COMS 4773: Boosting

Daniel Hsu

March 16, 2024

1 Weak learning versus strong learning

The problem of boosting originated in the context of PAC learning. A concept class $\mathcal{C} \subseteq \{-1,1\}^{\mathcal{X}}$ is *(strongly) PAC learnable* if there is a learning algorithm with the following property. For any $c \in \mathcal{C}$, any $\epsilon, \delta \in (0,1)$, and any data distribution P with labels generated by c, the learner returns an ϵ-approximation of $c \in \mathcal{C}$ with probability at least $1-\delta$. Such a learning algorithm is called a *strong PAC learner.* (Of course, the sample and time complexity should be polynomial in $1/\epsilon$, $1/\delta$, $\dim(\mathcal{X})$, $\operatorname{size}(c)$, etc.)

A *weak PAC learner* is much like a strong learner, except that it might only work for certain non-trivial values of ϵ and δ, say, $\epsilon \geq \epsilon_0$ and $\delta \geq \delta_0$. By non-trivial, we mean that ϵ_0 is non-trivially bounded away from $1/2$, and δ_0 is non-trivially bounded away from 1. If there is a weak learner for $\mathcal{C}$, then we say $\mathcal{C}$ is *weakly PAC learnable.*

Is weak PAC learnability equivalent to strong PAC learnability? This question was asked by Kearns and Valiant (1989), and answered affirmatively by Schapire (1990). Schapire proved the equivalence by designing a *boosting algorithm*: an algorithmic reduction from strong learning to weak learning. The boosting algorithm achieves the strong learning guarantee for a concept class $\mathcal{C}$ by using repeatedly invoking a weak learner for $\mathcal{C}$ as a subroutine. After Schapire's original proof was found, several other boosting algorithms have been developed, with the AdaBoost algorithm of Freund and Schapire (1997) probably being the most famous.

The key idea in boosting algorithms is to exploit the requirement that weak learners succeed under any marginal distribution over $\mathcal{X}$. All boosting algorithms work by invoking the weak learner under different distributions, and then use (or combine) the outputs of the weak learners (called "weak hypotheses") in some way.

2 In-sample weak learning

It is a little simpler to thinking about weak versus strong learning in the context of a fixed training dataset—i.e., "in-sample"—and then worry later about how to get "out-of-sample" results.

So let $(x_1,y_1),\dots,(x_n,y_n)$ be a training dataset from $\mathcal{X} \times \{-1,1\}$. A *weak learner* for this dataset is an algorithm that, given any *(re-)weighting* $p = (p_1,\dots,p_n) \in \Delta^{n-1}$ of the training examples, returns a hypothesis $h \colon \mathcal{X} \to \{-1,1\}$ with non-trivial *p-weighted (empirical) accuracy*:

$$\operatorname{acc}_n(h;p) := \sum_{i=1}^{n} p_i \cdot \mathbb{1}\{h(x_i) = y_i\} \geq \frac{1+\gamma}{2}$$

for some $\gamma > 0$. The number γ is called the *advantage* of h over random guessing (with respect to the re-weighting p). Here, $(1+\gamma)/2$ corresponds to what we previously called $1-\epsilon_0$. The assumption that there is a weak learner for the dataset is called the *weak learning assumption (WLA).*

There is another way to think about the WLA. Consider all possible hypotheses that the weak learner could output, and call this set $\mathcal{H}$. For simplicity, assume $\mathcal{H}$ is finite, say, $\mathcal{H} = \{h_1, \ldots, h_d\}$. Define matrix $A \in \mathbb{R}^{n \times d}$ by

$$A_{i,j} := \mathbb{1}\{h_j(x_i) = y_i\}.$$

The WLA says that for any $p \in \Delta^{n-1}$, there is some $j \in [d]$ such that

$$p^\mathsf{T} A e_j \geq \frac{1+\gamma}{2},$$

where e_j is the j-th elementary basis vector. In other words,

$$\min_{p \in \Delta^{n-1}} \max_{q \in \{e_1, \ldots, e_d\}} p^\mathsf{T} A q \geq \frac{1+\gamma}{2}. \tag{1}$$

Since linear functions over the probability simplex Δ^{d-1} are extremized at the vertices of the simplex, (1) is equivalent to

$$\min_{p \in \Delta^{n-1}} \max_{q \in \Delta^{d-1}} p^\mathsf{T} A q \geq \frac{1+\gamma}{2}. \tag{2}$$

