CentraleSupélec 2015-2016 Date : 19 octobre 2015
SIR & GTL

Ensemble Methods
Bureau d'Études (Machine Learning)

Objectives

This *Bureau d'Études* aim at studying ensemble methods using `python` and especially `scikit-learn`. The first part consists in studying bagging and boosting on a toy problem, the goal being notably to see what can be gained from combining base learners (the kind of base learners and how they are combined depending on the ensemble method). The second part consists in applying ensemble methods on a face recognition problem.

About `Python` and `scikit-learn`

For this *Bureau d'Études*, a very basic knowledge of `Python` is assumed. A good set of tutorials is:

- `http://www.scipy-lectures.org/`.

If you want to install `Python` on your own machine, Anaconda is a good and free Python distribution (including the Spyder editor):

- `https://www.continuum.io/downloads`.

To do the proposed exercises, it is highly advised to use `scikit-learn`, which is a Machine Learning library:

- `http://scikit-learn.org/stable/`.

Notably, using the documentation will be necessary :

- `http://scikit-learn.org/stable/user_guide.html`.

1 Toy problem

For this part, the following tutorials can be useful:

- `http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html`;
- `http://scikit-learn.org/stable/auto_examples/plot_learning_curve.html`.

The aim here is to study bagging and boosting on a binary classification toy problem where classes are two interleaving half circle with additional Gaussian noise (`sklearn.datasets.make_moons`). Therefore, whenever a risk is mentioned, it will be the risk based on the binary loss ($R(f) = \mathbb{E}[\mathbb{I}_{\{Y \neq f(X)\}}]$). The advantage of using such a toy dataset is that samples can be generated on demand (and thus, quantities such as the risk can be easily estimated).

1.1 Dataset

The documentation of the considered dataset is:

- `http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html`.

In all this part, the noise parameter will be set to 0.25.

1. To get an idea of what will be used for learning, plot the dataset for a small value of n, the number of samples (say $n = 100$).
2. To get an idea of what are the decision boundaries, plot the dataset for a large value of n (say $n = 1000$).

The "Bayes classifier" is the minimizer of the risk, without constraint on the hypothesis space. One can show that it is given by $\text{sgn}(\mathbb{E}[Y|X = x])$ (if the possible labels are ± 1). Can the risk of the Bayes classifier be null?

1.2 Bagging

In all this section, the base learner will be a fully grown classification tree.

1.2.1 Base learner

3. First, train the base learner on a large number of points (say $n = 10^4$ for example), plot the decision boundaries and estimate the risk.

Estimating the risk can be done by sampling a large number of testing points (independently from the training set), say $n = 10^4$, and by computing the empirical risk on this testing set.

4. Do the same thing with $n = 100$ training points.

Recall that the minimizer of the empirical risk is a random quantity. We would like to know how the risk evolves as a function of n.

5. Shows how the risk evolves (with mean $\pm$ standard deviation) as a function of n, for n varying from 50 to 500 (or better 1000) by steps of 50. Compute the statistics from at least 30 independent experiments.

Each experiment consists in sampling a dataset for each value of n and estimating the related risk (using again an independently generated testing set, with a large number of points). Each experiments therefore provides a curve showing the risk of the empirical minimizer as a function of n. For each independent experiment, the curve will be different (again, the minimizer is random). Therefore, show the mean curve ($\pm$ standard deviation).

1.2.2 Ensemble

Here, we will consider an ensemble of $M = 50$ base learners.

6. Train the bagged ensemble on $n = 100$ points, plot the decision boundaries and estimate the risk.

7. Shows how the risk evolves (same conditions as before). Compare this result to the one of the base learner.

1.3 Boosting

In all this section, the base learner will be a decision stump.

1.3.1 Base learner

8. First, train the base learner on a large number of points (say $n = 10^4$ for example), plot the decision boundaries and estimate the risk.

9. Do the same thing with $n = 100$ training points.

10. Shows how the risk evolves (same conditions as before).

1.3.2 Ensemble

Here, the considered boosting algorithm is AdaBoost with $M = 50$ base learners (use the `SAMME` algorithm, not the `SAMME.R` algorithm, see the documentation).

11. First, train the base learner on a large number of points (say $n = 10^4$ for example), plot the decision boundaries and estimate the risk.

12. Train the boosted ensemble on $n = 100$ points, plot the decision boundaries and estimate the risk.

13. Shows how the risk evolves (same conditions as before). Compare this result to the one of the base learner.

2 Face recognition

This part is based on the following tutorial:

- `http://scikit-learn.org/stable/auto_examples/applications/face_recognition.html`.

Based on this example, the goal is to compare the following algorithms:

- fully grown decision tree;
- shallow tree (say with a maximum of 12 nodes);
- Bagging;
- Random Forest;
- X-tree;
- Adaboost (boosting the shallow trees).

The algorithms will be trained on data (75% of the dataset used for training, the rest for testing, with a stratified K-fold cross-validation):

- unpreprocessed;
- preprocessed with an eigenface decomposition (a PCA).

For each approach, compute also:

- the training time;
- the testing time.

For each of these cases, compute (for the testing set):

- the empirical risk;
- the confusion matrix;
- the precision, the recall and the f1-score.

Some hints (a good set of parameters can also be found by doing a cross-validated grid search, see `sklearn.grid_search.GridSearchCV`, but it takes time):

- observe that the classes are not well balanced, consider the `class_weight` option;
- consider a large enough number of base learner (say 500 for all methods, even more for boosting);
- do not grow full trees for parallel approaches (for example, consider a maximum of 5 samples per node);
- to speed up AdaBoost, you can consider randomized splitting for the base learner.