ELEN0062 - Introduction to machine learning
Project 1 - Classification algorithms

François Rozet (s161024)
Adrien Schoffeniels (s162843)

October 21, 2019

1 Decision tree

1.1 Decision boundary

(a) For both datasets (`make_data1` and `make_data2`) and for each maximum depth (`max_depth`) value, a decision boundary graph[1] has been produced using the `plot_boundary` function, yielding Figures 1 and 2.

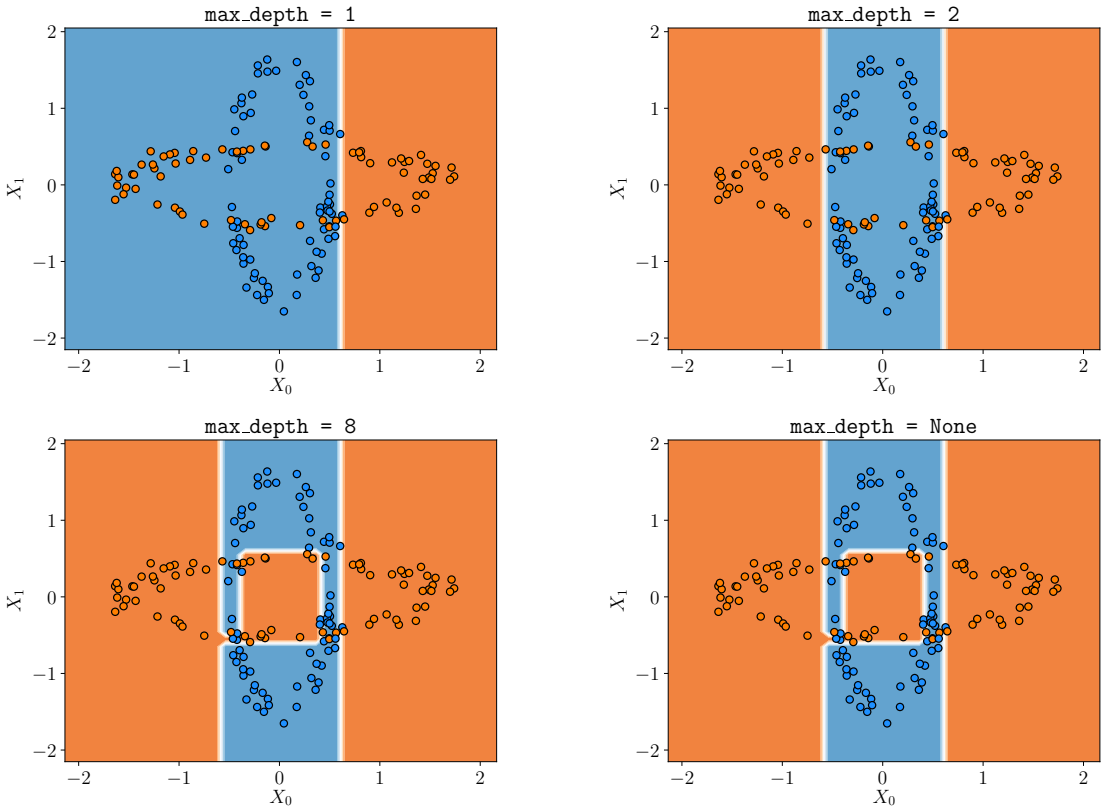

Figure 1 – Decision boundary plots of decision tree models with fixed `max_depth` for dataset `make_data1`.

[1]For the sake of compactness, the decision boundary plots won't all be displayed. Moreover, in order to keep the boundaries visible, it has been chosen to display only the 150 first objects from the testing set.

