# Adaboost

Kangcheng Hou

Zhejiang University

May 17, 2018

# Introduction I

We are trying to build a classification model. We have training data points

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$$

. Every data points has a weight that changes over training process. For example, in the first round, data point $(\mathbf{x}_i, y_i)$ has weight $w_{1,i}$. With $M$ iterations, we have $M$ classifiers $D_1, \ldots, D_M$ trained on data points

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$$

with weight

$$\{(w_{1,1}, \ldots, w_{1,N}), \ldots, (w_{M,1}, \ldots w_{M,N})\}$$

# Introduction II

Our algorithm is thus as follows:

1. initialize all weight $w_{1,1}, \ldots, w_{1,N}$ as $\frac{1}{N}$.
2. for k = 1 : K
3. train a classifier $C_k$ according to $w_{k,1}, \ldots, w_{k,N}$
4. get the training error $E_k$ of classifier $C_k$ w.r.t $w_{k,1}, \ldots, w_{k,N}$
5. get the weight of the classifier $C_k$, $\alpha_k = \frac{1}{2} \ln \frac{1-E_k}{E_k}$
6. update the training data weights
$$w_{k+1,i} \propto w_{k,i} \begin{cases} e^{-\alpha_k} & C_k(\mathbf{x}_i) = y_i \\ e^{\alpha_k} & C_k(\mathbf{x}_i) \neq y_i \end{cases}$$
7. done!

We use

$$g(x) = \sum_k \alpha_k C_k(x)$$

as the final classfier.

# Derivation I

So the natural question to ask is why such algorithm? The basic adea is as follows: after the $m-1$ iteration, the boosted classifier is a linear combination of the weak classifiers,

$$C_{(m-1)}(x) = \sum_{i=1}^{K} \alpha_i k_i(x)$$

At the $m$ iteration we want to extend this to a better boosted classifier by adding a weak classfier:

$$C_{(m)}(x) = C_{(m-1)}(x) + \alpha_m k_m(x)$$

So the question becomes how do we choose the $\alpha_m$ and $k_m(x)$? We define the total error $E$ of $C_m$ as

$$E = \sum_{i=1}^{N} e^{-y_i C_m(x_i)}$$

# Derivation II

Note that we are trying to optimize w.r.t $\alpha_m$ and $k_m(x)$, so we define $w_i^(m) = e^{-y_i C_{m-1}(x_i)}$ and we have

$$E = \sum_{i=1}^{N} w_i^{(m)} e^{-y_i \alpha_m k_m(x_i)}$$

We do further transformation of $E$ as:

$$\begin{aligned} E &= \sum_{y_i = k_m(x_i)} w_i^{(m)} e^{-\alpha_m} + \sum_{y_i \neq k_m(x_i)} w_i^{(m)} e^{\alpha_m} \\ &= \sum_{i=1}^{N} w_i^{(m)} e^{-\alpha_m} + \sum_{y_i \neq k_m(x_i)} w_i^{(m)} (e^{\alpha_m} - e^{-\alpha_m}) \end{aligned}$$

# Derivation III

So the only part that depends on $k_m$ is $\sum_{y_i \neq k_m(x_i)} w_i^{(m)}$, we optimize $k_m(x)$ w.r.t this error. Note that the choice of $k_m(x)$ does not depend on $\alpha_m$. And we further determine the weight $\alpha_m$ that minimizes $E$ with the $k_m$ that we just determined.

$$\frac{dE}{d\alpha_m} = \frac{d(\sum_{y_i = k_m(x_i)} w_i^{(m)} e^{-\alpha_m} + \sum_{y_i \neq k_m(x_i)} w_i^{(m)} e^{\alpha_m})}{d\alpha_m}$$

which is

$$\alpha_m = \frac{1}{2} \ln(\frac{1 - \epsilon_m}{\epsilon_m})$$

Thus we first optimize w.r.t the $k_m(x)$ and find the best $\alpha_m$ w.r.t $E$ given that $k_m(x)$ is fixed.

# Viewpoint from Learning Theory I