Adaptive Boosting - AdaBoosting

Given several weak learning algorithms, one can apply boosting techniques to derive a more powerful learning algorithm. AdaBoosting is a framework that shows how to do the following.

- Generate multiple weak hypotheses from any (weak) learning algorithm.
- Compute the more powerful algorithm from these hypotheses.

Given training data, a new hypothesis is generated by applying the weak learning algorithm to a sample of the training data. The key idea of AdaBoosting is to set weights to the training data so that wrongly classified data points are more likely to be considered by the new hypothesis. The implementation for the case in which the weak learner is producing either 1 or -1 (classifying into two classes) is shown below.

Input: the training data, given as N pairs (x_i, y_i), where x_i is the attributes vector, and y_i is the desired output, either 1 or -1. The number of iterations T.

Output: A function $f_T(x)$ that can be used to classify the attributes vector x.

If $f_T(x) < 0$ classify x as -1. If $f_T(x) > 0$ classify x as 1.

Initialization: Associate a probability $p_i = \frac{1}{N}$ with the ith example.

Iterate: For $t = 1, \ldots T$ compute the hypothesis h_t, a weight α_t, and an update to the probabilities $p_1, \ldots, p_N$ by the following steps:

a. Select (at random with replacements) a subset S_t of the training examples. The ith example is selected with probability p_i.

b. Compute the hypothesis h_t by applying the weak classifier to S_t.

c. Calculate the weighted training error ϵ_t of h_t:

$$\epsilon_t = \sum_{y_i \neq h_t(x_i)} p_i$$

c1. If $\epsilon_t \geq 0.5$ go back to the iteration step. This shouldn't happen too often.

c2. If $\epsilon_t = 0$ the weak classifier is not weak. Consider a re-run of the algorithm with more examples or with a weaker classifier.

d. Compute α_t, the "goodness" weight of h_t:

$$\alpha_t = \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t}$$

e. Update the probabilities. Set:

$$q_i = \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

$$\text{new } p_i = \frac{p_i q_i}{Z_t} = \frac{p_i e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$$

where Z_t is a normalization factor chosen so that $\sum_i p_i = 1$.

Termination:

$$f_T(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

The most important property of the AdaBoost algorithm is that f_{t+1} performs better than f_t in terms of an upper bound on the error in classifying the training data. In fact, choosing T large enough drives the error on the training data to 0. Experiments show that f_{t+1} almost always also performs better than f_t on test data.

To measure the algorithm performance define the following error indicator:

$$\text{wrong}(f, x_i, y_i) = \begin{cases} 1 & \text{if } f \text{ does not classify } x_i \text{ as } y_i \\ 0 & \text{if } f \text{ classifies } x_i \text{ as } y_i \end{cases}$$

The fractional error of f_T on the N training examples is:

$$E_T = \frac{\sum_{i=1}^{N} \text{wrong}(f_T, x_i, y_i)}{N}$$

Theorem: Let $g_t > 0$ be defined by the relation $\epsilon_t = \frac{1}{2} - g_t$. Then

$$E_T \leq \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)} = \prod_{t=1}^{T} \sqrt{1-4g_t^2} \leq e^{-2\sum_{t=1}^{T} g_t^2}$$

Proof: First observe that the following inequality always holds:

$$\text{wrong}(f_T, x_i, y_i) \leq e^{-y_i f_T(x_i)}$$

(If f_T is right, the left side is 0. If f_T is wrong, the left side is 1 and $-y_i f_T(x_i) > 0$.) Therefore:

$$E_T \leq \frac{1}{N} \sum_{i=1}^{N} e^{-y_i f_T(x_i)} = \frac{1}{N} \sum_{i=1}^{N} e^{-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)} \tag{1}$$

Let p_i^t denote the value of p_i at the beginning of iteration t. Then:

$$\begin{aligned} p_i^{T+1} &= \frac{p_i^T e^{-\alpha_T y_i h_T(x_i)}}{Z_T} = \frac{p_i^{T-1} e^{-\alpha_{T-1} y_i h_{T-1}(x_i)} e^{-\alpha_T y_i h_T(x_i)}}{Z_T Z_{T-1}} = \frac{p_i^1 e^{-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)}}{\prod_{t=1}^{T} Z_t} \\ &= \frac{\frac{1}{N} e^{-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)}}{\prod_{t=1}^{T} Z_t} \end{aligned}$$

Combining this with (1) we have:

$$E_T \leq \sum_{i=1}^{N} p_i^{T+1} \prod_{t=1}^{T} Z_t = \prod_{t=1}^{T} Z_t \tag{2}$$

This bound on E_T is in terms of the Z_t. It remains to express the bound in terms of ϵ_t and g_t. We have:

$$Z_t = \sum_{i=1}^{N} p_i q_i = \sum_{h_t(x_i)=y_i} p_i e^{-\alpha_t} + \sum_{h_t(x_i)\neq y_i} p_i e^{\alpha_t} = (1-\epsilon_t)e^{-\alpha_t} + \epsilon_t e^{\alpha_t}$$

Direct computation shows that Z_t is minimized for $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$ which gives:

$$Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$$

This proves the first bound in the theorem. To obtain the bound in terms of g_t, substitute $g_t = \frac{1}{2} - \epsilon_t$. The rightmost bound can be proved using the fact that $e^x + x \geq 1$ with $x = 4g_t^2$.