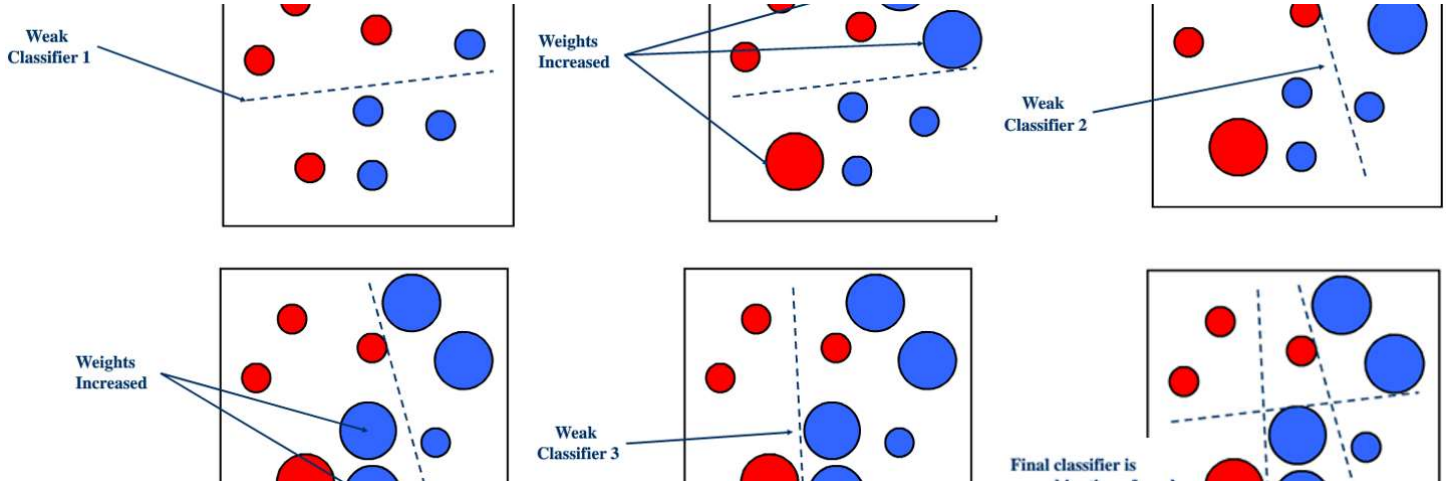


AdaBoost算法

Eureka

关注他

184 人赞同了该文章

专栏文章汇总

文章结构：

0： 回顾boosting算法的基本原理

1： 提升方法AdaBoost算法

- 1.1 提升方法的基本思路
- 1.2 AdaBoost算法（分类）
- 1.3 AdaBoost算法（回归）

2： Adaboost算法的训练误差分析（了解）

3: AdaBoost算法的解释

- 3.1 前向分布算法
- 3.2 前向分布算法与AdaBoost

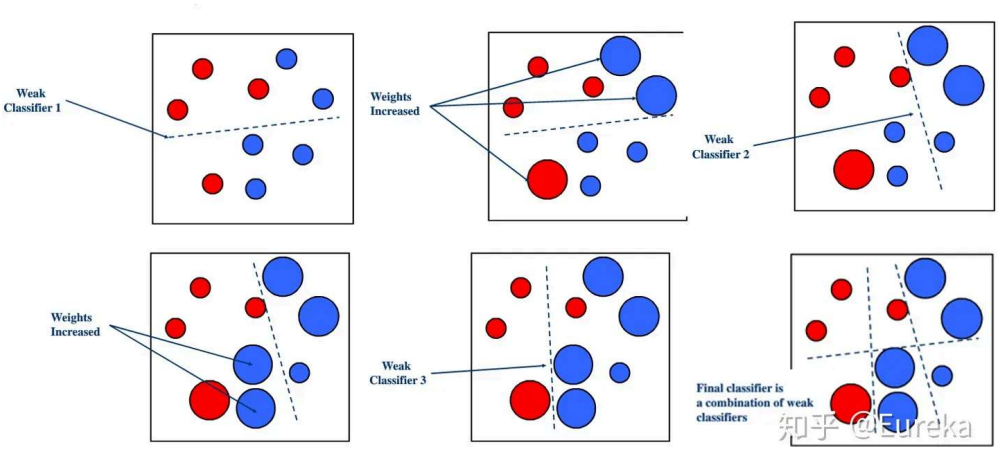
4： Adaboost算法的正则化

5： Adaboost小结

6： 参考文献

> 在集成学习-Boosting,Bagging与Stacking 中，讲到了集成学习按照个体学习器之间是否存在依赖关系可以分为两类，第一个是个体学习器之间存在强依赖关系，另一类是个体学习器之间不存在强依赖关系。前者的代表算法就是是boosting系列算法。在boosting系列算法中，Adaboost是最著名的算法之一。Adaboost既可以用作分类，也可以用作回归。

0： 回顾boosting算法的基本原理



