Boosting

David S. Rosenberg

1 AdaBoost (Freund-Schapire '95)

[slight notational(?) differences from lecture notes]

1. $D_1(1) = \cdots = D_1(n) = \frac{1}{n}$
2. $F_0(x) \equiv 0$
3. for $t = 1, \ldots, T$
   (a) Choose $f_t \in G$ [base classifier choice, approximately minimizes ε_t below]
   (b) $\varepsilon_t := \sum_{i=1}^{n} D_t(i) 1(f_t(x_i) \neq y_i)$ [weighted error of f_t]
   (c) $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$ [update weight]
   (d) $F_t = F_{t-1} + \alpha_t f_t$ [classifier at tth round]
   (e) $Z_t = 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$ [chosen s.t. Z_t is such that $\sum_{i=1}^{n} D_{t+1}(i) = 1$]
   (f) $D_{t+1}(i) = \frac{1}{Z_t} D_t(i) \times \begin{cases} e^{\alpha_t} & \text{if } f_t(x_i) \neq y_i \\ e^{-\alpha_t} & \text{otherwise} \end{cases}$, where

Final output: $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(s)\right)$

2 AdaBoost.M1 [Equivalenet version given in Hastie book and lecture notes]

As given in HTF Algorithm 10.1, AdaBoost is:
Given training set $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.

1. Initialize observation weights $w_i = 1/n$, $i = 1, 2, \ldots, n$.

2. For $m = 1$ to M:
   (a) Fit classifier $G_m(x)$ to $\mathcal{D}$ using weights w_i.
   (b) Compute weighted 0-1 empirical risk:
   $$\mathrm{err}_m = \frac{1}{W}\sum_{i=1}^{n} w_i 1(y_i \neq G_m(x_i)) \quad \text{where } W = \sum_{i=1}^{n} w_i.$$
   (c) Compute $\alpha_m = \ln\left(\frac{1-\mathrm{err}_m}{\mathrm{err}_m}\right)$
   (d) Set $w_i \leftarrow w_i \cdot \exp\left[\alpha_m 1(y_i \neq G_m(x_i))\right], \quad i = 1, 2, \ldots, N$
3. Ouptut $G(x) = \operatorname{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$.

We now show this is equivalent to the traditional formulation we gave above:

1. $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$ [update weight]
2. $F_t = F_{t-1} + \alpha_t f_t$ [classifier at tth round]
3. $Z_t = 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$ [chosen s.t. Z_t is such that $\sum_{i=1}^{n} D_{t+1}(i) = 1$]
4. $D_{t+1}(i) = \frac{1}{Z_t} D_t(i) \times \begin{cases} e^{\alpha_t} & \text{if } f_t(x_i) \neq y_i \\ e^{-\alpha_t} & \text{otherwise} \end{cases}$, where

Final output: $H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(s)\right)$

So, α's are the same, up to a constant factor, which doesn't change the final prediction at all.

HTF updates are:

$$w_i \leftarrow w_i \exp\left[\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) 1(y_i \neq G_m(x_i))\right]$$

while traditional is (ignoring normalization)

$$\begin{aligned} D_{t+1}(i) &\leftarrow D_t(i) \times \begin{cases} e^{\alpha_t} & \text{if } f_t(x_i) \neq y_i \\ e^{-\alpha_t} & \text{otherwise} \end{cases} \\ &= D_i \exp\left[\alpha_t \left[2 \times 1(y_i \neq G_m(x_i)) - 1\right]\right] \\ &= D_i \exp\left[\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)\left[1(y_i \neq G_m(x_i)) - \frac{1}{2}\right]\right] \\ &= D_i \exp\left[\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) 1(y_i \neq G_m(x_i))\right] \exp\left[-\frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)\right] \end{aligned}$$

and notice that the last term is independent of i, and thus is lost in the normalization. So they're the same.

3 AdaBoost Minimizes Empirical Risk

[we should change notation to look more like lecture notes...]

