AdaBoost

from Weak to Strong

Key Points of AdaBoost

- Linear ensemble of many weak classifiers with different weights.

- Adjust weights of training samples according to the prediction results in each iteration.

Linear ensemble

- Linear ensemble:

$$f(x) = \sum_{m=1}^{M} \alpha_m G_m(x)$$

M : the number of iterations

α_m: weight in m^{th} iteration

$G_m(x)$: **basic classifier** trained in m^{th} iteration

Linear ensemble

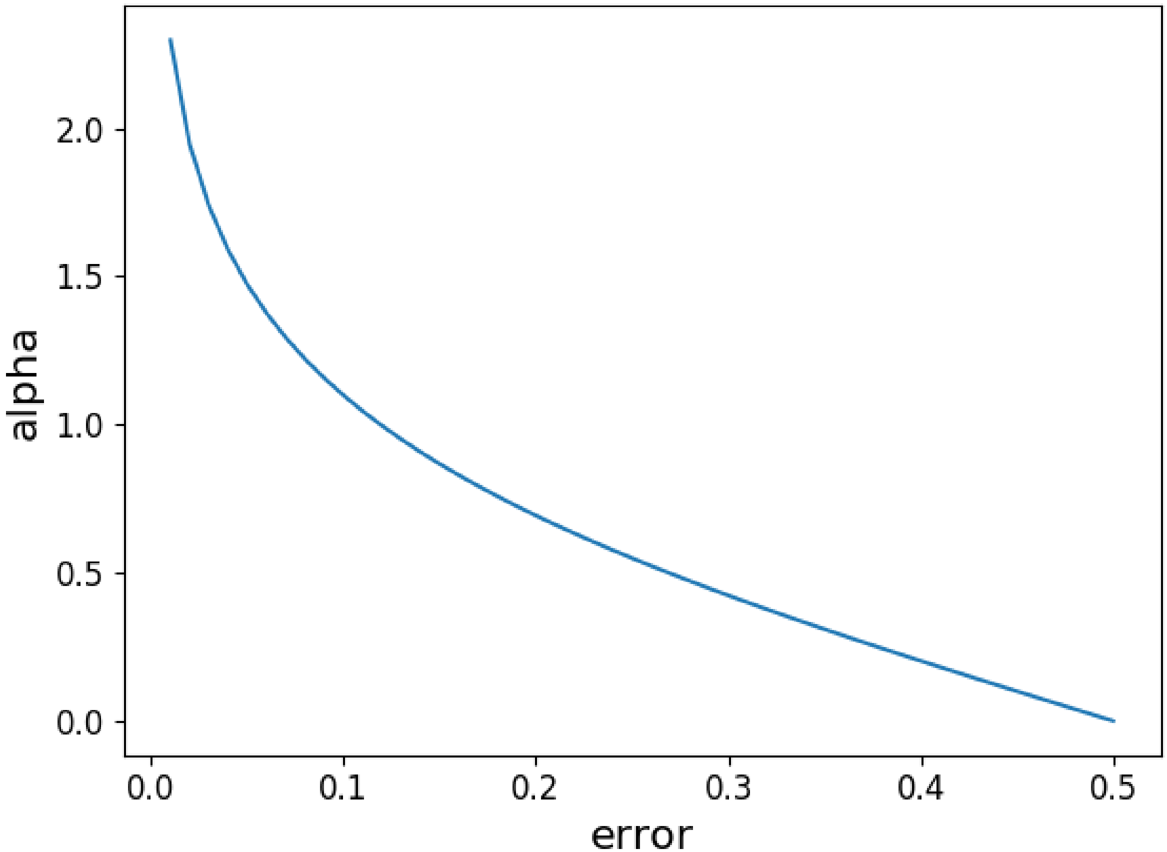
- Weight of classifier in m^{th} iteration:
$$\alpha_m = \frac{1}{2}\log\frac{1-e_m}{e_m}$$
- Weighted sum error in m^{th} iteration:
$$e_m = \sum_{i=1}^{N} w_{mi} I(G_m(x_i) \neq y_i)$$

N: number of training samples

x_i, y_i: features and label of i^{th} sample

w_{mi}: weight of i^{th} sample in m^{th} iteration

Linear ensemble



Linear ensemble

- If weak classifiers give similar classification results on training samples, similar weights will be assigned, performance limitedly improved after ensemble.

- To obtain different classifiers, **change input** by updating weight of each sample.

