AdaBoost

AdaBoost, which stands for "**Ada**ptive **Boost**ing", is an ensemble learning algorithm that uses the boosting paradigm.

We assume that we are given a training set S and a pool of hypothesis functions $\mathcal{H}$ from which we are to pick T hypotheses in order to form an ensemble H. H then makes a decision using the individual hypotheses $h_1, \dots, h_T$ in the ensemble as follows:

$$H(x) = \alpha_1 h_1(x) + \cdots + \alpha_T h_T(x) \tag{1}$$

That is, H uses a linear combination of the decisions of each of the h_i hypotheses in the ensemble. The AdaBoost algorithm sequentially chooses h_i from $\mathcal{H}$ and assigns this hypothesis a weight α_i. We let H_t be the classifier formed by the first t hypotheses. That is,

$$H_t(x) = \alpha_1 h_1(x) + \cdots + \alpha_t h_t(x)$$

$$= H_{t-1}(x) + \alpha_t h_t(x)$$

For technical reasons (as seen in the derivation of the AdaBoost algorithm), we define

$$H_0(x) = 0 \tag{2}$$

That is, the empty ensemble will always output 0. The idea of the AdaBoost algorithm is that the tth hypothesis will correct for the errors that the first $t-1$ hypotheses make on the training set. More specifically, after we select the first $t-1$ hypotheses, we determine which instances in S our $m-1$ hypotheses perform poorly on and make sure that the tth hypothesis performs well on these instances. The pseudocode for AdaBoost is described in Algorithm 1. A high-level overview of the algorithm is described below:

1. Initialize a training set distribution

At each iteration $1, \dots, T$ of the AdaBoost algorithm , we define a probability distribution $\mathcal{D}$ over the training instances in S. We let $\mathcal{D}_t$ be the probability distribution at the tth iteration and $\mathcal{D}_t(x_i)$ be the probability assigned to instance x_i according to $\mathcal{D}_t$.

As the algorithm proceeds, each iteration will design $\mathcal{D}_t$ so that it assigns higher probability mass to instances that the first $t-1$ hypotheses performed poorly on. That is, the worse the performance on x_i, the higher will be $\mathcal{D}_t(x_i)$.

Algorithm 1 AdaBoost

Precondition: A training set $S := (x_1, y_1), \ldots, (x_n, y_n)$, hypothesis space $\mathcal{H}$, and number of iterations T.

for $x_i \in S$ **do**
    $\mathcal{D}_1(x_i) \leftarrow \frac{1}{n}$
end for
$H \leftarrow \emptyset$
for $t = 1, \ldots, T$ **do**
    $h_t \leftarrow \min_{h \in \mathcal{H}} P_{i \sim \mathcal{D}_t}(h(x_i) \neq y_i)$ ▷ find good hypothesis on weighted training set
    $\epsilon_t \leftarrow P_{i \sim \mathcal{D}_t}(h_t(x_i) \neq y_i)$ ▷ compute hypothesis's error
    $\alpha_t \leftarrow \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$ ▷ compute hypothesis's weight
    $H \leftarrow H \cup \{(\alpha_t, h_t)\}$ ▷ add hypothesis to the ensemble
    for $x_i \in S$ **do** ▷ update training set distribution
        $\mathcal{D}_{t+1}(x_i) \leftarrow \frac{\mathcal{D}_t(x_i)\ e^{-\alpha_t y_i h_t(x_i)}}{\sum_{j=1}^n \mathcal{D}_t(x_j)\ e^{-\alpha_t y_j h_t(x_j)}}$
    end for
end for
return H

However, at the onset of the algorithm, we set $\mathcal{D}_1$ to be the uniform distribution over the instances. That is,

$$\mathcal{D}_1(x) = \frac{1}{n}$$

for all x where n is the number of instances in S.

2. Find a new hypothesis to add to the ensemble

At the tth iteration, we find a new hypothesis h_t that performs well on the tuple $(S, \mathcal{D}_t)$. That is, it should have low expected error, denoted ϵ_t, according to $\mathcal{D}_t$. More specifically, ϵ_t is defined using the 0-1 loss:

$$\epsilon_t = P_{i \sim \mathcal{D}_t}(y_i \neq h_t(x_i))$$

3. Assign the new hypothesis a weight

Once we compute h_t, we assign h_t a weight α_t based on its performance. More specifically, we give it the weight

$$\alpha_t := \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right) \tag{3}$$

We will soon explain the theoretical justification of this precise weight assignment, but intuitively we see that the the higher ϵ_t, the the larger will be the denominator and the smaller the numerator in $\frac{1-\epsilon_t}{\epsilon_t}$. Thus, the smaller will be its value. Since the logarithm is monotonically increasing, the smaller will be $\frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$. Thus, if the h_t has a high ϵ_t, meaning it has a high error, then the smaller will be the weight that we place on this hypothesis.

