

# 第8章 提升方法 AdaBoost 算法

一 提升方法的基本思路:

将弱可学习方法 提升为强可学习方法

PAC 学习框架 (泛化误差上界)

二 集成学习 两个主要类别

序列方法 学习下一个模型依赖于上一个模型的结果

并行方法

三

AdaBoost — 分类问题

提升树 < 分类  
回归

模型的表现  
整体

8.1 AdaBoost 算法 样本点 → 权重

输入: 训练数据集 弱学习算法

输出: 最终分类器  $G(x) = \text{sign} \left[ \sum_{m=1}^M \alpha_m G_m(x) \right]$

(1) 初始化训练数据的权值分布 — 每个样本点的权重

$$D_1 = (w_{11}, \dots, w_{12}, \dots, w_{1N})$$

初始化为一样  $\frac{1}{N}$

$$w_{1i} = \frac{1}{N}, i=1, 2, \dots, N$$

(2) 对  $M=1, 2, \dots, m$

(a) 使用具有权值分布  $D_m$  的训练数据集学习, 得到基本分类器

$$G_m(x): x \rightarrow \{-1, +1\}$$

(b) 计算  $G_m(x)$  在训练数据集上的分类误差率

$$e_m = P(G_m(x_i) \neq y_i) = \sum_{i=1}^N w_{mi} I(G_m(x_i) \neq y_i)$$

一定小于 0.5, 若大于 0.5, 则将分类结果取负号 — 8.1

(c) 计算  $G_m(x)$  的系数

$$\alpha_m = \frac{1}{2} \log \frac{1 - e_m}{e_m} \quad - 8.2$$

自然对数

(d) 更新训练数据集的权值分布

$$D_{m+1} = (w_{m+1,1}, \dots, w_{m+1,N}) \quad - 8.3$$

$$w_{m+1,i} = \frac{w_{mi}}{\sum_m} \exp(-\alpha_m y_i G_m(x_i)) \quad \begin{matrix} \text{分类错误的点} \\ \text{在下一个模型中权重} \uparrow \\ \hookrightarrow \end{matrix} \quad i=1, 2, \dots, N \quad - 8.4$$

$\Sigma$  为规范因子 — 使得  $w_{m+1,i}$  对于所有  $i$  求和为 1

~~$$\Sigma_m = \sum_{i=1}^N w_{mi} \exp(-\alpha_m y_i G_m(x_i)) \quad - 8.5$$~~

它使  $D_{m+1}$  成为一个概率分布

(3) 构建基本分类器的线性组合

$$f(x) = \sum_{m=1}^M \alpha_m G_m(x)$$

得到最终分类器

$$G(x) = \text{Sign}(f(x)) = \text{Sign} \left[ \sum_{m=1}^M \alpha_m G_m(x) \right]$$

## 8.2 AdaBoost 算法的训练误差分析

定理 8.1 (AdaBoost 的训练误差界)

AdaBoost 算法最终分类器的训练误差界为

$$\frac{1}{N} \sum_{i=1}^N I(g(x_i) \neq y_i) \leq \frac{1}{N} \sum_i \exp(-y_i f(x_i)) = \prod_{m=1}^M \sum_m$$

训练误差

每步的规范化  
因子的乘积

定理 8.2 (二分类问题 AdaBoost 的训练误差界)

$$\prod_{m=1}^M \sum_m = \prod_{m=1}^M [2 \cdot \sqrt{e_m(1-e_m)}] = \prod_{m=1}^M \sqrt{1-4\gamma_m^2} \leq \exp\left(-2 \sum_{m=1}^M \gamma_m^2\right)$$

这里,  $\gamma_m = \frac{1}{2} - e_m$ , 每轮训练集上的分类误差率

$\gamma_m$

★ 可以解释为当前模型对随机猜测结果的提升程度

训练误差随着训练轮数↑而↓

且以指数速度↓, 即轮数↑, 误差下降很快

## 8.3 AdaBoost 算法的解释

问题: 二分类问题

模型: 加法模型  $f(x) = \sum_{m=1}^M \beta_m b(x; \gamma_m)$

M事先给定

b(损失函数事先给定)

$$f(x) = \sum_{m=1}^M \beta_m b(x; \gamma_m) \quad \text{要求出 } \beta_m \text{ 与 } \gamma_m$$

