Section 17

Classification problem. Boosting.

Suppose that we have the data $(X_1, Y_1), \ldots, (X_n, Y_n)$ that consist of pairs (X_i, Y_i) such that X_i belongs to some set $\mathcal{X}$ and Y_i belongs to a set $\mathcal{Y} = \{+1, -1\}$. We will think of Y_i as a label of X_i so that all points in the set $\mathcal{X}$ are divided into two classes corresponding to labels ± 1. For example, X_is can be images or representations of images and Y_is classify whether the image contains a human face or not. Given this data we would like to find a classifier

$$f : \mathcal{X} \to \mathcal{Y}$$

which given a point $X \in \mathcal{X}$ would predict its label Y. This type of problem is called classification problem. In general, there may be more than two classes of points which means that the set of labels may consist of more than two points but, for simplicity, we will consider the simplest case when we have only two labels ± 1.

We will take a look at one approach to this problem called boosting and, in particular, prove one interesting property of the algorithm called AdaBoost.

Let us assume that we have a family of classifiers

$$\mathcal{H} = \{h : \mathcal{X} \to \mathcal{Y}\}.$$

Suppose that we can find many classifiers in $\mathcal{H}$ that can predict labels Y_i better than "tossing a coin" which means that they predict the correct label at least half of the time. We will call $\mathcal{H}$ a family of *weak classifiers* because we do not require much of them, for example, all these classifiers can make mistakes on, let's say, 30% or even 45% of the sample.

The idea of boosting consists in trying to combine these weak classifiers so that the combined classifier predicts the label correctly most of the time. Let us consider one particular algorithm called Adaboost.

Given weights $w(1), \ldots, w(n)$ that add up to one we define the weighted classification error of the classifier h by

$$w(1)I(h(X_1) \neq Y_1) + \ldots + w(n)I(h(X_n) \neq Y_n).$$

AdaBoost algorithm. We start by assigning equal weights to the data points:

$$w_1(1) = \ldots = w_1(n) = \frac{1}{n}.$$

Then for $t = 1, \ldots, T$ we repeat the following cycle:

1. Find $h_t \in \mathcal{H}$ such that weighted error

$$\varepsilon_t = w_t(1)I(h_t(X_1) \neq Y_1) + \ldots + w_t(n)I(h_t(X_n) \neq Y_n)$$

   is as small as possible.

2. Let $\alpha_t = \frac{1}{2}\log\frac{1-\varepsilon_t}{\varepsilon_t}$ and update the weights:

$$w_{t+1}(i) = w_t(i)\frac{e^{-\alpha_t Y_i h_t(X_i)}}{Z_t},$$

   where

$$Z_t = \sum_{i=1}^{n} w_t e^{-\alpha_t Y_i h_t(X_i)}$$

   is the normalizing factor to ensure that updated weights add up to one.

After we repeat this cycle T times we output the function

$$f(X) = \alpha_1 h_1(X) + \ldots + \alpha_T h_T(X)$$

and use $\text{sign}(f(X))$ as the prediction of label Y.

First of all, we can assume that the weighted error ε_t at each step t is less than 0.5 since, otherwise, if we make a mistake more than half of the time we should simply predict the opposite label. For $\varepsilon_t \leq 0.5$ we have,

$$\alpha_t = \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t} \geq 0.$$

Also, we have

$$Y_i h_t(X_i) = \begin{cases} +1 & \text{if } h_t(X_i) = Y_i \\ -1 & \text{if } h_t(X_i) \neq Y_i. \end{cases}$$

Therefore, if h_t makes a mistake on the example (X_i, Y_i) which means that $h_t(X_i) \neq Y_i$ or, equivalently, $Y_i h_t(X_i) = -1$ then

$$w_{t+1}(i) = \frac{e^{-\alpha_t Y_i h_t(X_i)}}{Z_t} w_t(i) = \frac{e^{\alpha_t}}{Z_t} w_t(i).$$

On the other hand, if h_t predicts the label Y_i correctly then $Y_i h_t(X_i) = 1$ and

$$w_{t+1}(i) = \frac{e^{-\alpha_t Y_i h_t(X_i)}}{Z_t} w_t(i) = \frac{e^{-\alpha_t}}{Z_t} w_t(i).$$

Since $\alpha_t \geq 0$ this means that we increase the relative weight of the ith example if we made a mistake on this example and decrease the relative weight if we predicted the label Y_i

correctly. Therefore, when we try to minimize the weighted error at the next step $t+1$ we will pay more attention to the examples misclassified at the previous step.

Theorem: *The proportion of mistakes made on the data by the output classifier* $\text{sign}(f(X))$ *is bounded by*

$$\frac{1}{n}\sum_{i=1}^{n} I(\text{sign}(f(X_i))) \neq Y_i) \leq \prod_{t=1}^{T} \sqrt{4\varepsilon_t(1-\varepsilon_t)}.$$

Remark: If the weighted errors ε_t will be strictly less than 0.5 at each step meaning that we predict the labels better than tossing a coin then the error of the combined classifer will decrease exponentially fast with the number of rounds T. For example, if $\varepsilon_t \leq 0.4$ then $4\epsilon_t(1-\epsilon_t) \leq 4(0.4)(0.6) = 0.96$ and the error will decrease as fast as 0.96^T.