Weights Updating

- Weight of i^{th} sample in $(m+1)^{th}$ iteration:

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} exp(-\alpha_m y_i G_m(x_i))$$

- where Z_m is the regularization term, it makes the sum of new weights equal to 1.

$$Z_m = \sum_{i=1}^{N} w_{mi} exp(-\alpha_m y_i G_m(x_i))$$

Weights Updating

- Weight of i^{th} sample in $(m+1)^{th}$ iteration:

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} exp(-\alpha_m \boxed{y_i G_m(x_i)})$$

$$\boldsymbol{y_i == G_m(x_i)}$$

- where Z_m is the regularization term, it makes the sum of new weights equal to 1.

$$Z_m = \sum_{i=1}^{N} w_{mi} exp(-\alpha_m y_i G_m(x_i))$$

Weights Updating

- Weight of i^{th} sample in $(m+1)^{th}$ iteration:

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} exp\underbrace{\boxed{(-\alpha_m y_i G_m(x_i))}}_{\boldsymbol{negative}}$$

- where Z_m is the regularization term, it makes the sum of new weights equal to 1.

$$Z_m = \sum_{i=1}^{N} w_{mi} exp(-\alpha_m y_i G_m(x_i))$$

Weights Updating

- Weight of i^{th} sample in $(m+1)^{th}$ iteration:

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} \underbrace{\boxed{exp(-\alpha_m y_i G_m(x_i))}}_{\textbf{smaller than 1}}$$

- where Z_m is the regularization term, it makes the sum of new weights equal to 1.

$$Z_m = \sum_{i=1}^{N} w_{mi} exp(-\alpha_m y_i G_m(x_i))$$

Weights Updating

- Weight of i^{th} sample in $(m+1)^{th}$ iteration:

$$w_{m+1,i} = \boxed{\frac{w_{mi}}{Z_m} exp(-\alpha_m y_i G_m(x_i))}$$

smaller than the previous weight

- where Z_m is the regularization term, it makes the sum of new weights equal to 1.

$$Z_m = \sum_{i=1}^{N} w_{mi} exp(-\alpha_m y_i G_m(x_i))$$

Weights Updating

- Weight of i^{th} sample in $(m+1)^{th}$ iteration:

$$w_{m+1,i} = \boxed{\frac{w_{mi}}{Z_m} exp(-\alpha_m y_i G_m(x_i))}$$

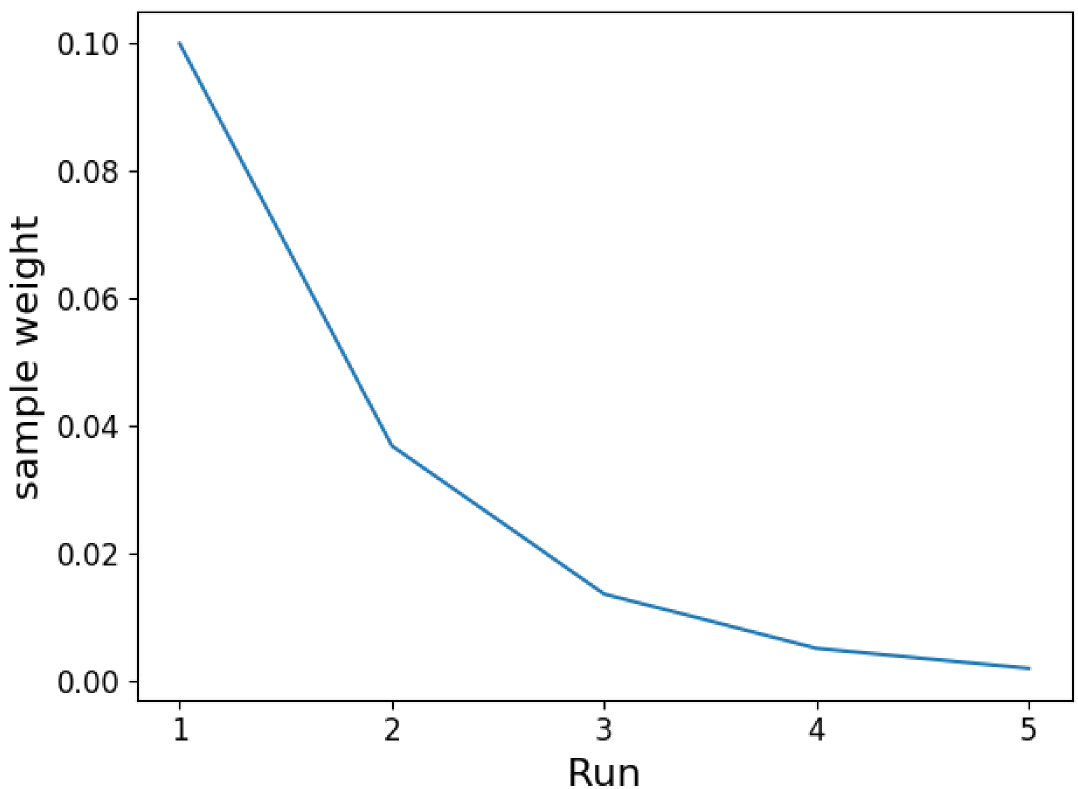
$$\boldsymbol{y_i \neq G_m(x_i) \rightarrow larger\ than\ the\ previous\ weight}$$