Boosting算法的工作机制是首先从训练集用初始权重训练出一个弱学习器1，根据弱学习的学习误差率表现来更新训练样本的权重，使得之前弱学习器1学习误差率高的训练样本点的权重变高，使得这些误差率高的点在后面的弱学习器2中得到更多的重视。然后基于调整权重后的训练集来训练弱学习器2.，如此重复进行，直到弱学习器数达到事先指定的数目T，最终将这T个弱学习器通过集合策略进行整合，得到最终的强学习器。

不过有几个具体的问题Boosting算法没有详细说明。
1. 如何计算学习误差率e?
2. 如何得到弱学习器权重系数 α ?
3. 如何更新样本权重D?
4. 使用何种结合策略?

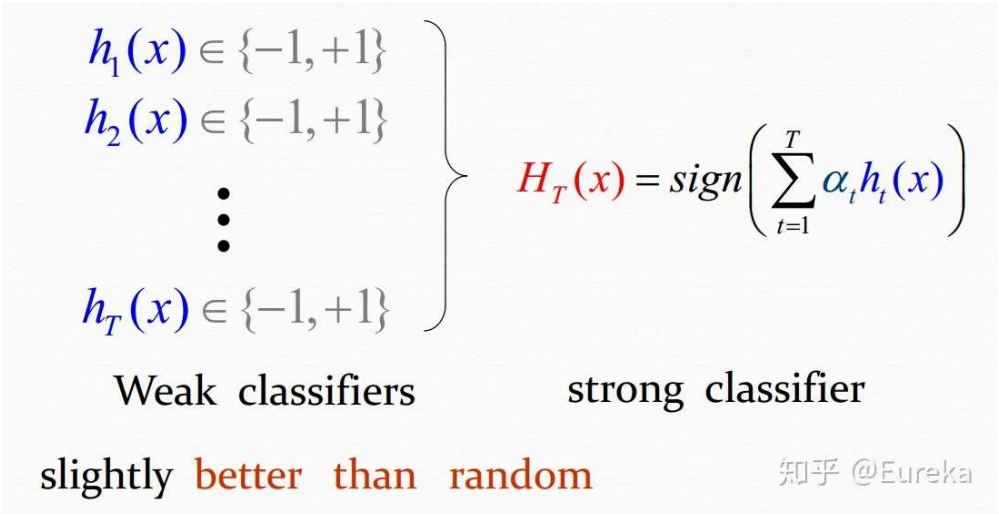
下面我们看看Adaboost是怎么解决的。

1： 提升方法AdaBoost算法

1.1 提升方法的基本思路

强可学习(strongly learnable)和弱可学习(weakly learnable)

1. 在概率近似正确（probably approximately correct, PAC)学习的框架中，一个概念（类），如果存在一个多项式的学习算法能够学习它，并且正确率很高，称这个概念是强可学习的;
2. 一个概念（类），如果存在一个多项式的学习算法能够学习它，学习的正确率仅比随机猜测略好，则称这个概念是弱可学习的。
3. 在PAC学习的框架下，一个概念是强可学习的充分必要条件是这个概念是弱可学习。



