Boosting 算法

王璐

Boosting 算法源于计算机科学家 Michael Kearns 的一个发问：一个弱学习算法能否被改造成一个强学习算法？具体来说，如果一个弱学习算法做分类的准确率只比随机猜测略高，有没有可能用它来构造一个错误率无限接近 0 的分类器？这个问题后来被 Schapire 和 Freund 解决了，他们发明了 AdaBoost 算法 (Freund et al., 1999)，是数据挖掘的 top 10 算法之一。

1 AdaBoost 算法

AdaBoost 是 adaptive boosting 的简称，它可以对任一做分类的弱学习算法 A 的效果进行增强。AdaBoost 的解决思路是利用算法 A 产生一系列分类结果，然后想办法巧妙地结合这些输出结果，降低训练集的出错率。但是算法 A 往往是确定的，如果总是给它相同的输入，它也只会输出相同的结果，所以每次产生新的分类结果时，我们需要对 A 的输入做一点改变，增加一些 "新信息"。AdaBoost 的做法是调整每次输入训练集的样本权重，它会提高前一轮分类错误的样本权重，降低前一轮分类正确的样本权重。最终容易分类的样本的权重可能会变得非常小，较难被正确分类的样本可能会占据所有权重。

用 $d_{t,i}$ 表示第 t 轮样本 $(\boldsymbol{x}_i, y_i)$ 的权重，向量 $\boldsymbol{d}_t = (d_{t,1}, \ldots, d_{t,n})$ 通常被称为为权重向量。用 $h^{(t)}(\boldsymbol{x}_i)$ 表示第 t 轮算法 A 对样本 $\boldsymbol{x}_i$ 的分类结果，规定 $y_i, h^{(t)}(\boldsymbol{x}_i) \in \{-1, 1\}, \forall i$. AdaBoost 对 $\boldsymbol{d}_t$ 的更新方式为:

$$
\begin{aligned}
d_{1,i} &= \frac{1}{n},\ \forall i \\
d_{t+1,i} &= \frac{d_{t,i}}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } y_i = h^{(t)}(\boldsymbol{x}_i) \\ e^{\alpha_t} & \text{if } y_i \neq h^{(t)}(\boldsymbol{x}_i) \end{cases} \\
&= \frac{d_{t,i}}{Z_t} e^{-\alpha_t y_i h^{(t)}(\boldsymbol{x}_i)}
\end{aligned} \tag{1}
$$

其中 $\alpha_t > 0$, Z_t 是归一化常数，保证第 $(t+1)$ 轮所有样本的权重和为 1 ($\sum_i d_{t+1,i} = 1$). 从(1)可以看出，AdaBoost 在每一轮会减小上一轮分类正确的样本权重，增加上一轮分类错误的样本权重。

AdaBoost 最终输出的结果是每一轮分类结果的线性组合：

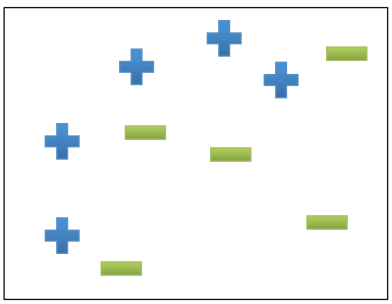
$$f(\boldsymbol{x}_i) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h^{(t)}(\boldsymbol{x}_i)\right) \tag{2}$$

其中

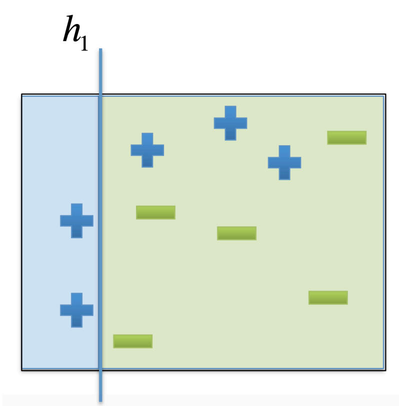
$$\begin{aligned} \alpha_t &= \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \\ \epsilon_t &= P_{i\sim \boldsymbol{d}_t}\left[h^{(t)}(\boldsymbol{x}_i) \neq y_i\right] = \sum_i d_{t,i}\mathbf{1}_{[h^{(t)}(\boldsymbol{x}_i)\neq y_i]}. \end{aligned} \tag{3}$$

