Boosting

① 概念：

※问题：随机森林中 m 棵决策树独立采样建立，互相之间没有关系。

※期望：ⓐ能否把前 m-1 棵树的信息用到第 m 棵树的建立上

ⓑ可否在建立 m 棵决策树时，得到投票时各棵树的权重

※思路：ⓐ在生成 m 个弱预测模型时，每一棵都依据损失函数的梯度方向 (Gradient Boosting) ⇒ XGBoost 的思想。

ⓑ直观理解：一个样本第一次分对，第二次分错，第三次这个分错的样本权值被调高。如此反复，总能收敛到一个效果不错的分类器。⇒ AdaBoost 的思想

ⓒ LR 的梯度：$\theta^* := \theta - \alpha \nabla J(\theta)$

Boost 的梯度：给决策树做一个目标函数，沿着偏导方向下降，得到另一棵决策树。

② Boosting 算法的目标

※目标：对训练样本 $(\vec{x}_1, y_1), (\vec{x}_2, y_2), (\vec{x}_3, y_3), \cdots, (\vec{x}_n, y_n)$ 目标是找到近似函数 $F(\vec{x})$，使得损失函数 $L(y, F(x))$ 的损失值最小。

$L(y, F(x))$ 的典型定义 $\begin{cases} L(y, F(\vec{x})) = \frac{1}{2}(y - F(\vec{x}))^2 & \leftarrow \text{最小二乘} \\ L(y, F(\vec{x})) = |y - F(\vec{x})| \end{cases}$

假定误差服从高斯分布（见线性回归）

用公式来表达该目标，就是

$$F^*(\vec{x}) = \underset{F}{\arg\min}\, E_{(x,y)}[L(y, F(\vec{x}))]$$

*表示最优　找到这样的 F 使得　最小　损失函数在 (x,y) 区域上的数学期望

其中 $\begin{cases} L(y, F(\vec{x})) = \frac{1}{2}(y - F(\vec{x}))^2 \text{ 或 } |y - F(\vec{x})| \\ F(\vec{x}) = \sum_{i=1}^{M} \gamma_i f_i(x) + \text{const} \end{cases}$

弱分类器权值　弱分类器预估函数

XGBoost

③ Boosting算法框架：用梯度提升的方法找到最优解 $F_0(x), F_1(x), \cdots, F_n(x)$ 使得损失函数在训练集上的数学期望最小

※ 起始步骤：给定常函数 $F_0(x) = C$，求 $F_0(\vec{x}) = \underset{\gamma}{\arg\min} \sum_{i=1}^{n} L(y_i, \gamma)$

γ：找到最优的 γ 其实就是最优的 C^*；L：损失函数；y_i：样本 y 值；γ：无预估值，其实就是 C

$\boxed{\text{这样的函数总能找到}}$，以 $L(y, F(\vec{x})) = |y - F(\vec{x})|$ 为例，

C^* 就是样本 y 值的中位数。

↓解法：不妨假设样本 $y_1, y_2, \cdots y_n$ 递增排序，$x_k \le \gamma$，$x_{k+1} \ge \gamma$

$$J(\gamma) = \sum_{i=1}^{n} |x_i - \gamma| = \sum_{i=1}^{k} (\gamma - x_i) + \sum_{i=k+1}^{n} (x_i - \gamma)$$

优化目标

$$\frac{\partial J(\gamma)}{\partial \gamma} = \sum_{i=1}^{k} (1) + \sum_{i=k+1}^{n} (-1) \xrightarrow{\text{定义}} 0$$

若要让偏导为0，前k个样本数目与后 $n-k$ 个样本数目必须为0，即 γ 为中位数

※ 迭代步骤：给定前 $t-1$ 个弱分类器(或回归器)生成的 $f_{t-1}(\vec{x})$，推导出 $f_t(\vec{x})$

$$J(f_t) = \sum_{i=1}^{n} \underbrace{L}_{\text{损失函数}}(y_i, \underbrace{\hat{y}_i^{(t-1)}}_{\text{分类器 }0\sim t-1\text{ 预测结果}} + \underbrace{f_t(x_i)}_{\text{分类器 }t\text{ 预测结果}}) + \underbrace{\Omega(f_t)}_{\text{第分类器t的正则项（防止过拟合）}} + \underbrace{C}_{\text{公式推导时析出的常数项}}$$