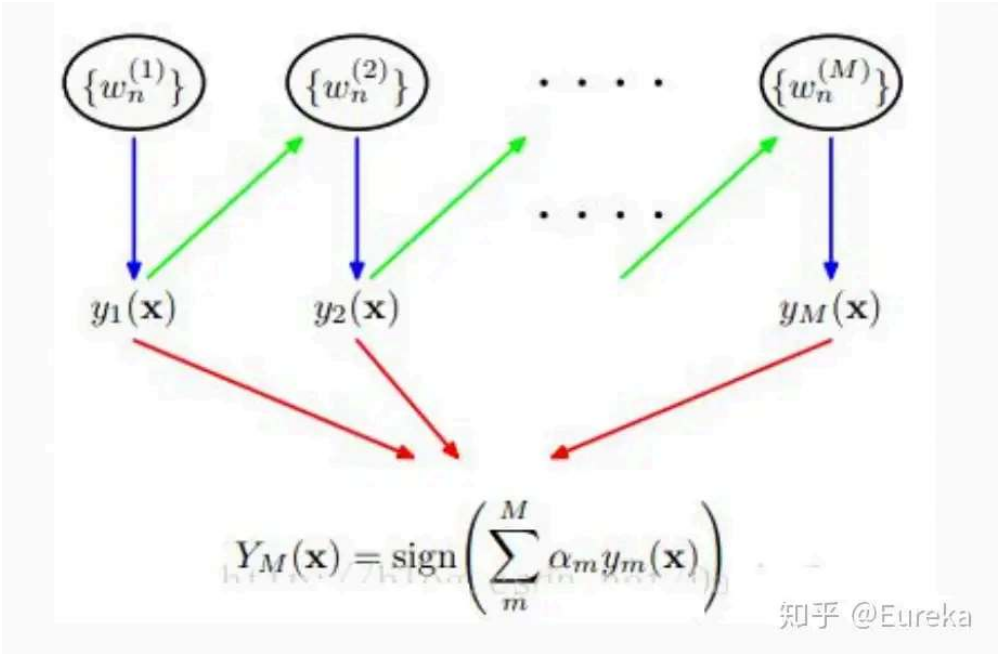
怎样获得不同的弱分类器?

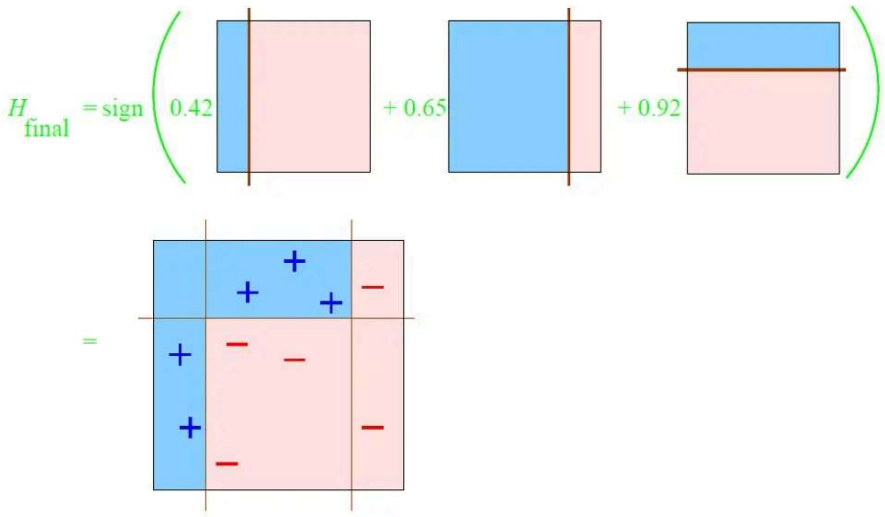
1. 使用不同的弱学习算法得到不同基本学习器
2. 使用相同的弱学习算法，但用不同的参数
3. 相同输入对象的不同表示凸显事物不同的特征
4. 使用不同的训练集：装袋（bagging）也称为自举汇聚法(boostrap aggregating)与提升（boosting）

怎样组合弱分类器?