即 ϵ_t 是第 t 轮分类错误样本的权重和。假设每一轮的弱分类器总可以保证 $\epsilon_t < 1/2$, 因此 $\alpha_t > 0$. 注意这里出现过两个权重，$\boldsymbol{d}_t$ 代表样本的权重，α_t 是做预测时对结果做线性组合的权重。

下面用一个简单的例子演示 AdaBoost 的工作流程。

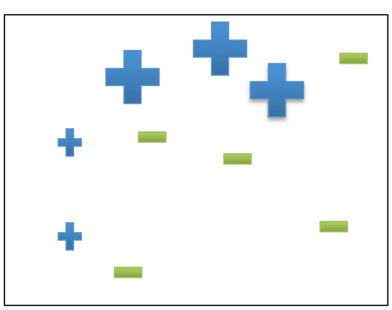


开始的时候所有样本权重相等。

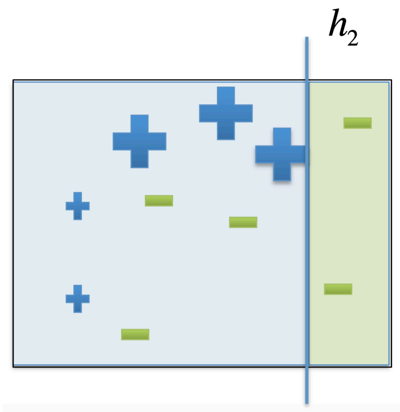


运行算法 A 将每个样本的分类结果记为 $h_1(x_i)$.

计算得 $\alpha_1 = 0.42$.

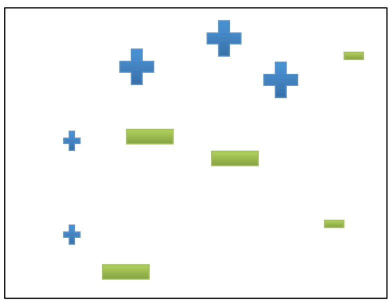


增大错误分类的样本权重，减小正确分类的样本权重。

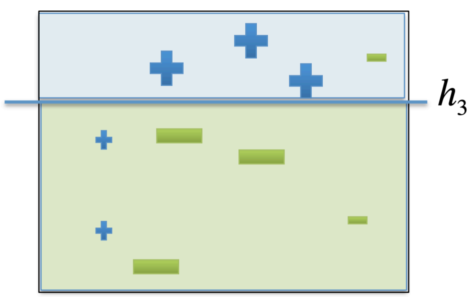


将调整权重后的样本输入算法 A 得到新的分类结果 h_2.

此时 $\alpha_2 = 0.66$.



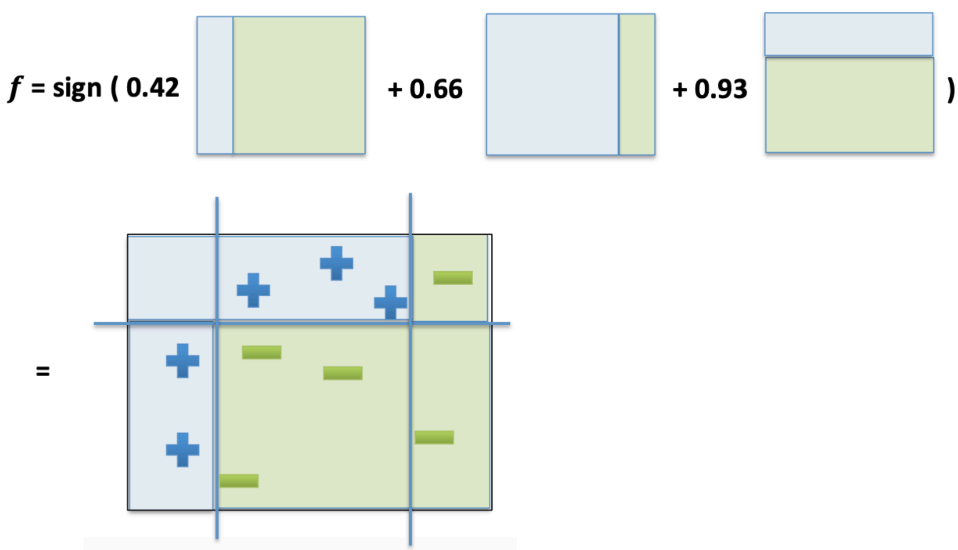
增大错误分类的样本权重，减小正确分类的样本权重。



将调整权重后的样本再次输入算法 A 得到新的分类结果 h_3.