优化目标

把损失函数看作增量形式，用泰勒展开式展开，忽略三阶无穷小

① $f(x + \Delta x) \approx f(x) + f'(x)\Delta x + \frac{1}{2} f''(x) \Delta x^2$

$x \leftrightarrow \hat{y}_i^{(t-1)}$，$\Delta x \leftrightarrow f_t(x_i)$　定义 g_i, h_i

由①,②得到 ← ② $g_i \overset{\text{def}}{=\!=} \dfrac{\partial L(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)}}$　③ $h_i \overset{\text{def}}{=\!=} \dfrac{\partial^2 (L(y_i, \hat{y}_i^{(t-1)}))}{\partial \hat{y}_i^{(t-1)}}$

↓

③ $$J(f_t) \approx \sum_{i=1}^{n} \Big[\underbrace{L(y_i, \hat{y}_i^{(t-1)})}_{\text{相当于 } f(x)} + \underbrace{g_i \cdot f_t(\vec{x}_i)}_{\text{相当于 } f'(x)\Delta x} + \underbrace{\tfrac{1}{2} h_i f_t^2(\vec{x}_i)}_{\text{相当于 } \frac{1}{2} f''(x)\Delta x^2} \Big] + \Omega(f_t) + C$$

■ 求解式子③即可推导出 $f_t(\vec{x})$

先解决 $f_t(\vec{x}_i)$，$\Omega(f_t)$ 求值的问题

*求解 $f_t(x_i)$：假定决策树t的叶子节点数为T，每个叶子节点权值为 $\vec{w} = (w_1, w_2, \dots, w_T)$，样本 $\vec{x_i}$ 落在叶节点 $q(x)$ 中，定义

$f_t(\vec{x_i}) = w_{q(x)}$ 即节点 $q(x)$ 的权值

*求解 $\underbrace{\Omega(f_t)}_{\text{正则化参数}}$：有很多种定义，一种方法是考虑 $\boxed{\text{叶节点数}}$ 和 $\boxed{\text{叶权值平方和}}$

叶节点数：不希望太多 ⇕ 不希望树太深（4~8层为宜）

叶权值平方和：不希望叶节点分到…；太多平方和…

控制模型复杂度

事先给定的超参数

$$\Omega(f_i) = \gamma \cdot |T_t| + \lambda \cdot \frac{1}{2} \sum_{j=1}^{T} w_j^2$$

$\gamma \cdot |T_t|$：叶子个数；$\sum_{j=1}^{T} w_j^2$：叶节点权值的平方和.

*计算式子(3)：

前t-1棵树的预测值，与 $f_t(x_i)$ 无关，扔到常数项C中

$$\underset{\text{目标函数}}{J(f_t)} \approx \sum_{i=1}^{n} \left[\boxed{L(y_i, \hat{y}_i^{(t-1)})} + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) + \boxed{C}$$

$$= \sum_{i=1}^{n} \left[g_i \boxed{f_t(x_i)} + \frac{1}{2} h_i \boxed{f_t^2(x_i)} \right] + \boxed{\Omega(f_t)} + C,$$

用上两步的推导结论求解.

$$= \sum_{i=1}^{n} \left[g_i \boxed{w_{q(x_i)}} + \frac{1}{2} h_i \boxed{w_{q(x_i)}^2} \right] + \boxed{\gamma \cdot |T| + \lambda \cdot \frac{1}{2} \sum_{j=1}^{T} w_j^2} + C,$$

加法结合律：由按样本分组改为按叶节点分组.

$$= \sum_{j=1}^{T} \left[\left[\sum_{i \in I_j} g_i \right] w_j + \frac{1}{2} \left[\sum_{i \in I_j} h_i \right] w_j^2 \right] + \gamma \cdot |T_t| + \lambda \cdot \frac{1}{2} \sum_{j=1}^{T} w_j^2 + C$$

$$= \sum_{j=1}^{T} \left[\boxed{\left[\sum_{i \in I_j} g_i \right]} w_j + \frac{1}{2} \boxed{\left[\sum_{i \in I_j} h_i + \lambda \right]} w_j^2 \right] + \gamma \cdot |T_t| + C.$$

$\sum_{i \in I_j} g_i$ 命名为 G_j；$\sum_{i \in I_j} h_i + \lambda$ 命名为 H_j

$$= \sum_{j=1}^{T} \left(\boxed{G_j} w_j + \frac{1}{2} \boxed{H_j} w_j^2 \right) + \gamma \cdot |T_t| + C.$$

w_j：第t棵决策树叶节点j的权重.

