Introduction to machine learning

Project 1

BERNARD Simon (s161519)
KLAPKA Ivan (s165345)

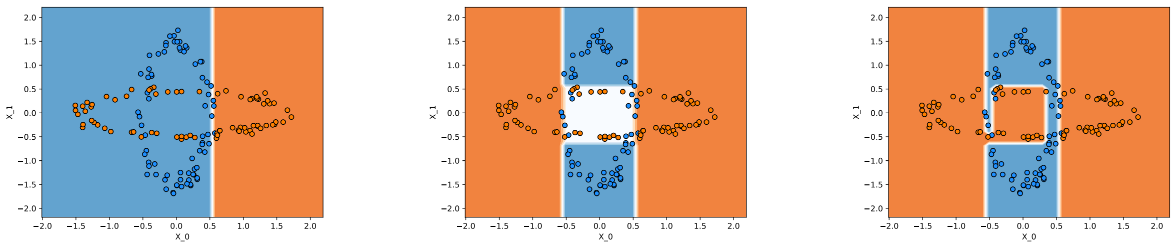
1st year of master in computer science and engineering
Academical year 2019-2020

1 Decision Tree

1.1 Tree depth effects on the decision boundary

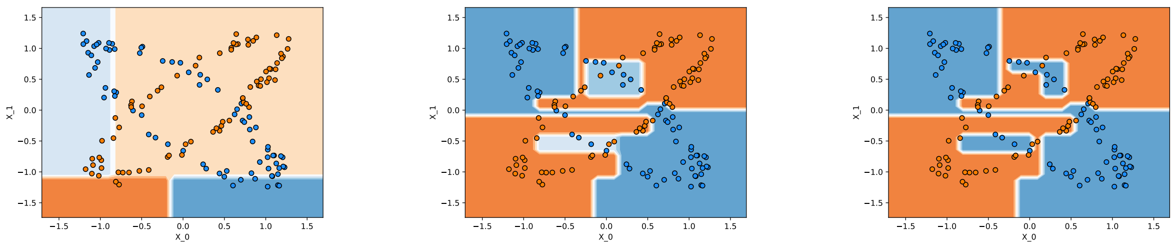
For the clarity of this report we will call the model built on the first dataset the "first model", and the model built on the second dataset the "second model".

(a) Illustration and explanation



Depth 1 Depth 4 Depth 8

Figure 1 – Decisions boundaries of the decision tree on the first dataset



Depth 2 Depth 8 Unbounded

Figure 2 – Decisions boundaries of the decision tree on the second dataset

In the figures above, each line represents a decision node present in the decision tree. As the model add new node after the previous one when the depth is increased, boundary existing in lower depth does not disappear in higher depth-model.

(b) Underfitting/Overfitting

The first model seems to underfit up to the depth 4 (especially the center part that is badly/not determined), this can be proven by the fact that it is still possible to add lines and makes the model perform significantly better on any testing set. For the 8 and the unbounded depth (which are identical), the first model reaches a state where it is able to fit every data of the training set and still maintain a good accuracy on the test set (92.8%) proving that it is not overfitting the dataset.

The second model looks underfitted up to the depth 4 (especially at depth 2 where the standard deviation is very high, as it can be seen on figure 4). The depth 8 and unbounded seems to neither be underfitting nor overfitting, since the average test set accuracies still increases slowly and seems to reach a maximum.

(c) Confidence when unbounded

An unbounded tree allows the model to build new nodes until all sample of the training set are correctly classified, but this could lead to overfitting and bad result on test set. The two models seems to be spared from this effect but training them on larger sample (and larger depth) would probably induces some overfitting.