得出 $\alpha_3 = 0.93$.

AdaBoost 最终输出的结果是每一轮分类结果的线性组合：



2 AdaBoost 统计解释

AdaBoost 最早由 Freund 和 Schapire 提出，之后有 5 个研究团队几乎同时给出了 AdaBoost 的统计解释 (Breiman, 1997; Friedman et al., 2000; Rätsch et al., 2001; Duffy and Helmbold, 1999; Mason et al., 2000)。从统计角度理解 AdaBoost 会发现，它本质上是一个坐标下降算法。

假设我们有 n 个训练样本 $\{(x_i, y_i) : y_i \in \{-1, 1\}\}_{i=1}^n$ 和 p 个弱分类器 $\{h_j : h_j(x) \in \{-1, 1\}\}_{j=1}^p$. 考虑用这些弱分类器的线性组合构造一个新的分类算法 f:

$$f(x) = \sum_{j=1}^p \lambda_j h_j(x). \tag{4}$$

该算法 f 在训练集上的错误率 (misclassification error) 定义为:

$$\text{Mis. err } = \frac{1}{n}\sum_{i=1}^n \mathbf{1}_{[y_i f(x_i) \le 0]}. \tag{5}$$

直接最小化(5)寻找最优的 f 是比较困难的，人们通常选择最小化(5)的一个凸上界 (convex upper bound) 函数，比如指数损失 (exponential loss) 函数

$$\frac{1}{n}\sum_{i=1}^n e^{-y_i f(x_i)} \tag{6}$$

就是(5)的一个光滑可导的上界函数，如图1所示。那么如何选择(4)中的 $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_p)^\top$ 使 f 的指数损失函数(6)最小?

定义 $n \times p$ 矩阵 M, 其元素为 $M_{ij} = y_i h_j(x_i)$. 由于 $y_i, h_j(x_i) \in \{-1, 1\}$, $\forall i, j$, 所以 M 由 1 和-1 构成。如果 $M_{ij} = 1$, 说明第 j 个分类器对样本 i 分类正确。

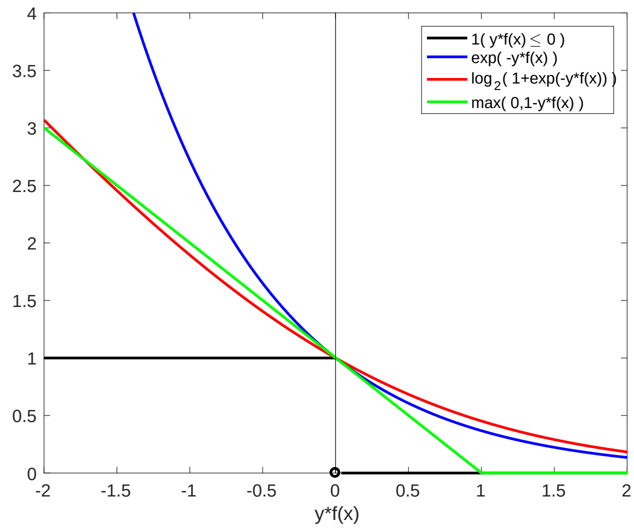
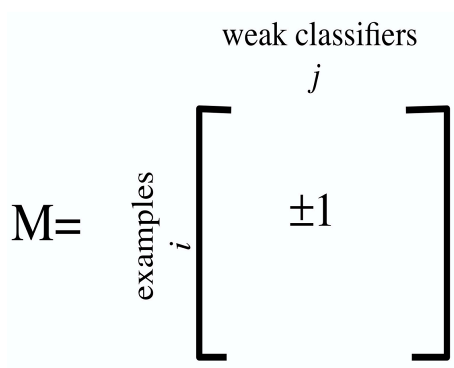


Figure 1: 函数 $\mathbf{1}\left(yf(x)\le 0\right)$ 及几种常用上界函数: exponential loss $e^{-yf(x)} \Rightarrow$ AdaBoost; logistic loss $\log_2\left(1+e^{-yf(x)}\right) \Rightarrow$ logistic regression $(y\in\{-1,1\})$; hinge loss $\max\left(0,1-yf(x)\right) \Rightarrow$ SVM.