1. 多专家组合：一种并行结构，所有的弱分类器都给出各自的预测结果，通过“组合器”把这些预测结果转换为最终结果。eg.投票（voting）及其变种、混合专家模型
2. 多级组合：一种串行结构，其中下一个分类器只在前一个分类器预测不够准（不够自信）的实例上进行训练或检测。eg.级联算法（cascading）

1.2 AdaBoost算法（分类）





AdaBoost算法:

输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中 $x_i \in \mathcal{X} \subseteq R^n, y_i \in \mathcal{Y}=\{+1,-1\}, i=1,2,\cdots,N$；弱学习算法

输出：分类器 $G(x)$

1. 初始化训练数据的权值分布

$$D_1=(w_{11},w_{12},\cdots,w_{1N}), \quad w_{1i}=\frac{1}{N}, \quad i=1,2,\cdots,N$$

2. 对 $m=1,2,\cdots,M$

2.1 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

$$G_m(x): \mathcal{X} \to \{-1,+1\}$$

2.2 计算 $G_m(x)$ 在训练数据集上的分类误差率

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

2.3 计算 $G_m(x)$ 的系数

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

2.4 更新训练数据集的权值分布

$$
\begin{aligned}
&D_{m+1}=\left(w_{m+1,1}, \cdots, w_{m+1,i}, \cdots, w_{m+1,N}\right) \\
&w_{m+1,i}=\frac{w_{mi}}{Z_m} \exp\left(-\alpha_m y_i G_m\left(x_i\right)\right), \\
&\quad=\begin{cases}\frac{w_{mi}}{Z_m} \exp\left(-\alpha_m\right), G_m\left(x_i\right)=y_i \\ \frac{w_{mi}}{Z_m} \exp\left(\alpha_m\right), G_m\left(x_i\right) \neq y_i\end{cases} \qquad i=1,2, \cdots, N
\end{aligned}
$$

其中， Z_m 是规范化因子

$$
Z_m=\sum_{i=1}^N w_{mi} \exp\left(-\alpha_m y_i, G_m\left(x_i\right)\right)
$$

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

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

注：对于Adaboost多元分类算法，其实原理和二元分类类似，最主要区别在弱分类器的系数上。比如Adaboost SAMME算法，它的弱分类器的系数：

$$
\alpha_m=\frac{1}{2} \log \frac{1-e_m}{e_m}+\log (R-1)
$$

其中R为类别数。从上式可以看出，如果是二元分类， $R=2$ ，则上式和我们的二元分类算法中的弱分类器的系数一致。

算法说明：

1.计算 $G_m(x)$ 在训练数据集上的分类误差率

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

这里 $\sum_{i=1}^N w_{mi}=1$ ，这表明 $G_m(x)$ 在加权的训练数据集上的分类错误率是被 $G_m(x)$ 误分类样本的权值之和，由此可以看出数据全值分布 D_m 与基本分类器 $G_m(x)$ 的分类误差率的关系。

2.当 $e_m \leqslant 1/2$ 时， $\alpha \geqslant 0$ ，并且 α 随着 e_m 的减少而增大，所以误分类误差率越小的基本分类器在最终分类器的作用越大。

3.更新训练数据的权值分布为下一轮做准备，式可以写成：

$$w_{m+1,i} = \begin{cases} \frac{w_{mi}}{Z_m} e^{-\alpha_m}, & G_m(x_i) = y_i \\ \frac{w_{mi}}{Z_m} e^{\alpha_m}, & G_m(x_i) \neq y_i \end{cases}$$

由此可知，被基本分类器 $G_m(x)$ 误分类样本的权值得以扩大，而被正确分类的样本的权值得以缩小。

4.这里系数 α_m 表示了基本分类器 Gm 的重要性，所以 α_m 之和并不为1。 $f(x)$ 的符号决定实例 x 类， $f(x)$ 的绝对值表示分类的确信度。

1.3 AdaBoost算法（回归）

上面是Adaboost做分类，下面来看看**Adaboost做回归**时误差率、权重系数如何选择：

先看看回归问题的误差率的问题，对于第 m 个弱学习器，计算他在训练集上的最大误差:

$$E_m = max\,|y_i - G_m(x_i)| \quad i = 1, 2, \ldots, N$$

然后计算每个样本的相对误差:

$$e_{mi} = \frac{|y_i - G_m(x_i)|}{E_m}$$

这里是误差损失为线性时的情况，如果我们用平方误差，则 $e_{mi} = \frac{(y_i - G_m(x_i))^2}{E_m^2}$,如果我们用的是指数误差，则 $e_{mi} = 1 - exp\left(\frac{-y_i + G_m(x_i)}{E_m}\right)$ 。