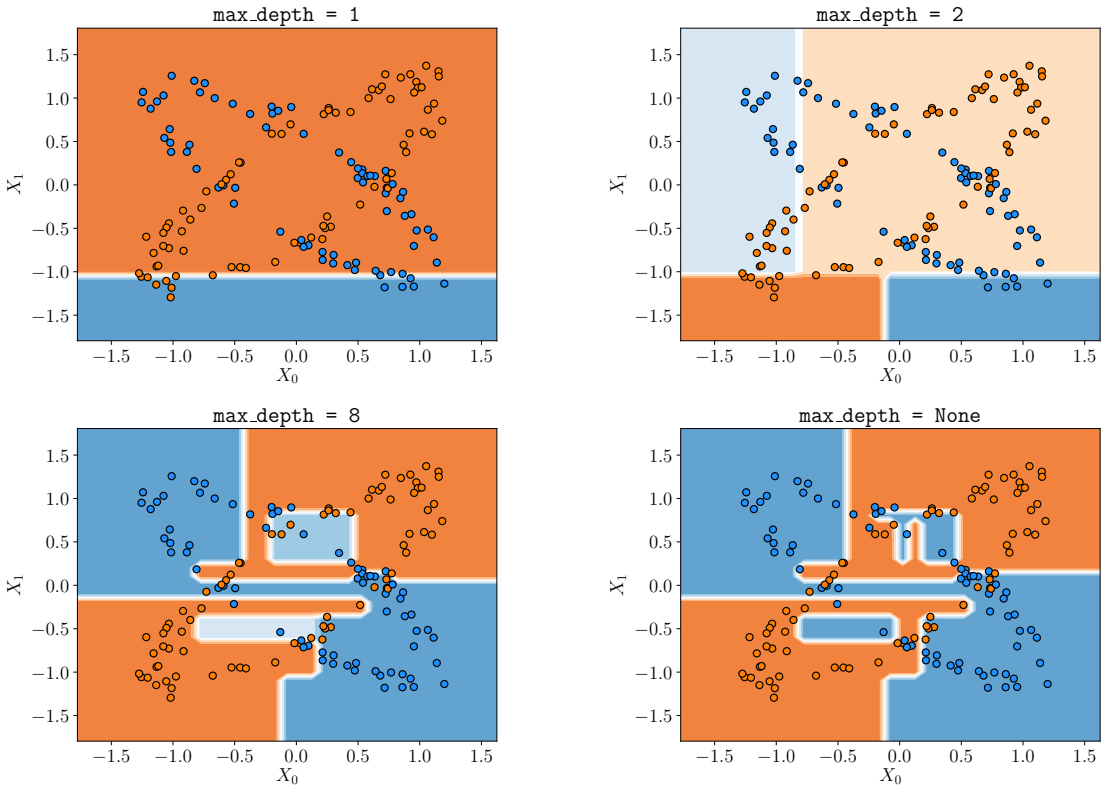

Figure 2 – Decision boundary plots of decision tree models with fixed `max_depth` for dataset `make_data2`.

As one can see in these Figures, as the maximum depth grows, the number and complexity of boundaries do as well. Nothing surprising since the number of leafs in a decision tree is (up to) exponentially proportional to its depth.

However, in the case of `make_data1`, a complexity rise between the decision tree with a maximum depth of 8 and the unconstrained one isn't observable. In fact, using the `get_depth` method of the `DecisionTreeClassifier` object, it can be shown that these two trees have the exact same depth, which means that 8 is a sufficient depth to classify *perfectly* the `make_data1` training set.

As far as the confidence is concerned, the classifiers over `make_data1` seem to do better than the ones over `make_data2`, especially with low depths. Further discussion in section 1.3.

(b) For both datasets, the decision trees with maximum depth of 1 and 2 seem to underfit since the boundaries are too simple to account for the data. With a maximum depth of 8, the regions begin to specialize too much, yet not enough dramaticaly to state it is overfitting, conversely to the unconstrainded decision tree for `make_data2` which is, indeed, overfitting.

(c) Unconstrained, the fitting algorithm won't stop growing the tree until it perfectly classifies the training set. At that point, each region is *pure* and the model predicts the proportion of training objects of each class in the region, i.e. inevitably 1 and 0.

1.2 Testing set accuracies

In order to compare the reliability of each model, the average testing set accuracies over five datasets generations were computed. The results are shown in Table 1.

Dataset	Max. depth	Average accuracy	Standard deviation
`make_data1`	1	0.684 000	0.005 191
	2	0.865 514	0.006 390
	4	0.890 703	0.017 153
	8	0.928 324	0.005 572
	None	0.928 649	0.005 071
`make_data2`	1	0.499 784	0.007 119
	2	0.653 405	0.110 842
	4	0.792 216	0.024 847
	8	0.856 649	0.017 440
	None	0.862 486	0.010 384

Table 1 – Average testing set accuracies (over five generations of the dataset) along with their standard deviations for each depth.

As expected, the average accuracy increases with the depth of the decision tree. However, it does not decrease for those that have been stated as overfitting. It means that the training and testing set distributions are very close. Indeed, both datasets barely are spread around the ellipses. Also, one can see that the standard deviation of the accuracy is correlated to the overall confidence of the classifier.