- where Z_m is the regularization term, it makes the sum of new weights equal to 1.

$$Z_m = \sum_{i=1}^{N} w_{mi} exp(-\alpha_m y_i G_m(x_i))$$

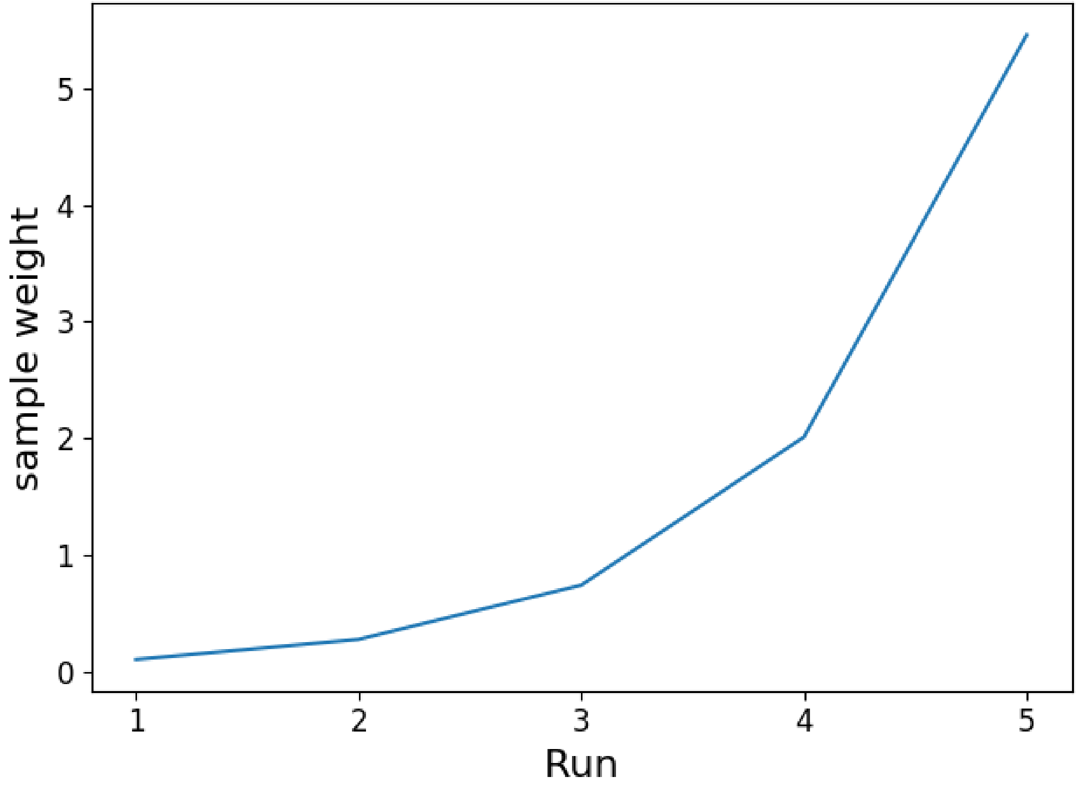
Weights Updating

- Assum w_1 = 0.1, $\alpha = 1$, the sample is correctly classified in each run. After 4 runs:



Weights Updating

- Assum w_1 = 0.1, $\alpha = 1$, the sample is falsely classified in each run. After 4 runs:



Final Classifier

- Linear ensemble:

$$f(x) = \sum_{m=1}^{M} \alpha_m G_m(x)$$

- Final classifier:

$$G(x) = sign\big(f(x)\big)$$

Pros and Cons

- Advantages:
  - Simple and elegant.
  - Fairly good generalization.
- Disadvantage:
  - Sensitive to noise and outliers.