策略: 损失函数为指数函数

$$L(y, f(x)) = \exp[-y \cdot f(x)]$$

预测值正确为  $e^{-1}$

错误为  $e^1$

算法: 前向分步算法

算法：前向分步算法

$$f_0(x) = 0$$

$$f_1(x) = f_0(x) + \beta_1 b(x; y_1)$$

$$\beta_1, y_1 = \underset{y_1, \beta_1}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^N L[y_i, f_1(x_i)]$$

用最小化经验  
风险  
来求  $\beta, y_1$

$$f_2(x) = f_1(x) + \beta_2 b(x; y_2)$$

$$\beta_2, y_2 = \underset{y_2, \beta_2}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^N L[y_i, f_2(x_i)]$$

$$f_m(x) \rightarrow f(x)$$

$$b(x; y_m) \rightarrow G_m(x)$$

$$\beta_m \rightarrow \alpha_m$$

可以对 AdaBoost 进行变型，一般是在 框架不变的情况下 对策略进行替换

## 8.4 提升树

基本分类器：分类树或回归树

提升树模型

$$f_M(x) = \sum_{m=1}^M T(x; \theta_m)$$

对回归树来说  
 $\theta_m$  对应了一个空间上的  
 划分，这个划分对应了  
 每个叶子节点与其上的  
 拟合值

前向分步算法：

$$f_m(x) = f_{m-1}(x) + T(x; \theta_m)$$

$$\hat{\theta}_m = \arg \min \sum_{i=1}^N L(y_i, f_{m-1}(x_i) + T(x_i; \theta_m))$$

$$L[y - f(x)] = \frac{1}{2} [y - f(x)]^2$$

为了求导方便  
— 平方误差损失

算法 8.3 (回归问题的提升树算法)

采用平方误差损失 用前向分步算法求解

输入：训练数据集

输出：提升数树  $f_M(x)$

(1) 初始化  $f_0(x) = 0$

(2) 对  $m=1, 2, \dots, M$

(a) 按  $T(x; \theta) = \sum_{j=1}^J c_j I(x \in R_j)$  计算残差

$$r_{mi} = y_i - f_{m-1}(x_i), i=1, 2, \dots, N$$

(b) 拟合残差  $r_{mi}$  学习一个回归树，得到  $T(x; \theta_m)$

(c) 更新  $f_m(x) = f_{m-1}(x) + T(x; \theta_m)$

(3) 得到回归问题提升树

$$f_M(x) = \sum_{m=1}^M T(x; \theta_m)$$

注意：提升树更新参数 $\theta$ 时，不是直接根据最小化经验风险来求，而是通过转化为计算残差这个数推导

$$\begin{aligned}
 L[y, f(x)] &= [y - f(x)]^2 \\
 &= [y - f_{m-1}(x) - T(x; \theta)]^2 \\
 &= [r_{m-1} - T(x; \theta)]^2 \\
 &= L[r_{m-1}, T(x; \theta_m)]
 \end{aligned}$$

第m棵树  
有J个叶子节点  
空间的划分

有时候残差不是那么好算（没有用对数/平方损失）

有了梯度提升方法

算法 8.4 (梯度提升算法)

输入：训练数据集；损失函数 $L(y, f(x))$

输出：回归树 $f(x)$

(1) 初始化

$$f_0(x) = \arg \min_c \sum_{i=1}^N L(y_i, c)$$

(2) 对 $m=1, 2, \dots, M$

(a) 对 $i=1, 2, \dots, N$ ，计算

$$r_{mi} = - \left[ \frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right] \quad f(x) = f_{m-1}(x)$$

