Boosting as Coordinate Descent

Karl Stratos

1 Steepest Descent

Let $f : \mathbb{R}^d \to \mathbb{R}$ be a differentiable function to minimize. Let $x \in \mathbb{R}^d$ where the gradient is nonzero: $\nabla f(x) \neq 0_d$. The rate of change of f at x along any $v \in \mathbb{R}^d$ is given by the directional derivative $\langle v, \nabla f(x) \rangle \in \mathbb{R}$. We seek the direction that yields the most negative rate of change:

$$v^* = \operatorname*{arg\,min}_{v \in \mathbb{R}^d : \, ||v|| \leq 1} \langle v, \nabla f(x) \rangle \tag{1}$$

where $||\cdot|| : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is some norm. Note that it is necessary to constrain the norm, otherwise there is no finite solution (we can blow up the objective to negative infinity by scaling). We constrain the norm size to be at most 1 without loss of generality, assuming that we will use an exact step size:

$$\eta^* = \operatorname*{arg\,min}_{\eta \in \mathbb{R}} f(x + \eta v^*)$$

We obtain our next location as

$$x_{\text{next}} = x + \eta^* v^*$$

By varying the choice of the norm in (1), we can derive various descent directions.

Lemma 1.1 (Gradient descent)**.** A solution of (1) using the l_2 norm is

$$v^* = -\frac{\nabla f(x)}{||\nabla f(x)||_2}$$

Proof. By the Cauchy-Schwarz inequality, for any $v \in \mathbb{R}^d$ with $||v||_2 \leq 1$ we have

$$\begin{aligned} \langle v, \nabla f(x) \rangle &\geq -||v||_2 \times ||\nabla f(x)||_2 \\ &\geq -||\nabla f(x)||_2 \end{aligned}$$

Choosing $v^* = -\frac{\nabla f(x)}{||\nabla f(x)||_2}$ achieves this minimum. □

Lemma 1.2 (Coordinate descent)**.** A solution of (1) using the l_1 norm is

$$l^* = \operatorname*{arg\,max}_{l=1}^{d} \left| \frac{\partial f(z)}{\partial z_l} \bigg|_{z=x} \right| \qquad\qquad v^* = -\textbf{sign}\left(\frac{\partial f(z)}{\partial z_{l^*}} \bigg|_{z=x} \right) \times e_{l^*} \tag{2}$$

where $\textbf{sign}(c) \in \{\pm 1\}$ returns the sign of $c \neq 0$ and $e_l \in \{0,1\}^d$ denotes the l-th standard basis vector in $\mathbb{R}^d$.

Proof. (1) is now

$$\min_{v \in \mathbb{R}^d} \sum_{l=1}^{d} v_l \times \frac{\partial f(z)}{\partial z_l} \bigg|_{z=x} \qquad \text{such that} \qquad \sum_{l=1}^{d} |v_l| \leq 1$$

This is a linear program over a convex polytope (l_1 ball of radius 1).[1] By the fundamental theorem of linear programming, the optimal value is attained at one of the verticies $\{\pm e_l\}_{l=1}^{d}$. We find a vertex v^* that minimizes the objective by choosing the dimension l^* with the largest $|\frac{\partial f(z)}{\partial z_{l^*}}|_{z=x}|$, then setting $v^* = -e_{l^*}$ if $\frac{\partial f(z)}{\partial z_{l^*}}|_{z=x} > 0$ and $v^* = e_{l^*}$ otherwise. □

[1] There is a standard trick to canonicalize the absolute-valued constraint: introduce auxiliary variables $t_1 \dots t_d \in \mathbb{R}$ and assert $\sum_{l=1}^{d} t_l \leq 1$, $t_l \geq v_l$ and $t_l \geq -v_l$ for all $l = 1 \dots d$.