到这一步，优化目标已经可以由第t棵决策树叶节点的权重 w_j $(j=1,2,\dots|T|)$ 表

* 求解最优值：对优化目标（式子③）关于 $\vec{W}=(w_1, w_2, \cdots w_{|T|})$ 求偏导

$$\frac{\partial J(f_t)}{\partial w_j} = G_j + (H_j+\lambda)w_j \overset{令}{=} 0 \Rightarrow w_j = -\frac{G_j}{H_j+\lambda}$$

从而得到第t棵树各个叶子节点的权重（第t-1棵树 T_{t-1} 的叶节点权重也是这么来的，当 $t=0$ 时 T_0 是常函数不存在权重问题）

* 把求解得到的 w_j 回代到目标函数 $J(f_t)$ 中

$$J(f_t) = \sum_{j=1}^{|T|}\left(G_j w_j + \frac{1}{2}H_j w_j^2\right) + \gamma|T_t| + C$$

$$= \sum_{j=1}^{|T|}\left(-\frac{G_j^2}{H_j+\lambda} + \frac{1}{2}\cdot\frac{H_j G_j^2}{(H_j+\lambda)^2}\right) + \gamma|T_t| + C$$

$$= -\frac{1}{2}\sum_{j=1}^{|T|}\frac{G_j^2}{H_j+\lambda} + \gamma\cdot|T|$$

即为最小损失值公式

* 能够计算损失值，那么给定分割点 Φ（把样本分成 L，R 两部分），就能计算信息增益。

$$\underset{分割点}{Gain(\Phi)} = J(f_t) - [J_R(f_t) + J_L(f_t)]$$

$$= \frac{1}{2}\left[-\frac{(G_L+G_R)^2}{H_L+H_R+\lambda} - \left[-\frac{G_L^2}{H_L+\lambda} - \frac{G_R^2}{H_R+\lambda}\right]\right] - \gamma$$

$$= \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma$$

枚举所有可分割点，增益大于阈值，选增益最大的点进行分割，增益小于阈值，不再分割；或者遇到纯节点也不再分割

* 注意，在推导式子③时用的 $f_t(\vec{x_i}) = W_{q(\vec{x})}$ 是假设我们知道 $q(\vec{x})$ 和 $W_{q(\vec{x})}$ 的值，而现在我们已经能得到 $q(\vec{x})$ 和 $W_{q(\vec{x})}$ 的真实值，因此可用于下一轮迭代

$$J(f_{t+1}) = \sum_{i=1}^{n} L\left(y_i, \hat{y}_i^{(t)} + f_{t+1}(\vec{x_i})\right) + \Omega(f_{t+1}) + C$$ 以推导 $f_{t+1}(\vec{x_i})$

※ 模型调参

<a> 正则项 $\Omega(f_i)$：对复杂模型增加惩罚项，如模型复杂度正比于叶节点数目或叶结点预测值的平方和等

<b> 叶结点数目控制了树的层数，一般选择 $4 \le J \le 8$

<c> 叶结点包含最少样本数目限制：防止出现过小叶节点，降低预测方差

<d> 梯度提升迭代次数 M：增加 M 可降低训练集损失值，但有过拟合风险

※ 其它训练技巧

<a> 衰减 Shrinkage $F_m(\vec{x}) = F_{m-1}(\vec{x}) + \nu \cdot \gamma_m f_m(\vec{x}_i) \quad 0 < \nu \le 1$

ν 称为学习率，$\nu=1$ 即为前面推导的原始模型，推荐 $\nu<0.1$ 的学习率，但太小也会导致迭代次数增加

<b> 随机梯度提升：Stochastic Gradient Boosting

每次迭代对伪残差样本采用无放回的降采样，用部分样本来训练基函数的参数。令训练样本数占所有伪残差样本的比例为 f：$f=1$ 即为原始模型；推荐 $0.5 \le f \le 0.8$