Proof. Using that $I(x \leq 0) \leq e^{-x}$ as shown in figure 17.1 we can bound the indicator of making an error by

$$I(\text{sign}(f(X_i)) \neq Y_i) = I(Y_i f(X_i) \leq 0) \leq e^{-Y_i f(X_i)} = e^{-Y_i \sum_{t=1}^{T} \alpha_t h_t(X_i)}. \tag{17.0.1}$$

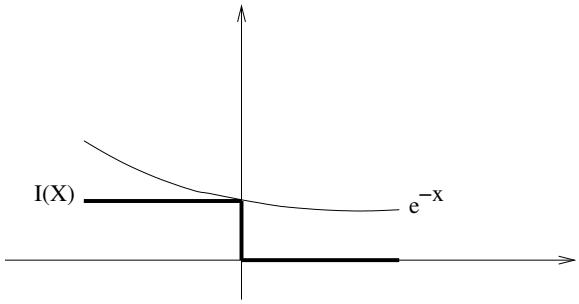

Figure 17.1: Example.

Next, using the step 2 of AdaBoost algorithm which describes how the weights are updated we can express the weights at each step in terms of the weights at the previous step and we can write the following equation:

$$\begin{aligned} w_{T+1}(i) &= \frac{w_T(i)e^{-\alpha_T Y_i h_T(X_i)}}{Z_T} = \frac{e^{-\alpha_T Y_i h_T(X_i)}}{Z_T}\frac{w_{T-1}(i)e^{-\alpha_{T-1} Y_i h_{T-1}(X_i)}}{Z_{T-1}} \\ &= \text{repeat this recursively over } t \\ &= \frac{e^{-\alpha_T Y_i h_T(X_i)}}{Z_T}\frac{e^{-\alpha_{T-1} Y_i h_{T-1}(X_i)}}{Z_{T-1}} \cdots \frac{e^{-\alpha_1 Y_i h_1(X_i)}}{Z_1} w_1(i) = \frac{e^{-Y_i f(X_i)}}{\prod_{t=1}^{T} Z_t}\frac{1}{n}. \end{aligned}$$

This implies that

$$\frac{1}{n}e^{-Y_i f(X_i)} = w_{T+1}(i)\prod_{t=1}^{T} Z_t.$$

Combining this with (17.0.1) we can write

$$\frac{1}{n}\sum_{i=1}^{n} I(\text{sign}(f(X_i) \neq Y_i)) \leq \sum_{i=1}^{n}\frac{1}{n}e^{-Y_i f(X_i)} = \prod_{t=1}^{T} Z_t \sum_{i=1}^{n} w_{T+1}(i) = \prod_{t=1}^{T} Z_t. \tag{17.0.2}$$

Next we will compute

$$Z_t = \sum_{i=1}^{n} w_t(i) e^{-\alpha_t Y_i h_t(X_i)}.$$

As we have already mentioned above, $Y_i h_t(X_i)$ is equal to -1 or $+1$ depending on whether h_t makes a mistake or predicts the label Y_i correctly. Therefore, we can write,

$$\begin{aligned} Z_t &= \sum_{i=1}^{n} w_t(i) e^{-\alpha_t Y_i h_t(X_i)} = \sum_{i=1}^{n} w_t(i) I(Y_i = h_t(X_i)) e^{-\alpha_t} + \sum_{i=1}^{n} w_t(i) I(Y_i \neq h_t(X_i)) e^{\alpha_t} \\ &= e^{-\alpha_t} (1 - \underbrace{\sum_{i=1}^{n} w_t(i) I(Y_i \neq h_t(X_i))}_{\varepsilon_t}) + e^{\alpha_t} \underbrace{\sum_{i=1}^{n} w_t(i) I(Y_i = h_t(X_i))}_{\varepsilon_t} \\ &= e^{-\alpha_t}(1 - \varepsilon_t) + e^{\alpha_t} \varepsilon_t. \end{aligned}$$

Up to this point all computations did not depend on the choice of α_t but since we bounded the error by $\prod_{t=1}^{T} Z_t$ we would like to make each Z_t as small as possible and, therefore, we choose α_t that minimizes Z_t. Simple calculus shows that we should take $\alpha_t = \frac{1}{2} \log \frac{1-\varepsilon_t}{\varepsilon_t}$ which is precisely the choice made in AdaBoost algorithm. For this choice of α_t we get

$$Z_t = (1 - \varepsilon_t) \sqrt{\frac{\epsilon_t}{1 - \epsilon_t}} + \varepsilon_t \sqrt{\frac{1 - \epsilon_t}{\epsilon_t}} = \sqrt{4 \varepsilon_t (1 - \varepsilon_t)}$$

and plugging this into (17.0.2) finishes the proof of the bound.

□