University of Liege

FACULTÉ DES SCIENCES APPLIQUÉES
Ecole d'ingénieurs et d'informaticiens

LIÈGE université

Project report

ELEN0062 - Introduction to machine learning
Project 1 - Classification algorithms

Maxime Lamborelle (s151701)
Thibault Renaud (s150899)
MASTER 1 Engineering

Professors : P. Geurts,
L. Wehenkel
Assistant : JM. Begon

2018-2019

1 Decision tree

1.1 Decision boundary

Without specifying a depth (`max_depth = None`), the decision tree clearly overfits the data. Indeed, the specification of the `DecisionTreeClassifier` function specifies that if `max_depth = None`, then nodes are expanded until all leaves are pure or until all leaves contain less than `min_samples_split` which is the minimum number of samples required to split an internal node.

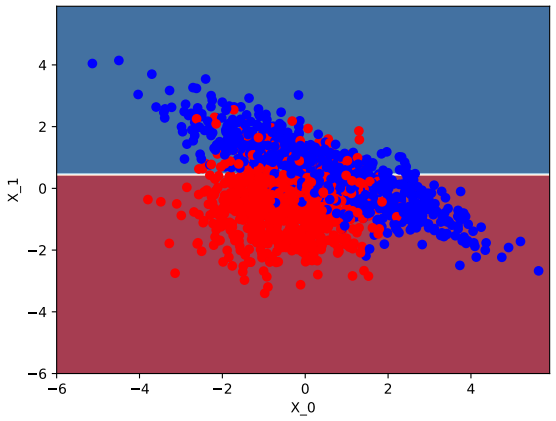
The lower the depth, the lower the precision of the decision boundary in the learning sample. Indeed, by increasing the depth, we increase the fitting of the model for the learning sample. That's what we can observe in the following figures. But we will also fit noise (see next point).

We also see that a depth of 1 clearly leads to an underfitting. It is normal since the algorithm chooses the best attribute (`X_0` or `X_1`) and just find the best splitting of this attribute. Then, the higher the depth, the less underfitting, and the more overfitting as said before.

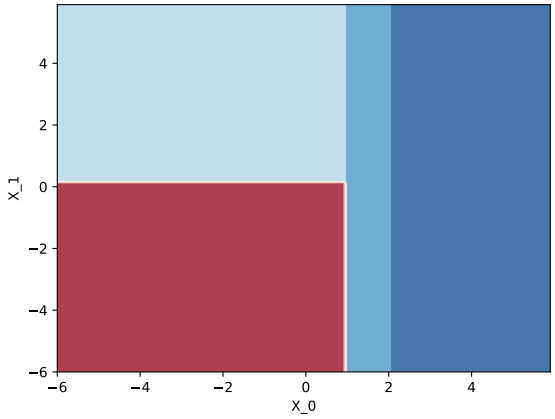
1.2 Average Test Set Accuracy

We notice that the accuracy increases until a depth of 4, then it decreases. When we deal with values over 4, the decision tree overfits the data, i.e the algorithm starts fitting noise, and then the accuracy decreases. The standard deviation is quite low. So we can expect to obtain similar results if we launch the program again since objects in dataset form quite homogeneous blocks (depending on classes). We have done some tests over 25 generations of the dataset and we have obtained the following results :

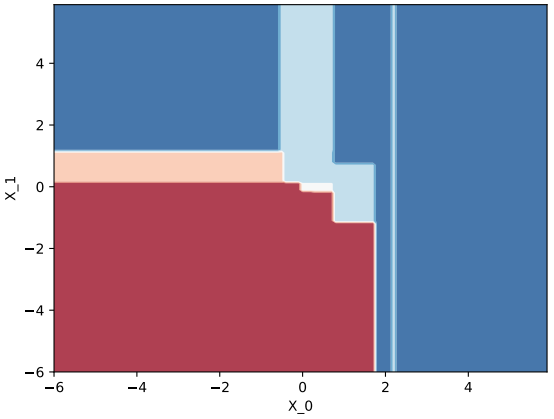
Depth	Average Accuracy	Standard deviation
1	0.674267	0.030703
2	0.80467	0.024585
4	0.832267	0.021781
8	0.812533	0.026688
None	0.787467	0.020576