By the Von Neumann Min-Max Theorem,

$$\min_{p \in \Delta^{n-1}} \min_{q \in \Delta^{d-1}} p^\mathsf{T} A q = \max_{q \in \Delta^{d-1}} \min_{p \in \Delta^{n-1}} p^\mathsf{T} A q, \tag{3}$$

so (1) (or (2)) is equivalent to

$$\max_{q \in \Delta^{d-1}} \min_{p \in \Delta^{n-1}} p^\mathsf{T} A q \geq \frac{1+\gamma}{2}. \tag{4}$$

Again, since linear functions over Δ^{n-1} are extremized at the vertices, (4) is equivalent to

$$\max_{q \in \Delta^{d-1}} \min_{i \in [n]} e_i^\mathsf{T} A q \geq \frac{1+\gamma}{2}. \tag{5}$$

This means that there exists some $q = (q_1, \ldots, q_d) \in \Delta^{d-1}$ such that, for all $i \in [n]$,

$$\begin{aligned}
\sum_{j=1}^{d} q_j \, \mathbb{1}\{h_j(x_i) = y_i\} &= \sum_{j=1}^{d} q_j \frac{1 + y_i h_j(x_i)}{2} \\
&= \frac{1 + y_i \sum_{j=1}^{d} q_j h_j(x_i)}{2} \\
&\geq \frac{1+\gamma}{2},
\end{aligned}$$

which implies that

$$\operatorname{sign}\left(\sum_{j=1}^{d} q_j h_j(x_i)\right) = y_i \quad \text{for all } i \in [n].$$

So, in the "feature space" where the feature vector for $x \in \mathcal{X}$ is $(h_1(x), \ldots, h_d(x))$, the WLA is equivalent to *linearly separability* of the dataset.

3 In-sample boosting

Since WLA is equivalent to linear separability, a strong (in-sample) learner can be implemented by finding a linear separator for a given dataset. Finding linear separators is equivalent to solving linear programs, which can be done in polynomial time. The final hypothesis might not be just one of the hypotheses in $\mathcal{H}$; rather, it would generally be a linear classifier built on top of hypotheses from $\mathcal{H}$.

There are two main problems with this approach: one computational, the other statistical.

- It is unclear how to enumerate all of the hypotheses that a weak learner might output; this makes it difficult to use standard linear programming algorithms to find a linear separator.
- The class of linear classifiers in a d-dimensional feature space has VC dimension d. So the sample complexity might be linear in d, which is exponentially worse than the sample complexity needed to learn $\mathcal{H}$, which only grows logarithmically in d.

Both of these problems are even worse if $\mathcal{H}$ is infinite. So this equivalence to linear separability might not seem so useful after all, except perhaps as a conceptual device.

Fortunately, the Von Neumann Min-Max Theorem (3) has an algorithmic proof due to Freund and Schapire that is based on executing the Hedge algorithm in a particular instance of the Online Allocation problem. From the execution of Hedge in this context, there is a way to obtain (in the terminology of (3)) $p \in \Delta^{n-1}$ and $q \in \Delta^{d-1}$ that approximately achieve the optimal value in (3).

The following boosting algorithm is derived from this line of reasoning.

- Input: training examples $(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \{-1, 1\}$; hyperparameters $\eta > 0$, $T > 0$.
- Initially, set $p_1 := (1/n, \dots, 1/n) \in \Delta^{n-1}$.
- For $t = 1, 2, \dots, T$:
  - Invoke the weak learner with weighting $p_t := (p_{t,1}, \dots, p_{t,n}) \in \Delta^{n-1}$ on dataset $((x_i, y_i))_{i=1}^n$ to obtain hypothesis $h_t \colon \mathcal{X} \to \{-1, 1\}$.
  - Define loss vector $\ell_t = (\ell_{t,1}, \dots, \ell_{t,n}) \in \{0, 1\}^n$ by
  $$\ell_{t,i} := \mathbb{1}\{h_t(x_i) = y_i\}.$$
  - Compute new weighting $p_{t+1} := (p_{t+1,1}, \dots, p_{t+1,n}) \in \Delta^{n-1}$:
  $$p_{t+1,i} := \frac{p_{t,i} \exp(-\eta \ell_{t,i})}{Z_{t+1}} \quad \text{for all } i \in [n],$$
  where
  $$Z_{t+1} := \sum_{i=1}^n p_{t,i} \exp(-\eta \ell_{t,i}).$$
- Return: final hypothesis $\hat{h}(x) = \operatorname{sign}(g(x))$ where $g(x) = \frac{1}{T} \sum_{t=1}^T h_t(x)$.