此时

$$y_i f(x_i) = \sum_{j=1}^{p} \lambda_j y_i h_j(x_i) = (M\boldsymbol{\lambda})_i$$

其中 $(M\boldsymbol{\lambda})_i$ 代表向量 $M\boldsymbol{\lambda}$ 的第 i 个分量。则 $\boldsymbol{\lambda}$ 对应的 f 的指数损失为

$$L(\boldsymbol{\lambda}) = \frac{1}{n}\sum_{i=1}^{n} e^{-y_i f(x_i)} = \frac{1}{n}\sum_{i=1}^{n} e^{-(M\boldsymbol{\lambda})_i}. \tag{7}$$

接下来我们用坐标下降算法最小化(7): 在每步迭代 t, 选择 $\boldsymbol{\lambda}$ 的一个分量进行更新，即每个分类器对应一个方向，假设选择的是第 j_t 个分量，则沿 j_t 方向移动最优步长 α_t. 此时在每步迭代中，我们需要先找到方向 j 使得损失函数(7)在该方向下降得最快，即方向导数最小。用 $\boldsymbol{e}_j \in \mathbb{R}^p$ 表示单位基向量，即只有第 j 个元素为 1，其余为 0. 将 $\boldsymbol{\lambda}$ 在第 t 步的取值记为 $\boldsymbol{\lambda}_t$, 则第 t 步损失函数(7)在

第 j 个方向的方向导数为

$$
\begin{aligned}
\left.\frac{\partial L(\boldsymbol{\lambda}_t+\alpha\boldsymbol{e}_j)}{\partial\alpha}\right|_{\alpha=0} &= \left.\frac{\partial}{\partial\alpha}\left[\frac{1}{n}\sum_{i=1}^n e^{-(M(\boldsymbol{\lambda}_t+\alpha\boldsymbol{e}_j))_i}\right]\right|_{\alpha=0} \\
&= \left.\frac{\partial}{\partial\alpha}\left[\frac{1}{n}\sum_{i=1}^n e^{-(M\boldsymbol{\lambda}_t)_i-\alpha M_{ij}}\right]\right|_{\alpha=0} \\
&= \left.\frac{1}{n}\sum_{i=1}^n(-M_{ij})e^{-(M\boldsymbol{\lambda}_t)_i-\alpha M_{ij}}\right|_{\alpha=0} \\
&= -\frac{1}{n}\sum_{i=1}^n M_{ij}e^{-(M\boldsymbol{\lambda}_t)_i}.
\end{aligned}
$$

我们希望选取的方向导数越小越好，因此第 t 步选取的方向 j_t 为

$$
\begin{aligned}
j_t &\in \operatorname{argmin}_j\left[\left.\frac{\partial L(\boldsymbol{\lambda}_t+\alpha\boldsymbol{e}_j)}{\partial\alpha}\right|_{\alpha=0}\right] \\
&\in \operatorname{argmin}_j\left[-\frac{1}{n}\sum_{i=1}^n M_{ij}e^{-(M\boldsymbol{\lambda}_t)_i}\right] \\
&\in \operatorname{argmax}_j\left[\frac{1}{n}\sum_{i=1}^n M_{ij}e^{-(M\boldsymbol{\lambda}_t)_i}\right].
\end{aligned}
\tag{8}
$$

为了计算方便，我们将样本 i 经过归一化的指数损失记为：

$$
d_{t,i} = e^{-(M\boldsymbol{\lambda}_t)_i}/Z_t,\ \text{其中} Z_t = \sum_{i=1}^n e^{-(M\boldsymbol{\lambda}_t)_i}. \tag{9}
$$

