ELEN0062 - Introduction to machine learning
Project 1 - Classification algorithms

October 3rd, 2018

In this first project, you will glean a first experience with some machine learning algorithms. More specifically, you will experiment with the decision tree and nearest neighbor algorithms as well as implement a simple linear classification algorithm.

For each algorithm, we ask you to deliver a separate Python script. Make sure that your experiments are reproducible (e.g., by fixing manually random seeds). Add a *brief* report (pdf format, 4 pages maximum not counting the figures) 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 platform (`http://submit.montefiore.ulg.ac.be`) before *October 31, 23:59 GMT+2.* Concatenate 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. More precisely, the examples are sampled from two gaussian distribution.

In the following, you will work mainly with `make_dataset2` (see Figure 1), where one gaussian is circular and the other is elliptic. `make_dataset1`, a dataset where two identical gaussians are shifted, will only be required for question 3.

You can generate datasets of 1500 samples. The first 1200 will be used as training set and the remaining ones as testing set.

The second file contains a function which depicts the decision boundary of a trained classifier. Note that you should use a dataset independent of the training set to visualize the boundary.

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

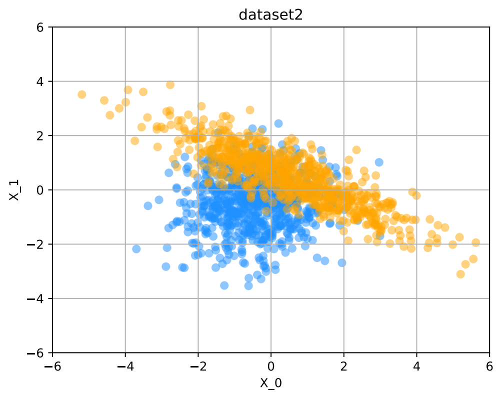

Figure 1: Second dataset.

1 Decision tree (`dt.py`)

In this section, we will study decision tree models (see the `DecisionTreeClassifier` class from `sklearn.tree`). More specifically, we will observe how model complexity impacts the classification boundary. To do so, we will build several decision tree models with `max_depth` values of $1, 2, 4, 8$ and `None` (which corresponds to an unconstrained depth). Answer the following questions in your report.

1. Observe how the decision boundary is affected by tree depth:
    (a) illustrate and explain the decision boundary for each depth;
    (b) discuss when the model is clearly underfitting/overfitting and detail your evidence for each claim;
    (c) explain why the model seems more confident when the depth is unconstrained.
2. Report the average test set accuracies (over five generations of the dataset) along with the standard deviations for each depth. Briefly comment on them.

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

In this section, we will study nearest neighbors models (see the `KNeighborsClassifier` class from `sklearn.neighbors`). More specifically, we will observe how model complexity impacts the classification boundary. To do so, we will build several nearest neighbor models with `n_neighbors` values of $1, 5, 25, 125, 625$ and 1200. Answer the following questions in your report.

1. Observe how the decision boundary is affected by the number of neighbors:
    (a) illustrate the decision boundary for each value of `n_neighbors`.
    (b) comment on the evolution of the decision boundary with respect to the number of neighbors.
2. Use a ten-fold cross validation strategy to optimize the value of the `n_neighbors` parameter:
    (a) explain your methodology;
    (b) report the score you obtain and the optimal value of `n_neighbors`. Do they corroborate your decision boundary-based intuition? Justify your answer.

3 Linear discriminant analysis (`lda.py`)

Discriminant analyses assume that each class density function $f_k(x)$ $(k = 1, \dots, K)$ is a multivariate Gaussian:

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

where $x \in \mathbb{R}^p$ is the feature vector and μ_k and Σ_k are respectively the mean and covariance matrix corresponding to class k.

Linear discriminant analysis (LDA) further assumes that the covariance matrices are equal: $\Sigma_k = \Sigma \quad \forall k$. This property is called homoscedasticity.

Applying Bayes' theorem yields that the probability of an example x belonging to class k is given by

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

where $\pi_k = P(y = k)$ $(k = 1, \dots, K)$.

And the final class is given by $\text{argmax}_k\, P(y = k|x)$.

Learning a LDA model thus requires to estimates the prior probabilities π_k $(k = 1, \dots, K)$, the means of each classes μ_k $(k = 1, \dots, K)$ and the common covariance matrix Σ. These can be inferred from the data by the usual maximum likelihood estimators. Contrary to k-nearest neighbors and decision trees, LDA does not have any hyper-parameters (but relies on the Gaussian and homoscedastic assumptions).

Implement your own LDA estimator according to the above description and following the scikit-learn convention (`http://scikit-learn.org/dev/developers/`) by filling the `lda.py` file. Note that the implementation should not be restricted to two classes.

Answer the following questions in your report.

1. In the two classes case, show that the decision boundary of LDA is linear. Comment on the shape of the decision frontier in the case of multiple classes.
2. Illustrate the decision boundary on *both* datasets and briefly comment on the results.
3. Report the average accuracy (over five generations of the dataset) along with the standard deviation for *both* datasets.
4. Based on the illustrations of the decision boundary and the average accuracies, comment on the similarities and divergences you observe between the two datasets.
   Suggestion: compare the results you obtain for LDA and for the nearest neighbor cases.