最终得到第 m 个弱学习器的误差率:

$$e_m = \sum_{i=1}^{N} w_{mi} e_{mi}$$

我们再来看看如何得到弱学习器权重系数 α 。这里有：

$$\alpha_m = \frac{e_m}{1 - e_m}$$

对于更新更新样本权重 D ，第 $m+1$ 个弱学习器的样本集权重系数为:

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} \alpha_m^{1 - e_{mi}}$$

这里 Z_m 是规范化因子:

$$Z_m = \sum_{i=1}^{N} w_{mi} \alpha_m^{1 - e_{mi}}$$

最后是结合策略，和分类问题稍有不同，采用的是对加权的弱学习器取中位数的方法，最终的强回归器为:

$$f(x)=\sum_{i=1}^{N}(ln\frac{1}{\alpha_m})g(x)$$

其中， $g(x)$ 是所有 $\alpha_m G_m(x), m=1,2,....M$ 的中位数。

AdaBoost算法:
输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$ ，其中 $x_i \in \mathcal{X} \subseteq R^n, y_i \in \mathcal{Y}=\{+1,-1\}, i=1,2,\cdots,N$
输出：分类器 $G(x)$
1. 初始化训练数据的权值分布

$$D_1=(w_{11},w_{12},\cdots,w_{1N}), \quad w_{1i}=\frac{1}{N}, \quad i=1,2,\cdots,N$$

2. 对 $m=1,2,\cdots,M$
2.1 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 $G_m(x)$
2.2 计算训练集上的最大误差

$$E_m=max|y_i-G_m(x_i)| \quad i=1,2,...,N$$

2.3 计算每个样本的相对误差:

如果是线性误差，则$e_{mi}=\frac{|y_i-G_k(m_i)|}{E_m}$
如果是平方误差，则$e_{mi}=\frac{(y_i-G_m(x_i))^2}{E_m^2}$
如果是指数误差，则$e_{mi}=1-exp\left(\frac{-y_i+G_k(m_i)}{E_k}\right)$

2.4 计算回归误差率:

$$e_m=\sum_{i=1}^{N}w_{mi}e_{mi}$$

2.5 计算弱学习器的系数

$$\alpha_m=\frac{e_m}{1-e_m}$$

2.6 更新样本集的权重分布为

$$w_{m+1,i}=\frac{w_{mi}}{Z_m}\alpha_m^{1-e_{mi}}$$

其中， Z_m 是规范化因子

$$Z_k = \sum_{i=1}^{N} w_{ki} \alpha_k^{1-e_{ki}}$$

3. 构建基本分类器的线性组合

$$f(x) = \sum_{m=1}^{M} (ln\frac{1}{\alpha_m}) g(x)$$

其中， $g(x)$ 是所有 $\alpha_m G_m(x), m = 1, 2, ... M$ 的中位数。

2： Adaboost算法的训练误差分析（了解）

(**Adaboost的训练误差界**) Adaboost算法最终分类器的训练误差界为：

$$\frac{1}{N} \sum_{i=1}^{N} I(G(x_i) \neq y_i) \leqslant \frac{1}{N} \sum_{i} exp(-y_i f(x_i)) = \prod_{m} Z_m$$

这里一定理说明可以在每一轮选取适当的 G_m 使得 Z_m 最小，从而使得误差下降的最快。

(**二分类问题Adaboost的训练误差界**)

$$\prod_{m=1}^{M} Z_M = \prod_{m=1}^{M} [2\sqrt{e_m(1-e_m)}] = \prod_{m=1}^{M} \sqrt{(1-4\gamma_m^2)} \leqslant exp(-2\sum_{m=1}^{M} \gamma_m^2)$$