(b) 对  $r_{mi}$  拟合一个回归树，得到第  $m$  棵树的叶结点

区域  $R_{mj}$ ,  $j=1, 2, \dots, J$

(c) 对  $j=1, 2, \dots, J$ , 计算

$$C_{mj} = \arg \min_c \sum_{x_i \in R_{mj}} L(y_i, f_{m-1}(x_i) + c)$$

(d) 更新  $f_m(x) = f_{m-1}(x) + \sum_{j=1}^J C_{mj} I(x \in R_{mj})$

只加上相应区域的

(3) 得到回归树  $M$  棵树

$$\hat{f}(x) = f_M(x) = \sum_{m=1}^M \sum_{j=1}^J C_{mj} I(x \in R_{mj})$$

每棵树上有  $J$  个区域

问题.

① 为什么可以用负梯度去代替残差

② 为什么只能求  $\theta_m$  中的  $R_{mj}$ ,  $C_{mj}$  还需要将  $R_{mj}$  代入才可求出

问题 ①

若  $L$  为平方损失函数，则 (用一阶泰勒展开来证)

$$\frac{[y_i - f(x_i)]^2}{f'(x_i)} = -2[y_i - f(x_i)] \quad r_{mi} = 2[y_i - f(x_i)]$$

(有 2 说明不能控制系数)

$2[y_i - f(x_i)]$  只能代表残差,

但不能等同残差.

∴ 有 (c) 步

∴ 在 C 步中，我们要根据原始的  $y_i$  与  $f_{mi}$  来求解 C.

∴ 在 b 步中，我们已经确定了叶子划分区域  $R_{mi}$

∴ C 步中，我们可以只考虑一个 ~~所有~~ 叶子节点

求  $C_{mi}$  时

∴ 要找到这个叶子节点中的样本点。

再求这些样本点的经验损失。

对这些样本点 真实值  $y_i$ ,  $f_{mi}(x_i)$  + 上一步得到的 C  
 $\Rightarrow C_{mi}$  (找到 C, 使这些样本点的经验损失最小)

用前向分步算求解以指数函数为损失函数的加法  
等价于 AdaBoost 模型

$$f(x) = \sum_{m=1}^M \alpha_m G_m(x)$$

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

$$f_{m-1}(x) = \sum_{j=1}^{m-1} \alpha_j G_j(x)$$

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

$$\alpha_m, G_m = \arg \min_{\alpha, G} \sum_{i=1}^N \exp \{-y_i [f_{m-1}(x_i) + \alpha G(x_i)]\}$$

$$= \arg \min \sum \overline{w}_{m,i} \exp \{-\alpha y_i G(x_i)\}$$

$$= \arg \min \sum \overline{w}_{m,i} \exp (-\alpha y_i G(x_i))$$

$$\begin{aligned} & \min \sum_{i \in M_1} \overline{w}_{m,i} \exp(-\alpha) + \sum_{i \in M_2} \overline{w}_{m,i} \exp(\alpha) \\ & \quad \text{分类正确} \qquad \qquad \qquad \text{分类错误} \\ & = \sum_{i \in M_1} \overline{w}_i \exp(-\alpha) + \sum_{i \in M_2} \overline{w}_i \exp(-\alpha) + \sum_{i \in M_2} \overline{w}_i [\exp(\alpha) - \exp(-\alpha)] \end{aligned}$$

$$= \exp(-\alpha) \sum_i \overline{w}_i + [\exp(\alpha) - \exp(-\alpha)] \sum_i \overline{w}_i I(y_i \neq G(x_i))$$

$$G_m^* = \arg \min_a \sum_i \overline{w}_i I(y_i \neq G(x_i))$$

得到  $G_m^*$  后求  $\alpha_m$

$$\alpha_m = \arg \min_{\alpha} \sum_i \bar{w}_i \exp(-\alpha \cdot y_i G^*(x_i))$$

$$= \sum_{i \in M_1} \bar{w}_i \exp(-\alpha) + \sum_{i \in M_2} \bar{w}_i \exp(\alpha)$$

$$= (e^\alpha - e^{-\alpha}) \sum_i \bar{w}_i I(y_i \neq G(x_i)) + e^{-\alpha} \sum_i \bar{w}_i$$