2 Application to Boosting

Assume a finite hypothesis class $\mathcal{H} = \{h_1 \dots h_H\}$ of binary classifiers $h_l : \mathcal{X} \to \{\pm 1\}$. We will assume that $\mathcal{H}$ is flip-closed: given any dataset, if $h_p \in \mathcal{H}$ obtains accuracy p, we have $h_{1-p} \in \mathcal{H}$ that obtains accuracy $1 - p$.[2] Now any ensemble can be expressed by a parameter $\alpha \in \mathbb{R}^H$ as

$$g_\alpha(x) := \langle \alpha, h(x) \rangle \in \mathbb{R}$$

Given labeled examples $(x_1, y_1) \dots (x_N, y_N) \in \mathcal{X} \times \{\pm 1\}$, we consider the empirical exponential loss of $g_\alpha : \mathcal{X} \to \mathbb{R}$:

$$J(\alpha) := \sum_{i=1}^{N} \exp\left(-y_i g_\alpha(x_i)\right) = \sum_{i=1}^{N} \exp\left(-\sum_{l=1}^{H} \alpha_l y_i h_l(x_i)\right)$$

The partial derivative of J with respect to α_l for some particular $l \in \{1 \dots H\}$ is

$$\begin{aligned}
\frac{\partial J(\alpha)}{\partial \alpha_l} &= \sum_{i=1}^{N} \exp\left(-y_i g_\alpha(x_i)\right)\left(-y_i h_l(x_i)\right) \\
&\propto \sum_{i=1}^{N} D_\alpha(i)\left(-y_i h_l(x_i)\right) && \left(D_\alpha(i) := \frac{\exp\left(-y_i g_\alpha(x_i)\right)}{\sum_{j=1}^{N} \exp\left(-y_j g_\alpha(x_j)\right)}\right) \\
&= \sum_{i=1:\ h_l(x_i) \neq y_i}^{N} D_\alpha(i) - \sum_{i=1:\ h_l(x_i) = y_i}^{N} D_\alpha(i) \\
&= \epsilon_\alpha(h_l) - (1 - \epsilon_\alpha(h_l)) && \left(\epsilon_\alpha(h) := \sum_{i=1:\ h(x_i) \neq y_i}^{N} D_\alpha(i)\right) \\
&= 2\epsilon_\alpha(h_l) - 1
\end{aligned}$$

The coordinate l^* with the largest value of $|\frac{\partial J(\alpha)}{\partial \alpha_{l^*}}|$ at $\alpha \in \mathbb{R}^H$ is thus

$$l^* = \arg\max_{l=1}^{H} \left|\epsilon_\alpha(h_l) - \frac{1}{2}\right| = \arg\min_{l=1}^{H} \ \epsilon_\alpha(h_l)$$

where the second equality holds under the premise that $\mathcal{H}$ is flip-closed. Assuming $\nabla J(\alpha) \neq 0_H$ we must have $\epsilon_\alpha(h_{l^*}) < \frac{1}{2}$. The steepest descent direction (3) at $\alpha \in \mathbb{R}^H$ is

$$v^* = -\underbrace{\mathbf{sign}\left(\epsilon_\alpha(h_{l^*}) - \frac{1}{2}\right)}_{-1} \times e_{l^*} = e_{l^*}$$

Here, we see that even if $\mathcal{H}$ is not flip-closed, the algorithm will automatically include flipped classifiers. The optimal step size is $\eta^* = \arg\min_{\eta \in \mathbb{R}} \ F(\eta)$ where

$$F(\eta) := J(\alpha + \eta e_{l^*}) = \sum_{i=1}^{N} \exp\left(-\sum_{l=1}^{H} \alpha_l y_i h_l(x_i)\right) \exp\left(-\eta y_i h_{l^*}(x_i)\right)$$

Assuming $\epsilon_\alpha(h_{l^*}) > 0$, we can easily verify that

$$\eta^* = \frac{1}{2} \log\left(\frac{1 - \epsilon_\alpha(h_{l^*})}{\epsilon_\alpha(h_{l^*})}\right) \geq 0$$

Thus $\alpha_{\text{next}} = \alpha + \eta^* v^*$ corresponds to adding a single weighted classifier to g_α as follows:

$$g_{\alpha_{\text{next}}} = g_\alpha + \frac{1}{2} \log\left(\frac{1 - \epsilon_\alpha(h_{l^*})}{\epsilon_\alpha(h_{l^*})}\right) \times h_{l^*} \qquad\qquad h_{l^*} = \arg\min_{h \in \mathcal{H}} \ \epsilon_\alpha(h)$$

Initializing $\alpha = 0_H$ and taking T steps of coordinate descent is exactly the AdaBoost algorithm.

[2] This assumption is without any loss of generality since we can always expand $\mathcal{H}$ to include $h_{1-p} = -h_p$. In fact, the derivation below will work without making this assumption, but it will be messier.