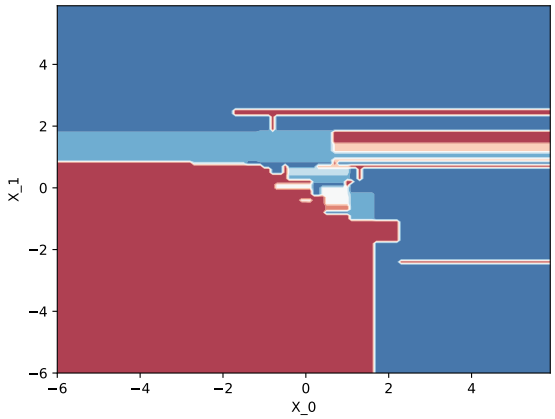
(a) depth 1



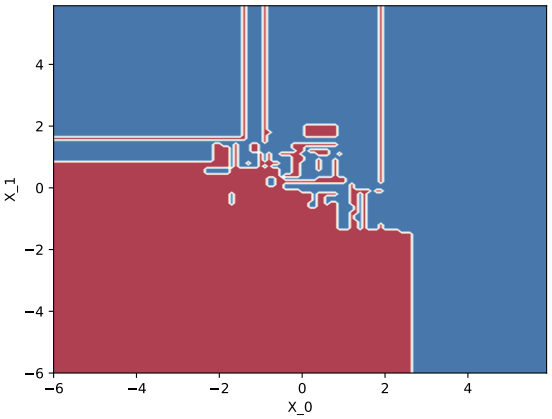
(b) depth 2



(c) depth 4



(d) depth 8



(e) depth = None

FIGURE 1 – Decision tree splits

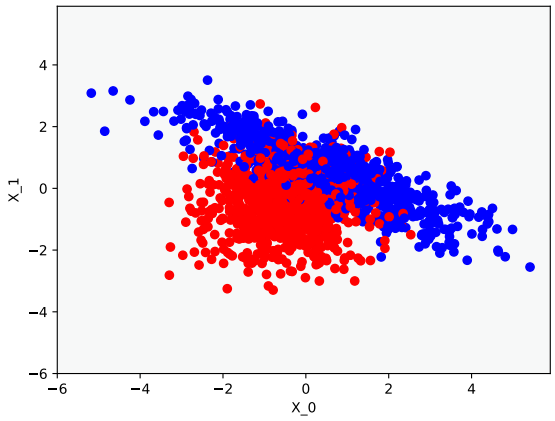
2 K-nearest neighbors

2.1 Decision boundary

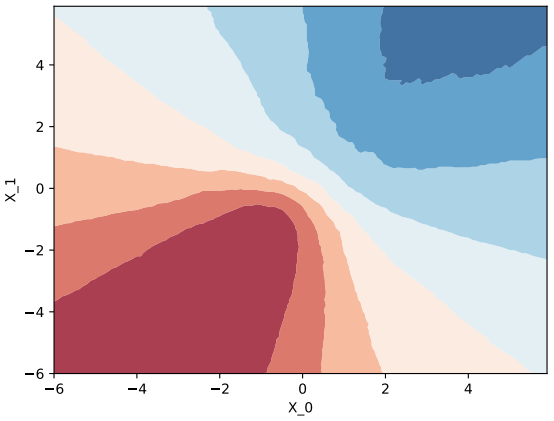
We can see on the following figures that the smaller the number of neighbors, the more overfitting. That is what we expected since a smaller number of neighbors means that the model will be very precise, and will overfit the data when reaching a k small enough. In the same way, if the number of neighbors is high, we expect the model to be less precise and therefore data will be underfitted (for `n_neighbors` large enough). Of course if we take `n_neighbors` equal to the size of the learning sample, then there will be only one class represented depending on which class has the most occurrences in the learning sample. The accuracy is then very low.

