

Adaboost

He He

CDS, NYU

April 6, 2021

Boosting

Bagging Reduce variance of a low bias, high variance estimator by ensembling many estimators trained in parallel.

Boosting Reduce the error rate of a high bias estimator by ensembling many estimators trained in sequential.

- A **weak/base learner** is a classifier that does slightly better than chance.
- Weak learners are like “rules of thumb”:
 - “Viagra” \implies spam
 - From a friend \implies not spam
- **Key idea:**
 - Each weak learner focuses on different examples (*reweighted data*)
 - Weak learners have different contributions to the final prediction (*reweighted classifier*)

AdaBoost: Setting

- *Binary* classification: $\mathcal{Y} = \{-1, 1\}$
- Base hypothesis space $\mathcal{H} = \{h : \mathcal{X} \rightarrow \{-1, 1\}\}$.
- Typical base hypothesis spaces:
 - **Decision stumps** (tree with a single split)
 - Trees with few terminal nodes
 - Linear decision functions

Weighted Training Set

Each base learner is trained on weighted data.

- Training set $\mathcal{D} = ((x_1, y_1), \dots, (x_n, y_n))$.
- Weights (w_1, \dots, w_n) associated with each example.
- **Weighted empirical risk:**

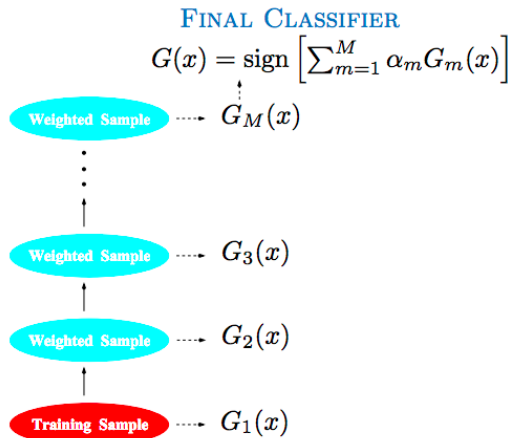
$$\hat{R}_n^w(f) \stackrel{\text{def}}{=} \frac{1}{W} \sum_{i=1}^n w_i \ell(f(x_i), y_i) \quad \text{where } W = \sum_{i=1}^n w_i$$

- Examples with larger weights have more influence on the loss.

AdaBoost - Rough Sketch

- Training set $\mathcal{D} = ((x_1, y_1), \dots, (x_n, y_n))$.
- Start with equal weight on all training points $w_1 = \dots = w_n = 1$.
- Repeat for $m = 1, \dots, M$:
 - Find base classifier $G_m(x)$ that tries to fit weighted training data (but may not do that well)
 - Increase weight on the points $G_m(x)$ misclassifies
- So far, we've generated M classifiers: $G_1, \dots, G_M : \mathcal{X} \rightarrow \{-1, 1\}$.

AdaBoost: Schematic



From ESL Figure 10.1

AdaBoost - Rough Sketch

- Training set $\mathcal{D} = \{(x_1, y_1), \dots, (x_n, y_n)\}$.
- Start with equal weight on all training points $w_1 = \dots = w_n = 1$.
- Repeat for $m = 1, \dots, M$:
 - Base learner fits weighted training data and returns $G_m(x)$
 - Increase weight on the points $G_m(x)$ misclassifies
- Final prediction $G(x) = \text{sign} \left[\sum_{m=1}^M \alpha_m G_m(x) \right]$. (recall $G_m(x) \in \{-1, 1\}$)
- What are desirable α_m 's?
 - nonnegative
 - larger when G_m fits its weighted \mathcal{D} well
 - smaller when G_m fits weighted \mathcal{D} less well

Adaboost: Weighted Classification Error

- Weights of base learners depend on their performance. How to evaluate each base learner?
- In round m , base learner gets a weighted training set.
 - Returns a base classifier $G_m(x)$ that minimizes weighted 0–1 error.
- The **weighted 0-1 error** of $G_m(x)$ is

$$\text{err}_m = \frac{1}{W} \sum_{i=1}^n w_i 1(y_i \neq G_m(x_i)) \quad \text{where } W = \sum_{i=1}^n w_i.$$

- Notice: $\text{err}_m \in [0, 1]$.

AdaBoost: Classifier Weights

- The weight of classifier $G_m(x)$ is $\alpha_m = \ln \left(\frac{1 - \text{err}_m}{\text{err}_m} \right)$.



- Higher weighted error \implies lower weight
- When is $\alpha_m < 0$?

Adaboost: Example Reweighting

- We train G_m to minimize weighted error, and it achieves err_m .
- Then $\alpha_m = \ln\left(\frac{1-err_m}{err_m}\right)$ is the weight of G_m in final ensemble.

We want the base learner to focus more on examples misclassified by the previous learner.

- Suppose w_i is weight of example i before training:
 - If G_m classifies x_i correctly, then w_i is unchanged.
 - Otherwise, w_i is increased as

$$\begin{aligned}w_i &\leftarrow w_i e^{\alpha_m} \\ &= w_i \left(\frac{1-err_m}{err_m}\right)\end{aligned}$$

- For $err_m < 0.5$ (weak learner), this always increases the weight.

