Adaboost 算法的原理与推导

0 引言

一直想写Adaboost来着，但迟迟未能动笔。其算法思想虽然简单：听取多人意见，最后综合决策，但一般书上对其算法的流程描述实在是过于晦涩。昨日11月1日下午，在我组织的机器学习班 第8次课上讲决策树与Adaboost，其中，Adaboost讲得酣畅淋漓，讲完后，我知道，可以写本篇博客了。

无心啰嗦，本文结合机器学习班决策树与Adaboost 的PPT，跟邹讲Adaboost指数损失函数推导的PPT（第85~第98页）、以及李航的《统计学习方法》等参考资料写就，可以定义为一篇课程笔记、读书笔记或学习心得，有何问题或意见，欢迎于本文评论下随时不吝指出，thanks。

1 Adaboost的原理

1.1 Adaboost是什么

AdaBoost，是英文"Adaptive Boosting"（自适应增强）的缩写，由Yoav Freund和Robert Schapire在1995年提出。它的自适应在于：前一个基本分类器分错的样本会得到加强，加权后的全体样本再次被用来训练下一个基本分类器。同时，在每一轮中加入一个新的弱分类器，直到达到某个预定的足够小的错误率或达到预先指定的最大迭代次数。

具体说来，整个Adaboost 迭代算法就3步：

1. 初始化训练数据的权值分布。如果有N个样本，则每一个训练样本最开始时都被赋予相同的权值：1/N。
2. 训练弱分类器。具体训练过程中，如果某个样本点已经被准确地分类，那么在构造下一个训练集中，它的权值就被降低；相反，如果某个样本点没有被准确地分类，那么它的权值就得到提高。然后，权值更新过的样本集被用于训练下一个分类器，整个训练过程如此迭代地进行下去。
3. 将各个训练得到的弱分类器组合成强分类器。各个弱分类器的训练过程结束后，加大分类误差率小的弱分类器的权重，使其在最终的分类函数中起着较大的决定作用，而降低分类误差率大的弱分类器的权重，使其在最终的分类函数中起着较小的决定作用。换言之，误差率低的弱分类器在最终分类器中占的权重较大，否则较小。

1.2 Adaboost算法流程

给定一个训练数据集T={(x1,y1), (x2,y2)...(xN,yN)}，其中实例$x \in \mathcal{X}$，而实例空间$\mathcal{X} \subset \mathbb{R}^n$，yi属于标记集合{-1,+1}，Adaboost的目的就是从训练数据中学习一系列弱分类器或基本分类器，然后将这些弱分类器组合成一个强分类器。

Adaboost的算法流程如下：

- 步骤***1.*** 首先，初始化训练数据的权值分布。每一个训练样本最开始时都被赋予相同的权值：1/N。

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

- 步骤***2.*** 进行多轮迭代，用m = 1,2, ..., M表示迭代的第多少轮

a. 使用具有权值分布Dm的训练数据集学习，得到基本分类器（选取让误差率最低的阈值来设计基本分类器）：

> $$G_m(x): \ \chi \to \{-1,+1\}$$

b. 计算Gm(x)在训练数据集上的分类误差率

> $$e_m = P(G_m(x_i) \neq y_i) = \sum_{i=1}^{N} w_{mi} I(G_m(x_i) \neq y_i)$$
>
> 由上述式子可知，*Gm(x)*在训练数据集上的 误差率*em*就是被*Gm(x)*误分类样本的权值之和。

c. 计算Gm(x)的系数，am表示Gm(x)在最终分类器中的重要程度（目的：得到基本分类器在最终分类器中所占的权重。注：这个公式写成**am=1/2ln((1-em)/em)** 更准确，因为底数是自然对数**e**，故用**In**，写成log容易让人误以为底数是2或别的底数，下同）：