Tests of AdaBoost

- Implement AdaBoost algorithm using Python.
- Apply decision tree as basic classifier which is generated from scikit-learn library.
- Tests on two datasets:
  - Simulated Dataset – Hastie_10_2
  - Real Dataset – Wisconsin Diagnostic Breast Cancer

  20% samples in test set.

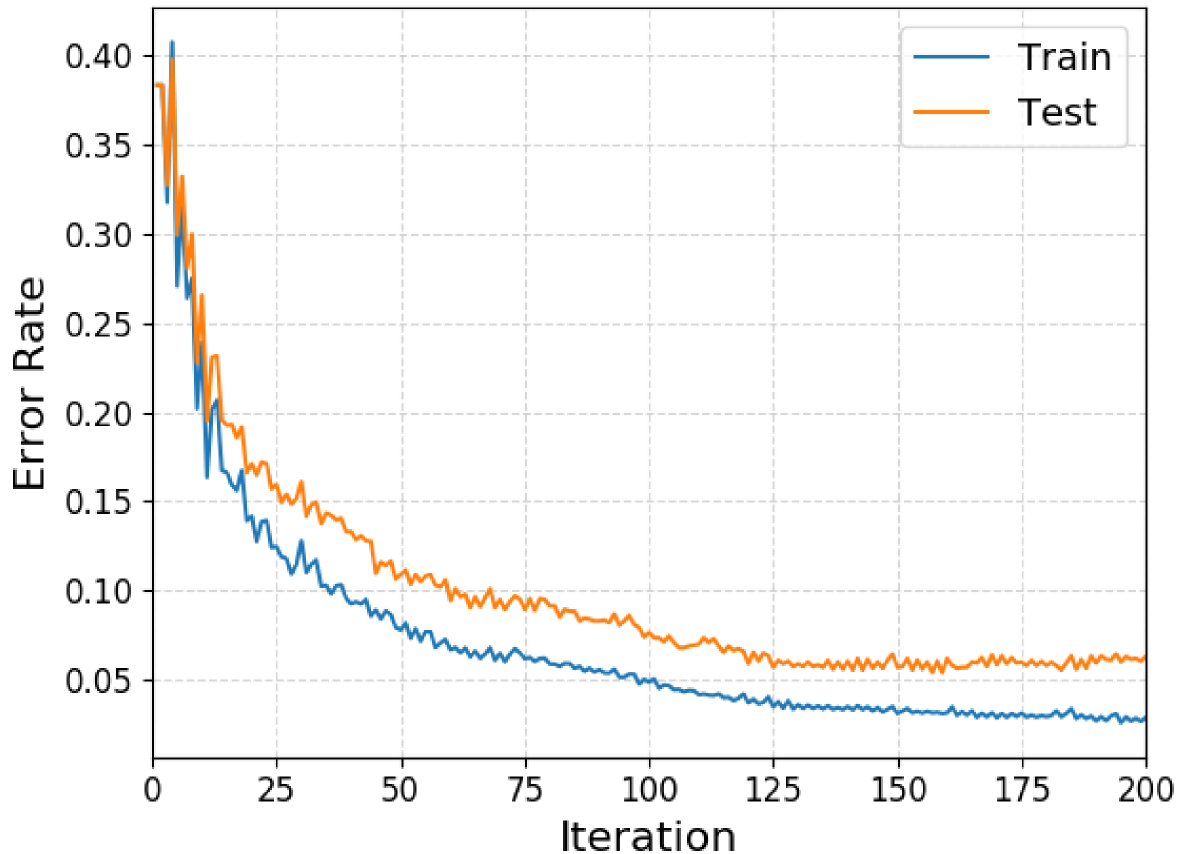
Simulated Dataset

- 10,000 samples.
- Each sample has 10 features.
- Each feature over all samples are randomly generated from Gaussian distribution.
- Label of sample is defined as:

$$y_i = \begin{cases} 1, & Z_i > 9.34 \\ -1, & otherwise \end{cases}, \qquad where\ Z_i = \sum_{j=1}^{10} {X_{i,j}}^2$$

- Two classes are balanced.