要使其最小，对  $\alpha$  求导

$$(e^\alpha + e^{-\alpha}) \sum_i \bar{w}_i I(y_i \neq G(x_i)) - e^{-\alpha} \sum_i \bar{w}_i = 0$$

等号两边都乘上  $e^\alpha$  得

$$(e^{2\alpha} + 1) \cdot \sum_i \bar{w}_i I = \sum_i \bar{w}_i$$

$$e^{2\alpha} = \frac{\sum \bar{w}_i}{\sum \bar{w}_i I} - 1$$

$$\Rightarrow \alpha = \frac{1}{2} \ln \frac{\sum \bar{w}_i - \sum \bar{w}_i I}{\sum \bar{w}_i I}$$

分子分母都除以  $\sum \bar{w}_i$  得

$$\Rightarrow 1 - \frac{\sum \bar{w}_i I}{\sum \bar{w}_i} e_m$$

$$= (b-1) \frac{\sum \bar{w}_i I}{\sum \bar{w}_i} + (b-1) \frac{\sum \bar{w}_i}{\sum \bar{w}_i} - b$$

到此， $G(x)$  与  $\alpha$  都与 AdaBoost 对应上。

但还差权重的更新，回过头来看  $\bar{w}_i I$

$$\bar{w}_{mi} = \exp(-y_i f_{m-1}(x_i))$$

$$= \exp(-y_i \sum_{j=1}^{m-1} \alpha_j g_j(x_i))$$

$$= \prod_j \exp(-y_i \alpha_j g_j(x_i))$$

# Adaboost 训练误差

先：权重与样本分布的理解：

例：有3个数据集，有3个样本点，假设此数据集是分布

则

$$(x_1, y_1) \quad (x_2, y_2) \quad (x_3, y_3) \quad D_1$$

$$\frac{1}{3} \quad \frac{1}{3} \quad \frac{1}{3} \rightarrow \text{独立同分布}$$

更新权重（即改变样本分布）★

$$(x_1, y_1) \quad (x_2, y_2) \quad (x_3, y_3) \quad D_2$$

$$\frac{2}{3} \text{ 即 } \frac{4}{6} \quad \frac{1}{6} \quad \frac{1}{6} \rightarrow \text{不是独立同分布}$$

$$\underbrace{(x_1, y_1) \quad (x_2, y_2) \quad \dots \quad (x_n, y_n)}_{4\uparrow} \quad (x_3, y_3) \rightarrow \text{看作独立同分布}$$

训练误差 证：

$$G_i(x_i) = \text{sign}[f(x)]$$

$$\begin{aligned} & \frac{1}{N} \sum_i I(G_i(x_i) \neq y_i) \\ \stackrel{\textcircled{1}}{\leq} & \frac{1}{N} \sum \exp(-y_i f(x_i)) = \prod_{m=1}^M \sum_m \end{aligned}$$

先看①

对于正确分类点： $I(G_i(x_i) \neq y_i) = 0 \leq \exp(-y_i f(x_i)) = e^{-1}$

对于误分类点： $I(G_i(x_i) \neq y_i) = 1 \leq \exp(-y_i f(x_i)) = e$

∴ 对于每个点，即求和的每一项，左边都小于右边

∴ ①成立

再看②

$\Sigma_m$  出现在 8.4 与 8.5 两式中

对 8.4 式子有

$$\Sigma_m \cdot w_{m,i} = w_{m,i} \exp(-\alpha_m \cdot y_i G_m(x_i)) \quad — 1$$