这里， $\gamma_m = \frac{1}{2} - e_m$

推论：如果存在 $\gamma > 0$ ，对所有 m 有 $\gamma_m \geqslant \gamma$ ，则

$$\frac{1}{N} \sum_{i=1}^{N} I(G(x_i) \neq y_i) \leqslant exp(-2M\gamma^2)$$

这表明在此条件下，AdaBoost的训练误差是以指数速度下降的。

3: AdaBoost算法的解释

刚才上一节我们讲到了分类Adaboost的弱学习器权重系数公式和样本权重更新公式。但是没有解释选择这个公式的原因，让人觉得是魔法公式一样。其实它可以从Adaboost的损失函数推导出来。

从另一个角度，AdaBoost还有一种解释，即可认为AdaBoost算法是模型为加法模型、损失函数为指数函数、学习算法为前向分布算法时的二类分类学习方法。

3.1 前向分布算法

加法模型 additive model

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

其中，$b(x;\gamma_m)$ 为基函数，β_m 为基函数系数，γ_m 为基函数参数。

在给定训练数据及损失函数 $L(y,f(x))$ 的条件下，学习加法模型 $f(x)$ 成为经验风险极小化问题

$$\min_{\beta_m,\gamma_m}\sum_{i=1}^{N}L\left(y_i,\sum_{m=1}^{M}\beta_m b(x_i;\gamma_m)\right)$$

学习加法模型，从前向后每一步只学习一个基函数及其系数，即每步只优化

$$\min_{\beta,\gamma}\sum_{i=1}^{N}L(y_i,\beta b(x_i;\gamma))$$

> **前向分布算法 forward stagewise algorithm**:
> 输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，损失函数 $L(y,f(x))$；
> 基函数集 $\{b(x;\gamma)\}$
> 输出：加法模型 $f(x)$
> 1. 初始化 $f_0(x)=0$
> 2. 对 $m=1,2,\cdots,M$
> 2.1 极小化损失函数
>
> $$(\beta_m,\gamma_m)=\arg\min_{\beta,\gamma}\sum_{i=1}^{N}L(y_i,f_{m-1}(x_i)+\beta b(x_i;\gamma))$$
>
> 得到参数 β_m,γ_m
> 2.2 更新
>
> $$f_m(x)=f_{m-1}(x)+\beta_m b(x;\gamma_m)$$
>
> 3. 得到加法模型
>
> $$f(x)=f_M(x)=\sum_{m=1}^{M}\beta_m b(x;\gamma_m)$$

3.2 前向分布算法与AdaBoost

定理：AdaBoost算法是前向分布加法算法的特例。这时模型是由基本分类器组成的加法模型，损失函数是指数函数。

证明：

加法模型等价于AdaBoost的最终分类器

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

由基本分类器 $G_m(x)$ 及其系数 α_m 组成， $m=1,2,\ldots,M$ 。前向分布算法逐一学习基本函数，这一过程与AdaBoost算法逐一学习基本分类器的过程一致。下面证明前向分布算法的损失函数是指数损失函数（exponential loss function）

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

时，其学习的具体操作等价于AdaBoost算法学习的具体操作。

假设经过$m-1$轮迭代前向分布算法已经得到 $f_{m-1}(x)$ ：

$$f_{m-1}(x)=f_{m-2}(x)+\alpha_{m-1}G_{m-1}(x)=\alpha_1 G_1(x)+\cdots+\alpha_{m-1}G_{m-1}(x$$

在第 m 轮得到 $\alpha_m,G_m(x)$ 和 $f_m(x)$.

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

目标是使前向算法得到的$\alpha *{m}$和$G*{m}$使$f_{m}(x)$在训练数据集T上的指数损失最小，即：

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

可以表示成：

$$(\alpha_m,G_m(x))=arg\min_{\alpha,G}\sum_{i=1}^{N}\overline{w}_{mi}exp[-y_i\alpha G(x_i)]$$

其中， $\overline{w}_{mi}=exp[-y_i f_{m-1}(x_i)]$ ，因为 $\overline{w}_{mi}$ 不依赖于 α ，也不依赖于 G 。