AdaBoost: Algorithm

Given training set $\mathcal{D} = \{(x_1, y_1), \dots, (x_n, y_n)\}$.

- ① Initialize observation weights $w_i = 1, i = 1, 2, \dots, n$.
- ② For $m = 1$ to M :
 - ① Base learner fits weighted training data and returns $G_m(x)$
 - ② Compute *weighted empirical 0-1 risk*:

$$\text{err}_m = \frac{1}{W} \sum_{i=1}^n w_i \mathbf{1}(y_i \neq G_m(x_i)) \quad \text{where } W = \sum_{i=1}^n w_i.$$

- ③ Compute *classifier weight*: $\alpha_m = \ln \left(\frac{1 - \text{err}_m}{\text{err}_m} \right)$.
 - ④ Update *example weight*: $w_i \leftarrow w_i \cdot \exp[\alpha_m \mathbf{1}(y_i \neq G_m(x_i))]$
- ③ Return *voted classifier*: $G(x) = \text{sign} \left[\sum_{m=1}^M \alpha_m G_m(x) \right]$.

AdaBoost with Decision Stumps

- After 1 round:

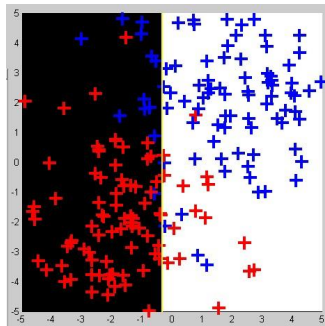


Figure: Plus size represents weight. Blackness represents score for red class.

AdaBoost with Decision Stumps

- After 3 rounds:

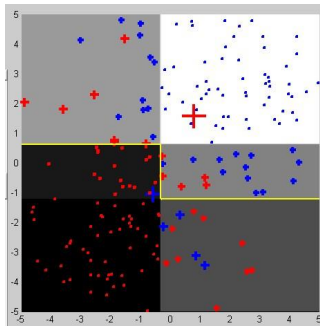


Figure: Plus size represents weight. Blackness represents score for red class.

AdaBoost with Decision Stumps

- After 120 rounds:

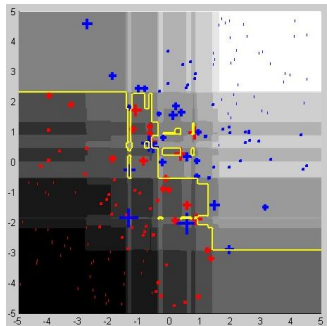
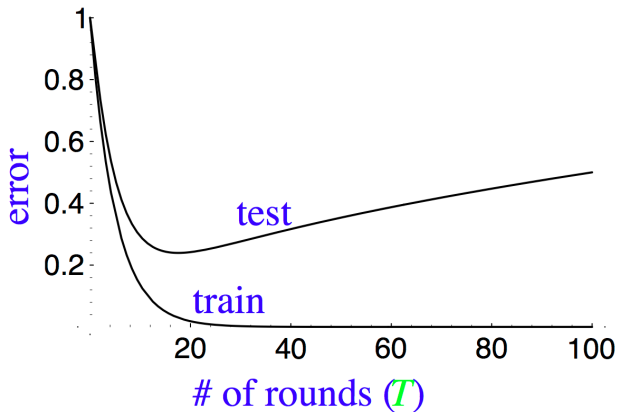


Figure: Plus size represents weight. Blackness represents score for red class.

Typical Train / Test Learning Curves

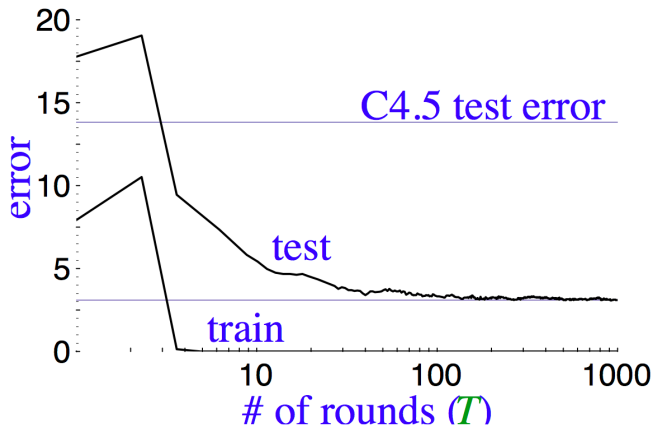
- Might expect too many rounds of boosting to overfit:



From Rob Schapire's NIPS 2007 Boosting tutorial.

Learning Curves for AdaBoost

- In typical performance, AdaBoost is surprisingly resistant to overfitting.
- Test continues to improve even after training error is zero!



Summary

- Shallow decision tree + boosting
 - “best off-the-shelf classifier in the world”—Leo Brieman
 - Used in the first successful real-time face detector (Viola and Jones, 2001)
 - **XGBoost**: very popular in competitions
- Next week
 - What is the objective function of Adaboost?
 - Generalize to other loss functions.