后面会证明它们与 AdaBoost 中的权重向量 $\boldsymbol{d}_t$ 是等价的。根据(8)

$$
j_t \in \operatorname{argmax}_j\left[\frac{Z_t}{n}\sum_{i=1}^n M_{ij}d_{t,i}\right] = \operatorname{argmax}_j(\boldsymbol{d}_t^\top M)_j. \tag{10}
$$

当选定了方向 j_t 后，沿该方向移动的最优步长是多少？根据(7), $L(\boldsymbol{\lambda}_t+\alpha\boldsymbol{e}_{j_t})$ 是 α 的凸函数，因此只需找到使 j_t 对应的方向导数为 0 的步长 α_t.

$$
\begin{aligned}
0 &= \left.\frac{\partial L(\boldsymbol{\lambda}_t+\alpha\boldsymbol{e}_{j_t})}{\partial\alpha}\right|_{\alpha=\alpha_t} \\
0 &= \frac{1}{n}\sum_{i=1}^n(-M_{ij_t})e^{-(M\boldsymbol{\lambda}_t)_i-\alpha_t M_{ij_t}} \\
0 &= -\frac{1}{n}\sum_{i:M_{ij_t}=1} e^{-(M\boldsymbol{\lambda}_t)_i}e^{-\alpha_t} + \frac{1}{n}\sum_{i:M_{ij_t}=-1} e^{-(M\boldsymbol{\lambda}_t)_i}e^{\alpha_t} \\
0 &= -\frac{Z_t}{n}\sum_{i:M_{ij_t}=1} d_{t,i}e^{-\alpha_t} + \frac{Z_t}{n}\sum_{i:M_{ij_t}=-1} d_{t,i}e^{\alpha_t}
\end{aligned}
\tag{11}
$$

定义 $d_+ \triangleq \sum_{i:M_{ij_t}=1} d_{t,i}$, $d_- \triangleq \sum_{i:M_{ij_t}=-1} d_{t,i}$. 由(11)得

$$\begin{aligned} 0 &= d_+ e^{-\alpha_t} - d_- e^{\alpha_t} \\ \alpha_t &= \frac{1}{2}\ln\frac{d_+}{d_-} = \frac{1}{2}\ln\frac{1-d_-}{d_-}. \end{aligned} \tag{12}$$

因此使指数损失函数(7)最小的坐标下降算法可以总结为 Algorithm 1.

Algorithm 1 最小化指数损失函数(7)的坐标下降算法

$\boldsymbol{\lambda}_1 = \mathbf{0}$

$d_{1,i} = 1/n$, $i = 1, \dots, n$

for $t = 1 : T$ **do**

$\quad j_t \in \text{argmax}_j (\boldsymbol{d}_t^\top M)_j$

$\quad d_- = \sum_{i:M_{ij_t}=-1} d_{t,i}$

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

$\quad \boldsymbol{\lambda}_{t+1} = \boldsymbol{\lambda}_t + \alpha_t \boldsymbol{e}_{j_t}$

$\quad d_{t+1,i} = e^{-(M\boldsymbol{\lambda}_{t+1})_i}/Z_{t+1}$, 其中 $Z_{t+1} = \sum_{i=1}^n e^{-(M\boldsymbol{\lambda}_{t+1})_i}$

end for

为什么 Algorithm 1和 AdaBoost 是等价的？注意到该算法输出的 $\lambda_{T+1,j}$ 其实是在 j 方向上移动的总步长，即

$$\lambda_{T+1,j} = \sum_{t=1}^T \alpha_t \mathbf{1}_{[j_t=j]}. \tag{13}$$

则

$$f(x) = \sum_{j=1}^p \lambda_{T+1,j} h_j(x) = \sum_{j=1}^p \sum_{t=1}^T \alpha_t \mathbf{1}_{[j_t=j]} h_j(x) = \sum_{t=1}^T \alpha_t h_{j_t}(x). \tag{14}$$

