Course: Machine learning

By: Pavlo Mozharovskyi

Tutorial to Lecture 3: Boosting algorithms

Task 1: Explore the performance of the original AdaBoost algorithm.

1. Program the original AdaBoost algorithm with the classification tree (use, *e.g.*, `R`-function `rpart` from `R`-package `rpart`). Let maximum tree depth and number of boosting iterations be the input parameters.

2. Fixing the maximum tree depth, compare performance of the original AdaBoost algorithm for different numbers of boosting iterations (say, $\{10, 25, 50, 100\}$) for the distributional setting 1 in Table 1 (Appendix from Tutorial 2). For this, train the classifiers on a randomly drawn training sample (of size 200) and evaluate its performance on a randomly drawn test sample (of size 1 000); repeat 100 times and visualize the results on a common boxplot diagram. Repeat for different maximum tree depths (say, in range $\{1, 2, 3, 4, 5\}$).

3. Repeat for the distributional setting 2 in Table 1 (Appendix from Tutorial 2).

Task 2: Comparative study of the performance of the original AdaBoost algorithm. Consider the following classification algorithms:

- Linear discriminant analysis (use `R`-function `lda`).

- Quadratic discriminant analysis (use `R`-function `qda`).

- Robustified equal-prior quadratic discriminant analysis defined as:

$$g(\boldsymbol{x}) = \underset{i\in\{0,1\}}{\arg\min}(\boldsymbol{x} - \bar{\boldsymbol{x}}_i)^T \boldsymbol{S}_i^{-1}(\boldsymbol{x} - \bar{\boldsymbol{x}}_i)$$

  with $\bar{\boldsymbol{x}}_i$ and $\boldsymbol{S}_i$ being robust mean and covariance estimates for class "i" (use `R`-function `covMcd` from `R`-package `robustbase`).

- k-nearest neighbors classifier (choose the number k of nearest neighbors by leave-one-out cross-validation, restrict k for better speed; use `R`-functions `knn` and `knn.cv` from `R`-package `class`).

- AdaBoost classifiers for different tree maximum depths (say, $\{1, 2, 3, 4, 5\}$) fixing the number of trees (say, to 100).

For each of them, train the classifier on 200 training observations and check its performance on 1 000 test observations, drawing classes in equal portions from the distributional setting 1 in Table 1 (Appendix from Tutorial 2). Perform 100 iterations with random draws, represent the error rates in form of boxplots (use R-function `boxplot` from R-package `graphics`). Interpret the results.

Repeat for the distributional setting 2 in Table 1 (Appendix from Tutorial 2).

Repeat for another distributional setting of your choice.

Task 3: Comparative study of implementation times.

Compare the speed of implementation of the LogitBoost in R-packages `gbm` (R-function `gbm`) and `xgboost` (R-function `xgboost`). On a training sample consisting of 2300 (first) training observations from the `spam` data (use R-package `kernlab`), estimate (average) training times of the classifiers for combinations of different maximum tree depths (say, $\{1, 2, 3, 4, 5\}$) and numbers of boosting iterations (say, $\{500, 1000, 2000\}$), by means of the R-function `microbenchmark` (R-package `microbenchmark`). Use the R-function `xgboost` with at least two kernels. Output the two corresponding tables containing time (in seconds), one for `gbm` and one for `xgboost`, with maximum tree depths in rows and numbers of boosting iterations in columns.