十、决策树与Adaboost

一、普通决策树

ID3：信息增益

C4.5：信息增益比（防止类别较多的属性造成的影响）

Cart树，基尼指数

- CART

$$Gini(p)=\sum_{k=1}^{K}p_k(1-p_k)=1-\sum_{k=1}^{K}p_k^2$$

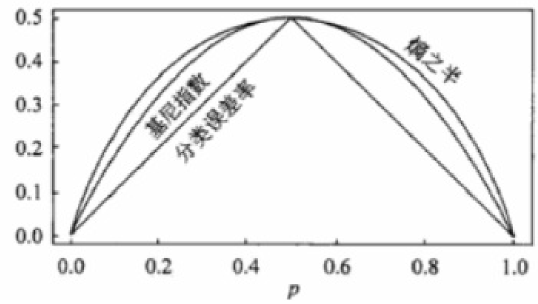
$$=1-\sum_{k=1}^{K}\left(\frac{|C_k|}{|D|}\right)^2$$

基尼指数与熵的关系

- 考察Gini系数的图像、熵、分类误差率三者之间的关系
  - 将f(x)=-lnx在x=1处一阶展开，忽略高阶无穷小，得到f(x)≈1-x

$$H(X)=-\sum_{k=1}^{K}p_k\ln p_k$$

$$\approx\sum_{k=1}^{K}p_k(1-p_k)$$



二、Bagging与AdaBoost区别

Bagging：并行，多个分类器进行民主选择　Bagging+DT=RF

- 随机森林在bagging基础上做了修改。
  - 从样本集中用Bootstrap采样选出n个样本；
  - 从所有属性中随机选择k个属性，选择最佳分割属性作为节点建立CART决策树；
  - 重复以上两步m次，即建立了m棵CART决策树
  - 这m个CART形成随机森林，通过投票表决结果，决定数据属于哪一类

随机森林的随机性：①随机放回抽样②随机选择属性

随机森林与样本有权重，但是跟分类器没权重（随机森林泛化能力大大增强）

Boost算法：串行，多个分类器的权重不同，结果叠加。

- 使用具有权值分布Dm的训练数据集学习，得到基本分类器

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

- 计算Gm(x)在训练数据集上的分类误差率

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

- 计算Gm(x)的系数 $\alpha_m = \frac{1}{2}\log\frac{1-e_m}{e_m}$

- 更新训练数据集的权值分布

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

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

- 这里，Zm是规范化因子

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

  - 它的目的仅仅是使Dm+1成为一个概率分布

$$w_{m+1,i} = \frac{w_{mi}}{Z_m}\exp(-\alpha_m y_i G_m(x_i)) \Rightarrow Z_m w_{m+1,i} = w_{mi}\exp(-\alpha_m y_i G_m(x_i)) \Rightarrow Z_1 w_{2,i} = w_{1i}\exp(-\alpha_1 y_i G_1(x_i))$$

分类器系数am，如果误差率为50%，即跟投硬币一样，没有任何意义，因此权重会变成0，误差率大于误差率小都有意义。（2）分类误差率是所有误分类样本的权重之和（3）更新权重的这一步，误分类的样本权重增加，分类正确的样本权重降低

三、GBDT

使用梯度对损失函数做近似

- 梯度提升方法寻找最优解F(x)，使得损失函数在训练集上的期望最小。方法如下：
- 首先，给定常函数F0(x)：

$$F_0(\vec{x}) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma)$$

- 以贪心的思路扩展得到Fm(x)：

$$F_m(\vec{x}) = F_{m-1}(\vec{x}) + \arg\min_{f \in H} \sum_{i=1}^{n} L(y_i, F_{m-1}(\vec{x}_i) + f(\vec{x}_i))$$

梯度近似

- 贪心法在每次选择最优基函数f时仍然困难
  - 使用梯度下降的方法近似计算
  - 将样本带入基函数f得到f(x1),f(x2),...,f(xn)，从而L退化为向量L(y1,f(x1)),L(y2,f(x2)),...,L(yn,f(xn))

$$F_m(\vec{x}) = F_{m-1}(\vec{x}) - \gamma_m \sum_{i=1}^{n} \nabla_f L(y_i, F_{m-1}(\vec{x}_i))$$

- 上式中的权值γ为梯度下降的步长，使用线性搜索求最优步长：

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(\vec{x}_i) - \gamma \cdot \nabla_f L(y_i, F_{m-1}(\vec{x}_i)))$$

提升算法

- 初始给定模型为常数 $F_0(\vec{x}) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma)$
- 对于m=1到M
  - 计算伪残差 $r_{im} = \left[\frac{\partial L(y_i, F(\vec{x}_i))}{\partial F(\vec{x}_i)}\right]_{F(\vec{x})=F_{m-1}(\vec{x})} \quad i = 1,2,\cdots,n$
  - 使用数据 计算拟合残差的基函数$f_m(x)$
  - 计算步长 $\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(\vec{x}_i) - \gamma \cdot f_m(\vec{x}_i))$
    - 一维优化
  - 更新模型 $F_m(\vec{x}) = F_{m-1}(\vec{x}) - \gamma_m f_m(\vec{x}_i)$