如果令 AdaBoost 中的 $h^{(t)} = h_{j_t}$ 且两者的 $\{\alpha_t\}$ 相同，则(2)与(14)等价。

我们首先检查 AdaBoost 每轮使用的弱分类器 $h^{(t)}$ 与算法1每步选择的分类器 h_{j_t} 是否一样。一

个合理的假设是 AdaBoost 每轮在 p 个弱分类器中选择使(3)中定义的出错率 ϵ_t 最小的分类器，即

$$\begin{aligned}
j_t &\in \operatorname{argmin}_j \sum_i d_{t,i}\mathbf{1}_{[h_j(x_i)\neq y_i]} \\
&= \operatorname{argmin}_j \left[\sum_{i:M_{ij}=-1} d_{t,i}\right] = \operatorname{argmax}_j \left[-\sum_{i:M_{ij}=-1} d_{t,i}\right] \\
&= \operatorname{argmax}_j \left[1-2\sum_{i:M_{ij}=-1} d_{t,i}\right] \\
&= \operatorname{argmax}_j \left[\left(\sum_{i:M_{ij}=1} d_{t,i}+\sum_{i:M_{ij}=-1} d_{t,i}\right)-2\sum_{i:M_{ij}=-1} d_{t,i}\right] \\
&= \operatorname{argmax}_j \left[\sum_{i:M_{ij}=1} d_{t,i}-\sum_{i:M_{ij}=-1} d_{t,i}\right] \\
&= \operatorname{argmax}_j \left(\boldsymbol{d}_t^\top M\right)_j .
\end{aligned} \tag{15}$$

比较(10)和(15)可以看到，如果 AdaBoost 和算法1每步使用的 $\boldsymbol{d}_t$ 相同，则 AdaBoost 每步选择的分类器与算法1相同。

接下来检查 AdaBoost 每步的权重向量 $\boldsymbol{d}_t$ 与算法1是否相同。假设 AdaBoost 每步选择的分类器与算法1相同，即 $h^{(t)}=h_{j_t}, \forall t$, 且 AdaBoost 每轮使用的 α_t 与算法1每步移动的步长 α_t 相等，则 AdaBoost 中，

$$\begin{aligned}
d_{t+1,i} &= \frac{d_{t,i}e^{-M_{ij_t}\alpha_t}}{Z_t} = \frac{\frac{1}{n}\prod_{s=1}^t e^{-M_{ij_s}\alpha_s}}{\prod_{s=1}^t Z_s} = \frac{e^{-\sum_{s=1}^t M_{ij_s}\alpha_s}}{n\prod_s Z_s} \\
&= \frac{e^{-\sum_{s=1}^t\sum_{j=1}^p M_{ij}\mathbf{1}_{[j_s=j]}\alpha_s}}{n\prod_s Z_s} = \frac{e^{-\sum_{j=1}^p M_{ij}\lambda_{t+1,j}}}{n\prod_s Z_s} = \frac{e^{-(M\boldsymbol{\lambda}_{t+1})_i}}{n\prod_s Z_s}.
\end{aligned} \tag{16}$$

其中倒数第二个等号使用了等式(13). 注意(16)中的分母一定等于 $\sum_{i=1}^n e^{-(M\boldsymbol{\lambda}_{t+1})_i}$, 因为 AdaBoost 的权重向量 $\boldsymbol{d}_{t+1}$ 的和为 1。比较(16)和(9)可得，当 AdaBoost 每步选择的分类器及 α_s 与算法1相同, AdaBoost 的权重向量 $\boldsymbol{d}_{t+1}$ 和算法 1的 $\boldsymbol{d}_{t+1}$ 是一样的。

如果 AdaBoost 与算法1每步选择的分类器和 $\boldsymbol{d}_t$ 都相同，那么 AdaBoost 每步使用的 α_t 与算法1每步移动的步长 α_t 相等吗？先检查 AdaBoost 中的 ϵ_t, 根据(3),

$$\epsilon_t = \sum_i d_{t,i}\mathbf{1}_{[h_{j_t}(x_i)\neq y_i]} = \sum_{i:h_{j_t}(x_i)\neq y_i} d_{t,i} = \sum_{i:M_{ij_t}=-1} d_{t,i} = d_- \tag{17}$$

则在 AdaBoost 中,

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) = \frac{1}{2}\ln\left(\frac{1-d_-}{d_-}\right).$$