1.3 Differences between the two datasets

The first, and only, difference one can observe between `make_data1` and `make_data2` is their spatial distribution. While the axes of `make_data1` ellipses are aligned with X_0 and X_1 axes, none of `make_data2` ellipses are. In fact, by looking closely, it can be established that they are the exact same dataset (with the same *seed*) but rotated 45° from each other.

This angle explains why decision trees classifies better the first dataset than the second (cf. Figures 1, 2 and Table 1), since they partition the space with axis-aligned cuts.

2 K-nearest neighbors

2.1 Decision boundary

(a) For both datasets (`make_data1` and `make_data2`) and for each `n_neighbors` values, a decision boundary graph has been produced using the `plot_boundary` function, yielding Figures 3 and 4.

(b) As one can see in Figures 3 and 4, as `n_neighbors` grows, the boundaries are getting less and less sharp and the classifier confidence drops. Indeed, for `n_neighbors =`

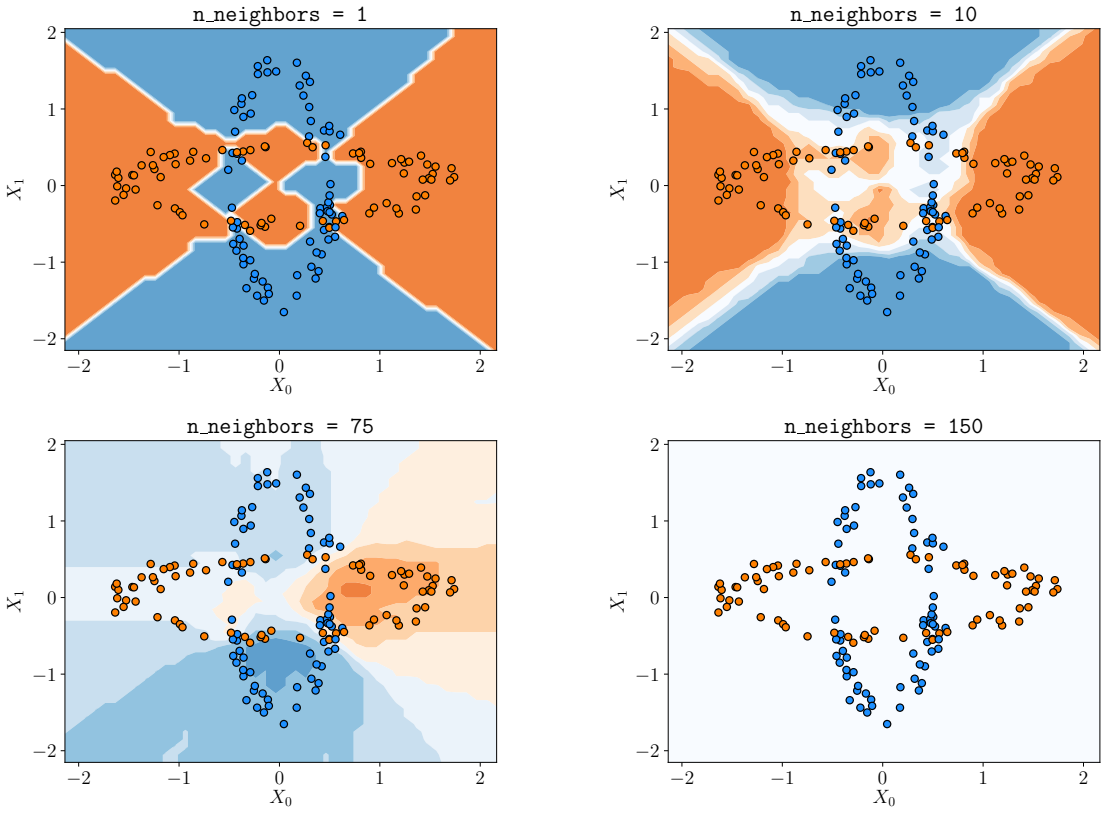

Figure 3 – Decision boundary plots of nearest neighbors models with fixed `n_neighbors` for dataset `make_data1`.

`1`, the model is 100 % confident about its prediction (since it only takes the closest neighbor into account). For example, in the *crossing* regions of the two ellipses, the model is still arbitrary confident, which could be depicted as overfitting.

For slightly higher `n_neighbors` values (10 to 20), the model loses its confidence in crossing regions while keeping confidence everywhere else. This is actually how the model should perform and, therefore, it is probably the most accurate.

Conversely, for much higher `n_neighbors` (40 and above), the model underfits clearly : it gives wrong predictions for most of the testing and training set objects and is poorly confident.

Eventually, when `n_neighbors` reaches the size of the training set, the prediction becomes spatially uniform since it always takes all the training set into account.