4. Recompute the training set distribution

Once the new hypothesis is added to the ensemble, we recompute the training set distribution to assign each instance a probability proportional to how well the current ensemble H_t performs on the instance. More specifically, we compute $\mathcal{D}_{t+1}$ as follows:

$$\mathcal{D}_{t+1}(x_i) := \frac{\mathcal{D}_t(x_i)\ e^{-\alpha_t y_i h_t(x_i)}}{\sum_{j=1}^{n} \mathcal{D}_t(x_j)\ e^{-\alpha_t y_j h_t(x_j)}} \tag{4}$$

We will soon explain a theoretical justification for this precise probability assignment, but for now we can gain an intuitive understanding as follows: Note the term $e^{-\alpha_t y_i h_t(x_i)}$. If $h_t(x_i) = y_i$, then $y_i h_t(x_i) = 1$ which means that $e^{-\alpha_t y_i h_t(x_i)} = e^{-\alpha_t}$. If, on the other hand, $h_t(x_i) \neq y_i$, then $y_i h_t(x_i) = -1$ which means that $e^{-\alpha_t y_i h_t(x_i)} = e^{\alpha_t}$. Thus, we see that $e^{-\alpha_t y_i h_t(x_i)}$ is smaller if the hypothesis's prediction agrees with the true value. Thus, we see that higher probability is placed onto instances on which the previous hypothesis was wrong.

Repeat steps 2 through 4

Repeat steps 2 through 4 for $T - 1$ more iterations.

Theoretical Derivation

The AdaBoost algorithm can be derived if one attempts to formulate an algorithm that searches hypotheses of the form of Equation 1 in order to minimize the **exponential loss function**. The exponential loss is defined as

$$\ell_{\exp}(h, x, y) = e^{-y h(x)}$$

Note that

$$\ell_{\exp}(h,x,y) = \begin{cases} e : h(x) \neq y \\ \\ \frac{1}{e} : h(x) = y \end{cases}$$

There are many ways in which one might search for a hypothesis of the form of Equation 1 in order to minimize the exponential loss function. The AdaBoost algorithm does so using a sequential optimization procedure in which a we iteratively add a new hypothesis h_t together multiplied by weight α_t to H that minimizes exponential loss function. Stated differently, at iteration t, we are given

$$H_t(x) = H_{t-1}(x) + \alpha_t h_t(x)$$

and our goal is to choose a h_t and α_t that minimizes H_t according to the exponential loss function on the training data. This is a form of minimizing empirical loss. The empirical loss of H_t is

$$\begin{aligned} L_S(H_t) &= \frac{1}{n}\sum_{i=1}^{n} e^{-y_i H_t(x_i)} \\ &= \frac{1}{n}\sum_{i=1}^{n} e^{-y_i[H_{t-1}(x_i)+\alpha_t h_t(x_i)]} \\ &= \frac{1}{n}\sum_{i=1}^{n} e^{-y_i H_{t-1}(x_i)} e^{-y\alpha_t h_t(x_i)} \\ &= \frac{1}{n}\sum_{i=1}^{n} w_{t,i} e^{-y\alpha_t h_t(x_i)} && \text{let } w_{t,i} := e^{-y_i H_{t-1}(x_i)} \end{aligned}$$

Let us first try to find h_t for a fixed α_t

$$\begin{aligned} \underset{h_t\in\mathcal{H}}{\operatorname{argmin}} L_S(H_{t-1}(x) + \alpha_t h_t(x)) &= \underset{h_t\in\mathcal{H}}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n} w_{t,i} e^{-y\alpha_t h_t(x_i)} \\ &= \underset{h_t\in\mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{n} w_{t,i} e^{-y\alpha_t h_t(x_i)} \end{aligned}$$