1.2 Average test set accuracies and standard deviations for each depth

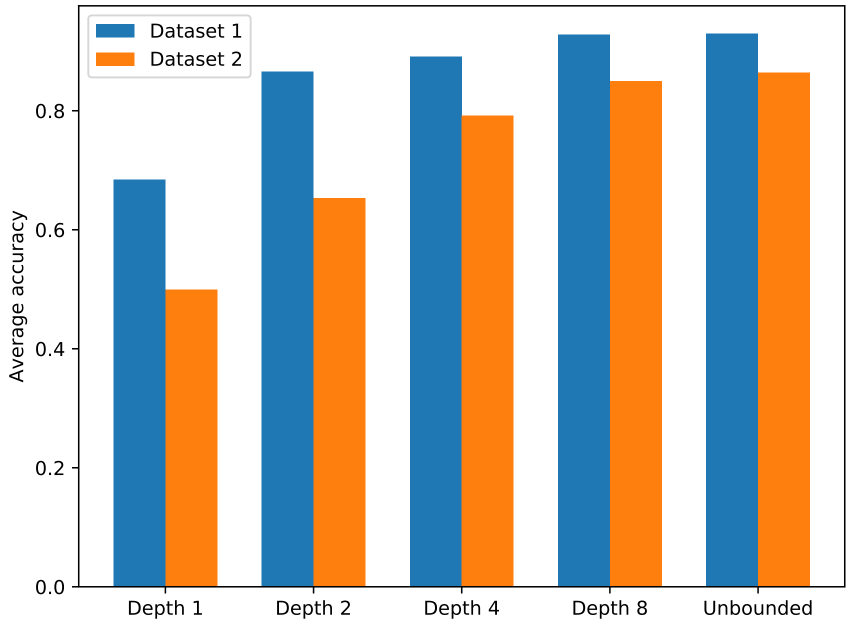


Figure 3 – Average accuracy by tree depth over 5 generations

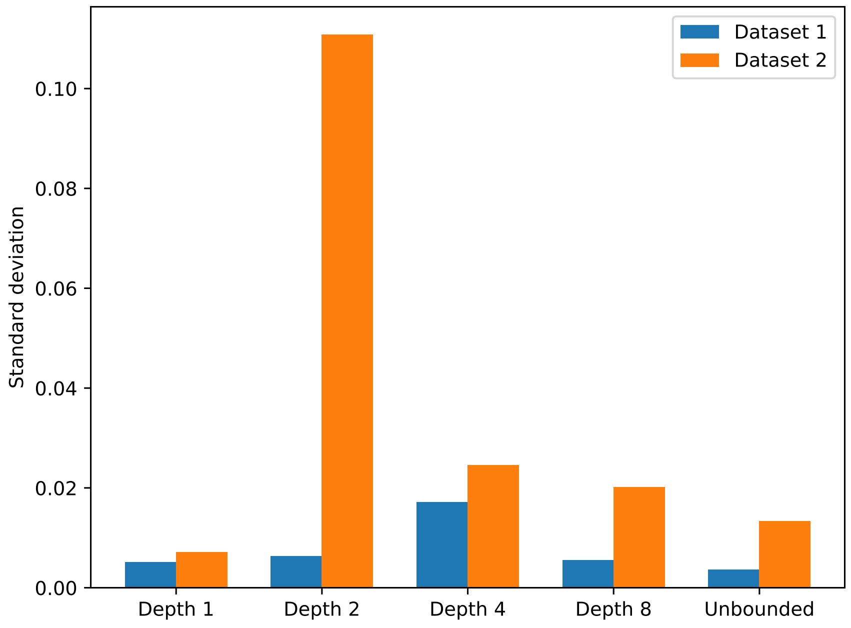


Figure 4 – Standard deviation by tree depth over 5 generations

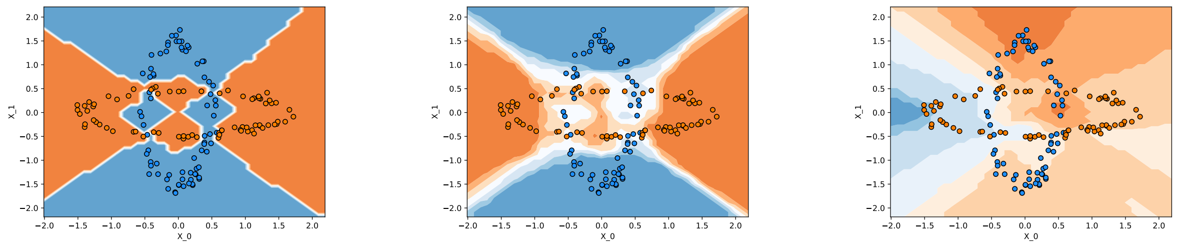
1.3 Differences between datasets

While the decision boundaries are parallel to x and y axis, it perform well when figures are oriented on those direction, but it has worse results when it has to guess a tilted line. This shows the importance of data collection and processing, indeed the model improves/weakens just by applying a pre-processing method (rotating coordinates by 45°).

2 K-nearest neighbors

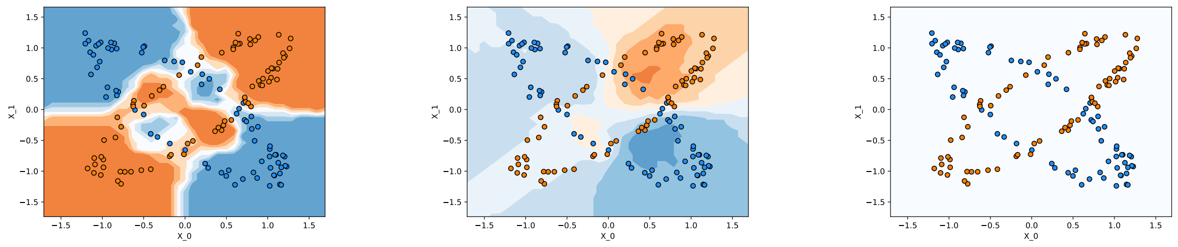
2.1 Number of neighbors effects on the decision boundary

