# ELEN0062 - Introduction to machine learning
# Project 1 - Classification algorithms

October 2016

In this first project, we ask you to write several Python scripts to answer the different questions below. One separate script is required for each of the three questions. Make sure that your experiments are reproducible (e.g., by fixing manually random seeds). Add a *brief* report (pdf format, 5 pages max.) giving your observations and conclusions.

Each project must be done by group of *two students* and submitted as a tar.gz file on Montefiore's submission plateform (`http://submit.run.montefiore.ulg.ac.be`) before *October 25, 23:59 GMT+2.* You must use your sXXXXXX ids as group name.

## Files

You are given several files, among which are `data.py` and `plot.py`. The first one generates binary classification datasets with two real input variables. In the following, you will work on two datasets generated by `make_data1` and `make_data2`, respectively. You can generate datasets of 2000 samples. The first 150 will be used as training set and the remaining ones as testing set.

The second file contains a function which depicts the dataset together with the decision boundary of a trained classifier.

The other files must be completed and archived together with the report.

## 1 Decision tree (`dt.py`)

1. Build a decision tree model (`sklearn.tree.DecisionTreeClassifier`) on the two training sets and compare visually the decision boundary with the testing set. Comment your observations.

2. Observe visually the effect of the `max_depth` parameter on the decision frontier for the two datasets.

3. For both datasets, evaluate the accuracy on the learning and testing sets for various values of the `max_depth` parameter and plot the corresponding error curves.

4. Use a ten-fold cross validation strategy to optimize the value of the `max_depth` parameter for both datasets. What are the best scores attained and best values for this parameter? Justify your answer.

5. Discuss the differences and similarities you observe between the two datasets.

## 2 K-nearest neighbors (`knn.py`)

1. Build a k-nearest neighbors model (`sklearn.neighbors.KNeighborsClassifier`) on the two training sets and compare visually the decision boundary with the testing set. Comment your observations.

2. Observe visually the effect of the `n_neighbors` parameter on the decision frontier for the two datasets.

3. For both datasets, evaluate the accuracy on the learning and testing sets for various values of the `n_neighbors` parameter and plot the corresponding error curves.

4. Use a ten-fold cross validation strategy to optimize the value of the `n_neighbors` parameter for both datasets. What are the best scores attained and best values for this parameter? Justify your answer.

5. Discuss the differences and similarities you observe between the two datasets.

# 3 Naive Bayes classifier (`naive_bayes.py`)

The Naive Bayes (NB) classifier is a probability-based method which computes its predictions according to the posterior probability:

$$\hat{f}(\mathbf{x}) = \arg\max_y Pr(y|x_1, \ldots, x_p), \tag{1}$$

where $\mathbf{x} = (x_1, \ldots, x_p)$ denotes the $p$ input variables. To compute this posterior, the NB classifier postulates that the input variables are conditionally independent given the output. More formally,

$$P\left(\mathcal{X}_i|\mathcal{Y}, \mathcal{Z}\right) = P(\mathcal{X}_i|\mathcal{Y}) \quad \forall \mathcal{Z} \in \mathcal{P}(\{X_1, \ldots, X_{i-1}, X_{i+1}, \ldots X_p\}) \tag{2}$$

where $\mathcal{P}(S)$ designate the power set of $S$. Given this hypothesis, the predictions of the NB classifier can be obtained as follows:

$$\hat{f}(\mathbf{x}) = \text{argmax}_y \, Pr(y) \prod_{i=1}^{p} Pr(x_i|y) \tag{3}$$

To estimate (3), the prior (*i.e.* $Pr(y)$) and the likelihood (*i.e.* $Pr(x_i|y)$) probabilities must be estimated. For discrete variables – which is the case of $\mathcal{Y}$, one chooses the corresponding ratio over the learning set. For continuous variables, an additional assumption must be made with respect to the underlying distribution. One common choice is to presume Gaussianity:

$$Pr(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(x_i - \mu_i)^2}{2\sigma_i^2}\right) \tag{4}$$

where the population's mean $\mu_i$ and variance $\sigma_i^2$ are estimated by the sample mean and variance.

Despite its simplicity, this learning algorithm has proven itself quite useful in several areas and in text-based problems in particular.

1. Show that (3) is equivalent to (1) under the NB independence assumption.

2. Implement your own NB estimator according to the above description and following the scikit-learn convention (`http://scikit-learn.org/dev/developers/`). *Suggestion: Fill in the class whose template is given in* `naive_bayes.py`.

3. Compute the testing set accuracy on both datasets and interpret the results.