

# AdaBoost Algorithm

Hossein Falaki  
Computer Science Department,  
University of California, Los Angeles  
falaki@cs.ucla.edu

## 1 Introduction

This note introduces the AdaBoost algorithm. We explain the algorithm and derive the updating scheme.

## 2 AdaBoost Algorithm

Similar to SVM AdaBoost works by combining several “votes.” Instead of using support vectors (i.e., important examples), AdaBoost uses *weak learners*.

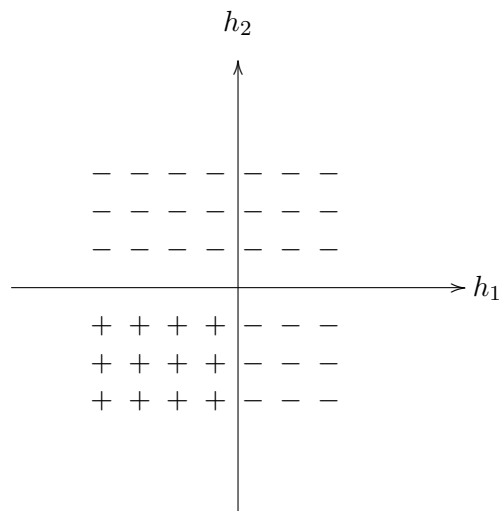


Figure 1: Neither  $h_1$  nor  $h_2$  is a perfect learner, but AdaBoost combines them to obtain a “good” learner

Figure ?? illustrates how AdaBoost combines two learners,  $h_1$  and  $h_2$ . It initially chooses the learner that classifies more data correctly. In the next

step, the data is re-weighted to increase the “importance” of misclassified samples. This process continues and at each step the weight of each week learner among other learners is determined. Therefore the algorithm is as follows:

1. Set all sample weights equal, and find  $h_1$  to maximize  $\sum_i y_i h(x_i)$ .
2. Perform re-weighting to increase the weight of the misclassified samples.
3. Find the next  $h$  to maximize  $\sum_i y_i h(x_i)$ . Find the weight of this classifier,  $\alpha$ .
4. go to step 2.

The final classifier will be:

$$\text{sgn}\left(\sum_{i=1}^T \alpha_i h_t(X)\right)$$

### 3 Objective Function

To be able to find the weight corresponding to each classifier we need to formulate an objective function and find  $\alpha$  to minimize it. The actual objective function that we seek to minimize is:

$$\sum_i 1_{y_i \neq \text{sgn}(\sum_{t=1}^T \alpha_t h_t(x_i))} \quad (1)$$

This function is non-linear and difficult to minimize, therefore the following linear function that provides an upper bound can be used:

$$\text{Obj} = \sum_{i=1}^n e^{-y_i (\sum_{t=1}^T \alpha_t h_t(x_i))} \quad (2)$$

This objective function prefers  $y_i = \text{sgn}(\sum_t \alpha_t h_t(x_i))$ :

- If  $\text{sgn}(y_i) = \text{sgn}(\sum_{t=1}^T \alpha_t h_t(x_i))$ , then  $y_i (\sum_{t=1}^T \alpha_t h_t(x_i))$  is a large positive, therefore Obj is small.
- If  $\text{sgn}(y_i) \neq \text{sgn}(\sum_{t=1}^T \alpha_t h_t(x_i))$ , then  $y_i (\sum_{t=1}^T \alpha_t h_t(x_i))$  is a large negative, therefore Obj is large.

This function prefers larger margins (because it is an upper bound), therefore it offers better training when used. As already pointed out, it is smooth and differentiable in all places.

## 4 The Updating Scheme

In this section, we derive the method to find  $\alpha_t$  at each stage of the algorithm. Consider the algorithm running at some stage:

- current committee vote:  $old(x) = \sum_t \alpha_t h_t(X)$
- current objective:  $\sum_i e^{-y_i \sum_t \alpha_t h_t(x_i)}$

We add a new classifier,  $h_{new}(x)$  to the committee with a new weight,  $\alpha_{new}$ :

- new committee vote:  $old(x) + \alpha_{new} \times h_{new}(x)$
- new objective:  $\sum_i e^{-y_i(old(x) + \alpha_{new} h_{new}(x))}$

Consider the ratio of the old objective to the new objective:

$$\frac{\text{old objective}}{\text{new objective}} = \sum_{i=1}^n w_i e^{-y_i \alpha_{new} h_{new}(x_i)}$$

where

$$w_i = \frac{e^{-y_i old(x_i)}}{\sum_i e^{-y_i old(x_i)}} = \frac{e^{-y_i old(x_i)}}{\text{sum}}$$

It can be easily seen that:

$$\text{sum}_i w_i = \frac{\text{sum}}{\text{sum}} = 1$$

In effect  $w_i$  is the weight that can be assigned to each sample for the new weak learner. It satisfies the requirements of sample weights:

- if  $y_i = \text{sgn}(old(x_i))$ , then the weight is small.
- if  $y_i \neq \text{sgn}(old(x_i))$ , then the weight is large.

To derive the new learner's weight,  $\alpha_{new}$ , assume that the old objective is fixed. We are interested to find  $\alpha_{new}$  such that the with a fixed  $h_{new}$ , the objective function is minimized. The ratio of the old and new objectives can be written as:

$$\frac{\text{old objective}}{\text{new objective}} = \underbrace{\sum_{i: y_i = h_{new}(x)} w_i e^{-\alpha_{new}}}_{1-\epsilon} + \underbrace{\sum_{i: y_i \neq h_{new}(x)} w_i e^{+\alpha_{new}}}_{\epsilon}$$

where  $\epsilon = \sum_{i: y_i \neq h_{new}(x)} w_i$  is the error rate on the re-weighted samples. Therefore:

$$\frac{\text{old objective}}{\text{new objective}} = (1 - \epsilon)e^{-\alpha_{new}} + \epsilon e^{\alpha_{new}}$$

Knowing the inequality between arithmetic and geometric means:

$$(1 - \epsilon)e^{-\alpha_{new}} + \epsilon e^{\alpha_{new}} \geq 2\sqrt{(1 - \epsilon)\epsilon}$$

Therefore the minimum of the sum happens when:

$$(1 - \epsilon)e^{-\alpha_{new}} = \epsilon e^{\alpha_{new}}$$

Therefore the optimal  $\alpha_{new}$  is:

$$\alpha_{new} = \frac{1}{2} \log \frac{1 - \epsilon}{\epsilon} \tag{3}$$

## 5 Acknowledgment

This note is based on Prof. Ying Nian Wu's lectures on Theoretical Statistics at UCLA. Figures 1 and 2 are from the Internet.