The weightings p_t are computed as in Hedge, with loss vectors ℓ_t determined by the hypotheses h_t returned by the weak learner. The main theorem for Hedge implies the following: for any $u \in \Delta^{n-1}$,

$$\frac{1}{T}\sum_{t=1}^{T}\langle p_t, \ell_t\rangle \leq \frac{1}{T}\sum_{t=1}^{T}\langle u, \ell_t\rangle + \frac{\ln n}{\eta T} + \frac{\eta}{2}. \tag{6}$$

Let us interpret the inequality in (6). First, the left-hand side satisfies

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\langle p_t, \ell_t\rangle &= \frac{1}{T}\sum_{t=1}^{T}\left(\sum_{i=1}^{n} p_{t,i}\,\mathbb{1}\{h_t(x_i) = y_i\}\right) \\ &= \frac{1}{T}\sum_{t=1}^{T}\operatorname{acc}_n(h_t; p_t) \\ &\geq \frac{1+\gamma}{2}\end{aligned}$$

where the inequality follows from the WLA. Now suppose $u = e_i$ for some $i \in [n]$. Then

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\langle e_i, \ell_t\rangle &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{1}\{h_t(x_i) = y_i\} \\ &= \frac{1}{T}\sum_{t=1}^{T}\frac{1 + y_i h_t(x_i)}{2} \\ &= \frac{1 + y_i g(x_i)}{2}.\end{aligned}$$

Therefore, via (6), we have

$$\frac{1 + y_i g(x_i)}{2} \geq \frac{1+\gamma}{2} - \frac{\ln n}{\eta T} - \frac{\eta}{2} \quad \text{for all } i \in [n].$$

This is equivalent to

$$y_i g(x_i) \geq \gamma - \frac{2\ln n}{\eta T} - \eta \quad \text{for all } i \in [n].$$

If we choose $\eta = \gamma/2$ and $T > (8\ln n)/\gamma^2$ (say), then we have $y_i g(x_i) > 0$ for all $i \in [n]$. In particular, $x \mapsto \operatorname{sign}(g(x))$ classifies all training examples correctly.

Alternative analysis. Recall that the main guarantee for Hedge can also be written in terms of the relative entropy of an arbitrary comparator $u \in \Delta^{n-1}$ from p_1:

$$\frac{1}{T}\sum_{t=1}^{T}\langle p_t, \ell_t\rangle \leq \frac{1}{T}\sum_{t=1}^{T}\langle u, \ell_t\rangle + \frac{\operatorname{RE}(u, p_1)}{\eta T} + \frac{\eta}{2}. \tag{7}$$

Define $S := \{i \in [n] : y_i g(x_i) \leq 0\}$; let $M := |S|$, and let u be the uniform distribution over S. Then

$$\operatorname{RE}(u, p_1) = \sum_{i=1}^{n} u_i \ln\frac{u_i}{1/n} = -\ln M + \ln n.$$

So, if $\eta = \gamma/2$, (7) implies

$$M \leq n\exp\left(-\frac{\gamma^2 T}{8}\right). \tag{8}$$

4 Out-of-sample performance

The boosting algorithm from Section 3 addresses the computational problem that was raised. What about the statistical problem? That is addressed next.

The type of hypothesis returned by the boosting algorithm is a (thresholded) *linear combination* of hypotheses from $\mathcal{H}$. Even if $\mathcal{H}$ has large cardinality or is infinite, a saving grace is that T may be relatively small, making the linear combination *sparse*.

