<https://zhuanlan.zhihu.com/p/75217528>

<https://cloud.tencent.com/developer/article/1513111>

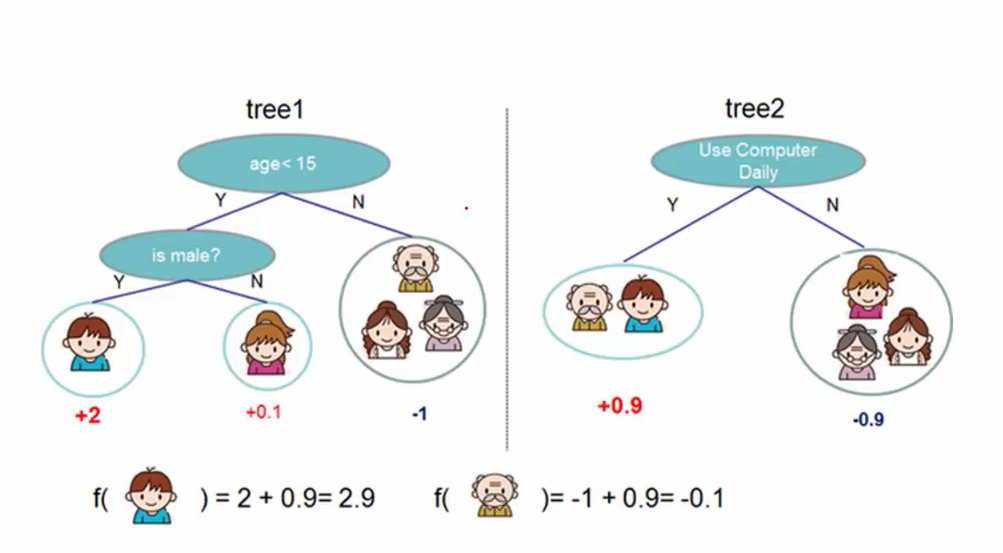
**什么是XGBoost？**

XGBoost是Exterme Gradient Boosting（极限梯度提升）的缩写，它是基于决策树的集成机器学习算法，它以梯度提升（Gradient Boost）为框架。XGBoost是由GBDT发展而来，同样是利用加法模型与前向分步算法实现学习的优化过程，但与GBDT是有区别的。主要区别包括以下几点：

* 目标函数：XGBoost的损失函数添加了正则化项，使用正则用以控制模型的复杂度，正则项里包含了树的叶子节点个数、每个叶子节点权重（叶结点的socre值）的平方和，L2正则。
* 优化方法：GBDT在优化时只使用了一阶导数信息，XGBoost在优化时使用了一、二介导数信息。
* 缺失值处理：XBGoost对缺失值进行了处理，通过学习模型自动选择最优的缺失值默认切分方向。
* 防止过拟合: XGBoost除了增加了正则项来防止过拟合,还支持行列采样的方式来防止过拟合。
* 结果：它可以在最短时间内用更少的计算资源得到更好的结果。

## XGBoost的基学习器

XGBoost的可以使用Regression Tree（CART）作为基学习器，也可以使用线性分类器作为基学习器。以CART作为基学习器时，其决策规则和决策树是一样的，但CART的每一个叶节点具有一个权重，也就是叶节点的得分或者说是叶节点的预测值。CART的示例如下图

