Machine learning
Compare different supervised learning algorithms

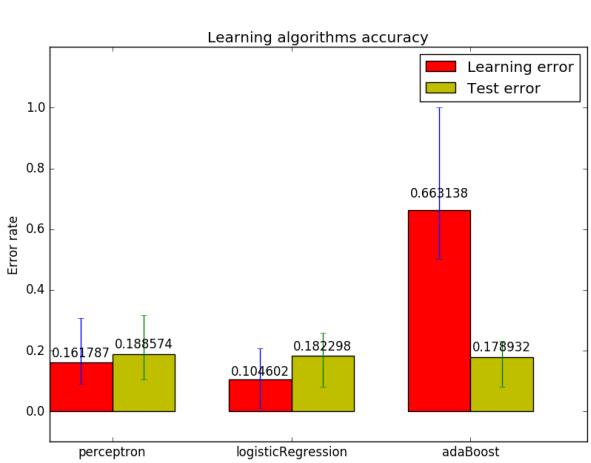
Riyane SID-LAKHDAR

June 5, 2017

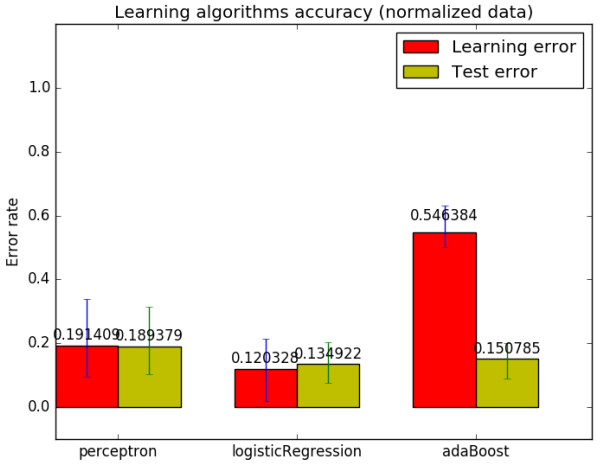
Abstract

This document presents the study that we have leaded in order to compare 3 supervised-learning algorithms, namely *Perceptron*, *Logistic Regression*, *AdaBoost*.
We present the results that we have obtained in term of learning and testing error, as well as the way they have peen processed. We also show the scope and the limit of such a benchmark.

Contents



(a) Row data



(b) Normalized data

Figure 1: Compare the accuracy of 3 supervised-learning algorithms: *Perceptron*, *Logistic Regression*, *AdaBoost*.

1 Compare the accuracy of the learning algorithms

In order to compare the accuracy of the considered learning algorithm, we have first built different set of samples. Each sample set is split into a learning and a testing set [1]. Then we have run each learning algorithm on each training set. The Figure 1 shows for each leaning algorithm the learning and testing error:

- The bar charts represent the mean error over all the test sets
- The error bar top represents the maximum value of the error over all the test sets
- The error bar bottom represents the minimum value of the error over all the test sets

Thanks to the Figure 1, one can first notice that the *Logistic Regression* learning algorithm is the most accurate considered algorithm on the considered data sets. Indeed and mainly when we consider the normalized data, the *Logistic Regression* algorithm has the lowest learning and test error considering the average, minimum and maximum errors.

Another noteworthy point on the Figure 1 is that the *AdaBoost* algorithm does not improve the test error of the *Perceptron* and the *Logistic Regression* algorithms,

[1] Randomly split with a uniform probability distribution of .6 to be in the learning set

however it has been proven to be a *boosting* algorithm [2](see previous homework) and that it combines them.
One of the possible reasons for this performance drawback is the number of iterations performed by the *AdaBoost* and the two other algorithms:
We know for instance that if there exist an optimal parameter (minimizes the learning error), then the *Perceptron* algorithm will converge to in at most $\frac{R}{\rho}^2$ steps [3]. But in most considered training sets, R is smaller than 1 (mainly for the normalized data sets). Thus this threshold iteration number will be relatively small. Meanwhile, we have proved (see previous homework) that the empirical error of the *AdaBoost* algorithm decreases exponentially with the number of iterations (number of weak classifiers) [4], the number of weak classifiers used in our implementation (3 weak classifiers) is not sufficient regarding the number of iterations that is performed (maximum 10 iterations).

2 Confidence in the obtained results

In order to assess the confidence in the previously presented results, we have computed the standard deviation of the error of each learning algorithm (after 7 consecutive runs on all the considered data set). The results are presented on Figure 2.

First, we can notice that standard deviation of the error of all the considered algorithms is smaller than $6 * 10^{-3}$. Thus, the previously computed error are pretty constant on all the considered data sets and for different iterations. Hence, the previous results have an interesting confidence property.

Second, we can notice that the standard deviation of the *Logistic Regression* algorithm is close to zero (smaller than the considered granularity). Thus, this algorithm is once again the most accurate one in terms of constancy.

3 Experiences

All the materials that have been used to produce the previously presented experiences may be found at the following address:
https://github.com/simbadSid/machineLearning_supervisedLearningComparison.git
The noteworthy directories of this project are the following:

- *resource:* Contains all the row data that are used for the training and the test steps. It also contains the data generated by our implementation using this

[2]Create a *"strong"* classifier by combining the *"weak"* classifiers *Perceptron* and Logistic Regression

[3]Novikoff, 1962, where R is the maximum training vector size and ρ the minimum labeled scalar product of a training vector and the unitary optimal parameter

[4]given that each classifier is slightly better than a coin toast

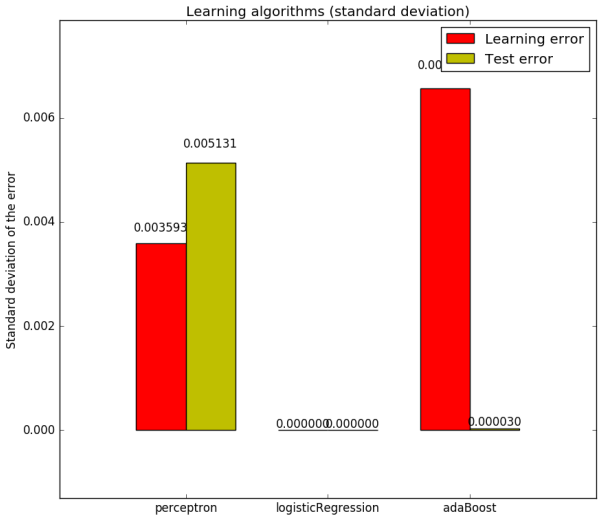


Figure 2: Standard deviation of the learning algorithm error after 7 run on each considered data set.

row data (normalized data, swap data between training and test sets)

- *src:* Contains all the source code that has been used for the experience. A unique "Makefile" has been implemented at the root of the directory. Our code has been separated as follows:
  - *learningAlgo:* contains an implementation of each considered algorithm. Each implementation has been modified (compared to the one given **at this link**) in order to communicate with the implemented result printer(see next point). We have also uniformed the outputs and corrected some bugs noticed in the original codes (memory licks dynamic miss-rooted dynamic linking).
  - *compareAlgo:* contains the python code that trains and test all the given learning algorithms. It also may build different set of data with different properties. The working and the feature of this program may be accessed by running the command
    python compareAccuracyLearningAlgo.py -help