> $$\alpha_m = \frac{1}{2} \log \frac{1 - e_m}{e_m}$$
>
> 由上述式子可知，*em <= 1/2*时，*am >= 0*，且*am*随着*em*的减小而增大，意味着分类误差率越小的基本分类器在最终分类器中的作用越大。

d. 更新训练数据集的权值分布（目的：得到样本的新的权值分布），用于下一轮迭代

$$D_{m+1}=\left(w_{m+1,1},w_{m+1,2}\cdots w_{m+1,i}\cdots,w_{m+1,N}\right),$$
$$w_{m+1,i}=\frac{w_{mi}}{Z_m}\exp\left(-\alpha_m y_i G_m\left(x_i\right)\right),\ i=1,2,\cdots,N$$

> 使得被基本分类器*Gm(x)*误分类样本的权值增大，而被正确分类样本的权值减小。就这样，通过这样的方式，*AdaBoost*方法能“重点关注”或“聚焦于”那些较难分的样本上。

其中，Zm是规范化因子，使得Dm+1成为一个概率分布：

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

- 步骤**3.** 组合各个弱分类器

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

从而得到最终分类器，如下：

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

1.3 Adaboost的一个例子

下面，给定下列训练样本，请用AdaBoost算法学习一个强分类器。

序号	1	2	3	4	5	6	7	8	9	X
X	0	1	2	3	4	5	6	7	8	9
Y	1	1	1	-1	-1	-1	1	1	1	-1

求解过程：初始化训练数据的权值分布，令每个权值W1i = 1/N = 0.1，其中，N = 10，i = 1,2, ..., 10，然后分别对于m = 1,2,3, ...等值进行迭代。

拿到这10个数据的训练样本后，根据 X 和 Y 的对应关系，要把这10个数据分为两类，一类是“1”，一类是“-1”，根据数据的特点发现：“0 1 2”这3个数据对应的类是“1”，“3 4 5”这3个数据对应的类是“-1”，“6 7 8”这3个数据对应的类是“1”，9是比较孤独的，对应类“-1”。抛开孤独的9不讲，“0 1 2”、“3 4 5”、“6 7 8”这是3类不同的数据，分别对应的类是

1、-1、1，直观上推测可知，可以找到对应的数据分界点，比如2.5、5.5、8.5 将那几类数据分成两类。当然，这只是主观臆测，下面实际计算下这个具体过程。

迭代过程**1**

对于m=1，在权值分布为**D1**（10个数据，每个数据的权值皆初始化为0.1）的训练数据上，经过计算可得：

1.

a. 阈值**v**取**2.5**时误差率为0.3（x < 2.5时取1，x > 2.5时取-1，则**6 7 8**分错，误差率为0.3），
b. 阈值v取5.5时误差率最低为0.4（x < 5.5时取1，x > 5.5时取-1，则3 4 5 6 7 8皆分错，误差率0.6大于0.5，不可取。故令x > 5.5时取1，x < 5.5时取-1，则0 1 2 9分错，误差率为0.4），
c. 阈值v取8.5时误差率为0.3（x < 8.5时取1，x > 8.5时取-1，则3 4 5分错，误差率为0.3）。

可以看到，无论阈值v取2.5，还是8.5，总得分错3个样本，故可任取其中任意一个如2.5，弄成第一个基本分类器为：

$$G_1(x)=\begin{cases}1, & x<2.5\\ -1, & x>2.5\end{cases}$$

上面说阈值v取2.5时则6 7 8分错，所以误差率为0.3，更加详细的解释是：因为样本集中

1.

a. 0 1 2对应的类（Y）是1，因它们本身都小于2.5，所以被G1(x)分在了相应的类“1”中，分对了。
b. 3 4 5本身对应的类（Y）是-1，因它们本身都大于2.5，所以被G1(x)分在了相应的类“-1”中，分对了。
c. 但6 7 8本身对应类（Y）是1，却因它们本身大于2.5而被G1(x)分在了类"-1"中，所以这3个样本被分错了。
d. 9本身对应的类（Y）是-1，因它本身大于2.5，所以被G1(x)分在了相应的类“-1”中，分对了。