2.2 Ten-fold cross validation strategy

(a) In order to find the optimal value of `n_neighbors`, a ten-fold cross validation strategy has been used. First, the dataset (`make_data2`) has been split into 10 subsets. Each subset was then used as a testing set and its complement as a training set. Eventually, the average accuracy over the 10 subsets was computed for all `n_neighbors` ranging from 1 to 100, yielding Figure 5. It wasn't useful to try greater values since the average accuracy decreases for such values.

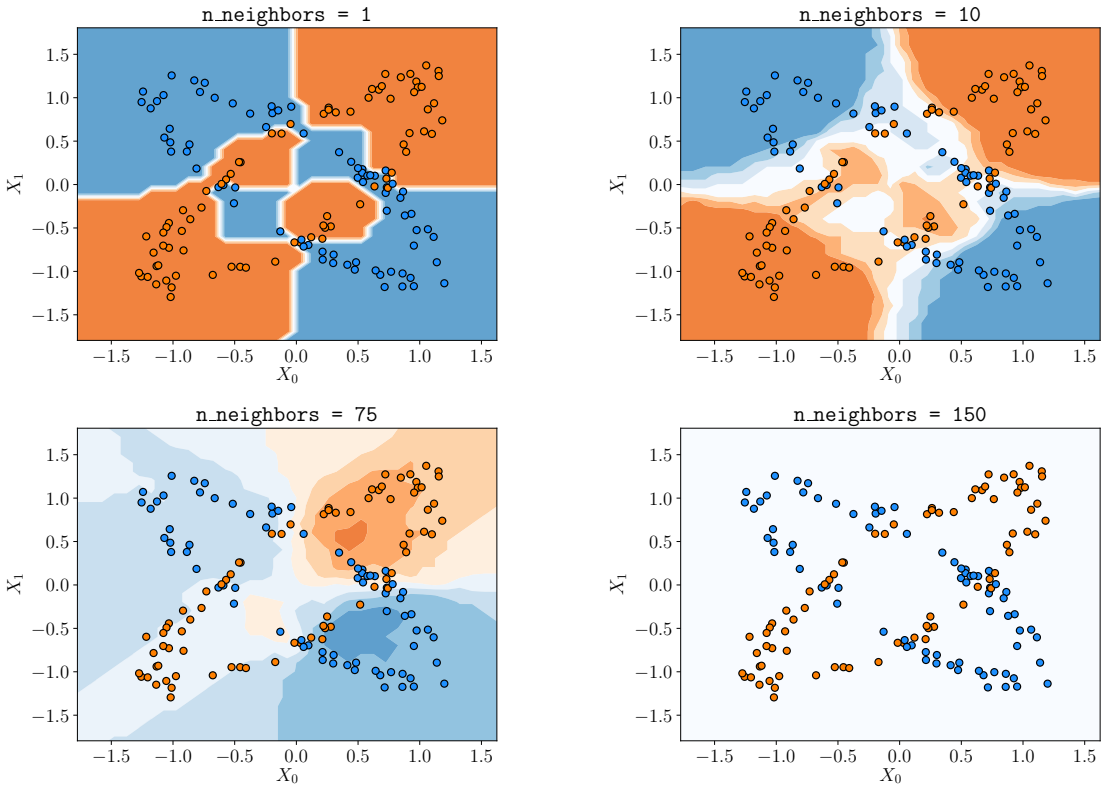

Figure 4 – Decision boundary plots of nearest neighbors models with fixed `n_neighbors` for dataset `make_data2`.

(b) The optimal value for `n_neighbors` is 15 with an average accuracy of 0.9635. This result corroborate the decision boundary based intuition : a good model needs to take a sufficient amount of neighbors into account in order to be less confident in crossing regions, yet not too much such that it remains confident in *pure* regions.

2.3 Optimal value for `make_data1`

Because `make_data1` and `make_data2` are the same datasets expressed in different (rotated) euclidean spaces (cf. section 1.3), the Euclidean distance between objects is conserved from one to the other. Therefore, since the k-nearest neighbors classifier is only based on that distance, the result of the ten-fold cross validation strategy should be exactly the same, that is 15. Indeed, as one can see by comparing Figures 3 and 4, the decision boundary is always the same, yet rotated, for both datasets.