If we take a look at the comparative scheme down here, we observe that the accuracy for `n_neighbors = 1200` is very low, as explained in the previous paragraph. From 1200 to 25, the accuracy of the model increases, reaching 85% of correct matches for a number of neighbors of 25. Then we can observe a decrease of the accuracy for `n_neighbors = 5, and 1`. Indeed, the model starts fitting the noise (overfits the data). The standard deviation is quite low and quite the same for all the `n_neighbors`, so we can expect to have similar results at each test. Like for the previous model, we have done some tests over 25 generations. Here are the results :

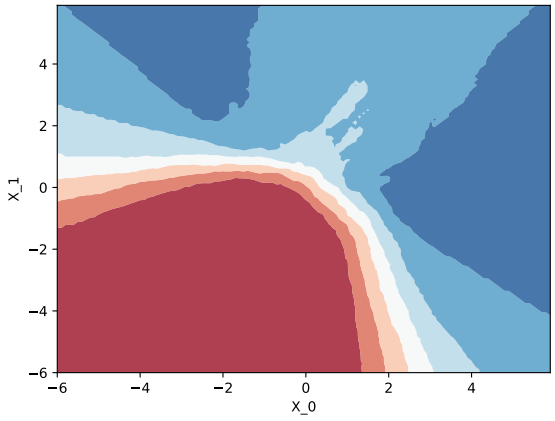
Nb of neighbors	Average Accuracy	Standard deviation
1200	0.4759999	0.01668
625	0.8178667	0.031253
125	0.8417333	0.023364
25	0.8504	0.021444
5	0.825733	0.0259
1	0.789867	0.023692



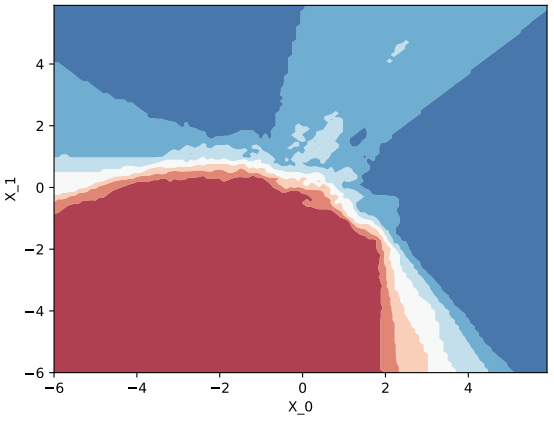
(a) Nb of neighbors = 1200



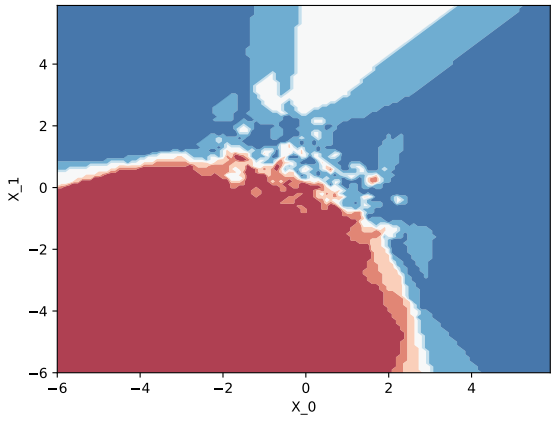
(b) Nb of neighbors = 625



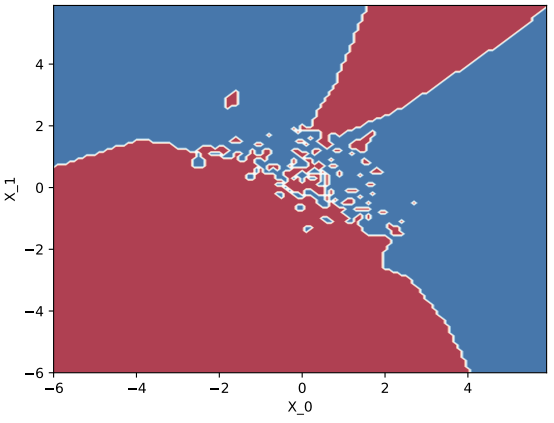
(c) Nb of neighbors = 125



(d) Nb of neighbors = 25



(e) Nb of neighbors = 5



(f) Nb of neighbors = 1

FIGURE 2 – K-nearest neighbors

2.2 Ten-fold cross validation strategy

A Methodology

We used the `cross_validate` function, specifying the estimator (`KNeighborsClassifier`), as well as X, y, and the cross-validation predicator `cv = 10`. This function splits the sample into 10 parts. First, the function takes as learning sample the first part, then the first two parts, and so on until 9 parts. After that, the function computes the average accuracy over the 9 different learning samples. Accuracy scores are described below.