从而得到G1(x)在训练数据集上的误差率（被G1(x)误分类样本“6 7 8”的权值之和）**e1=P(G1(xi)≠yi) = 3*0.1 = 0.3**。

然后根据误差率e1计算G1的系数：

$$\alpha_1 = \frac{1}{2}\log\frac{1-e_1}{e_1} = 0.4236$$

这个a1代表G1(x)在最终的分类函数中所占的权重，为0.4236。
接着更新训练数据的权值分布，用于下一轮迭代：

$$D_{m+1} = (w_{m+1,1}, w_{m+1,2} \cdots w_{m+1,i} \cdots, w_{m+1,N})$$

$$w_{m+i} = \frac{w_{mi}}{Z_m}\exp(-\alpha_m y_i G_m(x_i)), \quad i = 1,2,\cdots,N$$

值得一提的是，由权值更新的公式可知，每个样本的新权值是变大还是变小，取决于它是被分错还是被分正确。

即如果某个样本被分错了，则yi * Gm(xi)为负，负负得正，结果使得整个式子变大（样本权值变大），否则变小。

第一轮迭代后，最后得到各个数据新的权值分布**D2** = (0.0715, 0.0715, 0.0715, 0.0715, 0.0715, 0.0715, 0.1666, 0.1666, 0.1666, 0.0715)。由此可以看出，因为样本中是数据“6 7 8”被G1(x)分错了，所以它们的权值由之前的0.1增大到0.1666，反之，其它数据皆被分正确，所以它们的权值皆由之前的0.1减小到0.0715。

分类函数f1(x)= a1*G1(x) = 0.4236G1(x)。

此时，得到的第一个基本分类器sign(f1(x))在训练数据集上有3个误分类点（即6 7 8）。

从上述第一轮的整个迭代过程可以看出：被误分类样本的权值之和影响误差率，误差率影响基本分类器在最终分类器中所占的权重。

迭代过程**2**

对于m=2，在权值分布为**D2** = (0.0715, 0.0715, 0.0715, 0.0715, 0.0715, 0.0715, 0.1666, 0.1666, 0.1666, 0.0715)的训练数据上，经过计算可得：

1.
   a. 阈值v取2.5时误差率为0.1666*3（x < 2.5时取1，x > 2.5时取-1，则6 7 8分错，误差率为0.1666*3），
   b. 阈值v取5.5时误差率最低为0.0715*4（x > 5.5时取1，x < 5.5时取-1，则0 1 2 9分错，误差率为0.0715*3 + 0.0715），

c. 阈值**v**取**8.5**时误差率为0.0715*3（x < 8.5时取1，x > 8.5时取-1，则**3 4 5**分错，误差率为0.0715*3）。

所以，阈值v取8.5时误差率最低，故第二个基本分类器为：

$$G_2(x)=\begin{cases}1, & x<8.5\\ -1, & x>8.5\end{cases}$$

面对的还是下述样本：

序号	1	2	3	4	5	6	7	8	9	X
X	0	1	2	3	4	5	6	7	8	9
Y	1	1	1	−1	−1	−1	1	1	1	−1

很明显，G2(x)把样本“3 4 5”分错了，根据D2可知它们的权值为0.0715, 0.0715, 0.0715，所以G2(x)在训练数据集上的误差率e2=P(G2(xi)≠yi) = 0.0715 * 3 = 0.2143。

计算G2的系数：

$$\alpha_2=\frac{1}{2}\log\frac{1-e_2}{e_2}=0.6496$$

更新训练数据的权值分布：

$$D_{m+1}=\left(w_{m+1,1},w_{m+1,2}\cdots w_{m+1,i}\cdots,w_{m+1,N}\right)$$

$$w_{m+i}=\frac{w_{mi}}{Z_m}\exp\left(-\alpha_m y_i G_m(x_i)\right),\quad i=1,2,\cdots,N$$

D3 = (0.0455, 0.0455, 0.0455, 0.1667, 0.1667, 0.01667, 0.1060, 0.1060, 0.1060, 0.0455)。被分错的样本“3 4 5”的权值变大，其它被分对的样本的权值变小。

f2(x)=0.4236G1(x) + 0.6496G2(x)

