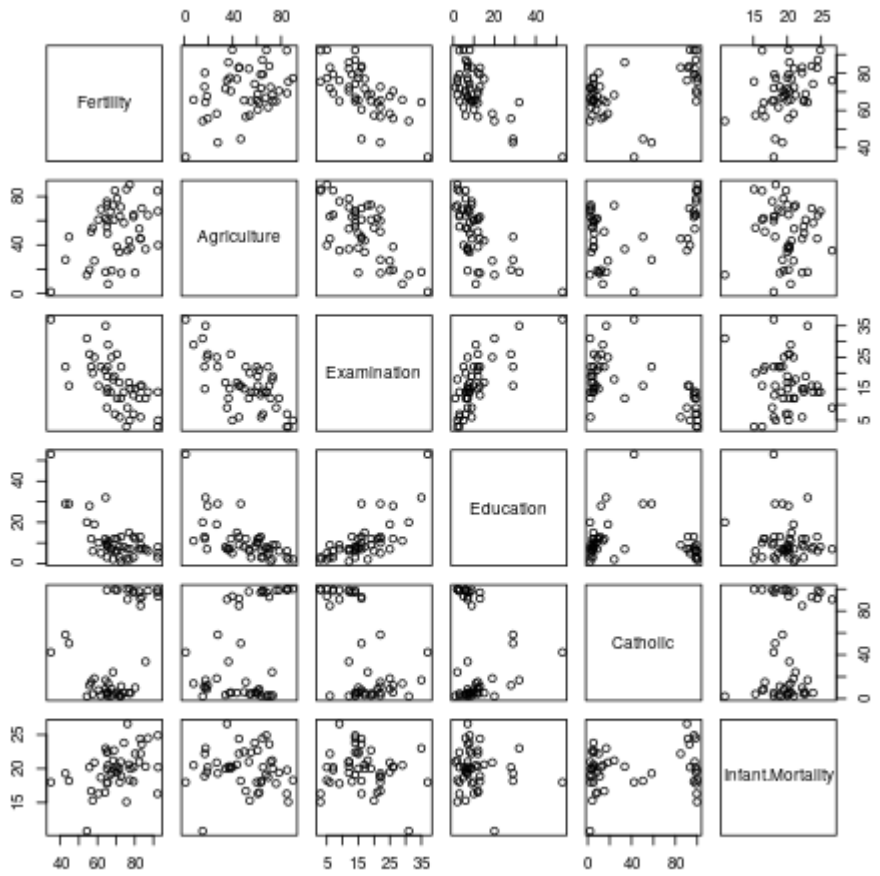
```
> data (swiss)
```

to load the data file. (Try data () to see which other built-in data sets are available if you would like to experiment with them.)

Typing cor (swiss) will give you a matrix of correlations between the six regional variables in the swiss data set: fertility (i.e. birth rate), level of agriculture, average exam results, educational level, prop. of Catholics in the local population, and infant mortality. Have a look for strong and weak correlations. Do the strong correlations make sense?

R provides a visual analogue of the correlation matrix. Try the command pairs (swiss):

```
> pairs (swiss)
```



This plots a scatterplot for the relationship between each pair of variables in the data. Look for a graph that shows a strong linear relationship and then check the corresponding correlation in the output of cor (swiss). This is a good way to get a feel for what various levels of correlation actually mean in practice.