ELEN0062 : Introduction to machine learning
Project 1 report: Classification algorithms

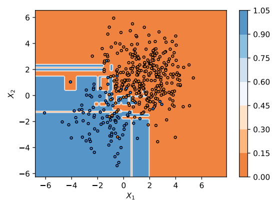
Aldeghi Florian(s183157) Baguette Brice (s181482)
Dasnois Louis (s181779)

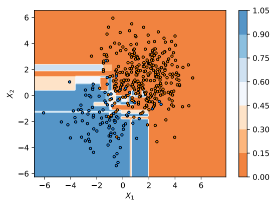
September 28, 2023

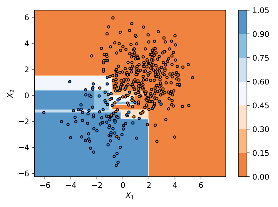
1 Decision tree

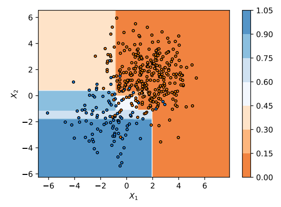
1. Observe how the decision boundary is affected by tree complexity

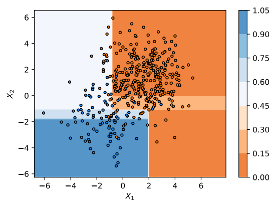
   (a) The decison boundaries are illustrated in figure 1.With a parameter of 500, the set of 1000 samples is split more or less 3 times. That means that we have 4 leaves in the tree. That's why the boundaries are not complex and are just big rectangles. We can see on the graph those 4 leaves for `min_samples_split` = 500. Deeper is the decision tree, more complex will be the boundary.
   For 128 and 64, the imprecision is still there with part of the graph that are not fully blue or orange. That means some underfitting because we don't split samples enough times. `min_samples_split` = 32 seems to be the best parameter. Colors fit well with the distribution even if a white zone stays on the graph.
   For parameters 8 and 2, the tree is so deep that it creates an overfitting represented by small blue area in the orange part and vice versa. Actually, with `min_samples_split` = 2, each positive or negative value creates a blue or orange area around it. This boundary is too complex.


(a) min_samples_split = 2


(b) min_samples_split = 8


(c) min_samples_split = 32


(d) min_samples_split = 64


(e) min_samples_split = 128

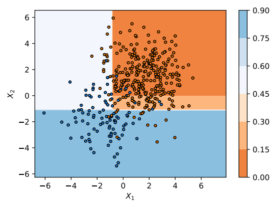

(f) min_samples_split = 500

Figure 1: Decision boundaries using decision tree models

   (b) For a small `min_samples_split` parameter, we observe an overfitting. In fact, when we observe the boundaries from the training sample and then we apply it to the testing sample, we observe more errors for a min_samples_split of 2 and 8 than for

32. Actually, the model is so accurate according to the learning sample that it learns too much and is no longer accurate for a test sample.
For a large `min_samples_split` parameter like 128 or 500, the model is totally underfitting because the learning phase ends too early. We can see that the boundaries for these graphs do not match with our best parameter of 32.

   (c) When the decision tree comes to a leaf, the prediction confidence is computed according to the ratio of the two classes present in this leaf. For example, if we have a leaf with 40% of blue and 60% of orange, the prediction will say orange but the ratio is only 60%, and so the prediction confidence is not good. If we have a pure leaf, the ratio is 1 and the prediction confidence is perfect.
   The model is more confident for small `min_samples_split` because the deeper the tree, the more times we use the separation criterion. Thus, the confidence ratio logically greater because it is the objective of this separation criterion.
   For a parameter of 2, each leaf has only one value and the ratio is always 1.

2. Based on what you have observed, how could you visually guess that boundaries are associated to decision trees and that the problem is unbalanced?

   (a) We can easily see that boundaries are associated to a decision tree because it is a set of rectangles. Actually, since we have two attributes, each separation in the decision tree cuts the x,y plan in two areas until we reach a leaf. Rectangles are each time cut in two and we have new rectangles.

   (b) If we had a balanced problem, the blue and orange proportion should more or less be the same. However, it is clearly seen that the orange proportion is much greater, especially when the `min_samples_split` parameter is small. Even if this doesn't hold with a parameter of 128 or 500, it is due to the very weak precision of those models. Once it is more precise with small parameter, the difference is clear.

3. Report the average test set accuracies (over five generations of the dataset) along with the standard deviations for each value of `min_samples_split`. Briefly comment on them.

