XGBoost属于boosting方法之一，这种方法的特点是构建多个基分类器，训练时采用串行的方式，各个基分类器之间有依赖。测试时，根据各层分类器的结果的加权得到最终结果。

XGBoost的算法思想：

残差：**A的实际值 - A的预测值 = A的残差**

1. **训练阶段不断地添加树，去拟合之前预测的残差。树中的每一个节点都进行特征分裂，直到满足特定的条件（树的最大深度、增益小于阈值等）；**
2. **训练过程结束得到t棵树，对于样本的预测，就是根据样本的特征在每棵树中归到一个叶子结点，每个叶子结点输出一个预测值；**
3. **将每棵树对应的预测值求和就是该样本的预测值。**

XGBoost的目标函数：

我们的目标就是使得建立的树群对样本的预测值$\hat{y}_i$ 尽量接近真实值y_i 。

损失函数：

$$L = \sum_{i=1}^{n} l(y_i, \hat{y}_i) \tag{1}$$

模型的预测精度由模型的偏差和方差共同决定，损失函数代表了模型的偏差，想要方差小则需要在目标函数中添加正则项，用于防止过拟合。所以目标函数由模型的损失函数L 与抑制模型复杂度的正则项Ω 组成，最终的目标函数定义如下：

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{i=1}^{t} \Omega(f_i) \tag{2}$$

后半部分是正则化项，将t棵树的复杂度求和（值越小复杂度越低，泛化能力越强）。

接下来是公式推导的部分：

t棵树的预测结果与t-1棵树的预测结果有关：其中$\hat{y}_i^{(t-1)}$ 是t-1步给出的预测值，常量；$f_t(x_i)$是第t棵树加入的预测值。

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i) \tag{3}$$

将正则化项进行拆分，由于前t-1棵树的结构已经确定，因此前t-1棵树的复杂度之和可以用一个常量表示，如下所示：

$$\begin{aligned} \sum_{i=1}^{t} \Omega(f_i) &= \Omega(f_t) + \sum_{i=1}^{t-1} \Omega(f_i) \\ &= \Omega(f_t) + constant \end{aligned} \tag{4}$$

将（3）和（4）带入（2）中：

$$
\begin{aligned}
Obj^{(t)} &= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \Omega(f_i) \\
&= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \sum_{i=1}^{t} \Omega(f_i) \\
&= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) + constant
\end{aligned}
\tag{5}
$$

注意（5）式子中，唯一的变量就是$f_t(x_i)$，也就是第t棵树加入的预测值，因此求这个目标函数的最小值就是找到相应的$f_t(x_i)$。

泰勒展开

对损失函数进行在$\hat{y}_i^{(t-1)}$处的**二阶泰勒展开：**

$$
f(x + \Delta x) \approx f(x) + f'(x)\Delta x + \frac{1}{2} f''(x)\Delta x^2 \tag{6}
$$

$$
l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) = l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \tag{7}
$$

Δx 对应第t棵树$f_t(x_i)$，g_i 为损失函数一阶导，h_i 是损失函数二阶导，x 对应$\hat{y}_i^{(t-1)}$，因此是对$\hat{y}_i^{(t-1)}$求导。

将展开后的损失函数（7）带入目标函数（5）中：

$$
Obj^{(t)} \simeq \sum_{i=1}^{n} [l(y_i, \hat{y}_i^{t-1}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) + constant \tag{8}
$$

其中$l(y_i, \hat{y}_i^{(t-1)})$也是常数项，因此去掉后剩余的目标函数为：

$$
Obj^{(t)} \simeq \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \tag{9}
$$

因此求出每一步损失函数的一阶导和二阶导的值（由于前一步的 $\hat{y}^{(t-1)}$ 是已知的，所以这两个值就是常数），然后最优化目标函数，就可以得到每一步的 $f(x)$，最后根据加法模型得到一个整体模型。

