AdaBoost算法原理详解

主要参考李宏毅老师机器学习课程中的讲解：

ensemble

以及周志华老师《机器学习》书中对应章节。

AdaBoost是Boosting族算法中的著名代表，其原理有多种理解方式，这里我们按照李宏毅老师课程中的思路来进行分析。

1. 算法实现

AdaBoost希望在训练过程中，后一个基学习器能与前一个基学习器形成互补，弥补之前做的不好的地方。AdaBoost通过对数据集进行重新加权的方式来达成这一目的，使得之前分类错误的样本在训练新的基学习器时受到更多的关注。在实际操作时，我们只要对loss函数进行重新调整即可。

下面我们来推导权重的计算方式。

标准的AdaBoost处理的是二分类问题，我们假设训练数据为$\left\{\left(x^1, \hat{y}^1\right), \cdots,\left(x^n, \hat{y}^n\right), \cdots,\left(x^N, \hat{y}^N\right)\right\}$，标签$\hat{y}=\pm 1$，不同的基分类器用$\left\{f_1, f_2, \ldots, f_T\right\}$表示，$u_i^j$表示在训练第$i$个基分类器时第$j$个样本对应的权重。对于第一个基分类器$f_1$，其误差率$\varepsilon_1$为：

$$\varepsilon_1=\frac{\sum_n u_1^n \delta\left(f_1\left(x^n\right) \neq \hat{y}^n\right)}{Z_1}$$

其中$Z_1=\sum_n u_1^n$是归一化系数。很显然$\varepsilon_1<0.5$，因为不论是多弱的分类器，我们总能将误差率控制在0.5以下（大于0.5时只要将输出取反即可）。

将样本的权重值从u_1^n变为u_2^n，使得分类器f_1失效，即

$$\frac{\sum_n u_2^n \delta\left(f_1\left(x^n\right) \neq \hat{y}^n\right)}{Z_2}=0.5 \tag{1.1}$$

这意外着在新的权重u_2^n下，f_1只是在做随机的预测。接下来我们在新的权重u_2^n下训练新的分类器f_2，这样得到的f_2即可与f_1形成互补，可以理解为f_2做到了f_1做不到的事情（尽管f_2不一定能做到f_1也能做到的事情）。

改变权重的原则是：

- 如果样本x^n被f_1错误的分类，即$f_1\left(x^n\right) \neq \hat{y}^n$，那么将其对应的权重$u_1^n$乘以一个数$d_1$，得到新的权重$u_2^n$，这意味着被分错的样本将受到更多关注；
- 如果样本x^n被f_1正确的分类，即$f_1\left(x^n\right)=\hat{y}^n$，那么将其对应的权重$u_1^n$除以$d_1$，得到新的权重$u_2^n$，这意味着被分对的样本将受到关注减少。

注意到，式（1.1）的分子可以改写为：

$$\sum_n u_2^n \delta\left(f_1\left(x^n\right) \neq \hat{y}^n\right)=\sum_{f_1\left(x^n\right) \neq \hat{y}^n} u_1^n d_1 \tag{1.2}$$

式（1.1）的分母可以改写为：

$$\begin{aligned} Z_2 &= \sum_n u_2^n \\ &= \sum_{f_1(x^n)\neq\hat{y}^n} u_2^n + \sum_{f_1(x^n)=\hat{y}^n} u_2^n \\ &= \sum_{f_1(x^n)\neq y^n} u_1^n d_1 + \sum_{f_1(x^n)=y^n} u_1^n / d_1 \end{aligned} \tag{1.3}$$

将式（1.2）和式（1.3）代回式（1.1），并颠倒分子分母的顺序，得

$$\frac{\sum_{f_1(x^n)\neq\hat{y}^n} u_1^n d_1 + \sum_{f_1(x^n)=\hat{y}^n} u_1^n / d_1}{\sum_{f_1(x^n)\neq\hat{y}^n} u_1^n d_1} = 2$$

整理得

$$\frac{\sum_{f_1(x^n)=\hat{y}^n} u_1^n / d_1}{\sum_{f_1(x^n)\neq\hat{y}^n} u_1^n d_1} = 1$$