对 8.5 式子有

$$\Sigma_m = \sum_i w_{m,i} \exp(-\alpha_m \cdot y_i G_m(x_i)) \quad — 2$$

对 1 式来说

连接了每个样本点在下一轮的权重值与本轮权值的一个关系

对 2 式来说

连接了在整个数据集上 N 个权值与分类效果(指数形式)之间的关系

$$\Sigma_1 \cdot w_{1,i} = w_{1,i} \exp[-\alpha_{1,m} \cdot y_i G_1(x_i)]$$

$$\Sigma_2 \cdot w_{2,i} = w_{2,i} \exp[-\alpha_{2,m} \cdot y_i G_2(x_i)]$$

⋮

$$\Sigma_{M-1} \cdot \overline{w_{M,i}} = w_{M,i} \exp[-\alpha_{M-1} \cdot y_i G_{M-1}(x_i)]$$

相乘

$$\prod_{m=1}^{M-1} \Sigma_m w_{M,i} = w_{i,i} \exp[-y_i \sum_{m=1}^{M-1} \alpha_m G_m(x_i)] \quad \text{比较}$$

左边还差  $\Sigma_M$ , 多了  $w_{M,i}$

右边对于  $f(x) = \sum_{m=1}^M \alpha_m G_m$  来说差了 1 个  $\alpha_M G_M(x_i)$   
所需的

左右两乘上

$$\prod_{m=1}^M \sum_i w_{m,i} \exp[-d_m G_m(x_i)] \quad (\text{原式左边})$$
$$= \frac{1}{N} \exp[-y_i f(x_i)] \quad (\text{原式右边})$$

我们最要的是对于整个数据集中的  
但上式是对于每个样本点来说的

∴ 求和

$$\prod_{m=1}^M \sum_i w_{m,i} \exp[-d_m G_m(x_i)] = \frac{1}{N} \sum \exp[-y_i f(x_i)]$$
$$= \sum_m$$

这个定理告诉我们训练误差可以被  $\sum_m$  控制住。

若想让训练误差  $\downarrow$ , 则意味着  $\sum_m \downarrow$

$$\sum_m = \sum_i w_{m,i} \exp[-d_m y_i G_m(x_i)]$$

↓                      ↓  
上一轮              本轮

首先知道  $w_m$  与  $G_m$ , 不知道  $d_m$

∴ 可以通过最小化  $\sum_m$  来求  $d_m$

$$d_m = \underset{d_m}{\operatorname{argmin}} \sum_m$$

$$\frac{\partial \sum_m}{\partial d_m} = -w_{m,i} \cdot y_i G_m(x_i) \exp[-d_m y_i G_m(x_i)]$$

$$d_m = \frac{1}{2} \ln \frac{1 - e_m}{e_m} \quad \text{定理 8.1}$$

定理8.2中

用一个多项式去逼近一个函数  
↑

$$\sqrt{1-4y^2} \leq \exp(-2y^2) \quad \text{用泰勒展开，在 } x=0 \text{ 地方}$$

$$e^x \sqrt{1-x}$$

$$f(x) = \sqrt{1-x} = (1-x)^{\frac{1}{2}}$$

$$f'(x) = -\frac{1}{2}(1-x)^{-\frac{1}{2}}$$

$$f''(x) = -\frac{1}{4}(1-x)^{-\frac{3}{2}}$$

$$\begin{aligned} f(x) &= f(0) + x \cdot f'(0) + \frac{1}{2}x^2 \cdot f''(0) + \dots \\ &= 1 - \frac{1}{2}x - \frac{1}{8}x^2 \end{aligned}$$

对于  $\sqrt{1-4y^2}$  有

$$f(4y^2) = 1 - 2y^2 - 2y^4$$

$$g(x) = e^x$$

$$g'(x) = e^x$$

$$g''(x) = e^x$$

$$g(x) = g(0) + g'(0) + \frac{1}{2}x^2 g''(0) + \dots$$

$$= 1 + x + \frac{1}{2}x^2 + \dots$$

对于  $e^{-2y^2}$  有

$$g(-2y^2) = 1 - 2y^2 + 2y^4$$

$$y = \frac{1}{2} - e_m \in (0, \frac{1}{2}]$$

---

$$\therefore 1 - 2y^2 - 2y^4 \leq 1 - 2y^2 + 2y^4$$

$$\therefore \sqrt{1-4y^2} \leq \exp(-2y^2)$$

当  $y=0$  时等号成立