B Score and Optimal Value of `n_neighbors`

We notice in the Figure 3 that the maximum accuracy does not correspond to a specific number of neighbors but is reached through different numbers of neighbors between 25 and 125, depending on each test. For these tests to be more general, we have done 10 iterations over which we have computed the accuracy for `n_neighbors = [5; 125]`. There is no obvious optimal value coming out of the graph below.

We can still notice that the number of neighbors brings similar results (i.e similar accuracy) for the k-nearest neighbors model or for the ten-fold cross validation model.

Nb of neighbors	Average Accuracy
1200	0.55866
625	0.81933
125	0.84
25	0.85
5	0.834
1	0.783333

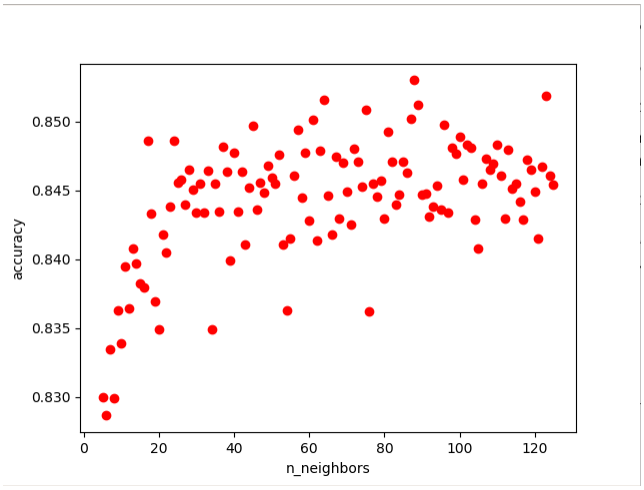


FIGURE 3 – ten-fold cross validation

3 Linear discriminant analysis

3.1 Shape of the decision boundary

The decision frontier, in the two classes cases (0 and 1), is the line where :

$$P(y = 0|x) = P(y = 1|x) \tag{1}$$

Where,

$$P(y = k|x) = \frac{f_k(x)\pi_k}{\sum_{l=1}^{K} f_l(x)\pi_l} \tag{2}$$

With,

$$f_k(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)} \tag{3}$$

After some mathematical transformations

$$\log \frac{P(y = 0|x)}{P(y = 1|x)} = 0 \tag{4}$$

$$\log \frac{f_0(x)}{f_1(x)} + \log \frac{\pi_0}{\pi_1} = 0 \tag{5}$$

$$\log \frac{\pi_0}{\pi_1} - \frac{1}{2}(\mu_0 + \mu_1)^T\Sigma^{-1}(\mu_0 - \mu_1) + x^T\Sigma^{-1}(\mu_0 - \mu_1) = 0 \tag{6}$$

That equation is linear in x, the frontier is then a hyperplane in p dimensions. In our case, $p = 2$ so the frontier is just a line like it can be seen on the FIGURE 4.