即

$$\sum_{f_1(x^n)=\hat{y}^n} u_1^n / d_1 = \sum_{f_1(x^n)\neq\hat{y}^n} u_1^n d_1 \tag{1.4}$$

同时，根据

$$\varepsilon_1 = \frac{\sum_{f_1(x^n)\neq\hat{y}^n} u_1^n}{Z_1}$$

可得

$$\sum_{f_1(x^n)\neq\hat{y}^n} u_1^n = Z_1 \varepsilon_1 \tag{1.5}$$

以及

$$\sum_{f_1(x^n)=\hat{y}^n} u_1^n = Z_1 \left(1-\varepsilon_1\right) \tag{1.6}$$

将式（1.5）和式（1.6）代回式（1.4），得

$$Z_1 \left(1-\varepsilon_1\right) / d_1 = Z_1 \varepsilon_1 d_1$$

整理后即可得到系数d_1的计算方式：

$$d_1 = \sqrt{\left(1-\varepsilon_1\right) / \varepsilon_1}$$

显然$d_1 > 1$。

于是我们可以得到：

$$d_t = \sqrt{(1-\varepsilon_t)/\varepsilon_t}$$

规定

$$\alpha_t = \ln\sqrt{(1-\varepsilon_t)/\varepsilon_t} \tag{1.7}$$

对于分类错误的样本，其权重更新为：

$$u_{t+1}^n = u_t^n \times d_t = u_t^n \times \exp(\alpha_t)$$

对于分类正确的样本，其权重更新为：

$$u_{t+1}^n = u_t^n / d_t = u_t^n \times exp(-\alpha_t)$$

可以整合一下，权重的更新规则为：

$$u_{t+1}^n = u_t^n \times \exp(-\hat{y}^n f_t(x^n)\alpha_t) \tag{1.8}$$

按照这种方式，我们可以得到一系列分类器$f_1(x), \ldots, f_t(x), \ldots, f_T(x)$，通过加权和的方式，可以得到最终结果：

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(x)\right) \tag{1.9}$$

式（1.7）告诉我们，错误率ε_t，α_t的值越大。所以式（1.9）的含义即是在最后考虑所有分类器时，错误率小的分类器会被给予更高的权重，这显然是合理的。

综上所述，整个AdaBoost的算法表述如下：

给定：训练数据$\left\{(x^1, \hat{y}^1, u_1^1), \cdots, (x^n, \hat{y}^n, u_1^n), \cdots, (x^N, \hat{y}^N, u_1^N)\right\}$，标签$\hat{y} = \pm 1$，初始权重$u_1^n = 1$（所有样本等权重）。

训练：从$t = 1, \ldots, T$循环：

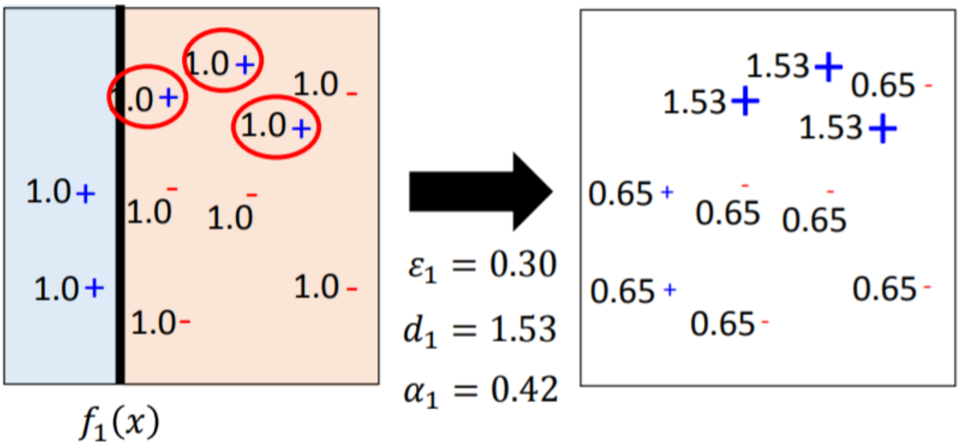
- 训练弱分类器$f_t(x)$，对应权重$\left\{u_t^1, \cdots, u_t^N\right\}$，对应错误率$\varepsilon_t$；
- 对循环$n = 1, \ldots, N$：
  - 如果x^n被$f_t(x)$错误分类，即$\hat{y}^n \neq f_t(x^n)$，则更新权重为
    $$u_{t+1}^n = u_t^n \times d_t = u_t^n \times \exp(\alpha_t)$$
    其中$d_t = \sqrt{(1-\varepsilon_t)/\varepsilon_t}$，$\alpha_t = \ln\sqrt{(1-\varepsilon_t)/\varepsilon_t}$；
  - 否则，更新权重为
    $$u_{t+1}^n = u_t^n / d_t = u_t^n \times exp(-\alpha_t)$$
- 得到分类器$f_1(x), \ldots, f_t(x), \ldots, f_T(x)$。

