Series 6

1. Should a space shuttle pilot use the autolander or land manually given information on weather and the state of the space shuttle (6 variables)?
Some experts discussed the "correct" decision for 256 possible settings (all possible settings given the discrete variables). Your task in this excercise is to transform the knowledge of these experts into a simple diagram that a pilot can easily use when landing.

a) Load the package `MASS`. Have a look at the data set `shuttle`.

b) For fitting and visualizing a tree we will need the functions `rpart, plot, text` and `print` in the package `rpart`. Have a look at the corresponding help files (`plot.rpart, ?text.rpart`, etc.)

c) Train a tree using rpart. Plot the result. Also look at a text representation of the tree using the `print` function and compare it with the plot. Note that in the plot the levels of all factors are abbreviated with a,b,c,etc. For example, "vis=a" means: If the variable "vis" takes on its first level (**in this case R uses alphabetic ordering of levels, check `str(dataset)` to check ordering of the levels**), go to the left, otherwise go to the right. Can you confirm this by looking at the textual representation?

d) How many cases are misclassified?

e) Should the autolander be used in a situation with `vis=yes` and `error=MM`? Solve by looking at the plot and at the text representation.

f) In this example, we don't want to optimize for prediction, since all possible situations were already enumerated by the experts. Thus, create a tree that perfectly describes the opinion of the experts (i.e., don't use pruning). Create a plot and a text representation of the resulting tree.

g) Create a postscript file of the tree you found.

2. Suppose there was a crime scene and some small glass shards are found on the floor. The investigor would like to know, where this glass comes from. For situations like these, laboratories can analyze the physical and chemical properties of the glass shards and make an educated guess when saying where such glass is typically found. In this excercise, we will try to pack the knowledge of 214 analyses of glass shards into a simple tree that can be printed and posted in the laboratory. We are mainly interested in the prediction performance, so we will use cost-complexity pruning.

a) Load package `MASS` and read the help file of data set `fgl`.

b) Fit a tree with the default settings. Plot it and look at the text representation. Load the package `rpart` first.

c) Fit a tree that fits the 214 analyses perfectly. Again plot it.

d) Although the perfectly fitting tree explains all past observations well, it is not necessarily optimal for prediction (due to overfitting). Investigate the crossvalidation error in relation to the complexity parameter.

e) Choose a complexity parameter that seems reasonable to you and prune the model accordingly.

3. In this excercise, we try to detect spam given some features of the email.

a) Have a look at the data set "spam" in the package "ElemStatLearn"

b) Fit a random Forest (`library(randomForest)`) with the default settings. (Use seed 123 in order to reproduce the solution). Be patient: this may take several seconds.

c) Plot the error rate vs. the number of fitted trees. How many trees are necessary? Refit the model with the chosen number of trees. How long does it take now?

d) Have a look at the output. What error rate do you expect for new predictions (OOB error rate)? What is the error rate in the 'spam'-class?

e) Suppose, we get a new email and want to predict the spam label. For simplicity, we refit the Random Forest on 2601 randomly chosen emails and save the remaining 2000 emails as test set. How does the OOB error compare with the error on the test set? (use ntree = 100, and set.seed = 123)

f) Suppose we don't want to compute all variables for each new incoming mail, but only use the best 5. Which 5 variables should we choose? Compare the OOB error using all variables, the best 5 and the worst 5 (according to decrease in accuracy; use ntree = 100 and seed = 123).

4. Random Forest fits many trees while randomly changing the samples and the selection of variables for each tree a little bit. At first sight, this seems to be stupid; but it has the effect of de-correlating the trees, so that the average prediction of all trees becomes (oftentimes) more precise than each individual tree. In this excercise, we will compare the prediction performance of a single tree and random forest.

a) Load the data on forensic glass again.

b) Fit a tree using the default settings of rpart.

c) Fit a Random Forest using the default settings of function 'randomForest' in package 'randomForest'.

d) Have a look at the output of the Random Forest. Which error rate would you expect for new samples?

e) In order to make a fair comparison of the two methods, we will do a leave-one-out cross-validation on the forensic glass data set and compute the misclassification rates for both methods. (The computation might take a few minutes)

f) Random Forest has usually a better prediction performance than single trees. Do we pay any price for this advantage?

g) If you are curious: How do Random Forest and the Tree compare to LDA in this example?

Preliminary discussion: 27.05.13.

Deadline: No hand-in.