※ 较小的 f 能够增强随机性，防止过拟合，并且收敛的快。

降采样的好处是能够使用剩余样本做模型验证

※ 例子：见PPT

※ 小结：以上为 XGBoost 使用的核心推导过程，相比传统的 GBDT，XGBoost 使用了二阶导信息，可以更快地在训练集上收敛

XGBoost { 属于"随机森林族"因此也具有抗过拟合的优势
使用了并行/多核计算，训练速度更快

④ AdaBoost

※思想：对于N个样本，给每个样本一个权重，初始权重都是 $1/N$

第 i 轮迭代中，如果分错，该权重会被调高；如分对被调低

第 $i+1$ 轮迭代中，之前分错的样本会得到更多重视

每轮迭代都会生成一个弱分类器及权重，最终组合成一个强分类器

※算法框架：

$$D_1 = \{W_n^{(1)}\} = \{\tfrac{1}{N}, \cdots\} \xrightarrow{迭代} \{W_n^{(2)}\} \xrightarrow{迭代} \cdots \xrightarrow{迭代} \{W_n^{(M)}\}$$

$y_1(x)$ 基本弱分类器1　　$y_2(x)$ 基本弱分类器2　　$y_M(x)$ 基本弱分类器M

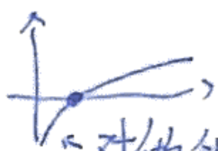
$$Y_M(x) = \mathrm{sign}\left(\sum_m^M \alpha_m y_m(x)\right)$$

构建基本分类器的线性组合，作为最终分类器

<a> 如何计算 α_m（模型权重）

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

做对概率 / 做错概率



对做错惩罚大

$\boxed{误差率}$ $e_m = P(\underbrace{G_m(\vec{X_i})}_{弱分类器预测值：-1或+1} \neq y_i) = \sum_{i=1}^N \underbrace{w_{m_i}}_{样本权值} \underbrace{I(G_m(\vec{X_i}) \neq y_i)}$

$I(test) = \begin{cases} 1 & test为true \\ 0 & test为false \end{cases}$

<b> 如何更新 $D_m = \{W_n^{(m)}\}$（样本权重）

样本标签　基本分类器预测值

$$\underset{\substack{\downarrow \\ 基本分类器序号}}{W_{m+1,}}\underset{\substack{\downarrow \\ 样本}}{_{i}} = \frac{W_{m_i}}{\underset{\substack{\downarrow \\ 规范化因子，为了使 W_{m+1,i} 成为一个概率值.}}{Z_m}} \cdot e^{-\alpha_m \cdot y_i \cdot G_m(\vec{X_i})}$$

α_m：基本分类器权重　y_i：取值 -1 或 $+1$　$G_m(\vec{X_i})$：取值 -1 或 $+1$

其中 $Z_m = \sum_{i=1}^N W_{m_i} e^{-\alpha_m y_i G_m(\vec{X_i})}$

※例子：见PPT ： 模型实例推导

※例子：见PPT ： Adaboost代码

※AdaBoost误差上限：AdaBoost是Adaptive Boost缩写，含义是只要基础分类器误差<0.5，就可以让AdaBoost误差下降，其理论依据就是AdaBoost误差上限

$$\frac{1}{N}\sum_{i=1}^{N} I(G(x_i)\neq y_i) \underset{ⓐ}{\leq} \frac{1}{N}\sum_i e^{-y_i\cdot f(x_i)} \underset{ⓑ}{=} \prod_m Z_m = \prod_m \left(\sum_{i=1}^{N} w_{mi} e^{-\alpha_m y_i G_m(x_i)}\right)$$

Z_m定义

*证明左半部分ⓐ：$G(x_i)\neq y_i$时，$y_i f(x_i)<0$.

*证明右半部分ⓑ：

$$\frac{1}{N}\sum_t e^{-y_i\cdot f(\vec{x}_i)} = \sum_t \frac{1}{N} e^{-\sum_{m=1}^{M}\alpha_m\cdot y_i\cdot G_m(\vec{x}_i)}$$

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

$$=\sum_t w_{1t}\cdot e^{-\sum_{m=1}^{M}\alpha_m\cdot y_i\cdot G_m(\vec{x}_i)} = \sum_t w_{1t}\prod_{m=1}^{M} e^{-\alpha_m\cdot y_i\cdot G_m(\vec{x}_i)}$$

$w_{11}=w_{12}=\cdots=w_{1N}=\frac{1}{N}$ 初始参数

$$=\sum_t w_{1t}\cdot e^{-\alpha_1\cdot y_t G_1(\vec{x}_t)}\prod_{m=2}^{M} e^{-\alpha_m y_t\cdot G_m(\vec{x}_t)}$$

$$=\sum_t Z_1\cdot w_{2t}\prod_{m=2}^{M} e^{-\alpha_m\cdot y_t\cdot G_m(\vec{x}_t)}$$