此时，得到的第二个基本分类器sign(f2(x))在训练数据集上有3个误分类点（即3 4 5）。

迭代过程**3**

对于m=3，在权值分布为**D3** = (0.0455, 0.0455, 0.0455, 0.1667, 0.1667, 0.01667, 0.1060, 0.1060, 0.1060, 0.0455)的训练数据上，经过计算可得：

1.

a. 阈值v取2.5时误差率为0.1060*3（x < 2.5时取1，x > 2.5时取-1，则6 7 8分错，误差率为0.1060*3），
b. 阈值**v**取**5.5**时误差率最低为0.0455*4（x > 5.5时取1，x < 5.5时取-1，则**0 1 2 9**分错，误差率为0.0455*3 + 0.0715），
c. 阈值v取8.5时误差率为0.1667*3（x < 8.5时取1，x > 8.5时取-1，则3 4 5分错，误差率为0.1667*3）。

所以阈值v取5.5时误差率最低，故第三个基本分类器为：

$$G_3(x) = \begin{cases} 1, & x > 5.5 \\ -1, & x < 5.5 \end{cases}$$

依然还是原样本：

序号	1	2	3	4	5	6	7	8	9	X
X	0	1	2	3	4	5	6	7	8	9
Y	1	1	1	-1	-1	-1	1	1	1	-1

此时，被误分类的样本是：0 1 2 9，这4个样本所对应的权值皆为0.0455，

所以G3(x)在训练数据集上的误差率**e3** = P(G3(xi)≠yi) = **0.0455*4** = 0.1820。

计算G3的系数：

$$\alpha_3 = \frac{1}{2}\log\frac{1-e_3}{e_3} = 0.7514$$

更新训练数据的权值分布：

$$D_{m+1} = (w_{m+1,1}, w_{m+1,2} \cdots w_{m+1,i} \cdots, w_{m+1,N})$$

$$w_{m+i} = \frac{w_{mi}}{Z_m}\exp(-\alpha_m y_i G_m(x_i)), \quad i = 1,2,\cdots,N$$

D4 = (0.125, 0.125, 0.125, 0.102, 0.102, 0.102, 0.065, 0.065, 0.065, 0.125)。被分错的样本“0 1 2 9”的权值变大，其它被分对的样本的权值变小。

f3(x)=0.4236G1(x) + 0.6496G2(x)+0.7514G3(x)

此时，得到的第三个基本分类器sign(f3(x))在训练数据集上有0个误分类点。至此，整个训练过程结束。

现在，咱们来总结下3轮迭代下来，各个样本权值和误差率的变化，如下所示（其中，样本权值D中加了下划线的表示在上一轮中被分错的样本的新权值）：

1. 训练之前，各个样本的权值被初始化为D1 = (0.1, 0.1,0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1)；
2. 第一轮迭代中，样本**“6 7 8”**被分错，对应的误差率为**e1**=P(G1(xi)≠yi) = 3*0.1 = 0.3，此第一个基本分类器在最终的分类器中所占的权重为**a1** = 0.4236。第一轮迭代过后，样本新的权值为**D2** = (0.0715, 0.0715, 0.0715, 0.0715, 0.0715, 0.0715, 0.0715, 0.1666, 0.1666, 0.1666, 0.0715)；
3. 第二轮迭代中，样本**“3 4 5”**被分错，对应的误差率为**e2**=P(G2(xi)≠yi) = 0.0715 * 3 = 0.2143，此第二个基本分类器在最终的分类器中所占的权重为**a2** = 0.6496。第二轮迭代过后，样本新的权值为**D3** = (0.0455, 0.0455, 0.0455, 0.1667, 0.1667, 0.01667, 0.1060, 0.1060, 0.1060, 0.0455)；
4. 第三轮迭代中，样本**“0 1 2 9”**被分错，对应的误差率为**e3** = P(G3(xi)≠yi) = 0.0455*4 = 0.1820，此第三个基本分类器在最终的分类器中所占的权重为**a3** = 0.7514。第三轮迭代过后，样本新的权值为**D4** = (0.125, 0.125, 0.125, 0.102, 0.102, 0.102, 0.065, 0.065, 0.065, 0.125)。