   The values in table 1 correspond well with the previous explanations. Actually, the

min_sample_split	2	8	32	64	128	500
average accuracy	0.8903	0.8878	0.9056	0.9024	0.9012	0.8718
standard deviation	0.01078239	0.01004291	0.00914002	0.00876584	0.01339254	0.01635726

Table 1: Accuracies of the decision trees

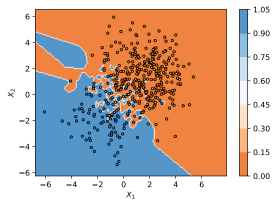
parameter 32 is the optimal point between learning sample accuracy and test sample accuracy. It corresponds to the point we have seen in the course.

2 K-nearest neighbors

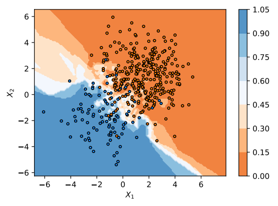
1. Observe how the decision boundary is affected by the number of neighbors

   (a) The decision boundaries for each hyper-parameter value are depicted in figure 2.

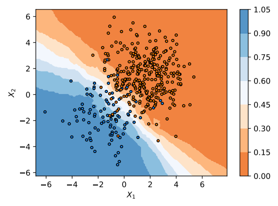
   (b) The higher the parameter, the more precise the model becomes because for a small `n_neighbors`, that creates blue areas in the orange part and vice versa. Actually, for `n_neighbors` = 1, each positive point have a blue area around it. However, when we take `n_neighbors` = 500, the model is not good because the learning sample is not large enough (more or less 250 positive values) and so, even if we are in the blue part, we will have 250 positive and 250 negative values to average which results in uncertainty rather than a confident blue prediction.



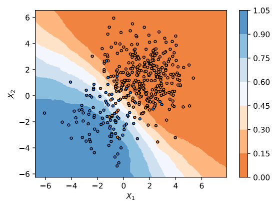
(a) n_neighbors = 1



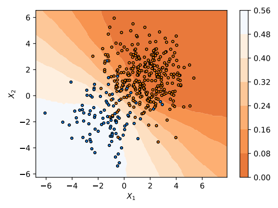
(b) n_neighbors = 5



(c) n_neighbors = 50



(d) n_neighbors = 100



(e) n_neighbors = 500

Figure 2: Decision boundaries using nearest neighbor models

2. Use a ten-fold cross validation strategy to optimize the value of the `n_neighbors` parameter

   (a) We use the ten-fold cross validation for each value of the parameter to obtain an estimate of the accuracy. That is, we divide the dataset into 10 parts, then train the model 10 times using 9/10 of the dataset and test it on the remaining 1/10, then take the average of the 10 results. Thanks to these estimates, we choose the optimal value of the `n_neighbors` parameter as the one with the greatest accuracy.

   (b) The best parameter is `n_neighbors` = 50 with an accuracy of 0.925. It is a very good accuracy and it fits well with our prediction. In fact, we said that small parameters are not accurate enough because they don't take enough points into account, and large value of the parameter becomes irrelevant with our unbalanced learning sample of 1000 values.

3. Observe the evolution of the optimal value of `n_neighbors` with respect to the size of the training set

   The evolution of mean accuracies and optimal values for `n_neighbors` are plotted in figures 3 and 4. We can see that the optimal value for k increases with the training sample size, which makes sense because bigger values of k help prevent overfitting, but if k is too big compared to the sample size, the method can't give accurate results since we have to take the average with points that are very far away.

3 Logistic regression

1. The decision boundary corresponds to when:

$$P(Y = +1|\mathbf{x_i}; w_0, \mathbf{w}) = \frac{1}{1 + \exp(-w_0 - \mathbf{w}^T \mathbf{x}_i)} = \frac{1}{2} \tag{1}$$

   This is equivalent to:

$$\mathbf{w}^T \mathbf{x}_i + w_0 = 0 \tag{2}$$

   which is a linear equation in $\mathbf{x}_i$. The boundary is therefore linear.

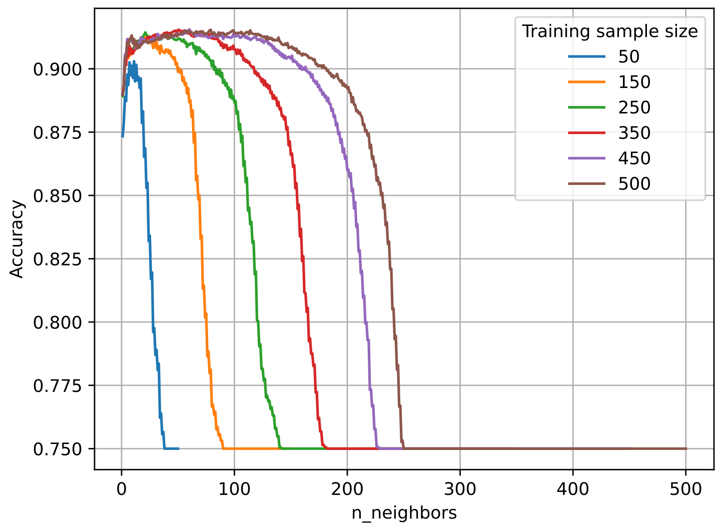


Figure 3: Evolution of mean test accuracies depending on the number of neighbors for different training sample sizes

