Boosting

Karl Stratos

1 Emsemble Learning

Suppose we have m labeled examples $S = \left\{x^{(i)}\right\}_{i=1}^{m}$ where $x^{(i)} \in \mathcal{X}$ is labeled as $y^{(i)} \in \{\pm 1\}$. Given any distribution D over S, we have an algorithm $\mathcal{A}$ to produce a classifier $h \leftarrow \mathcal{A}(S, D)$ which is better than random guessing on S according to the distribution:

$$P_{x\sim D}\left(h\left(x\right) \neq y\right) = \sum_{i=1}^{m} D(i) \left[\left[h\left(x^{(i)}\right) \neq y^{(i)}\right]\right] < \frac{1}{2}$$

The celebrated machine learning technique, **boosting**, iteratively trains T such "base" classifiers $h_1 \dots h_T$ in a way that the empirical error of their *ensemble* g_T decreases exponentially fast in T:

$$\widehat{R}(g_T) := \frac{1}{m} \sum_{i=1}^{m} \left[\left[g_T\left(x^{(i)}\right) \neq y^{(i)}\right]\right] \leq \exp\left(-2\gamma^2 T\right)$$

where $\gamma \in (0, 1/2]$ is a constant.

2 Derivation

We define the ensemble as a linear combination of T base classifiers $h_1 \dots h_T$:

$$g_T(x) := \sum_{t=1}^{T} \alpha_t h_t(x)$$

where the label of x is then determined by $\text{sign}(g_T(x))$. This leads to two subproblems.

1. How can we use a single algorithm $\mathcal{A}$ to obtain T distinct classifiers $h_1 \dots h_T$ that complement each other?
2. How should we determine the weights α_t of the classifiers?

A sensible approach to the first problem is the following. At the t-th round, we have a partial ensemble $g_t(x) := \sum_{s=1}^{t} \alpha_s h_s(x)$. We will train the next base classifier $h_{t+1} \leftarrow \mathcal{A}(S, D_{t+1})$ from a distribution D_{t+1} that puts larger weights on points in S that g_t found difficult. That is, we want the probability mass of point $x^{(i)} \in S$ under D_{t+1} to be

$$D_{t+1}(i) \propto \exp\left(-y^{(i)} g_t\left(x^{(i)}\right)\right) \tag{1}$$

This is natural because $y^{(i)} g_t\left(x^{(i)}\right)$ is the *margin* of g_t at $x^{(i)}$: its sign tells us the correctness of g_t and its magnitude tells us the confidence of g_t.

$$y^{(i)} g_t\left(x^{(i)}\right) = \begin{cases} > 0 & \text{if } x^{(i)} \text{ is correctly classified by } g_t \\ \leq 0 & \text{if } x^{(i)} \text{ is incorrectly classified by } g_t \end{cases}$$

This way, if $x^{(i)}$ is correctly classified by g_t with high confidence, $D_{t+1}(i)$ is small. If $x^{(i)}$ is incorrectly classified by g_t with high confidence, $D_{t+1}(i)$ is large.

With this in mind, we iteratively define for $t = 1 \dots T-1$

$$D_{t+1}(i) := \frac{D_t(i) \exp\left(-\alpha_t y^{(i)} h_t\left(x^{(i)}\right)\right)}{Z_t} \tag{2}$$

where $Z_t := \sum_{i=1}^{m} D_t(i) \exp\left(-\alpha_t y^{(i)} h_t\left(x^{(i)}\right)\right)$ is a normalizer and $D_1(i) := 1/m$. This is an excellent choice for several reasons:

Reason 1. It does what we want in (1). Simply expand $D_t \dots D_1$ to verify:

$$D_{t+1}(i) = \frac{\exp\left(-y^{(i)} g_t\left(x^{(i)}\right)\right)}{m \prod_{s=1}^{t} Z_s} \tag{3}$$

Reason 2. Because the numerator in (3) is an upper bound on the zero-one loss at $x^{(i)}$, we can use it to bound the empirical loss of g_t:

$$\begin{aligned} \widehat{R}(g_t) = \frac{1}{m} \sum_{i=1}^{m} \left[\left[y^{(i)} g_t\left(x^{(i)}\right) \leq 0 \right]\right] &\leq \frac{1}{m} \sum_{i=1}^{m} \exp\left(-y_i g_t\left(x^{(i)}\right)\right) \\ &\leq \frac{1}{m} \sum_{i=1}^{m} \left(m \prod_{s=1}^{t} Z_s \right) D_{t+1}(i) \\ &= \prod_{s=1}^{t} Z_s \end{aligned} \tag{4}$$

Reason 3. (4) implies that we want to minimize each of the normalizers $Z_1 \dots Z_T$ to make $\widehat{R}(g_T)$ small. Since Z_t is a function of α_t, this answers the second subproblem of choosing the weights:

$$\begin{aligned} Z_t &:= \sum_{i=1}^{m} D_t(i) \exp\left(-\alpha_t y^{(i)} h_t\left(x^{(i)}\right)\right) \\ &= \sum_{i=1:\ y^{(i)} h_t\left(x^{(i)}\right) = -1}^{m} D_t(i) \exp\left(\alpha_t\right) + \sum_{i=1:\ y^{(i)} h_t\left(x^{(i)}\right) = 1}^{m} D_t(i) \exp\left(-\alpha_t\right) \\ &= \epsilon_t \exp\left(\alpha_t\right) + \left(1 - \epsilon_t\right) \exp\left(-\alpha_t\right) \end{aligned}$$

where $\epsilon_t := P_{x \sim D_t}\left(h_t\left(x\right) \neq y\right) = \sum_{i=1:\ y^{(i)} h_t\left(x^{(i)}\right) = -1}^{m} D_t(i)$ is the error of h_t. Z_t is convex in α_t (a sum of convex functions) and minimized at $\alpha_t = \log \sqrt{(1-\epsilon_t)/\epsilon_t}$ as $Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$.