目标函数的正则项由生成的所有决策树的叶子结点数量和所有结点权重所组成的向量的L_2 范式共同决定：

$$
\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{10}
$$

Number of leaves

L2 norm of leaf scores

叶子结点权重向量的L_2范数

带入（9）中，**将原本便利所有样本变为遍历所有叶子结点的样本集**（因为每个样本最终都会落到叶子结点上）：

$$
\begin{aligned}
Obj^{(t)} &\simeq \sum_{i=1}^{n}\left[g_i f_t(x_i)+\frac{1}{2}h_i f_t^2(x_i)\right]+\Omega(f_t) \\
&=\sum_{i=1}^{n}\left[g_i w_{q(x_i)}+\frac{1}{2}h_i w_{q(x_i)}^2\right]+\gamma T+\frac{1}{2}\lambda\sum_{j=1}^{T}w_j^2 \\
&=\sum_{j=1}^{T}\left[(\sum_{i\in I_j}g_i)w_j+\frac{1}{2}(\sum_{i\in I_j}h_i+\lambda)w_j^2\right]+\gamma T
\end{aligned}
\tag{11}
$$

定义 $G_j=\sum_{i\in I_j}g_i$ ：叶子结点 j 所包含样本的一阶偏导数累加之和，常量；

定义 $H_j=\sum_{i\in I_j}h_i$ ：叶子结点 j 所包含样本的二阶偏导数累加之和，常量；

因此最终的目标函数为：

$$
Obj^{(t)}=\sum_{j=1}^{T}\left[G_j w_j+\frac{1}{2}(H_j+\lambda)w_j^2\right]+\gamma T \tag{12}
$$

其中真正的变量只有最后一棵树的叶子结点权重向量w_j 不确定。

目标函数求极值

对于最后一棵树的每一个叶子结点 j ，单独的都有：

$$
G_j w_j+\frac{1}{2}(H_j+\lambda)w_j^2 \tag{13}
$$

这个式子是只包含一个变量**叶子结点权重**w_j 的一元二次函数，可以直接求其极值点。

而叶子结点之间相互独立，因此，每个叶子结点子式都达到极值时，目标函数才达到极值点。可以求得叶子结点 j 对应的权值：

$$
w_j^*=-\frac{G_j}{H_j+\lambda} \tag{14}
$$

所以目标函数的极值可以求出，越小越好：

$$
Obj=-\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j+\lambda}+\gamma T \tag{15}
$$

至此公式推导结束

树的生成

每棵树是如何分裂的，选取什么特征作为切点很关键。

贪心算法

从树的深度为0开始：

1. 对每个叶节点枚举所有的可用特征；
2. 针对每个特征，把属于该节点的训练样本根据该特征值进行升序排列，通过线性扫描的方式来决定该特征的最佳分裂点，并**记录该特征的分裂收益**；
3. 选择收益最大的特征作为分裂特征，用该特征的最佳分裂点作为分裂位置，在该节点上分裂出左右两个新的叶节点，并为每个新节点关联对应的样本集；
4. 回到第1步，递归执行直到满足特定条件为止；

计算每个特征的分裂收益

分裂前目标函数：

$$Obj_1 = -\frac{1}{2}[\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}]+\gamma$$

分裂后目标函数：

$$Obj_2 = -\frac{1}{2}[\frac{G_L^2}{H_L+\lambda}+\frac{G_R^2}{H_R+\lambda}]+2\gamma$$

分裂后收益：

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

计算每次分裂的分割方案

线性扫描，计算所有分割方案对应的特征分裂的Gain，哪种最大选取哪种划分方法。对于所有的特征，我们只要做一遍从左到右的扫描就可以枚举出所有分割的梯度和GL和GR。然后用计算Gain的公式计算每个分割方案的分数就可以了。

限制树的生成和分裂

1. 对于引入分割后的复杂度过高，带来的增益小于阈值γ（正则项中叶子结点树T的稀疏）时，则忽略分割（类似于预剪枝）；
2. 树达到最大深度max_depth，停止建树，避免树太深导致学习局部样本从而过拟合；
3. 一个叶子结点的样本数太少了，防止过拟合；
4. 树的最大数量