首先求 $G_m^*(x)$ ，进一步展开：

$$\sum_{i=1}^{N}\overline{w}_{mi}exp[-y_i\alpha G(x_i)]=\sum_{i=1}^{N}\overline{w}_{mi}e^{-\alpha}I\{y_i=G(x_i)\}+\sum_{i=1}^{N}\overline{w}_{mi}e^{\alpha}I\{y_i\neq G(x_i)\}$$

$$=e^{-\alpha}\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i=G(x_i)\}+e^{\alpha}\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}+e^{-\alpha}$$
$$\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}-e^{-\alpha}\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}$$

$$=e^{-\alpha}\sum_{i=1}^{N}\overline{w}_{mi}+(e^{\alpha}-e^{-\alpha})\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}$$

所以最小化 $G(x)$ 由下式得到：

$$G_m^*(x)=arg\min_{G}\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}$$

之后我们求解 α_m^* ：

$$\sum_{i=1}^{N}\overline{w}_{mi}exp[-y_i\alpha G(x_i)] = \sum_{y_i=G_m(x_i)}\overline{w}_{mi}e^{-\alpha} + \sum_{y_i\neq G_m(x_i)}\overline{w}_{mi}e^{\alpha}$$

$$= e^{-\alpha}\sum_{i=1}^{N}\overline{w}_{mi} + (e^{\alpha}-e^{-\alpha})\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}$$

对 α 求导:

$$\frac{\partial}{\partial\alpha}(e^{-\alpha}\sum_{i=1}^{N}\overline{w}_{mi} + (e^{\alpha}-e^{-\alpha})\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\})$$

$$= -e^{-\alpha}\sum_{i=1}^{N}\overline{w}_{mi} + (e^{\alpha}+e^{-\alpha})\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\} = 0$$

即得:

$$\frac{e^{\alpha}+e^{-\alpha}}{e^{-\alpha}} = \frac{\sum_{i=1}^{N}\overline{w}_{mi}}{\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}}$$

$$\alpha_m^* = \frac{1}{2}log\frac{1-e_m}{e_m}$$

其中 e_m 是分类错误率:

$$e_m = \frac{\sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}}{\sum_{i=1}^{N}\overline{w}_{mi}} = \sum_{i=1}^{N}\overline{w}_{mi}I\{y_i\neq G(x_i)\}$$

这里的 α_m^* 与AdaBoost算法的 α_m 完全一致。

再看一下每一轮的权值更新，由:

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

以及 $\overline{w}_{mi} = exp[-y_i f_{m-1}(x_i)]$ ，可得:

$$\overline{w}_{m+1,i} = \overline{w}_{m,i}exp[-y_i\alpha_m G_x]$$

这与AdaBoost算法的样本权值的更新，只相差规范会因子，因此等价。

4： Adaboost算法的正则化

为了防止Adaboost过拟合，我们通常也会加入正则化项，这个正则化项我们通常称为步长 (learning rate)。定义为 ν ,对于前面的弱学习器的迭代

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

如果我们加上了正则化项，则有:

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

ν 的取值范围为 $0 \leq \nu \leq 1$ 。对于同样的训练集学习效果，较小的 ν 意味着我们需要更多的弱学习器的迭代次数。通常我们用步长和迭代最大次数一起来决定算法的拟合效果。

5： Adaboost小结

理论上任何学习器都可以用于Adaboost.但一般来说，使用最广泛的Adaboost弱学习器是决策树和神经网络。对于决策树，Adaboost分类用了CART分类树，而Adaboost回归用了CART回归树。

Adaboost的主要优点有：
1. Adaboost作为分类器时，分类精度很高
2. 在Adaboost的框架下，可以使用各种回归分类模型来构建弱学习器，非常灵活。
3. 作为简单的二元分类器时，构造简单，结果可理解。
4. 不容易发生过拟合

Adaboost的主要缺点有：对异常样本敏感，异常样本在迭代中可能会获得较高的权重，影响最终的强学习器的预测准确性。

6： 参考文献

李航. 统计学习方法[M]. 北京：清华大学出版社，2012

集成学习之Adaboost算法原理小结 - 刘建平Pinard - 博客园

编辑于 2018-09-08 17:35

机器学习 集成学习 adaboost