提升方法AdaBoost

综合多个分类器的预测，其他sb名字：集成学习、元算法

bagging

全称bootstrap aggregating，自聚汇聚法，是从原始数据中选择S个数据集的一种技术。每个数据集都是随机挑选出来的。训练得到S个分类器。
用投票的结果作为最后的预测。

boosting

与bagging不一样，boosting通过每轮改变训练数据的权重（关注被分错的数据），以及最后得到关于多个分类器的权重，最后用加权投票得到结果。
两者之间的差别：

Bagging和Boosting的区别：

1）样本选择上：

Bagging：训练集是在原始集中有放回选取的，从原始集中选出的各轮训练集之间是独立的.

Boosting：每一轮的训练集不变，只是训练集中每个样例在分类器中的权重发生变化.而权值是根据上一轮的分类结果进行调整.

2）样例权重：

Bagging：使用均匀取样，每个样例的权重相等

Boosting：根据错误率不断调整样例的权值，错误率越大则权重越大.

3）预测函数：

Bagging：所有预测函数的权重相等.

Boosting：每个弱分类器都有相应的权重，对于分类误差小的分类器会有更大的权重.

4）并行计算：

Bagging：各个预测函数可以并行生成

Boosting：各个预测函数只能顺序生成，因为后一个模型参数需要前一轮模型的结果.

而adaboost就是boosting中一个经典的方法，下面介绍

adaboost

算法 8.1（AdaBoost）

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

输出：最终分类器 $G(x)$.

（1）初始化训练数据的权值分布

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

（2）对 $m=1,2,\cdots,M$

（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

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

（b）计算 $G_m(x)$ 在训练数据集上的分类误差率

$$e_m=P(G_m(x_i)\neq y_i)=\sum_{i=1}^{N} w_{mi} I(G_m(x_i)\neq y_i) \tag{8.1}$$

（c）计算 $G_m(x)$ 的系数

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

这里的对数是自然对数.

（d）更新训练数据集的权值分布

$$D_{m+1} = (w_{m+1,1},\cdots,w_{m+1,i},\cdots,w_{m+1,N}) \tag{8.3}$$

$$w_{m+1,i} = \frac{w_{mi}}{Z_m}\exp(-\alpha_m y_i G_m(x_i)), \quad i=1,2,\cdots,N \tag{8.4}$$

这里，Z_m 是规范化因子

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

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

$$f(x) = \sum_{m=1}^{M} \alpha_m G_m(x) \tag{8.6}$$

得到最终分类器

$$G(x) = \mathrm{sign}(f(x)) = \mathrm{sign}\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right) \tag{8.7}$$

■

提升树

采用单一结点的决策树作为Adaboost的基函数，这样的方法叫做提升树，具体的学习过程如下：

算法 8.3（回归问题的提升树算法）

输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，$x_i\in\mathcal{X}\subseteq\mathbf{R}^n$，$y_i\in\mathcal{Y}\subseteq\mathbf{R}$；

输出：提升树 $f_M(x)$.

（1）初始化 $f_0(x)=0$

（2）对 $m=1,2,\cdots,M$

（a）按式 (8.27) 计算残差

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

（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

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

■

参考

http://blog.csdn.net/dark_scope/article/details/14103983

http://www.cnblogs.com/dudumiaomiao/p/6361777.html

http://blog.csdn.net/golden1314521/article/details/46548031