[From Peter Bartlett's lecture – I think they're using the original form of AdaBoost here]

Theorem 1. *We have*

$$\hat{P}\left(YF_T(x) \leq 0\right) = \frac{1}{n}\left|\{i : y_i F_T(x_i) \leq 0\}\right| \tag{3.1}$$

$$\leq \prod_{t=1}^{T} 2\sqrt{\varepsilon_t(1-\varepsilon_t)} \tag{3.2}$$

Furthermore, if we know that ϵ_t is slightly less than $\frac{1}{2}$, say $\epsilon_t \leq \frac{1}{2} - \gamma\, \forall t$, the product above is no more than $(1-4\gamma^2)^{\frac{T}{2}}$.

Proof. Instead of the event $YF_T(X) \leq 0$, look at the equivalent event $\exp(-YF_T(X)) \geq 1$. Also, note that

$$1(\exp(-YF_T(X)) \geq 1) \leq \exp(-YF_T(X))$$

So

$$\hat{P}\left(YF_T(X) \leq 0\right) = \hat{\mathbb{E}}1(\exp(-YF_T(X)) \geq 1) \tag{3.3}$$

$$\leq \hat{\mathbb{E}}\left[\exp(-YF_T(X))\right] \tag{3.4}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \exp\left(-y_i \sum_{t=1}^{T} \alpha_t f_t(x_i)\right) \tag{3.5}$$

$$= \frac{1}{n}\sum_{i}\prod_{t} \exp\left(-y_i \alpha_t f_t(x_i)\right) \tag{3.6}$$

We also know that, since $y_i, f(x_i) \in \{\pm 1\}$, their product is also in $\{\pm 1\}$. Applying this to the expression for D_{t+1} in the algorithm, we have

$$D_{t+1}(i) = \frac{1}{Z_t} D_t(i) \times \begin{cases} e^{\alpha_t} & \text{if } f_t(x_i) \neq y_i \\ e^{-\alpha_t} & \text{otherwise} \end{cases}$$

$$= \frac{1}{Z_t} D_t(i) \exp\left(-y_i \alpha_t f_t(x_i)\right)$$

Plugging in, we get

$$\hat{\mathbb{E}}\left[\exp(-YF_T(X))\right] = \frac{1}{n}\sum_i \prod_t \left(\frac{D_{t+1}(i)}{D_t(i)} Z_t\right) \tag{3.7}$$

$$= \frac{1}{n}\sum_i \left(\prod_t Z_t\right) \frac{D_{T+1}(i)}{D_1(i)} \tag{3.8}$$

$$= \prod_t Z_t \tag{3.9}$$

where in the final equality we use the fact that D_{T+1} is a distribution and sums over i to one.

Meanwhile, recalling that $\varepsilon_t = \sum_{i=1}^n D_t(i) 1(f_t(x_i) \neq y_i)$, we have

$$\begin{aligned} Z_t =& \sum_{i=1}^n D_t(i) e^{\alpha_t} 1(f_t(x_i) \neq y_i) + \sum_{i=1}^n D_t(i) e^{-\alpha_t} 1(f_t(x_i) = y_i) \\ =& e^{\alpha_t}\varepsilon_t + e^{-\alpha_t}(1-\varepsilon_t) \end{aligned}$$

If we choose α_t to minimize Z_t (by differentiating w.r.t. α_t...) we get

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

Which gives

$$Z_t = (1-\epsilon_t)\sqrt{\frac{\epsilon_t}{1-\epsilon_t}} + \epsilon_t\sqrt{\frac{1-\epsilon_t}{\epsilon_t}} \tag{3.10}$$

$$= 2\sqrt{\epsilon_t(1-\epsilon_t)} \tag{3.11}$$

We can plug this into 3.9 to get the desired result. □

We can extend the above theorem to include a margin as well.