$w_{2t}=\frac{w_{1t}}{Z_1}\cdot e^{-\alpha_1 y_t G_1(\vec{x}_t)}$

$$=Z_1\sum_t w_{2t}\prod_{m=2}^{M} e^{-\alpha_m y_t G_m(\vec{x}_t)}$$

$w_{3t}=\frac{w_{2t}}{Z_2} e^{-\alpha_2 y_t G_2(\vec{x}_t)}$

$$=Z_1 Z_2\sum_t w_{3t}\prod_{m=3}^{M} e^{-\alpha_m y_t G_m(\vec{x}_t)} = \cdots$$

$$=Z_1 Z_2\cdots Z_{M-1}\sum_t w_{Mt}\, e^{-\alpha_M y_t G_M(\vec{x}_t)}$$

$$=Z_1 Z_2\cdots Z_{M-1} Z_M = \prod_{m=1}^{M} Z_m$$

$w_{Mt}=\frac{w_{(M-1)t}}{Z_M} e^{-\alpha_M y_t G_M(\vec{x}_t)}$

*训练误差上界

$$\prod_{m=1}^{M} Z_m = \prod_{m=1}^{M}\left[2\sqrt{e_m(1-e_m)}\right] \overset{\boxed{令\gamma_m=\frac{1}{2}-e_m}}{=} \prod_{m=1}^{M}\sqrt{1-4\gamma_m^2} \leq e^{\boxed{-2\sum_{m=1}^{M}\gamma_m^2}}$$

推导见附录

$\boxed{e_m：分错次数}$

$$Z_m=\sum_{i=1}^{M} w_{mi}\, e^{-\alpha_m\cdot y_i\, G_m(\vec{x}_i)}$$

$$=\sum_{y_i=G_m(\vec{x}_i)} w_{mi} e^{-\alpha_m} + \sum_{y_i\neq G_m(\vec{x}_i)} w_{mi} e^{\alpha_m}$$

$$=(1-e_m)e^{-\alpha_m}+e_m e^{\alpha_m} = 2\sqrt{e_m(1-e_m)}$$

※附录二：证明 $\prod_{m=1}^{M}\sqrt{(1-4\gamma_m^2)} \le e^{-2\sum_{m=1}^{M}\gamma_m^2}$

$$e^{-x}+x-1 \ge 0 \Rightarrow e^{-x} \ge 1-x$$

代入 $x=4\gamma^2$ 得. $e^{-4\gamma^2} \ge 1-4\gamma^2 \Rightarrow e^{-2\gamma^2} \ge \sqrt{1-4\gamma^2}$

$$\Rightarrow \begin{cases} e^{-2\gamma_1^2} \ge \sqrt{1-4\gamma_1^2} \\ e^{-2\gamma_2^2} \ge \sqrt{1-4\gamma_2^2} \\ \vdots \\ e^{-2\gamma_m^2} \ge \sqrt{1-4\gamma_m^2} \end{cases} \Rightarrow \prod_{m=1}^{M} e^{-2\gamma_m^2} \ge \prod_{m=1}^{M}\sqrt{1-4\gamma_m^2}$$

$$\Rightarrow e^{-2\sum_{m=1}^{M}\gamma_m^2} \ge \prod_{m=1}^{M}\sqrt{1-4\gamma_m^2}.$$

※前面的 $\frac{1}{n}\sum_i e^{-y_i \cdot f(x_i)}$ 看着比较突兀，其实可以把它看作是损失函数取指数函数，认为模型服从若干基础分类器加权，然后用贪心法（前向分步算法）来推导，事后推导发现AdaBoost其实正符合这个思路

（推导过程见附录1）

⑤ Bagging 与 Boosting

Bagging ↓ 增强训练集精度，降低Variance

Boosting ↘ 减少偏差Bias，增加泛化能力

⑥ GBDT：使用关于分类器的一阶导数进行学习（与XGBoost不同）

XGBoost是GBDT（GBM）的一个C++实现

⑦ 数据预处理 — 清洗

重要性

高

※特征：重要性高，缺失率低

①通过计算填充

②通过经验或业务知识估计

※特征：重要性高，缺失率高

①从其它渠道补全

②用其它字段计算获取

③去除字段并在结果中标注

高

低

特征：重要性低，缺失率低

①不做处理或简单填充

※特征：不重要，缺失率高

去除该字段

低