# INFO-F-528 Machine learning methods for bioinformatics
# Exercise 4: classification

Gianluca Bontempi
Machine Learning Group, Computer Science Department,
Université Libre de Bruxelles

## 1 Binary classification by using regression

We can solve a binary classification problem using a regression approach, i.e. by using neural networks. In a classification problems the output variable is discrete and not necessarily ordered (i.e. the type of animal, the shape of a character). If the output variable is binary (for example TRUE/FALSE), the problem can be solved by replacing the two possibilities by 0 and 1 and then using a regression model (see Section 8.2 of the handbook)

#### Exercises

The script `classif.R` applies regression techniques to a binary classification task where the data set `BreastCancer.Rdata` made available by the R library `mlbench`. Such dataset contains the measures of nine clinical variables (e.g. cell size, cell shape) and the associated tumor class (benign, coded as 1, or malign, coded as 0) for 683 patients collected during a clinical study in the University of Wisconsin hospital.

The script is very similar to the one used for regression. The two differences are: first, the error computation determines now the proportion of wrong classifications. Second, a logistic function is used instead of a linear function for the neurons in the hidden layer.

- Install the packages `tree`, `lazy` and `e1071` which implement decision tree, local modeling and SVM models. Uncomment the lines which use these models in the script `classif.R` and compare the performance of the different methods on the data set `BreastCancer.Rdata`.
- Trace a ROC curve for the different classifiers.

## 1.1 The K-NN classifier

Suppose a training set is available and that the classification is required for a $[1, n]$ vector. Henceafter we will call this input vector a *query* point. The classification procedure of a K-NN classifier can be summarized in these steps:

1. Compute the distance between the query and the training samples according to a predefined metric.
2. Rank the neighbors on the basis of their distance to the query. *Hint: use the R function* `sort`.
3. Select a subset of the $K$ nearest neighbors. Each of these neighbors has an associated class.
4. Return the class which characterizes the majority of the $K$ nearest neighbors. *Hint: use the R function* `which.max`.

**Exercise**

1. Write a R function `KNN` which takes as inputs
   - an input matrix $X$ of size $N \times n$
   - an output class vector $Y$ of size $N \times 1$
   - a query input vector of size $1 \times n$
   - the integer $K$

   and returns the class prediction according to the K-nearest neighbor with an Euclidean distance metric.

### 1.1.1 Validation of the classifier

We have built a simple classifier able to return a prediction of the class once a query is provided. But how accurate is it? Can we be sure that its prediction will be accurate enough, before testing it on new patients? A validation of the classifier is therefore required. But in our context the validation can use only the available data.

**Exercises**

Test the KNN classifier model on the BreastCancer dataset.

1. Write a script `KNNvalid.R` which, given a $K$,
   - partitions the BreastCancer dataset in a training set containing two-thirds of the data and a test set,
   - validates the accuracy of the K-NN classifier by training-and-test,
   - performs a 10-fold cross-validation to validate the accuracy of the K-NN classifier,

- performs a leave-one-out to validate the accuracy of the K-NN classifier.

2. Using the script `KNNvalid.R`,

   - select which among the following options

     (a) $K = 1$
     (b) $K = 10$
     (c) $K = 20$

     represents the best number of neighbors for the KNN classifier.
   - In order to justify your choice report the percentage misclassification error in the training set, in cross-validation and leave-one-out.

3. think about another distance metric and check whether the new metric improves the generalization.

# 2 The LDA classifier

Linear discriminant analysis (LDA) allows to find linear combinations of features that best discriminate a set of classes.

**Example**

The function `lda.example()` illustrates the linear discriminant analysis on a 2-class artificially generated dataset. The solid black line represents the optimal separation (as if an infinite number of observations were given), whereas the blue line represents the separation obtained by estimating from the dataset the parameters of the multivariate gaussians corresponding to each class.

**Exercise**

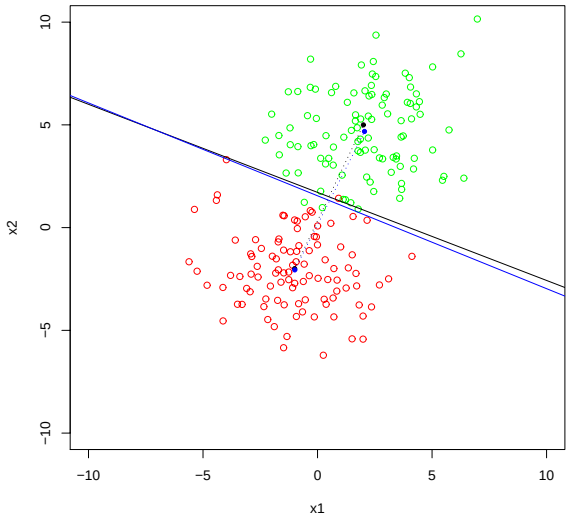
1. Write a R function which implements a LDA classifier by making the assumption that the two covariance matrices are diagonal. The inputs of the function are

   - an input matrix $X$ of size $N \times n$
   - an output class vector $Y$ of size $N \times 1$
   - a query input vector of size $1 \times n$

   and returns the class prediction.

2. Write a R function which implements a LDA classifier without making the assumption that the two covariance matrices are diagonal.

3. Compare the two implementations on the dataset BreastCancer.




# 3 The Naive Bayes classifier

Consider a classification problem with $n$ inputs and a random output variable $\mathbf{y}$ that takes values in the set $\{c_1, \ldots, c_K\}$. The Naive Bayes classifier for a 0-1 loss function returns

$$c_{NB}(x) = \arg \max_{k=1,\ldots,K} \text{Prob}\{\mathbf{y} = c_k\} \prod_{h=1}^{n} \text{Prob}\{x_h | \mathbf{y} = c_k\}$$

if the inputs are discrete and

$$c_{NB}(x) = \arg \max_{k=1,\ldots,K} \text{Prob}\{\mathbf{y} = c_k\} \prod_{h=1}^{n} p(x_h | \mathbf{y} = c_k)$$

if the inputs are continuous.

**Exercise**

1. Write a R function `NB` which takes as inputs
   - a real input matrix $X$ of size $N \times n$. Note that the input is continuous.
   - an output class vector $Y$ of size $N \times 1$
   - a query input vector of size $1 \times n$

and returns the class prediction according to the NB algorithm.

2. Test the NB classifier model on the BreastCancer dataset. Write a script `NBvalid.R` which

   - partitions the BreastCancer dataset in a training set containing two-thirds of the data and a test set,
   - validates the accuracy of the NB classifier by training-and-test,
   - performs a 10-fold cross-validation to validate the accuracy of the NB classifier,
   - performs a leave-one-out to validate the accuracy of the NB classifier.