结果整合：最终预测结果为所有分类器的加权和：$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(x)\right)$。

2. 简单例子

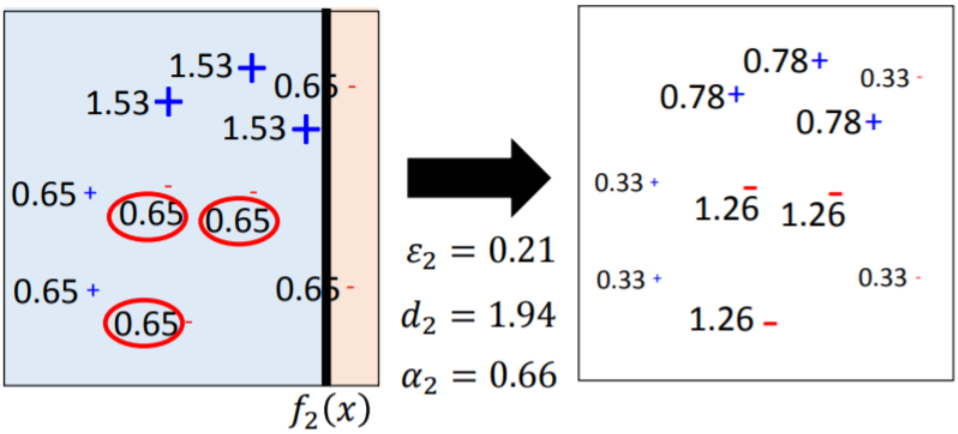
以训练三个决策树桩作为弱分类器，即$T = 3$为例，来说明AdaBoost的运作方式。

- $t=1$:



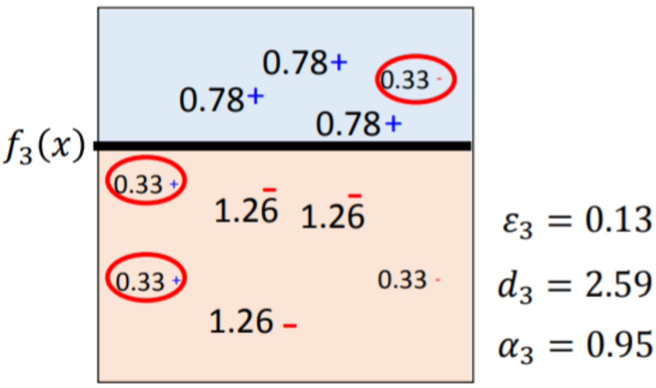
可以看到有三个样本被分类错误，对数据重新加权。

- $t=2$:



此时仍有三个样本被分类错误，再次进行重新加权。

- $t=3$:

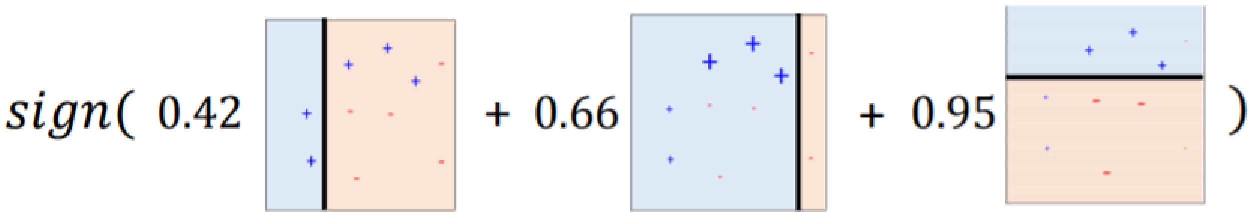


此时仍有三个样本被分类错误，但基分类器数量已达到要求，所以迭代终止。

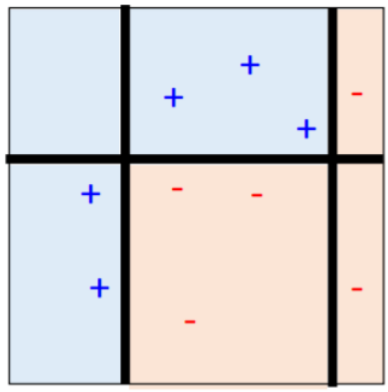
最终的分类器是三个基分类器的加权和：

$$H(x) = \mathrm{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(x)\right)$$

即如下分类器：



整合以后，最终的效果是这样的：



这就是AdaBoost分类器的工作流程。

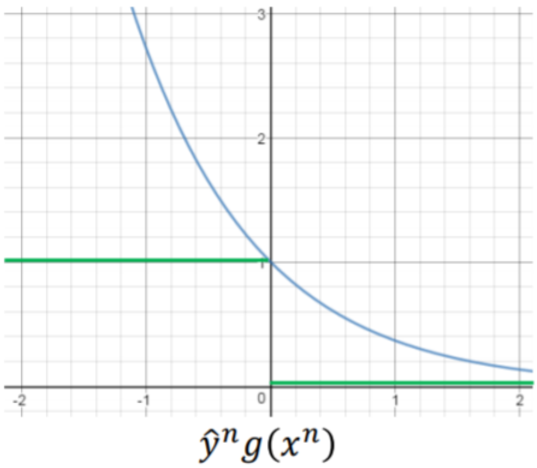
3. 误差分析

AdaBoost的强大之处在于，随着基分类器数量的增加（即T增加），最终整合的分类器$H(x)$的训练误差会越来越小，直到变为0。下面从数学上来证明这一点。

最终的总分类器$H(x)$的误差率ϵ为：

$$
\begin{aligned}
\epsilon &= \frac{1}{N}\sum_{n}\delta\left(H\left(x^{n}\right) \neq \hat{y}^{n}\right) \\
&= \frac{1}{N}\sum_{n}\delta\left(\hat{y}^{n} g\left(x^{n}\right) < 0\right) \\
&\leq \frac{1}{N}\sum_{n}\exp\left(-\hat{y}^{n} g\left(x^{n}\right)\right)
\end{aligned}
$$

最后一步的不等式用到了误差函数的上界，如下：



下面我们来证明：

$$\frac{1}{N}\sum_{n}\exp(-\hat{y}^n g(x^n))) = \frac{1}{N}Z_{T+1} \tag{3.1}$$

我们用Z_t来表示当训练f_t时所有权重的和，那么Z_{T+1}即为：

$$Z_{T+1} = \sum_n u_{T+1}^n$$

根据

$$\begin{aligned} u_1^n &= 1 \\ u_{t+1}^n &= u_t^n \times \exp(-\hat{y}^n f_t(x^n)\alpha_t) \end{aligned}$$

可以得到

$$u_{T+1}^n = \prod_{t=1}^{T}\exp(-\hat{y}^n f_t(x^n)\alpha_t)$$

则

$$\begin{aligned} Z_{T+1} &= \sum_n \prod_{t=1}^{T}\exp(-\hat{y}^n f_t(x^n)\alpha_t) \\ &= \sum_n \exp\left(-\hat{y}^n \sum_{t=1}^{T} f_t(x^n)\alpha_t\right) \\ &= \sum_n \exp(-\hat{y}^n g(x^n)) \end{aligned}$$

其中$g(x^n) = \sum_{t=1}^{T} f_t(x^n)\alpha_t$。

所以式（3.1）得证。

接下来只要证明Z_{T+1}会越来越小，即可证明误差会越来越小。

我们有：

$$
\begin{aligned}
Z_1 &= N \\
Z_t &= Z_{t-1}\varepsilon_t \exp(\alpha_t) + Z_{t-1}\left(1-\varepsilon_t\right)\exp(-\alpha_t)
\end{aligned}
$$