4 Exponential Loss with FSAM is AdaBoost

4.1 Exponential loss and adaboost (Section 10.4)

Turns out that the traditional AdaBoost.M1 algorithm (Alg. 10.1) is equivalent to forward stagewise additive modeling (Alg 10.2) using the loss function

$$L(y, f(x)) = \exp(-yf(x))$$

Then taking $G(x) = b(x;\gamma)$,

$$\begin{aligned}(\beta_m, G_m) &= \arg\min_{\beta,G} \sum_{i=1}^{N} \exp\left[-y_i \left(f_{m-1}(x_i) + \beta G(x_i)\right)\right] \\ &= \arg\min_{\beta,G} \sum_{i=1}^{N} w_i^{(m)} \exp\left[-\beta y_i G(x_i)\right]\end{aligned}$$

where $w_i^{(m)} := \exp(-y_i f_{m-1}(x_i))$. Note that $w_i^{(m)}$ only depends on the $m-1$'st classifier, thus we can consider $w^{(m)}$ as the weighting of the data for the m'th classifier.

Note that $y_i, G(x_i) \in \{-1, 1\}$. Thus for any fixed $\beta > 0$, we can break up the sum by possible values inside the exponential.

$$\begin{aligned}\sum_{i=1}^{N} w_i^{(m)} \exp\left[-\beta y_i G(x_i)\right] &= e^{-\beta} \sum_{i:y_i = G(x_i)} w_i^{(m)} + e^{\beta} \sum_{i:y_i \neq G(x_i)} w_i^{(m)} \\ &= e^{-\beta}\left[\sum_{i=1}^{N} w_i^{(m)} - \sum_{i=1}^{N} w_i^{(m)} 1(y_i \neq G(x_i))\right] \qquad (4.1) \\ &\quad + e^{\beta} \sum_{i=1}^{N} w_i^{(m)} 1(y_i \neq G(x_i)) \qquad (4.2) \\ &= (e^{\beta} - e^{-\beta}) \sum_{i=1}^{N} w_i^{(m)} 1(y_i \neq G(x_i)) + e^{-\beta} \sum_{i=1}^{N} w_i^{(m)} \qquad (4.3)\end{aligned}$$

Plugging this in, and keeping β fixed, we get

$$\begin{aligned}G_m &= \arg\min_{G} (e^{\beta} - e^{-\beta}) \sum_{i=1}^{N} w_i^{(m)} 1(y_i \neq G(x_i)) + e^{-\beta} \sum_{i=1}^{N} w_i^{(m)} \\ &= \arg\min_{G} \sum_{i=1}^{N} w_i^{(m)} 1(y_i \neq G(x_i))\end{aligned}$$

Notice that this last expression is independent of β.

Since the minimizing G_m is independent of β, we can plug it into 4.3 and minimize with respect to β. First, let's introduce

$$\begin{aligned}
\mathrm{err}_m &:= \frac{\sum_{i=1}^N w_i^{(m)} 1(y_i \neq G_m(x_i))}{W^{(m)}} \\
W^{(m)} &:= \sum_{i=1}^N w_i^{(m)}
\end{aligned}$$

the weighted error rate for G_m on the training data. Then minimizing 4.3 with respect to β is equivalent to minimizing

$$(e^{\beta} - e^{-\beta})\mathrm{err}_m + e^{-\beta}.$$

Differentiating w.r.t. β and equating to zero, we get

$$\begin{aligned}
\partial_\beta \left[(e^{\beta} - e^{-\beta})\mathrm{err}_m + e^{-\beta}\right] &= 0 \\
(e^{\beta} + e^{-\beta})\mathrm{err}_m - e^{-\beta} &= 0 \\
e^{\beta}\mathrm{err}_m + e^{-\beta}\mathrm{err}_m - e^{-\beta} &= 0 \\
e^{2\beta}\mathrm{err}_m + \mathrm{err}_m - 1 &= 0 \\
e^{2\beta} &= \frac{1 - \mathrm{err}_m}{\mathrm{err}_m} \\
\implies \beta_m &= \frac{1}{2}\ln\left(\frac{1 - \mathrm{err}_m}{\mathrm{err}_m}\right)
\end{aligned}$$

(How do we know this is a local min and not a local max? What if we allow negative weight values?)