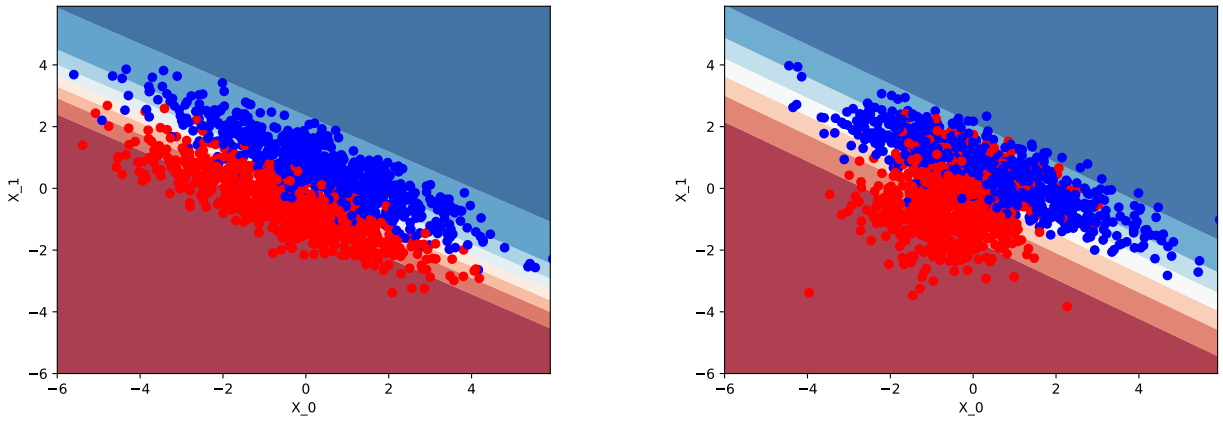
If there are more than 2 classes, the space $\mathbb{R}^p$ can be divided into regions that are classified as class 1, class 2, etc., these regions will be separated by hyperplanes. If $p = 2$ the different classes are separated by lines.

3.2 Decision Boundary, Average Accuracy, Standard Deviation

As expected (see previous sub-section), linear splits can be observed. Difference between the two datasets is well shown in the FIGURE 4. The way boundaries are computed is directly related to the distribution of objects in these datasets. This is why we observe a difference between boundaries of dataset 1 and boundaries of dataset 2.

The average accuracy of the dataset 1 is higher than the one of the dataset 2. This is logical since objects of dataset 1 form two homogeneous blocks (depending on classes) unlike in the dataset 2 where objects are quite mixed. Standard deviation is lower for the dataset 1 than for the 2 for the same reason that the accuracy is higher for the first dataset than for second.

	Average Accuracy	Standard deviation
Dataset 1	0.84133	0.00686
Dataset 2	0.82667	0.0159

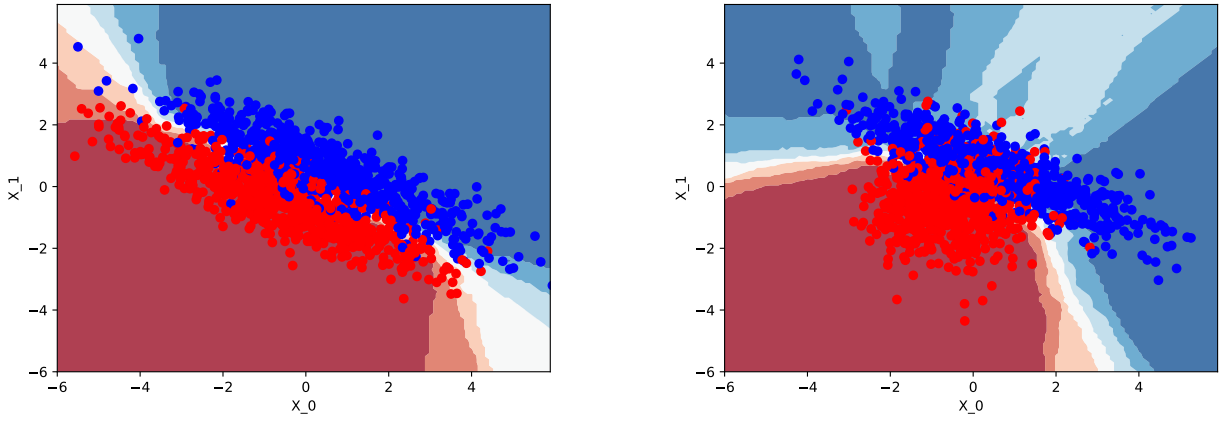


(a) Dataset 1 (b) Dataset 2

FIGURE 4 – Linear discriminant analysis

3.3 Comparison of the 2 datasets

We can see that the first dataset is composed of two elliptic Gaussian distributions with different means. Due to this distribution, even the nearest neighbor algorithm has a decision boundary which is almost linear. In the first dataset, the two classes do not overlay each other a lot, so the accuracy is better.



(a) Dataset 1 (b) Dataset 2

FIGURE 5 – 25-Nearest neighbors