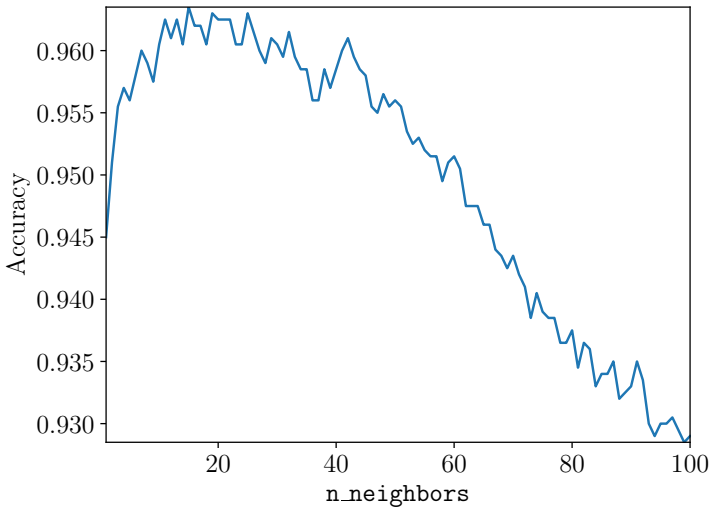

Figure 5 – Ten-fold cross validation average accuracy with respect to `n_neighbors` for dataset `make_data2`.

3 Naive Bayes classifier

3.1 Equivalence

Let the posterior probability be $P(\mathcal{Y}|\mathcal{X}_1, \ldots, \mathcal{X}_p)$. Using the conditional probability definition,

$$\begin{aligned} P(\mathcal{X}_1, \ldots, \mathcal{X}_p)\, P(\mathcal{Y} \mid \mathcal{X}_1, \ldots, \mathcal{X}_p) &= P(\mathcal{Y})\, P(\mathcal{X}_1, \ldots, \mathcal{X}_p \mid \mathcal{Y}) \\ &= P(\mathcal{Y})\, P(\mathcal{X}_1, \ldots, \mathcal{X}_{p-1} \mid \mathcal{Y}, \mathcal{X}_p)\, P(\mathcal{X}_p \mid \mathcal{Y}) \\ &= P(\mathcal{Y}) \prod_{i=1}^{p} P(\mathcal{X}_i \mid \mathcal{Y}, \mathcal{X}_p, \ldots, \mathcal{X}_{i+1})\,. \end{aligned}$$

But, under the NB independence assumption,

$$P(\mathcal{X}_i \mid \mathcal{Y}, \mathcal{X}_j) = Pr(\mathcal{X}_i \mid \mathcal{Y}) \quad \forall\, i, j \in \{1, \ldots, p\} \text{ and } i \neq j$$

and therefore

$$\begin{aligned} P(\mathcal{Y} \mid \mathcal{X}_1, \ldots, \mathcal{X}_p)\, P(\mathcal{X}_1, \ldots, \mathcal{X}_p) &= P(\mathcal{Y}) \prod_{i=1}^{p} P(\mathcal{X}_i \mid \mathcal{Y}, \mathcal{X}_p, \ldots, \mathcal{X}_{i+1}) \\ &= P(\mathcal{Y}) \prod_{i=1}^{p} P(\mathcal{X}_i \mid \mathcal{Y})\,. \end{aligned}$$

And, because $P(\mathcal{X}_1, \ldots, \mathcal{X}_p)$ is independent of $\mathcal{Y}$, it doesn't account in "argmax$_y$" which, with the above relation, proves the equivalence between (1) and (3) in the project statement.

3.3 Testing set accuracy on both datasets

The testing set accuracies on both datasets are shown in Table 2. One can see that the Naive Bayes classifier has a better accuracy with the first dataset[2].

Dataset	Accuracy
`make_data1`	0.797 838
`make_data2`	0.553 514

Table 2 – Testing set accuracy of both datasets using NB estimator.

Actually, this is due to the NB independence assumption : the corollary to this assumption is that, knowing the class $\mathcal{Y}$, the correlation of every pair $\mathcal{X}_i, \mathcal{X}_j$ $(i \neq j)$ is zero which is clearly misleading for `make_data2` (cf. Table 3) and, therefore, produces a poor model.

Dataset	$\mathcal{Y}$	$\mathrm{Cor}(\mathcal{X}_0, \mathcal{X}_1 \mid \mathcal{Y})$
`make_data1`	0	−0.021 705
	1	−0.025 280
`make_data2`	0	0.807 203
	1	−0.801 947

Table 3 – Conditional correlations of $\mathcal{X}_0$ and $\mathcal{X}_1$ in both datasets.

[2]In fact, the classifier accuracy on the second dataset is near 50 %, that is the accuracy a totally random classifier would obtain.