Now we divide this summation into the terms where $h(x_i) = y_i$ and where $h(x_i) \neq y_i$

$$= \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} \left\{ \sum_{i:h(x_i)=y_i} w_{t,i} e^{-\alpha_t} + \sum_{i:h(x_i)\neq y_i} w_{t,i} e^{\alpha_t} \right\}$$

$$= \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} \left\{ \left(\sum_{i=1}^{n} w_{t,i} e^{-\alpha_t} - \sum_{i:h(x_i)\neq y_i} w_{t,i} e^{-\alpha_t} \right) + \sum_{i:h(x_i)\neq y_i} w_i e^{\alpha_t} \right\}$$

$$= \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} w_{t,i} e^{-\alpha_t} + \sum_{i:h(x_i)\neq y_i} w_{t,i} (e^{\alpha_t} - e^{-\alpha_t}) \right\}$$

$$= \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} \left\{ K + \sum_{i:h(x_i)\neq y_i} w_{t,i} (e^{\alpha_t} - e^{-\alpha_t}) \right\} \qquad K := \sum_{i=1}^{n} w_i e^{-\alpha_t} \text{ is a constant}$$

$$= \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} \left\{ (e^{\alpha_t} - e^{-\alpha_t}) \sum_{i:h(x_i)\neq y_i} w_{t,i} \right\}$$

$$= \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} \left\{ (e^{\alpha_t} - e^{-\alpha_t}) \frac{1}{\sum_{j=1}^{n} w_{t,j}} \sum_{i:h(x_i)\neq y_i} w_{t,i} \right\}$$

$$= \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} \left\{ (e^{\alpha_t} - e^{-\alpha_t}) \sum_{i:h(x_i)\neq y_i} \frac{w_{t,j}}{\sum_{j=1}^{n} w_{t,j}} \right\}$$

$$= \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} \left\{ \sum_{i:h(x_i)\neq y_i} \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \right\}$$

We see that the h_t that minimizes the above equation is the h_t that minimizes the "weighted error" which we define as

$$\epsilon_{\text{weighted}} = \sum_{i:h(x_i)\neq y_i} \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$

We will now show that this weighted error is identical to the expected 0-1 loss over the distribution $\mathcal{D}_t$. That is, we want to show that

$$\sum_{i:h(x_i)\neq y_i} \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} = P_{i\sim\mathcal{D}_t}(y_i \neq h_t(x_i))$$

Proof:

First, we show that by Equation 4, it is true that

$$\mathcal{D}_t(x) = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \tag{5}$$

Proof of Equation 5

We prove this fact by induction. First, we prove the base case. We need to show that

$$\frac{w_{1,i}}{\sum_{j=1}^{n} w_{1,j}} = \frac{1}{n}$$

$$= \mathcal{D}_1(x_i)$$

This is proven as follows:

$$w_{1,i} := e^{-y_i H_0(x_i)}$$

$$= 1 \qquad \text{because } H_0(x_i) = 0 \text{ by Equation 2}$$

Next, we need to prove the inductive step. That is, we prove that

$$\mathcal{D}_t(x) = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \implies \mathcal{D}_{t+1}(x) = \frac{w_{t+1,i}}{\sum_{j=1}^{n} w_{t+1,j}}$$

This is proven as follows:

$$\mathcal{D}_{t+1}(x_i) := \frac{\mathcal{D}_t(x_i)\ e^{-\alpha_t y_i h_t(x_i)}}{\sum_{j=1}^n \mathcal{D}_t(x_j)\ e^{-\alpha_t y_j h_t(x_j)}} \qquad \text{by Equation 4}$$

$$= \frac{\frac{w_{t,i}}{\sum_{j=1}^n w_{t,j}}\ e^{-\alpha_t y_i h_t(x_i)}}{\sum_{j=1}^n \frac{w_{t,j}}{\sum_{k=1}^n w_{t,k}}\ e^{-\alpha_t y_j h_t(x_j)}} \qquad \text{by the inductive hypothesis}$$

$$= \frac{\frac{e^{-y_i H_{t-1}(x_i)}}{\sum_{j=1}^n e^{-y_j H_{t-1}(x_j)}}\ e^{-\alpha_t y_i h_t(x_i)}}{\sum_{j=1}^n \frac{e^{-y_j H_{t-1}(x_j)}}{\sum_{k=1}^n e^{-y_k H_{t-1}(x_k)}}\ e^{-\alpha_t y_j h_t(x_j)}} \qquad \text{by the fact that } w_{t,i} := e^{-y_i H_{t-1}(x_i)}$$