从上述过程中可以发现，如果某些个样本被分错，它们在下一轮迭代中的权值将被增大，同时，其它被分对的样本在下一轮迭代中的权值将被减小。就这样，分错样本权值增大，分对样本权值变小，而在下一轮迭代中，总是选取让误差率最低的阈值来设计基本分类器，所以误差率e（所有被Gm(x)误分类样本的权值之和）不断降低。

综上，将上面计算得到的a1、a2、a3各值代入G(x)中，G(x) = sign[f3(x)] = sign[a1 * G1(x) + a2 * G2(x) + a3 * G3(x)]，得到最终的分类器为：

G(x) = sign[f3(x)] = sign[0.4236G1(x) + 0.6496G2(x)+0.7514G3(x)]。

2 Adaboost的误差界

通过上面的例子可知，Adaboost在学习的过程中不断减少训练误差e，直到各个弱分类器组合成最终分类器，那这个最终分类器的误差界到底是多少呢？

事实上，Adaboost 最终分类器的训练误差的上界为：

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

下面，咱们来通过推导来证明下上述式子。

当G(xi)≠yi时，yi*f(xi)<0，因而exp(-yi*f(xi))≥1，因此前半部分得证。

关于后半部分，别忘了：

$$w_{m+1,i} = \frac{w_{mi}}{Z_m}\exp\left(-\alpha_m y_i G_m(x_i)\right)$$

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

整个的推导过程如下：

$$\begin{aligned}
&\frac{1}{N}\sum_i \exp\left(-y_i f(x_i)\right) \\
&= \frac{1}{N}\sum_i \exp\left(-\sum_{m=1}^{M}\alpha_m y_i G_m(x_i)\right) \\
&= w_{1i}\sum_i \exp\left(-\sum_{m=1}^{M}\alpha_m y_i G_m(x_i)\right) \\
&= w_{1i}\prod_{m=1}^{M}\exp\left(-\alpha_m y_i G_m(x_i)\right) \\
&= Z_1\sum_i w_{2i}\prod_{m=2}^{M}\exp\left(-\alpha_m y_i G_m(x_i)\right) \\
&= Z_1 Z_2\sum_i w_{3i}\prod_{m=3}^{M}\exp\left(-\alpha_m y_i G_m(x_i)\right) \\
&= Z_1 Z_2 \cdots Z_{M-1}\sum_i w_{Mi}\exp\left(-\alpha_M y_i G_M(x_i)\right) \\
&= \prod_{m=1}^{M} Z_m
\end{aligned}$$

这个结果说明，可以在每一轮选取适当的Gm使得Zm最小，从而使训练误差下降最快。接着，咱们来继续求上述结果的上界。

对于二分类而言，有如下结果：

$$\prod_{m=1}^{M} Z_m = \prod_{m=1}^{M}\left(2\sqrt{e_m(1-e_m)}\right) = \prod_{m=1}^{M}\sqrt{\left(1-4\gamma_m^2\right)} \le \exp\left(-2\sum_{m=1}^{M}\gamma_m^2\right)$$

其中，

$$\gamma_m = \frac{1}{2} - e_m$$

。

继续证明下这个结论。

由之前Zm的定义式跟本节最开始得到的结论可知：

$$
\begin{aligned}
Z_m &= \sum_{i=1}^{N} w_{mi} \exp\left(-\alpha_m y_i G_m(x_i)\right) \\
&= \sum_{y_i = G_m(x_i)} w_{mi} e^{-\alpha_m} + \sum_{y_i \neq G_m(x_i)} w_{mi} e^{\alpha_m} \\
&= (1 - e_m) e^{-\alpha_m} + e_m e^{\alpha_m} \\
&= 2\sqrt{e_m(1 - e_m)} \\
&= \sqrt{1 - 4\gamma_m^2}
\end{aligned}
$$

