ELEN0062 - Introduction to Machine Learning
Project 1 - Classical algorithms

September 2020

The goal of this first assignment is to get accustomed to the basics of machine learning and concepts such as under- and over-fitting. We ask you to write several python 3 scripts using Scikit-Learn to answer the different questions below. One separate script is required for each of the three questions. Make sure that your experiments are reproducible (*e.g.*, by fixing manually random seeds). We ask you to write a brief report (*pdf format*) giving your observations and conclusions. Answers are expected to be concise. You will need to write several scripts to answer some of the questions below, add all of them.

The assignment must be carried out by group[1] of *two students* and submitted as a tar.gz or zip file on Montefiore's submission plateform (`http://submit.montefiore.ulg.ac.be`) before October 18, 23:59 GMT+2. Note that attention will be paid to how you present your results. Careful thoughts in particular - but not limited to - should be given when it comes to plots.

Files

You are given several files, among which `data.py` and `plot.py`.

The first one contains two functions, `make_data1` and `make_data2`, that generate samples from two different binary classification problems, each with two real input variables (so that decision boundaries can be plotted). The second file contains functions that depict the dataset together with the decision boundary of a trained classifier. Read the function specifications for further details.

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

For the three questions below, and unless otherwise stated, we ask you to carry out experiments on both problems using a learning set of 250 samples to train the models and a test set of 10000 samples to evaluate them and to depict the corresponding decision boundaries. Both learning and test sets will be generated by the functions `make_data1` and `make_data2`.

1 Decision tree (`dt.py`)

Let us first study how the complexity of the decision tree model (see the `DecisionTreeClassifier` class from `sklearn.tree`) impacts the classification boundary. To do so, we will build several decision tree models with increasing `max_depth` values of 1, 2, 4, 8, and `None` (which corresponds to an unconstrained depth and therefore an unpruned tree). Answer the following questions.

1. For both problems, 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 under- and over-fitting 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 deviation for each depth. Briefly comment on them.
3. Based on both the decision boundaries and the test accuracies, discuss the differences between the two problems.

[1] See instructions on `https://people.montefiore.uliege.be/asutera/iml.php`.

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

Let us now study how the complexity of the nearest neighbors model (see the `KNeighborsClassifier` class from `sklearn.neighbors`) impacts the classification boundary. To do so, we will build several nearest neighbor models with different values of the `n_neigbors` parameter. Answer the following questions.

1. For both datasets, observe how the decision boundary is affected by the number of neighbors:
   (a) illustrate the decision boundary for $\{1, 5, 10, 75, 100, 150\}$ `n_neighbors`;
   (b) comment on the evolution of the decision boundary with respect to the number of neighbors.
2. Optimize the value of the `n_neighbors` parameter using a five-fold cross validation strategy (see lecture 1: introduction, slide 32) and obtain an unbiased estimate of the test accuracy for the second dataset:
   (a) explain your methodology;
   (b) report the optimal value of `n_neighbors` and the mean score over the five folds. Do they corroborate your decision boundary-based intuition? Justify your answer.
3. For both datasets, observe the evolution the optimal value of the number of neighbors with respect to the size of the learning sample set. To do so:
   (a) plot the evolution curve of test accuracies (on a TS of size 500) for every possible number of neighbors for LS of sizes $\{50, 200, 250, 500\}$;
   (b) plot the optimal value of `n_neighbors` with respect to the LS size.

   Comment on them.
4. Given the results of question 2.3 and a LS of size 250, what do you think of using five-fold cross-validation to determine the optimal value of `n_neighbors` as you did in question 2.2? Discuss.

3 Residual fitting (`residual.py`)

Residual fitting is a simple algorithm to fit iteratively a linear regression model (see Lecture 3: linear regression, slide 20). We propose to implement this algorithm and to use it here to address the two classification problems by encoding the two classes in numerical values 0/1. Answer the following questions.

1. In the algorithm in the lecture slides, proof that the best weight w_k for the attribute a_k introduced in the model at step k is $\rho_{a_k,\Delta_k y}\sigma_{\Delta_k y}$, where $\rho_{a_k,\Delta_k y}$ is the Pearson correlation between a_k and $\Delta_k y$ and $\sigma_{\Delta_k y}$ is the standard deviation of $\Delta_k y$.
2. Implement the algorithm as described in the slides.
3. Learn a residual fitting model on both datasets:
   (a) illustrate the decision boundaries;
   (b) report the test accuracies.

   Comment on these results.
4. Learn a residual fitting model on a modified version of the second dataset that includes three new attributes corresponding to $X_1 * X_1$, $X_2 * X_2$ and $X_1 * X_2$, in addition to the two original ones X_1 and X_2:
   (a) illustrate the decision boundaries[2];
   (b) report the test accuracies.

   Comment on these results and compare them with those obtained in question 3.3.

[2]Use `plot_boundary_extended`.