Classification

Introduction

In this lab, you will first perform classification on a number of (increasingly difficult) two-class problems in 2D. This allows you to visualize the samples and the decision boundaries, which helps you to form an understanding of the intricacies of the various classifiers. At the end, you will get to apply the same routines on a "real" biological dataset, for which visualization is often impossible - note that in *p* dimensions, a decision boundary is a *p*-1 dimensional surface!

To help you visualize decision boundaries in 2D, we have created an auxiliary function plot_classifier(data,predictor,par), where predictor is one of "knn", "lr", "tree" or "forest" and par is *k* (for *k*-nearest neighbour) or *ntree* (for a random forest). You can have a look at the function (if you feel adventurous), but at least source it and load the other required packages:

```
> source("http://www.bioinformatics.nl/bda/plot_classifier.R")
> library(ggplot2)
> library(chemometrics)
> library(class)
> library(tree)
> library(randomForest)
```

k-nearest neighbour classification

The first classifier to work with is the *k*-nearest neighbour method.

a. Load a simple 2D dataset:

```
> simple <-
read.table(url("http://www.bioinformatics.nl/bda/simple.txt"),
+                        sep="\t",header=FALSE)
```

Visualize it. How many samples does it have?

b. Use plot_classifier() to try the *k*-nearest neighbour classifier, for various values of *k*. Which value of *k* gives best results, by eye?

c. Try cross-validation to find the optimal *k*, using knnEval:

```
> ntrain <- round(nrow(simple)*3/4)
> train <- sample(1:nrow(simple), ntrain)
> res <- knnEval(simple[,1:2],simple[,3],train,kfold=10,
+                 knnvec=seq(1,25,1),legpos="bottomright")
```

d. Repeat b.-c. for the datasets "hard.txt", "banana.txt" and "spirals.txt", found on the same server.

e. Does scaling have an influence on the decision boundary for the hard dataset for the optimal *k*? Scale the data as follows:

```
> nhard <- hard; nhard[,1:2] <- scale(hard[,1:2])
```

f. Find the optimal *k* for the scaled hard data and calculate the accuracy of the results, both on the training set and the test set (use the code given in the book or on the slides). Also inspect the confusion matrices.

Other classifiers

a. Use plot_classifier() to plot the logistic regression decision boundary for the simple dataset. What do you notice?

b. Do the same for the hard, banana and spirals datasets. What do you see here? Is logistic regression sensitive to scaling?

c. Repeat a.-b. for the decision tree and the random forest. What do you notice?

Multiclass problems

a. Train the three classifiers that can handle multiple classes, on 100 samples taken from the iris dataset; use data("iris"") to load it. Note that you cannot use the plot_classifier() function anymore, as it only works on 2D data; rather, use the code in the book/slides. Generate confusion matrices on the remaining test set. Use *k*=1 for the *k*-nearest neighbour classifier and the default settings for the random forest. Do the results concur with what you would expect of this dataset?

b. Plot the tree you found above; what do you notice? Find the optimal size to prune the tree back to using cross-validation (use the code in the book/slides), and check whether this issue still occurs with the pruned tree.

Gene expression data

a. In this final part, you will train a classifier on a real gene expression dataset, of 128 T-cell and B-cell leukemias measured on an Affymetrix HG-U95Av2 microarray. More details can be found in Chiaretti *et al*., 2004 (PubMed ID 14684422). The goal is to see whether we can make sense of the features used for classification.

As the dataset is too large to work with in a reasonable amount of time, you will first select a (very) small subset of probesets that show variation:

```
> source("https://bioconductor.org/biocLite.R")
> biocLite("ALL")

> library(ALL)
> data(ALL)
> ALLdata <- data.frame(t(exprs(ALL)))
> mads <- apply(ALLdata,2,mad)
> ind = order(mads,decreasing=TRUE)
> n = 10
> data <- ALLdata[,ind[1:n]]
> data$label <- as.factor(t(substr(ALL@phenoData@data$BT,1,1)))
```

b. Select a training set containing 75% of the samples and train a logistic regression classifier. What is the performance on the training set and on the test set? Hint: translate the predictions into labels using: ifelse(predictions>0.5,"T","B").

c. Which probesets are assigned the largest weight in the model? Use GeneAnnot to learn about these probesets (remove the X at the first position!). Does it make sense that these are useful features?

Optional

d. What is the smallest number of probesets (n) you can use in logistic regression and still get the same test set performance?

e. Train a decision tree on the dataset with 10 features. Does it perform equally well? Plot the tree; does it support your findings for question d.?

f. Experiment with *k*-nearest neighbour classification; use cross-validation to find the optimal value for *k*.