(a) Illustrations of the decision boundary



1 neighbors 10 neighbors 100 neighbors

Figure 5 – Decisions boundaries of the K-nearest neighbors on the first dataset



5 neighbors 75 neighbors 150 neighbors

Figure 6 – Decisions boundaries of K-nearest neighbors on the second dataset

(b) Evolution of the decision boundary

For both datasets, it begins quite well, even though it is a bit too much dependant on the random location (determined by the seed) of the points of the ellipses. However, it starts to really become bad form 75 neighbors and above. It shows signs of underfitting, this is due to the fact that far and irrelevant points are taken into account for the decision. For 150 neighbors, all points are considered for the whole space and no decision can be taken (or just a uniform random distribution).

2.2 Ten-fold cross validation strategy

(a) Methodology

The ten-fold cross validation strategy consist in training the classifier on multiples instance of the data (10 in our case). On each instance we rearrange the part which will be used for training and the one used for testing. Once we have computed the accuracy (score) of each iteration, we take the average to get the final accuracy of the classifier. This strategy allows us to validate a classifier on different instance of the data and thus give us a better prediction of the accuracy on an independent dataset.

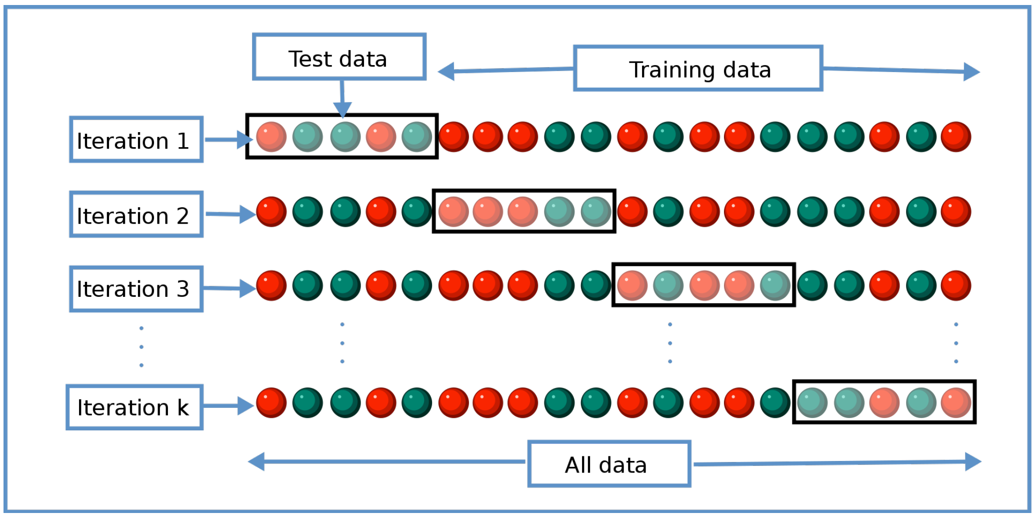


Figure 7 – K-fold-cross validation diagram[1]

In order to use find the optimal value of neighbor we compute the final accuracy of the 10-fold cross validation for all the possible number of neighbors. The optimal value for the number of neighbors corresponds to the maximum final accuracy. In our case we only computed the first 150 values of neighbors because we know that the accuracy will not increase with higher number of neighbors.