Boost
Input: labeled examples $S = \left\{x^{(i)}\right\}_{i=1}^{m}$ where $x^{(i)} \in \mathcal{X}$ is labeled as $y^{(i)} \in \{\pm 1\}$, algorithm $\mathcal{A}$ which produces a base learner $h \leftarrow \mathcal{A}(S, D)$ such that $\sum_{i=1}^{m} D(i)\left[\left[h\left(x^{(i)}\right) \neq y^{(i)}\right]\right] < \frac{1}{2}$ for for any distribution D over S, number of boosting rounds T

1. Set the initial distribution $D_1(i) \leftarrow 1/m$ for $i = 1 \dots m$.
2. For $t = 1 \dots T$,
   (a) Obtain $h_t \leftarrow \mathcal{A}(S, D_t)$.
   (b) Compute its error $\epsilon_t \leftarrow \sum_{i=1}^{m} D_t(i)\left[\left[h_t\left(x^{(i)}\right) \neq y^{(i)}\right]\right]$.
   (c) Set weight $\alpha_t \leftarrow \log\sqrt{(1-\epsilon_t)/\epsilon_t}$.
   (d) Set normalizer $Z_t \leftarrow 2\sqrt{\epsilon_t(1-\epsilon_t)}$.
   (e) Set the next distribution: for $i = 1 \dots m$,

$$D_{t+1}(i) \leftarrow \frac{D_t(i)\exp\left(-\alpha_t y^{(i)} h_t\left(x^{(i)}\right)\right)}{Z_t}$$

Output: $g_T \leftarrow \sum_{t=1}^{T} \alpha_t h_t$

Now that we know the exact value of α_t and how to define D_t, we can write down the algorithm **Boost**. It follows from (4) and the choice of α_t that $g_T \leftarrow \textbf{Boost}(S, \mathcal{A}, T)$ satisfies

$$\begin{aligned}
\widehat{R}(g_T) \leq \prod_{t=1}^{T} Z_t = \prod_{t=1}^{T} \sqrt{4\epsilon_t - 4\epsilon_t^2} &= \prod_{t=1}^{T} \sqrt{1-(1-2\epsilon_t)^2} \\
&\leq \prod_{t=1}^{T} \exp\left(-\frac{1}{2}(1-2\epsilon_t)^2\right) \\
&= \exp\left(-2\sum_{t=1}^{T}\left(\frac{1}{2}-\epsilon_t\right)^2\right) \\
&\leq \exp\left(-2\gamma^2 T\right)
\end{aligned}$$

where $\gamma := \min_{t=1}^{T} \frac{1}{2} - \epsilon_t$ lies in $(0, 1/2]$ by assumption and is dependent on the base classifier with the largest error.

2.1 Coordinate Descent Interpretation

The algorithm coincides with doing coordinate descent over $\alpha = (\alpha_1 \dots \alpha_T)$ on

$$F(\alpha) := \sum_{i=1}^{m} \exp\left(y^{(i)} \sum_{t=1}^{T} \alpha_t h_t\left(x^{(i)}\right)\right)$$

which is a convex upper bound on the zero-one loss $\sum_{i=1}^{m}\left[\left[y^{(i)}\sum_{t=1}^{T}\alpha_t h_t\left(x^{(i)}\right) \leq 0\right]\right]$. In this interpretation, we initialize the parameter at the origin $\alpha^{(0)} \in \mathbb{R}^T$ and select one coordinate e_t (i.e., the t-th standard basis vector in $\mathbb{R}^T$) at a time to move along by distance η to greedily minimize $F(\alpha^{(t-1)} + \eta e_t)$. Using the identity of D_t in (3)

and $\epsilon_t := \sum_{i=1:\, y^{(i)} h_t\left(x^{(i)}\right)=-1}^{m} D_t(i)$, we can verify that

$$\frac{\partial F(\alpha^{(t-1)} + \eta e_t)}{\partial \eta} = \epsilon_t \left(m \prod_{s=1}^{t-1} Z_s \right) \exp(\eta) - (1 - \epsilon_t) \left(m \prod_{s=1}^{t-1} Z_s \right) \exp(-\eta)$$

Then the rate of decrease of $F(\alpha^{(t-1)} + \eta e_t)$ at $\eta = 0$ is given by

$$-\frac{\partial F(\alpha^{(t-1)} + \eta e_t)}{\partial \eta}\bigg|_{\eta=0} = (1 - 2\epsilon_t) \left(m \prod_{s=1}^{t-1} Z_s \right)$$

so we decrease the objective fastest by choosing h_t with the smallest error ϵ_t. Setting the derivative to zero also shows that $\eta^* := \arg\min_{\eta \in \mathbb{R}} F(\alpha^{(t-1)} + \eta e_t)$ is given by $\eta^* = \log \sqrt{(1 - \epsilon_t)/\epsilon_t}$.

One nice aspect of this interpretation is that it can be used to derive variants of the boosting algorithm by simply changing the choice of $F(\alpha)$. Above, we used the exponential loss $\exp(-x)$, but we could use the logistic loss $\log(1 + \exp(-x))$, the hinge loss $\max(1 - x, 0)$, and so forth. As long as they are convex upper bounds on the zero-one loss $[[x \leq 0]]$, we can use this framework to easily come up with different step sizes (i.e., weights of the base classifiers).