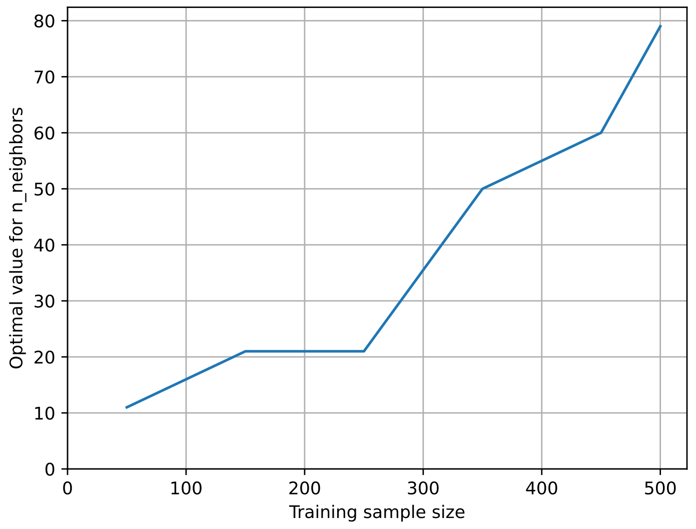


Figure 4: Optimal values for n_neighbors for different training sample sizes

2. Since the samples are independent, we have :

$$P(D|\theta) = \prod_{i=1}^{N} P(Y = y_i|\mathbf{x}_i, \theta) \tag{3}$$

Maximising this is equivalent to maximising its logarithm, since the logarithm is a stricly increasing function. We have, by the logarithm product rule:

$$\log \prod_{i=1}^{N} P(Y = y_i|\mathbf{x}_i, \theta) = \sum_{i=1}^{N} \log P(Y = y_i|\mathbf{x}_i, \theta)) \tag{4}$$

Multiplying by $-1/N$ gives us the negative log-likelihood:

$$-\frac{1}{N}\sum_{i=1}^{N}\log P(Y = y_i|\mathbf{x}_i, \theta) = \mathcal{L}(\theta) \tag{5}$$

which we should minimise if we want to maximise $P(D|\theta)$, since multiplying by $-1/N$ is a decreasing function.

3. We can start like this:

$$\begin{aligned}\nabla_\theta \mathcal{L}(\theta) &= \nabla_\theta \left[-\frac{1}{N}\sum_{i=1}^{N}\log P(Y = y_i|\mathbf{x}_i, \theta)\right] \\ &= -\frac{1}{N}\sum_{i=1}^{N}\nabla_\theta \log P(Y = y_i|\mathbf{x}_i, \theta) \\ &= -\frac{1}{N}\sum_{i=1}^{N}\frac{\nabla_\theta P(Y = y_i|\mathbf{x}_i, \theta)}{P(Y = y_i|\mathbf{x}_i, \theta)}\end{aligned} \tag{6}$$

When $y_i = 1$, we have:

$$\begin{aligned}\nabla_\theta P(Y = 1|\mathbf{x}_i, \theta) &= \nabla_\theta \frac{1}{1 + \exp(-w_0 - \mathbf{w}^T\mathbf{x}_i)} \\ &= \nabla_\theta \frac{1}{1 + \exp(-\theta\mathbf{x'}_i)} \\ &= \frac{-\nabla_\theta \exp(-\theta\mathbf{x'}_i)}{[1 + \exp(-\theta\mathbf{x'}_i)]^2} \\ &= \frac{\exp(-\theta\mathbf{x'}_i)\mathbf{x'}_i}{[1 + \exp(-\theta\mathbf{x'}_i)]^2} \\ &= P(Y = 1|\mathbf{x}_i, \theta)[1 - P(Y = 1|\mathbf{x}_i, \theta)]\mathbf{x'}_i\end{aligned} \tag{7}$$

And thus:

$$\frac{\nabla_\theta P(Y = 1|\mathbf{x}_i, \theta)}{P(Y = 1|\mathbf{x}_i, \theta)} = [1 - P(Y = 1|\mathbf{x}_i, \theta)]\mathbf{x'}_i \tag{8}$$