(b) Optimal number of neighbors

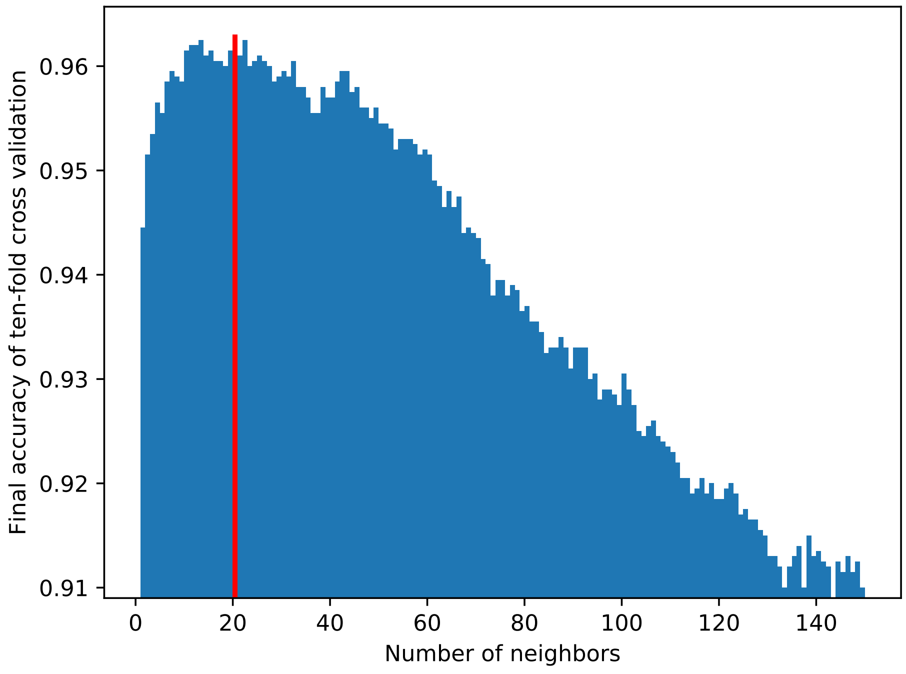


Figure 8 – Final accuracy by number of neighbors on the second dataset using ten-fold cross validation (maximum accuracy in red)

[1]Source (17/10/2019) : https://en.wikipedia.org/wiki/Cross-validation_(statistics)

In our case, the optimal value is 20 neighbors and gives us a final accuracy of 96.3% (highlighted in red on figure 8). Theses results correspond more or less to our expectations. It seems logical to assume that a low number of neighbors will lead to sub-optimal guesses because it gives too much importance to single points of data (overfitting). Noises in the data and overlapping of probability would make the model take bad decisions. On the other hand, a high number of neighbors would make the model less precise. This is due to the fact that increasing the number of neighbors decreases too much the importance of single points and thus failing to distinguish edges precisely (underfitting). In the extreme case of the number of neighbor being equal to the number of data point, the classifier will always guess the output with the largest population in the dataset. This is why we expected the optimal value to be larger that a few neighbors and smaller than a hundred of them.

2.3 Effect of the optimal value for the second dataset on the first dataset

The optimal value found on the second dataset is also the optimal value of the first dataset (with the same seed). This is due to the fact that the K-nearest neighbors method is based on the distance between points and that the distance does not change between dataset one and two of same seed. The distance does not change since the only difference between datasets is a rotation of 45 degrees.

3 Naive Bayes classifier

3.1 Proof of equivalence

$$\hat{f}(\mathbf{x}) = \text{argmax}_y \; Pr(y|x_1, ..., x_p)$$

Using the Bayes' theorem we get :

$$\hat{f}(\mathbf{x}) = \text{argmax}_y \; \frac{Pr(y)\, Pr(x_1, ..., x_p|y)}{Pr(x_1, ..., x_p)}$$

While the denominator is independent of y, it can be removed in the argmax$_y$

$$\hat{f}(\mathbf{x}) = \text{argmax}_y \; Pr(y)\, Pr(x_1, ..., x_p|y)$$

Then using the conditional probability's definition several times on $Pr(x_1, ..., x_p|y)$

$$\begin{aligned}\hat{f}(\mathbf{x}) &= \text{argmax}_y \; Pr(y)\, Pr(x_1|y)\, Pr(x_2, ..., x_p|y, x_1) \\ &= \text{argmax}_y \; Pr(y)\, Pr(x_1|y)\, Pr(x_2|y, x_1)\, Pr(x_3, ..., x_p|y, x_1, x_2) \\ &= \; ... \\ &= \text{argmax}_y \; Pr(y)\, Pr(x_1|y)\, Pr(x_2|y, x_1) \; ... \; Pr(x_k|y, x_1, ..., x_{k-1}) \; ... \; Pr(x_p|y, x_1, ..., x_{p-1})\end{aligned}$$

With the NB independence while all condition in the probabilities in our equation depend on y, all x_i can be removed in the dependencies

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

3.2 Implementation

See code

3.3 Testing set accuracy on both datasets

	Testing set accuracy
Dataset 1	0.797297
Dataset 2	0.514594

This model works by computing an estimate gaussian probability for each features knowing the class then just making a guess based on the highest probabilities. This means the model can be visualized by doing a superposition of the gaussians for each class on each axis.
With the first dataset, each ellipsis will have a large gaussian (high standard deviation) along the axis of their major radius and a small one (small standard deviation) along the other. This lead to a higher probability of having a blue on the middle and orange on the side for the x-axis (the opposite for the y axis) and thus a not so bad accuracy (79,72%).
For the second dataset with the tilt of 45°, datas are equally distributed for the two ellipsises on each axis. The model has no real idea on which one is more likely to be right, we have about 50% which is similar to a random pick.