Proposition 1. *Let $\mathcal{H} \subset \{-1,1\}^{\mathcal{X}}$ be a hypothesis class, and let $\mathcal{H}_T$ denote the set of hypotheses of the form*

$$x \mapsto \operatorname{sign}\left(\frac{1}{T}\sum_{t=1}^{T} h_t(x)\right)$$

where $h_1, \dots, h_T \in \mathcal{H}$. Then for any $x_1, \dots, x_n \in \mathcal{X}$,

$$|\mathcal{H}_T(x_{1:n})| \leq |\mathcal{H}(x_{1:n})|^T.$$

We now apply the uniform convergence theorem to get a bound on the error rate of the final hypothesis $\hat{h}$ returned by the boosting algorithm. Let $(X_1, Y_1), \dots, (X_n, Y_n)$ be the iid sample from the data distribution P over $\mathcal{X} \times \{-1,1\}$, and let $\hat{h}$ be the output of the boosting algorithm when provided these examples as training data. With probability at least $1 - \delta$,

$$\operatorname{err}(\hat{h}) \leq \operatorname{err}_n(\hat{h}) + C\left(\sqrt{\frac{Td_0 \log(n) + \log(1/\delta)}{n}}\right), \tag{9}$$

where d_0 is the VC dimension of $\mathcal{H}$. Under the WLA (with advantage $\gamma > 0$), using the boosting algorithm with $\eta = \gamma/2$ ensures

$$\operatorname{err}_n(\hat{h}) \leq \exp\left(-\frac{\gamma^2 T}{8}\right),$$

so $\operatorname{err}_n(\hat{h}) = 0$ if

$$T = \frac{8 \log n}{\gamma^2} + 1.$$

With this value of T, we find that $\operatorname{err}(\hat{h}) \leq \epsilon$ with probability at least $1 - \delta$, provided that

$$n \geq C'\left(\frac{d_0 \log^2(1/\epsilon)}{\gamma^2 \epsilon^2} + \frac{\log(1/\delta)}{\epsilon^2}\right).$$

(Using a slightly different uniform convergence theorem replaces the $1/\epsilon^2$ factors with $1/\epsilon$.)

5 Margins

The analysis in Section 4 shows that, with probability at least $1 - \delta$,

$$\operatorname{err}(\hat{h}) \leq \exp\left(-\frac{\gamma^2 T}{8}\right) + C\left(\sqrt{\frac{Td_0 \log(n) + \log(1/\delta)}{n}}\right).$$

This bound has a term that decreases with T and another term that increases with T. This suggests a trade-off that should be optimized by carefully choosing T (as done in the previous section).

However, in practice, it is occasionally observed with boosting that the error rate only decreases with T, even beyond the point at which $\operatorname{err}_n(\hat{h}) = 0$. To provide a theoretical explanation of this phenomenon, Schapire et al. (1998) gave a different analysis of boosting based on the notion of *margins*.

Recall that $\hat{h}(x) = \operatorname{sign}(g(x))$, where $g(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$. Since $h_t(x) \in \{-1,1\}$, it follows that $g(x) \in [-1,1]$. Although only the sign of $g(x)$ is used for prediction, the magnitude of $g(x)$ can be regarded as a measure of "confidence" in the prediction. Let us call $yg(x)$ the *margin* of g on $(x,y) \in \mathcal{X} \times \{-1,1\}$. The key high-level idea is that larger margins imply better generalization.

One way to take advantage of margins the theoretical analysis is to consider a loss function that is sensitive to margins. For each $\gamma \in [0,1]$, define the *γ-ramp loss* $\ell_\gamma \colon \mathbb{R} \to \mathbb{R}$ by

$$\ell_\gamma(z) := \begin{cases} 1 & \text{if } z \leq 0, \\ 1 - z/\gamma & \text{if } 0 < z \leq \gamma, \\ 0 & \text{if } z > \gamma. \end{cases}$$

For any distribution P over $\mathcal{X} \times \{-1,1\}$, any $g \colon \mathcal{X} \to \mathbb{R}$, and any $\gamma \in [0,1]$,

$$\operatorname{err}(\operatorname{sign} \circ g) = \mathbb{E}_{(X,Y)\sim P}[\ell_0(Yg(X))] \leq \mathbb{E}_{(X,Y)\sim P}[\ell_\gamma(Yg(X))].$$

We'll show that, with probability at least $1-\delta$,

$$\sup_{g \in \operatorname{conv}(\mathcal{H})} \mathbb{E}[\ell_\gamma(Yg(X))] - \frac{1}{n}\sum_{i=1}^{n} \ell_\gamma(Y_i g(X_i)) \leq C\sqrt{\frac{d_0 \log n}{\gamma^2 n} + \frac{\log(1/\delta)}{n}}. \tag{10}$$

Again, here d_0 is the VC dimension of $\mathcal{H}$. There is no dependence on the number of hypotheses from $\mathcal{H}$ in the convex combination g.