而这个不等式

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

可先由e^x和1-x的开根号，在点x的泰勒展开式推出。

值得一提的是，如果取γ1, γ2... 的最小值，记做γ（显然，γ≥γi>0，i=1,2,...m），则对于所有m，有：

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

这个结论表明，AdaBoost的训练误差是以指数速率下降的。另外，AdaBoost算法不需要事先知道下界γ，AdaBoost具有自适应性，它能适应弱分类器各自的训练误差率 。

最后，Adaboost 还有另外一种理解，即可以认为其模型是加法模型、损失函数为指数函数、学习算法为前向分步算法的二类分类学习方法，下个月即12月份会再推导下，然后更新此文。而在此之前，有兴趣的可以参看《统计学习方法》第8.3节或其它相关资料。

3 Adaboost 指数损失函数推导

事实上，在上文1.2节Adaboost的算法流程的步骤3中，我们构造的各个基本分类器的线性组合

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

是一个加法模型，而Adaboost算法其实是前向分步算法的特例。那么问题来了，什么是加法模型，什么又是前向分步算法呢？

3.1 加法模型和前向分步算法

如下图所示的便是一个加法模型

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

其中，$b(x;\gamma_m)$称为基函数，γ_m称为基函数的参数，β_m称为基函数的系数。

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，即损失函数极小化问题：

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

随后，该问题可以作如此简化：从前向后，每一步只学习一个基函数及其系数，逐步逼近上式，即：每步只优化如下损失函数：

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

这个优化方法便就是所谓的前向分步算法。

下面，咱们来具体看下前向分步算法的算法流程：

- 输入：训练数据集$T=\{(x_1,y_1),(x_2,y_2),...,(x_N,y_N)\}$
- 损失函数：$L(y,f(x))$
- 基函数集：$\{b(x;\gamma)\}$
- 输出：加法模型$f(x)$
- 算法步骤：
    - 1. 初始化$f_0(x)=0$
    - 2. 对于m=1,2,..M
- a)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和γ_m。

- b)更新

$$f_m(x) = f_{m-1}(x) + \beta_m b(x;\gamma_m)$$

-

- 3. 最终得到加法模型

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

就这样，前向分步算法将同时求解从m=1到M的所有参数（β_m、γ_m）的优化问题简化为逐次求解各个β_m、γ_m（1≤m≤M）的优化问题。

3.2 前向分步算法与Adaboost的关系

在上文第2节最后，我们说Adaboost 还有另外一种理解，即可以认为其模型是加法模型、损失函数为指数函数、学习算法为前向分步算法的二类分类学习方法。其实，Adaboost算法就是前向分步算法的一个特例，Adaboost 中，各个基本分类器就相当于加法模型中的基函数，且其损失函数为指数函数。

换句话说，当前向分步算法中的基函数为Adaboost中的基本分类器时，加法模型等价于Adaboost的最终分类器

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

你甚至可以说，这个最终分类器其实就是一个加法模型。只是这个加法模型由基本分类器$G_m(x)$及其系数α_m组成，m = 1, 2, ..., M。前向分步算法逐一学习基函数的过程，与Adaboost算法逐一学习各个基本分类器的过程一致。

下面，咱们便来证明：当前向分步算法的损失函数是指数损失函数

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

时，其学习的具体操作等价于**Adaboost**算法的学习过程。

假设经过m-1轮迭代，前向分步算法已经得到

$$f_{m-1}(x)$$

：

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

而后在第m轮迭代得到α_m、$G_m(x)$和$f_m(x)$。其中，$f_m(x)$为：

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

而α_m和$G_m(x)$未知。所以，现在咱们的目标便是根据前向分步算法训练α_m和$G_m(x)$，使得最终$f_m(x)$在训练数据集T上的指数损失最小，即

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

针对这种需要求解多个参数的情况，可以先固定其它参数，求解其中一两个参数，然后逐一求解剩下的参数。例如我们可以固定$G_1(x),\ldots,G_{m-1}(x)$和$\alpha_1,\ldots,\alpha_{m-1}$，只针对$G_m(x)$和$\alpha_m$做优化。