$$= \frac{\frac{1}{\sum_{j=1}^n e^{-y_j H_{t-1}(x_j)}}\ e^{-y_i H_{t-1}(x_i)}\ e^{-\alpha_t y_i h_t(x_i)}}{\frac{1}{\sum_{k=1}^n e^{-y_k H_{t-1}(x_k)}} \sum_{j=1}^n\ e^{-y_j H_{t-1}(x_j)}\ e^{-\alpha_t y_j h_t(x_j)}}$$

$$= \frac{e^{-y_i H_{t-1}(x_i) - \alpha_t y_i h_t(x_i)}}{\sum_{j=1}^n\ e^{-y_j H_{t-1}(x_j) - \alpha_t y_j h_t(x_j)}}$$

$$= \frac{e^{-y_i H_t(x_i)}}{\sum_{j=1}^n\ e^{-y_j H_t(x_j)}}$$

$$= \frac{w_{t+1,i}}{\sum_{j=1}^n w_{t+1,j}}$$

◇

Now,

$$\sum_{i:h(x_i)\neq y_i} \frac{w_{t,i}}{\sum_{j=1}^n w_{t,j}} = \sum_{i:h(x_i)\neq y_i} \mathcal{D}_t(x_i)$$

$$= P_{i\sim\mathcal{D}_t}(y_i \neq h_t(x_i))$$

□

Thus, we have shown that the best h_t for a fixed α_t at iteration t is the hypothesis that minimizes the expected 0-1 loss according to distribution $\mathcal{D}_t$.

We now prove that the α_t that minimizes the exponential loss is given by Equation 3.

Proof:

We want to show that Equation 3 is the solution to

$$\operatorname{argmin}_{\alpha_t} \left\{ \left(\sum_{i:h(x_i) \neq y_i} w_{t,i} \right) e^{\alpha_t} + \left(\sum_{i:h(x_i) = y_i} w_{t,i} \right) e^{-\alpha_t} \right\}$$

To do so, we take the derivative of this function and set it to zero in order to solve for α_t (the function is convex, though we don't prove it here).

$$\frac{d\left[\left(\sum_{i:h(x_i)\neq y_i} w_{t,i}\right)e^{\alpha_t} + \left(\sum_{i:h(x_i)=y_i} w_{t,i}\right)e^{-\alpha_t}\right]}{d\alpha_t} = 0$$

$$\implies \left(\sum_{i:h(x_i)\neq y_i} w_{t,i}\right)e^{\alpha_t} - \left(\sum_{i:h(x_i)=y_i} w_{t,i}\right)e^{-\alpha_t} = 0$$

$$\implies e^{2\alpha_t} = -\frac{\sum_{i:h(x_i)\neq y_i} w_{t,i}}{\sum_{i:h(x_i)=y_i} w_{t,i}}$$

$$\implies 2\alpha_t = \ln\left(-\frac{\sum_{i:h(x_i)\neq y_i} w_{t,i}}{\sum_{i:h(x_i)=y_i} w_{t,i}}\right)$$

$$\implies \alpha_t = \frac{1}{2}\ln\left(\frac{\sum_{i:h(x_i)=y_i} w_{t,i}}{\sum_{i:h(x_i)\neq y_i} w_{t,i}}\right)$$

$$\implies \alpha_t = \frac{1}{2}\ln\left(\frac{\sum_{i=1}^{n} w_{t,i} - \sum_{i:h(x_i)\neq y_i} w_{t,i}}{\sum_{i:h(x_i)\neq y_i} w_{t,i}}\right)$$

$$\implies \alpha_t = \frac{1}{2}\ln\left(\frac{\frac{1}{\sum_{i=1}^{n} w_{t,i}}}{\frac{1}{\sum_{i=1}^{n} w_{t,i}}}\frac{\sum_{i=1}^{n} w_{t,i} - \sum_{i:h(x_i)\neq y_i} w_{t,i}}{\sum_{i:h(x_i)\neq y_i} w_{t,i}}\right)$$

$$\implies \alpha_t = \frac{1}{2}\ln\left(\frac{1 - \frac{\sum_{i:h(x_i)\neq y_i} w_{t,i}}{\sum_{i=1}^{n} w_{t,i}}}{\frac{\sum_{i:h(x_i)\neq y_i} w_{t,i}}{\sum_{i=1}^{n} w_{t,i}}}\right)$$

$$\implies \alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

Thus, the optimal weight for hypothesis h_t is $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$.

This concludes the derivation of AdaBoost.

□