Before proving this bound, let us see how to use it. A slight modification of the argument at the end of Section 3 shows that, under the WLA (with advantage $\gamma > 0$), if $\eta = \gamma/4$, then

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{Y_i g(X_i) \leq \gamma/2\} \leq \exp\left(-\frac{\gamma^2 T}{32}\right).$$

Therefore, in this case, with probability at least $1-\delta$,

$$\begin{aligned} \operatorname{err}(\operatorname{sign} \circ g) &\leq \frac{1}{n}\sum_{i=1}^{n} \ell_{\gamma/2}(Y_i g(X_i)) + C\sqrt{\frac{d_0 \log n}{\gamma^2 n} + \frac{\log(1/\delta)}{n}} \\ &\leq \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{Y_i g(X_i) \leq \gamma/2\} + C\sqrt{\frac{d_0 \log n}{\gamma^2 n} + \frac{\log(1/\delta)}{n}} \\ &\leq \exp\left(-\frac{\gamma^2 T}{32}\right) + C\sqrt{\frac{d_0 \log n}{\gamma^2 n} + \frac{\log(1/\delta)}{n}}. \end{aligned}$$

The bound decreases with both T and γ.

The proof of (10) is based on Rademacher complexity. Consider the function class $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X} \times \{-1,1\}}$ consisting of functions of the form $f^g \colon \mathcal{X} \times \{-1,1\} \to \mathbb{R}$ for each $g \in \operatorname{conv}(\mathcal{H})$, where

$$f^g(x,y) := \ell_\gamma(yg(x)).$$

With probability at least $1-\delta$, we have

$$\sup_{g \in \operatorname{conv}(\mathcal{H})} \mathbb{E}[\ell_\gamma(Yg(X))] - \frac{1}{n}\sum_{i=1}^n \ell_\gamma(Y_i g(X_i)) \leq 2\,\mathbb{E}\operatorname{Rad}_n(\mathcal{F}) + \sqrt{\frac{2\log(1/\delta)}{n}}.$$

So it remains to bound the Rademacher complexity $\mathbb{E}\operatorname{Rad}_n(\mathcal{F})$. We can write $\operatorname{Rad}_n(\mathcal{F})$ as

$$\mathbb{E}_\sigma \sup_{g \in \operatorname{conv}(\mathcal{H})} \frac{1}{n}\sum_{i=1}^n \sigma_i \ell_\gamma(Y_i g(X_i)) = \mathbb{E}_\sigma \sup_{g \in \operatorname{conv}(\mathcal{H})} \frac{1}{n}\sum_{i=1}^n \sigma_i \phi_i(g(X_i)),$$

where we define $\phi_i(t) = \ell_\gamma(Y_i t)$ for each $i \in [n]$. It can be checked that each ϕ_i is L-Lipschitz for $L = 1/\gamma$. Therefore, by the Lipschitz contraction property of Rademacher averages, we have

$$\mathbb{E}_\sigma \sup_{g \in \operatorname{conv}(\mathcal{H})} \frac{1}{n}\sum_{i=1}^n \sigma_i \phi_i(g(X_i)) \leq \frac{1}{\gamma}\,\mathbb{E}_\sigma \sup_{g \in \operatorname{conv}(\mathcal{H})} \frac{1}{n}\sum_{i=1}^n \sigma_i g(X_i).$$

Next, observe that $\{(g(X_1),\dotsc,g(X_n)) : g \in \operatorname{conv}(\mathcal{H})\} = \operatorname{conv}(\{(h(X_1),\dotsc,h(X_n)) : h \in \mathcal{H}\})$. Therefore

$$\mathbb{E}_\sigma \sup_{g \in \operatorname{conv}(\mathcal{H})} \frac{1}{n}\sum_{i=1}^n \sigma_i g(X_i) = \mathbb{E}_\sigma \sup_{h \in \mathcal{H}} \frac{1}{n}\sum_{i=1}^n \sigma_i h(X_i).$$

Finally, since $\mathcal{H}$ has VC dimension d_0, Massart's lemma and Sauer's lemma implies

$$\mathbb{E}_\sigma \sup_{h \in \mathcal{H}} \frac{1}{n}\sum_{i=1}^n \sigma_i h(X_i) \leq \sqrt{\frac{2\ln\binom{n}{\leq d_0}}{n}}.$$

Putting everything together gives (10).