换言之，在面对$G_1(x),\ldots,G_{m-1}(x),G_m(x)$和$\alpha_1,\ldots,\alpha_{m-1},\alpha_m$ 这2m个参数都未知的情况下，可以：

1. 先假定$G_1(x),\ldots,G_{m-1}(x)$和$\alpha_1,\ldots,\alpha_{m-1}$已知，求解出$G_m(x)$和$\alpha_m$；
2. 然后再逐一求解其它未知参数。

且考虑到上式中的$exp(-y_i f_{m-1}(x_i))$既不依赖α也不依赖G，所以是个与最小化无关的固定值，记为$\overline{w}_{mi}$，即$\overline{w}_{mi}=\exp(-y_i f_{m-1}(x_i))$，则上式可以表示为（后面要多次用到这个式子，简记为$(\alpha_m,G_m(x))$）：

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

值得一提的是，$\overline{w}_{mi}$虽然与最小化无关，但$\overline{w}_{mi}$依赖于$f_{m-1}(x)$，随着每一轮迭代而发生变化。

接下来，便是要证使得上式达到最小的α_m^*和$G_m^*(x)$就是**Adaboost**算法所求解得到的α_m和$G_m(x)$。

为求解上式，咱们先求$G_m^*(x)$再求α_m^*。

首先求$G_m^*(x)$。对于任意$\alpha>0$，使上式$(\alpha_m,G_m(x))$最小的G(x)由下式得到：

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

别忘了，

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

。

跟1.2节所述的误差率的计算公式对比下：

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

可知，上面得到的$G_m^*(x)$便是Adaboost算法的基本分类器$G_m(x)$，因为它是在第m轮加权训练数据时，使分类误差率最小的基本分类器。换言之，这个$G_m^*(x)$便是Adaboost算法所要求的$G_m(x)$，别忘了，在Adaboost算法的每一轮迭代中，都是选取让误差率最低的阈值来设计基本分类器。

然后求α_m^*。还是回到之前的这个式子$(\alpha_m, G_m(x))$上：

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

这个式子的后半部分可以进一步化简，得：

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

接着将上面求得的

$$G_m^*(x)$$

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

代入上式中，且对α求导，令其求导结果为0，即得到使得$(\alpha_m, G_m(x))$一式最小的α，即为：

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

这里的α_m^*跟上文1.2节中α_m的计算公式完全一致。

此外，毫无疑问，上式中的

$$e_m$$

便是误差率：

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

即

$$e_m$$

就是被Gm(x)误分类样本的权值之和。

就这样，结合模型$f_m(x) = f_{m-1}(x) + \alpha_m G_m(x)$，跟$\overline{w}_{mi} = \exp(-y_i f_{m-1}(x_i))$，可以推出

$$\overline{w}_{m+1,i} = exp[-y_i f_m(x_i)] = exp[-y_i(f_{m-1}(x_i)+\alpha_m G_m(x))] = exp[-y_i f_{m-1}(x_i)]exp[-y_i \alpha_m G_m(x)]$$

从而有：

$$\overline{w}_{m+1,i} = \overline{w}_{m,i} \exp(-y_i \alpha_m G_m(x))$$

与上文1.2节介绍的权值更新公式

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} \exp(-\alpha_m y_i G_m(x_i)), \quad i = 1,2,\cdots,N$$

相比，只相差一个规范化因子，即后者多了一个

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

所以，整个过程下来，我们可以看到，前向分步算法逐一学习基函数的过程，确实是与Adaboost算法逐一学习各个基本分类器的过程一致，两者完全等价。

综上，本节不但提供了Adaboost的另一种理解：加法模型，损失函数为指数函数，学习算法为前向分步算法，而且也解释了最开始1.2节中基本分类器$G_m(x)$及其系数α_m的由来，以及对权值更新公式的解释，你甚至可以认为本节就是对上文整个1.2节的解释。