In the case of $y_i = 0$, we get (using the result of equation 7 again):

$$\begin{aligned}\frac{\nabla_\theta P(Y = 0|\mathbf{x}_i, \theta)}{P(Y = 0|\mathbf{x}_i, \theta)} &= \frac{\nabla_\theta [1 - P(Y = 1|\mathbf{x}_i, \theta)]}{P(Y = 0|\mathbf{x}_i, \theta)} \\ &= \frac{-P(Y = 1|\mathbf{x}_i, \theta)[1 - P(Y = 1|\mathbf{x}_i, \theta)]\mathbf{x'}_i}{P(Y = 0|\mathbf{x}_i, \theta)} \\ &= -P(Y = 1|\mathbf{x}_i, \theta)\mathbf{x'}_i\end{aligned} \tag{9}$$

Since $y_i \in \{0, 1\}$, we can summarize the results of equations 8 and 9 as:

$$\frac{\nabla_\theta P(Y = y_i|\mathbf{x}_i, \theta)}{P(Y = y_i|\mathbf{x}_i, \theta)} = [y_i - P(Y = 1|\mathbf{x}_i, \theta)]\mathbf{x'}_i \tag{10}$$

Plugging this back into equation 6, we finally get:

$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}[P(Y = 1|\mathbf{x}_i, \theta) - y_i]\mathbf{x'}_i \tag{11}$$

4. We propose to set the initial value of θ to $(0\ 0\ 0)$. In this way, it does not give any value to the outputs ($P(Y = 1|\mathbf{x}, \theta) = 1/2$ no matter the $\mathbf{x}$). It is a common way to proceed.

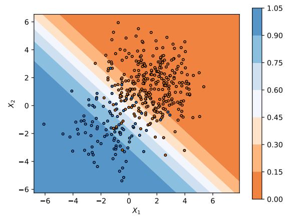
5. See figure 5. The number of iterations varies from 1 to 100 and the learning rate is equal to 1.
   The decision boundary fits well with the model. Actually, the real separation of the distribution of colors is the line y = -x that corresponds to the white area where the two distributions seems to have the same value.

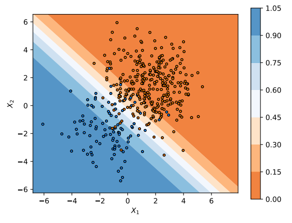
6. The average accuracy is 0.9237 and its standard deviation is 0.00587026 for 10 iterations and a learning rate of 1.

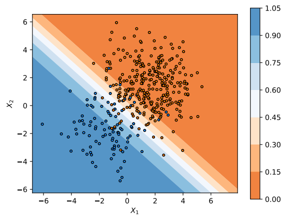
7. The average accuracies over 5 generations of the logistic regression for each number of iterations are shown in table 2. As we can see in figure 5, the decision boundary is getting sharper and sharper over the iterations, this is because the error rate decreases as well so the separation we get is more precise. The more iterations we get, the more precise we are obviously. So we can say that error rate is decreasing over iterations. It's because it keeps learning over the iterations. However it looks like the average accuracy tends to a limit and there's no real improvement between 50 or 100 iterations.

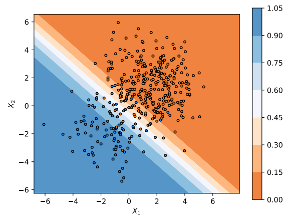
iterations	1	5	10	20	50	100
average accuracy	0.9168	0.9209	0.9237	0.9241	0.9257	0.9252
standard deviation	0.00785239	0.00745922	0.00587026	0.00543507	0.00619355	0.00613677

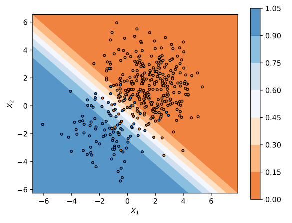
Table 2: Average accuracies of the logistic regression


(a) 1 iteration


(b) 5 iterations


(c) 10 iterations


(d) 20 iterations


(e) 50 iterations

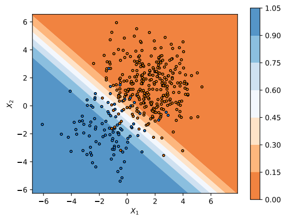

(f) 100 iterations

Figure 5: Decision boundaries using logistic regression

8. Nearest neighbors is a really good model in terms of accuracy if we take the best values of k for the training sample size. It has one of the best average accuracies over all models. However it is beaten by the logistic regression, which has a slightly better accuracy after a sufficient number of iterations. Once trained, the method is also way faster to compute and we don't need to find a good value of k that fits best the data and is dependent on the training sample size. It also has a linear boundary, which is ideal because in the case of two circular gaussian distributions, the optimal decision boundary would be a straight line.
   Compared to these two, decision trees perform rather poorly.