注意到$Z_{t-1}\varepsilon_t$代表Z_{t-1}中被分错的部分，$Z_{t-1}\left(1-\varepsilon_t\right)$代表$Z_{t-1}$中被分对的部分。进一步地，有

$$
\begin{aligned}
Z_t &= Z_{t-1}\varepsilon_t\sqrt{\left(1-\varepsilon_t\right)/\varepsilon_t} + Z_{t-1}\left(1-\varepsilon_t\right)\sqrt{\varepsilon_t/\left(1-\varepsilon_t\right)} \\
&= Z_{t-1} \times 2\sqrt{\varepsilon_t\left(1-\varepsilon_t\right)}
\end{aligned}
$$

由于$\varepsilon_t < 0.5$，因此总有$2\sqrt{\varepsilon_t\left(1-\varepsilon_t\right)} < 1$，所以$Z_t$总是比$Z_{t-1}$小。

简单递归，可得

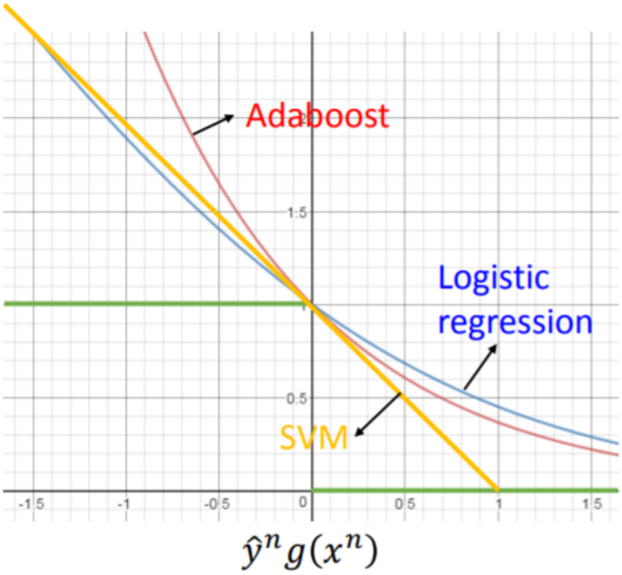
$$
Z_{T+1} = N\prod_{t=1}^{T} 2\sqrt{\varepsilon_t\left(1-\varepsilon_t\right)}
$$

所以我们可以得知，Z_{T+1}是越来越小的，因此误差也是越来越小的。

总结一下我们可以知道，AdaBoost实际上是在优化以下的loss函数：

$$
\frac{1}{N}Z_{T+1} = \prod_{t=1}^{T} 2\sqrt{\epsilon_t\left(1-\epsilon_t\right)} = \frac{1}{N}\sum_n \exp(-\hat{y}^n g\left(x^n\right))
$$

这个loss函数是理想二分类误差函数的上界，将其在图中画出即为：



上图还给出了其他分类器的loss曲线。

4. 梯度解释

我们还可以从梯度下降的角度来理解AdaBoost，或者说所谓的“加性模型”。

我们令

$$g_{t-1}(x)=\sum_{i=1}^{t-1}\alpha_i f_i(x)$$

从之前的分析我们可以知道，AdaBoost的宗旨实际上是希望找到一个函数$f_t(x)$以及a_t来提升$g_{t-1}(x)$。我们有：

$$g_t(x)=g_{t-1}(x)+\alpha_t f_t(x) \tag{4.1}$$

最终输出是：

$$H(x)=\operatorname{sign}(g_T(x))$$

根据第三节，我们可以得到$g(x)$的优化目标是最小化：

$$L(g)=\sum_n l\left(\hat{y}^n, g\left(x^n\right)\right)=\sum_n \exp\left(-\hat{y}^n g\left(x^n\right)\right)$$

所以我们需要找到一个$g(x)$，使得$L(g)=\sum_n \exp\left(-\hat{y}^n g\left(x^n\right)\right)$最小。如果我们已经知道了$g(x)=g_{t-1}(x)$，如何来更新$g(x)$呢？