Simulated Dataset – learning curves

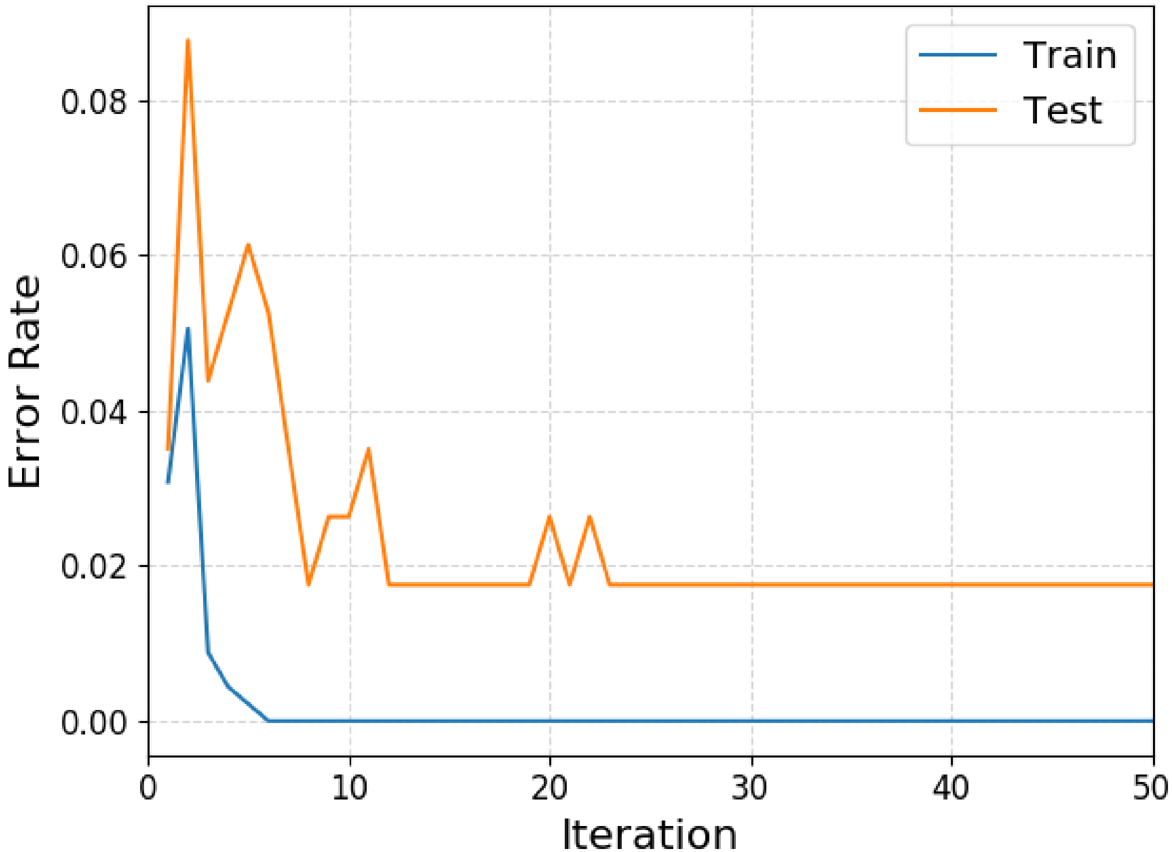


Test Acc: 0.9435

Wisconsin Diagnostic Breast Cancer (WDBC)

- 569 samples.
- Each sample has 30 features extracted from images of cell nuclei of breast mass.
- Number of “Benign”: 357
- Number of “Malignant”: 212
- See UCI Data Repo for more information.

Wisconsin Diagnostic Breast Cancer(WDBC) – learning curves



Test Acc: 0.9825

Implementation (If you want to know)

- The GitHub repo:
  https://github.com/quqixun/MLAlgorithms
- See here to create virtual python environment and install all required libraries.
- Follow instructions in folder "AdaBoost" to run code in command line.

More ensemble methods

- See the “ensemble” module of scikit-learn.
- eXtreme Gradient Boosting (XGBoost).
- Light Gradient Boosting Machine (LightGBM).

References

1. Freund, Y. and Schapire, R.E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, *55*(1), pp.119-139.
2. Schapire, R.E. and Singer, Y., 1999. Improved boosting algorithms using confidence-rated predictions. *Machine learning*, *37*(3), pp.297-336.
3. Simulated data for binary classification.