与(12)比较可得此时 AdaBoost 中的 α_t 与算法1相同。

Adaboost 和算法1每步迭代涉及三个要素：权重向量 $\boldsymbol{d}_t$, 分类器 $h^{(t)}$ 和参数 α_t. 以上我们证明了固定其中任意两个要素相等，则第三个要素在两个算法中也相等。注意到两个算法使用的初始值 $\boldsymbol{d}_1$ 相同，由(15)得 $h^{(1)} = h_{j_1}$, 则两个算法得到的 α_1 必然相等，因此权重向量 $\boldsymbol{d}_2$ 也相同，以此类推，两个算法每轮迭代的三要素都是相等的。所以 AdaBoost 等价于用坐标下降算法最小化一个指数损失函数。

Theorem 1. *如果存在 $\gamma_A > 0$ 使得 AdaBoost 每轮出错样本的权重和*

$$\epsilon_t = \sum_i d_{t,i}\mathbf{1}_{[h_{j_t}(x_i)\neq y_i]} = \frac{1}{2} - \gamma_t,\ \text{且}\gamma_t > \gamma_A, \forall t. \tag{18}$$

则 AdaBoost 在训练集上的错误率(5)以指数速率下降：

$$\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}_{[y_i f(x_i)\leq 0]} \leq e^{-2\gamma_A^2 T}. \tag{19}$$

Proof. 证明的思路是找到(7)中指数损失函数 $L(\boldsymbol{\lambda}_{t+1})$ 和 $L(\boldsymbol{\lambda}_t)$ 的递归关系，即找出每步迭代减小的训练集误差，然后把这些误差累加起来得出总误差的上界。根据与 AdaBoost 等价的算法1,

$$\begin{aligned} L(\boldsymbol{\lambda}_{t+1}) = L(\boldsymbol{\lambda}_t + \alpha_t \boldsymbol{e}_{j_t}) &= \frac{1}{n}\sum_{i=1}^{n} e^{-\left[M(\boldsymbol{\lambda}_t+\alpha_t\boldsymbol{e}_{j_t})\right]_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-(M\boldsymbol{\lambda}_t)_i - \alpha_t M_{ij_t}} \\ &= \frac{e^{-\alpha_t}}{n}\sum_{i:M_{ij_t}=1} e^{-(M\boldsymbol{\lambda}_t)_i} + \frac{e^{\alpha_t}}{n}\sum_{i:M_{ij_t}=-1} e^{-(M\boldsymbol{\lambda}_t)_i}. \end{aligned} \tag{20}$$

根据(9),

$$\begin{aligned} \sum_{i:M_{ij_t}=1} e^{-(M\boldsymbol{\lambda}_t)_i} &= \sum_{i:M_{ij_t}=1} d_{t,i}Z_t = d_+ Z_t \\ \sum_{i:M_{ij_t}=-1} e^{-(M\boldsymbol{\lambda}_t)_i} &= \sum_{i:M_{ij_t}=-1} d_{t,i}Z_t = d_- Z_t. \end{aligned} \tag{21}$$

将(21)代入(20)得

$$L(\boldsymbol{\lambda}_{t+1}) = \frac{Z_t}{n}d_+ e^{-\alpha_t} + \frac{Z_t}{n}d_- e^{\alpha_t}. \tag{22}$$

注意到

$$L(\boldsymbol{\lambda}_t) = \frac{1}{n}\sum_{i=1}^{n} e^{-(M\boldsymbol{\lambda}_t)_i} = \frac{Z_t}{n}. \tag{23}$$

将(23)代入(22)得

$$\begin{aligned} L(\boldsymbol{\lambda}_{t+1}) &= L(\boldsymbol{\lambda}_t)\left(e^{-\alpha_t}d_+ + e^{\alpha_t}d_-\right) \\ &= L(\boldsymbol{\lambda}_t)\left[e^{-\alpha_t}(1-d_-) + e^{\alpha_t}d_-\right]. \end{aligned} \tag{24}$$

又根据(12),

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-d_-}{d_-}\right) \Rightarrow e^{\alpha_t} = \left(\frac{1-d_-}{d_-}\right)^{1/2}, e^{-\alpha_t} = \left(\frac{1-d_-}{d_-}\right)^{-1/2}$$