答案很简单，梯度下降即可。我们有：

$$\begin{aligned} g_t(x) &= g_{t-1}(x)-\left.\eta \frac{\partial L(g)}{\partial g(x)}\right|_{\mathrm{g}(x)=g_{t-1}(x)} \\ &= g_{t-1}(x)+\sum_n \exp\left(-\hat{y}^n g_{t-1}\left(x^n\right)\right)\left(\hat{y}^n\right) \end{aligned} \tag{4.2}$$

观察一下式（4.1）和（4.2）即可发现，AdaBoost实际上是希望$\alpha_t f_t(x)$能和$-\eta \left.\frac{\partial L(g)}{\partial g(x)}\right|_{\mathrm{g}(x)=g_{t-1}(x)}$有相同的方向，这样更新的效果是相同的。这里我们可以认为α_t扮演了类似于学习率η的角色，那么即可认为，我们是在要求$f_t(x)$（Boosting的角度）尽可能要和$\sum_n \exp\left(-\hat{y}^n g_t\left(x^n\right)\right)\left(\hat{y}^n\right)$（梯度下降的角度）有相同的方向。换句话说，我们希望$f_t(x)$和$\sum_n \exp\left(-\hat{y}^n g_t\left(x^n\right)\right)\left(\hat{y}^n\right)$的乘积越大越好。所以，我们是在寻找$f_t(x)$，使得如下的目标函数最大化：

$$\sum_n \exp\left(-\hat{y}^n g_t\left(x^n\right)\right)\left(\hat{y}^n\right) f_t\left(x^n\right)$$

这个式子可以这样理解：对于每一个训练样本，我们都希望$\left(\hat{y}^n\right) f_t\left(x^n\right)$越大越好（视为某种对误差的度量，其值越大误差越小），而前面的$\exp\left(-\hat{y}^n g_t\left(x^n\right)\right)$可以理解为某种权重，我们可以将这个权重进行改写：

$$\begin{aligned} u_t^n &= \exp\left(-\hat{y}^n g_{t-1}\left(x^n\right)\right) \\ &= \exp\left(-\hat{y}^n \sum_{i=1}^{t-1} \alpha_i f_i\left(x^n\right)\right) \\ &= \prod_{i=1}^{t-1} \exp\left(-\hat{y}^n \alpha_i f_i\left(x^n\right)\right) \end{aligned}$$

可以发现，这个权重正好就是我们在AdaBoost中获得的权重！所以，在AdaBoost中，我们寻找弱分类器$f_t(x)$的过程就可以看作是梯度下降的过程。

然后剩下的问题就是如何确定α_t。假设我们已经找到了$f_t(x)$，那么现在希望找到α_t来让loss值$L(g)$最小。列出$L(g)$的表达式，有：

$$
\begin{aligned}
L(g) &= \sum_{n} \exp\left(-\hat{y}^{n}\left(g_{t-1}(x)+\alpha_{t} f_{t}(x)\right)\right) \\
&= \sum_{n} \exp\left(-\hat{y}^{n} g_{t-1}(x)\right) \exp\left(-\hat{y}^{n} \alpha_{t} f_{t}(x)\right) \\
&= \sum_{\hat{y}^{n} \neq f_{t}(x)}^{n} \exp\left(-\hat{y}^{n} g_{t-1}\left(x^{n}\right)\right) \exp\left(\alpha_{t}\right) \\
&\quad+\sum_{\hat{y}^{n}=f_{t}(x)} \exp\left(-\hat{y}^{n} g_{t-1}\left(x^{n}\right)\right) \exp\left(-\alpha_{t}\right)
\end{aligned}
$$

对α_t求导

$$
\frac{\partial L(g)}{\partial \alpha_{t}}=0
$$

即可得到

$$
\alpha_{t}=\ln \sqrt{\left(1-\varepsilon_{t}\right) / \varepsilon_{t}}
$$

这恰好就是AdaBoost要求的α_t。

综上所述，AdaBoost实际上就可以理解为是在做梯度下降。AdaBoost的整个算法是下面两步的迭代：

- 训练一个基分类器$f_t(x)$，即求解$f_t(x)$；
- 计算新的权重对训练集进行加权，即求解α_t。

从梯度下降的角度来看，求解函数$f_t(x)$可以理解为是在求解梯度，而求解α_t可以理解为是在自适应的确定学习率，这也是AdaBoost名称中“Adaptive”的由来。