6 Exponential loss

Consider a finite hypothesis class $\mathcal{H} = \{h_1,\dotsc,h_d\} \subseteq \{-1,1\}^{\mathcal{X}}$, and define the feature map $\phi \colon \mathcal{X} \to \mathbb{R}$ by

$$\phi(x) = (h_1(x),\dotsc,h_d(x)).$$

A linear classifier in this feature space is defined by a weight vector $\lambda = (\lambda_1,\dotsc,\lambda_d) \in \mathbb{R}^d$:

$$x \mapsto \operatorname{sign}(\langle \phi(x), \lambda \rangle).$$

We'll see that the boosting algorithm can be viewed as a coordinate-descent algorithm for minimizing an average "exponential loss" objective

$$L(\lambda) := \frac{1}{n}\sum_{i=1}^n \exp(-y_i \langle \phi(X_i), \lambda \rangle).$$

Note that $L(\lambda)$ is an upper-bound on the training error rate of the linear classifier.

The partial derivative of L with respect to the j-th variable is

$$\partial_j L(\lambda) = -\sum_{i=1}^{n} \exp(-y_i \langle \phi(x_i), \lambda \rangle) y_i h_j(x_i).$$

In coordinate descent, we repeatedly choose some $j \in [d]$ and adjust the j-th variable so as to improve the objective value.

Suppose in round t of coordinate descent, we have weight vector is $\lambda_{t-1} = (\lambda_{t-1,1}, \dots, \lambda_{t-1,d}) \in \mathbb{R}^d$. (For round $t = 1$, we start with $\lambda_0 = (0, \dots, 0) \in \mathbb{R}^d$.) Define probability vector $p_t = (p_{t,1}, \dots, p_{t,n}) \in \Delta^{n-1}$ by

$$p_{t,i} = \frac{\exp(-y_i \langle \phi(x_i), \lambda_{t-1} \rangle)}{Z_t} \quad \text{where} \quad Z_t = \sum_{i=1}^{n} \exp(-y_i \langle \phi(x_i), \lambda_{t-1} \rangle).$$

Then, the j-th partial derivative of L at λ_{t-1} is

$$\begin{aligned} \partial_j L(\lambda_{t-1}) &= -\sum_{i=1}^{n} \exp(-y_i \langle \phi(x_i), \lambda_{t-1} \rangle) y_i h_j(x_i) \\ &= -Z_t \sum_{i=1}^{n} p_{t,i} y_i h_j(x_i). \end{aligned}$$

Under the weak learning assumption, there exists $j \in [d]$ such that $-\partial_j L(\lambda_{t-1}) \geq Z_t \gamma > 0$. So assume we choose such a $j_t \in [d]$ with $-\partial_{j_t} L(\lambda_{t-1}) > 0$. Then we can decrease the objective by increasing λ_{t-1,j_t} a little bit—say, by adding $\eta/2 > 0$—and leaving all other components of λ_{t-1} unchanged:

$$\lambda_{t,j} := \begin{cases} \lambda_{t-1,j} + \eta/2 & \text{if } j = j_t, \\ \lambda_{t-1,j} & \text{if } j \neq j_t. \end{cases}$$

With this updated weight vector $\lambda_t = (\lambda_{t,1}, \dots, \lambda_{t,d}) \in \mathbb{R}^d$, we have a corresponding updated probability vector $p_{t+1} = (p_{t+1,1}, \dots, p_{t+1,n}) \in \Delta^{n-1}$ defined by

$$\begin{aligned} p_{t+1,i} &\propto p_{t,i} \exp\Big(-\frac{\eta}{2} y_i h_{j_t}(x_i)\Big) \\ &\propto p_{t,i} \exp(-\eta \ell_{t,i}), \end{aligned}$$

where $\ell_t = (\ell_{t,1}, \dots, \ell_{t,n}) \in \{0,1\}^n$ is the loss vector with

$$\ell_{t,i} = \mathbb{1}\{h_{j_t}(x_i) = y_i\}.$$

The choice of h_{j_t} and the way that p_t is updated to p_{t+1} is the same as in the boosting algorithm.

References

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, pages 433–444, 1989.

Robert E Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

Robert E Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.