代入(24)得

$$\begin{aligned} L(\boldsymbol{\lambda}_{t+1}) &= L(\boldsymbol{\lambda}_t)2\left[d_-(1-d_-)\right]^{1/2} \\ &= L(\boldsymbol{\lambda}_t)2\left[\epsilon_t(1-\epsilon_t)\right]^{1/2}. \end{aligned} \tag{25}$$

其中最后一步用到了等式(17). 由 $\boldsymbol{\lambda}_1 = \mathbf{0}$ 得 $L(\boldsymbol{\lambda}_1) = 1$. 反复利用递推关系(25)可得

$$L(\boldsymbol{\lambda}_{T+1}) = \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)}. \tag{26}$$

由定理条件(18)得

$$2\sqrt{\epsilon_t(1-\epsilon_t)} = 2\sqrt{\left(\frac{1}{2}-\gamma_t\right)\left(\frac{1}{2}+\gamma_t\right)} = \sqrt{1-4\gamma_t^2} \le \sqrt{e^{-4\gamma_t^2}} = e^{-2\gamma_t^2} < e^{-2\gamma_A^2}. \tag{27}$$

将(27)代入(26)，再由指数损失函数(7)是错误率(5)的上界函数可得

$$\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}_{[y_i f(x_i)\le 0]} \le L(\boldsymbol{\lambda}_{T+1}) \le \prod_{t=1}^{T} e^{-2\gamma_t^2} < e^{-2\gamma_A^2 T}.$$

□

3 AdaBoost 概率解释

在一些分类问题中，我们不仅希望对 y 做出准确预测，还希望计算出概率 $P(Y=1\mid x)$，比如发现石油的概率、失败的概率、收到垃圾邮件的概率等。下面的定理给出了从 AdaBoost 算法计算 $P(Y=1\mid x)$ 的方法。

Theorem 2 (Friedman et al. (2000))**.** 使指数损失函数的期望

$$E_Y\left[e^{-Yf(x)}\right]$$

最小的 $f(x)$ 为

$$f(x) = \frac{1}{2}\ln\frac{P(Y=1\mid x)}{P(Y=-1\mid x)}.$$

Proof.

$$E\left[e^{-Yf(x)}\right] = P(Y=1\mid x)e^{-f(x)} + P(Y=-1\mid x)e^{f(x)}$$

$$0 = \frac{dE\left[e^{-Yf(x)}\right]}{df(x)} = -P(Y=1\mid x)e^{-f(x)} + P(Y=-1\mid x)e^{f(x)}$$

$$P(Y=1 \mid x)e^{-f(x)} = P(Y=-1 \mid x)e^{f(x)}$$
$$\frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)} = e^{2f(x)} \Rightarrow f(x) = \frac{1}{2}\ln\frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)}.$$

□

根据定理2, 可以如下从 AdaBoost 输出的函数 f 中计算 $P(Y=1 \mid x)$:

$$f(x) = \frac{1}{2}\ln\left[\frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)}\right] = \frac{1}{2}\ln\left[\frac{P(Y=1 \mid x)}{1-P(Y=1 \mid x)}\right]$$
$$P(Y=1 \mid x) = \frac{e^{2f(x)}}{1+e^{2f(x)}}.$$

练习：证明使 logistic loss 的期望

$$E_Y\left[\log_2(1+e^{-Yf(x)})\right]$$

最小的 $f(x)$ 为

$$f(x) = \ln\frac{P(Y=1 \mid x)}{P(Y=-1 \mid x)}.$$

References

Breiman, L. (1997). Arcing the edge. Technical report, Technical Report 486, Statistics Department, University of California at ....

Duffy, N. and Helmbold, D. (1999). A geometric approach to leveraging weak learners. In *European conference on computational learning theory*, pages 18–33. Springer.

Freund, Y., Schapire, R., and Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612.

Friedman, J., Hastie, T., Tibshirani, R., et al. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407.

Mason, L., Baxter, J., Bartlett, P. L., and Frean, M. R. (2000). Boosting algorithms as gradient descent. In *Advances in neural information processing systems*, pages 512–518.

Rätsch, G., Onoda, T., and Müller, K.-R. (2001). Soft margins for adaboost. *Machine learning*, 42(3):287–320.