Now we've found the next basis function $G_m(x)$ and the next coefficient β_m. Interesting that β_m is the best coefficient, no matter what the value of $G_m(x)$? Anyway, the next approximation in our forward stagewise additive model is

$$f_m(x) = f_{m-1}(x) + \beta_m G_m(x).$$

[NoteRecalling that $w_i^{(m)} := \exp(-y_i f_{m-1}(x_i))$, we find that the weights at the next iteration are

$$\begin{aligned}
w_i^{(m+1)} &= \exp(-y_i f_m(x_i)) \\
&= \exp(-y_i(f_{m-1}(x_i) + \beta_m G_m(x_i))) \\
&= w_i^{(m)} \exp(-y_i \beta_m G_m(x_i))
\end{aligned}$$

Noting that $y_i G_m(x_i) = 2 \cdot 1(y_i \neq G_m(x_i)) - 1$, we find

$$\begin{aligned} w_i^{(m+1)} &= w_i^{(m)} \exp\left[-\beta_m (2 \cdot 1(y_i \neq G_m(x_i)) - 1)\right] \\ &= w_i^{(m)} e^{-2\beta_m 1(y_i \neq G_m(x_i))} e^{-\beta_m} \end{aligned}$$

Taking $\alpha_m := 2\beta_m = \log \frac{1-\mathrm{err}(m)}{\mathrm{err}(m)}$, and noting that $e^{-\beta_m}$ multiplies all the weights by the same amount, we can equivalently take the weight updates to be

$$w_i^{(m+1)} = w_i^{(m)} e^{-\alpha_m 1(y_i \neq G_m(x_i))}$$

We see that this algorithm is the same as the traditional AdaBoost.M1 on p. 301. The ony difference is that, while in AdaBoost, we were loose about the requirements for "fitting the weighted training data", in the forwards stagewise approach, we are explicity looking for

$$\arg\min_G \sum_{i=1}^N w_i^{(m)} 1((y_i \neq G(x_i))).$$

5 Population Minimizer of Exponential Loss

In the previous section we showed that AdaBoost.M1 is equivalent to forward stagewise additive modeling with an exponential loss. The exponential loss gave of certain computational benefits, but what are its statistical properties?

Consider

$$f^*(x) = \arg\min_{f(x)} \mathbb{E}_{Y|x} e^{-Yf(x)}$$

Note that we can solve this for each fixed x. Also note that $Y \in \{-1, 1\}$, so we can write

$$\mathbb{E}_{Y|x} e^{-Yf(x)} = e^{-f(x)} \mathbb{P}(Y = 1|x) + e^{f(x)} P(Y = -1|x)$$

Considering x to be fixed, and $f(x) \in \mathbf{R}$, we can find the population minimum by differentiating with respect to $f(x)$ (a scalar) and equating to zero. Doing this gives us

$$\begin{aligned} 0 = \partial_{f(x)} \mathbb{E}_{Y|x} e^{-Yf(x)} &= -f(x) e^{-f(x)} \mathbb{P}(Y = 1|x) + f(x) e^{f(x)} P(Y = -1|x) \\ \implies f(x) e^{f(x)} P(Y = -1|x) &= -f(x) e^{-f(x)} \mathbb{P}(Y = 1|x) \\ \implies e^{2f(x)} &= \frac{\mathbb{P}(Y = 1|x)}{\mathbb{P}(Y = -1|x)} \end{aligned}$$

so we get

$$f^*(x) = \frac{1}{2} \ln \frac{\mathbb{P}(Y = 1|x)}{\mathbb{P}(Y = -1|x)}.$$

Let $p = \mathbb{P}(Y = 1|x)$ and $f = f^*(x)$, and let's solve for p:

$$\begin{aligned} f &= \frac{1}{2} \ln \frac{p}{1-p} \\ e^{2f} &= \frac{p}{1-p} \\ p &= \frac{e^{2f}}{1+e^{2f}} \end{aligned}$$

Equivalently,

$$\mathbb{P}(Y = 1|x) = \frac{1}{1 + e^{-2f^*(x)}}$$

Thus the additive expansion produced by AdaBoost is estimating one-half the log-odds of $P(Y = 1|x)$! This justifies using its sign as the classification rule. (So we're estimating Bayes rule?)

Another loss criterion with the same populating minimizer is the binomial negative log-likelihood (or *deviance* or *cross-entropy*), where we interpret f as the logit transform.