Week 5 Practical (unassessed): Nonlinear Classification

HT2016 Statistical Data Mining and Machine Learning

In this exercise, we will consider various classification methods on a dataset with a nonlinear separation between the two classes.

You will need the following libraries.

```
library(MASS)
library(ROCR)
library(kernlab)
```

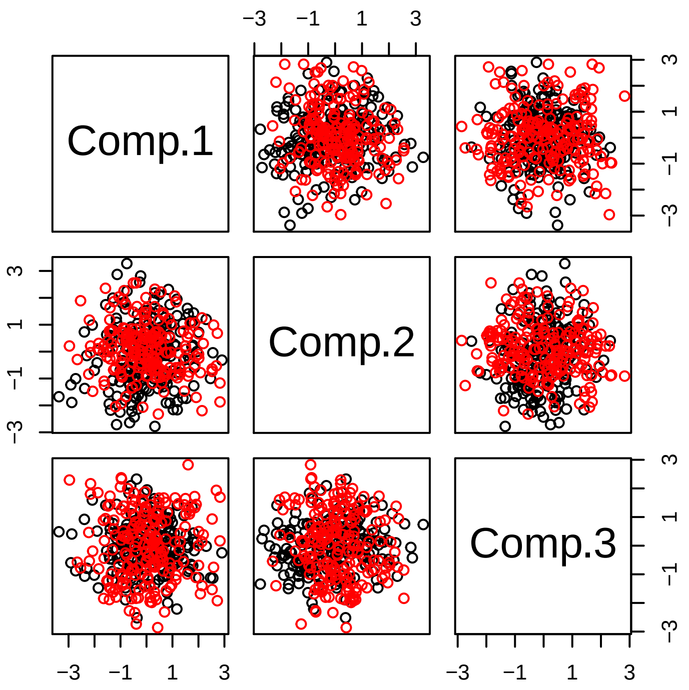
Download `week5practical.data` from http://www.stats.ox.ac.uk/~sejdinov/sdmml/data and load it with:

```
xy<-read.table("http://www.stats.ox.ac.uk/~sejdinov/sdmml/data/week5practical.data")
x<-data.matrix(xy[,1:10])
y<-factor(xy[,11])
n<-dim(x)[1]
p<-dim(x)[2]
```

This data consists of $n = 400$ observations in $p = 10$ dimensions. The labels belong to $\mathcal{Y} = \{1, 2\}$. Try some exploratory analysis first and see if you can spot anything that differentiates the two classes. This dataset was created so that the classification is easy when some hidden information is available but without it, all linear methods will do poorly. We will later see why.

PCA and LDA

Run principal components analysis on `x` and create the pairwise plot of the projections onto the first three principal components. The output should look like this:



So, not much going on there – PCA did not recover any interesting differences between the two classes.

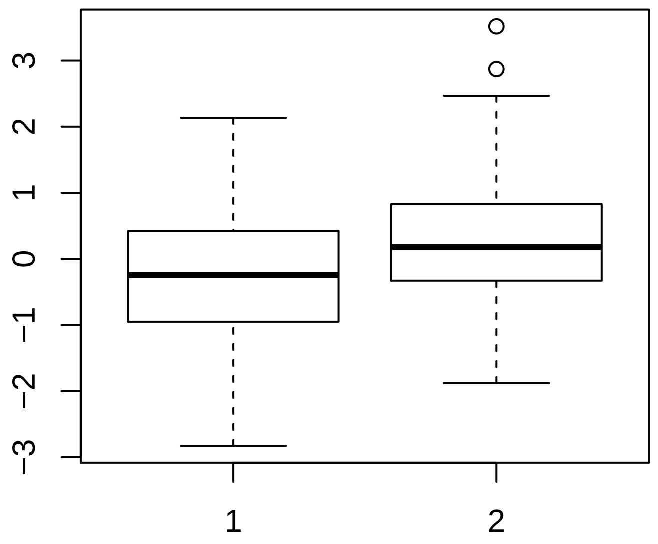
Let us try some supervised approaches. To do that we will first split the data into the training set (80% of the data) and the testing set (the rest). The classifiers will be learned on the training set, and we will estimate the classification accuracy on the testing set (unseen in training).

```
ntrain <- round(n*0.8) # number of training examples
tindex <- sample(n,ntrain) # indices of training examples
xtrain <- x[tindex,]
ytrain <- y[tindex]
xtest <- x[-tindex,]
ytest <- y[-tindex]
```

First, train the LDA model on the training data:

```
xtrain.lda<-lda(xtrain,ytrain)
```

The projections defined by LDA will be to a one-dimensional subspace on this data (since the number of classes is 2). Make a boxplot contrasting these projections for the two classes. It should look something like this:



Thus, LDA did not do anything useful either. It also does a poor job on predicting the class label on the testing set:

```
ypred = predict(xtrain.lda,xtest)$class
table(ytest,ypred)
```

```
##      ypred
## ytest  1  2
##     1 20 17
##     2 26 17
```

Support Vector Machines

Now let us have a look at Support Vector Machines. There are several implementations out there, but we will use the one provided in package `kernlab`. Have a look at the help with

```r
?ksvm
```

Let us first try a linear C-SVM and have a look at the output.

```r
svp <- ksvm(xtrain,ytrain,kernel="vanilladot",type="C-svc",cross=5)
```

```
##  Setting default kernel parameters
```

```r
svp
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 288
##
## Objective Function Value : -281.8252
## Training error : 0.40625
## Cross validation error : 0.509375
```

That did not go well either: training error and 5-fold cross-validation error are not better than chance. Now, try the option `kernel="rbfdot"` that uses the Gaussian RBF kernel defined as

$$k(x, x') = \exp(-\sigma\|x - x'\|^2).$$

A useful heuristic to choose σ is implemented for you by default in `kernlab`. Compute 5-fold cross-validation errors for parameters C taking values $2^{-3}, 2^{-2}, \ldots, 2^{10}$ and select C that gives smallest cross-validation error. Then check the performance on the testing set. What do you conclude?

ROC curves

Now, let us plot some Receiver Operating Characteristic (ROC) curves, and also compute the area under the ROC curve (AUC) that indicate the performance of the classifiers as the discrimination threshold is varied (higher AUC is better). You can use functions `prediction` and `performance` in `ROCR` package for this.

```r
ypredscore = predict(svp,xtest,type="decision")
pred <- prediction(ypredscore,ytest)
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf,col='red')
abline(a=0,b=1)
auc <- performance(pred,measure="auc")@y.values
auc
```

Epilogue

To understand better what happened, download the orthonormal matrix V which was used to hide a simple nonlinear decision boundary in a 10-dimensional space. Repeat the LDA classification after adding the appropriate additional feature into x.

```
V<-read.table("http://www.stats.ox.ac.uk/~sejdinov/sdmml/data/week5practicalV.data")
x_rotated<-x%*%t(V)
plot(x_rotated[,1],x_rotated[,10],col=y)
```

UCI Ionosphere Data

Time permitting, repeat the whole pipeline (PCA,LDA,SVM) on the UCI Ionosphere data (https://archive.ics.uci.edu/ml/datasets/Ionosphere}. Use 60% of the data for training and the rest for testing.

```
ion<-read.table('~/ionosphere.data',sep=',')
x<-scale(ion[,3:34])
y<-factor(ion[,35])
n<-dim(x)[1]
p<-dim(x)[2]
```