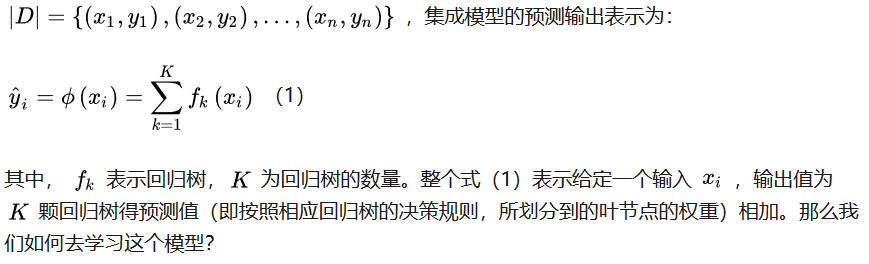


图中为两颗回归树（左右两个），其中树下方的输出值即为叶节点的权重（得分），

当输出一个样本进行预测时，根据每个内部节点的决策条件进行划分节点，最终被划分到的叶节点的权重即为该样本在该树上的预测输出值。

## XGBoost的模型

XGBoost模型的定义为：给定一个包含n个样本m个特征的数据集，



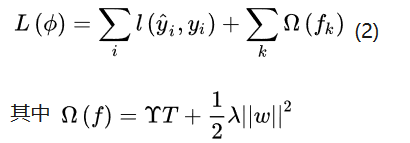
第i个样本的预测值为前k个决策树决策结果的累加，这个决策结果其实就是样本i在决策树k上最终划分到的叶子节点的权值w。这里需要注意理解，该式是指通过输入样本i的特征x得到决策树k的决策结果，和线性回归的wx的线性加和是完全不同的。

**模型的学习**

通常情况下，怎样去学习一个模型？

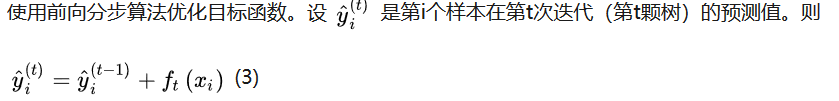
1. 定义目标函数，即损失函数以及正则项。
2. 优化目标函数。

按照这个套路，第一步定义XGBoost的目标函数。

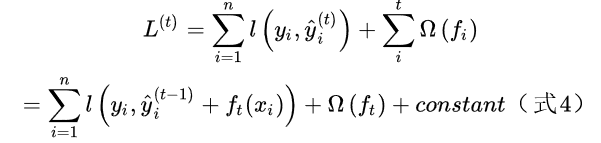


公式（2）右边第一部分是度量预测值与真实值之间的损失函数，第二部分表示对模型复杂度的惩罚项（正则项）。在正则项中表示正则项系数，T代表给定一棵树的叶子节点个数，表示每颗树叶节点上的输出分数的平方(相当于L2正则)，其中w表示叶子节点的权值，也可以说是决策结果。从目标函数的定义可以看出XGBoost对模型复杂度考虑了每颗树的叶节点个数，以及每颗树叶节点输出得分值得平方和。·

第二步，优化目标函数。在通常的模型中针对这类目标函数可以使用梯度下降的方式进行优化，但注意到fk表示的是一棵树，而非一个数值型的向量，所以不能使用梯度下降的方式去优化该目标函数。那么怎么优化这个目标函数呢？**前向分步算法！！！**ps：关于前向分步算法详细内容见李航博士的《统计学习方法》P114.

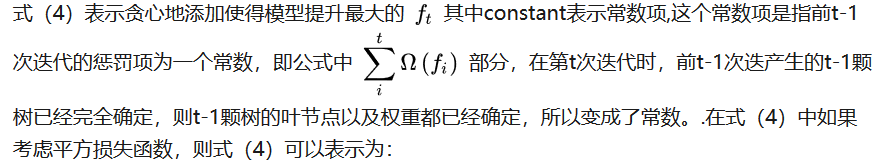


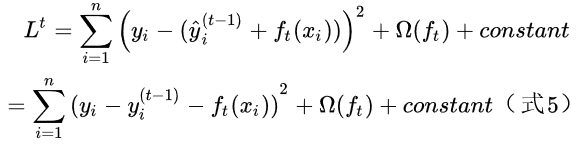
公式（3）表示样本i在t次迭代后的预测值=样本i在前t-1次迭代后的预测值+当前第t颗树预测值。则目标函数可以表示为



这里两个数需要理清，i代表样本，t代表树，对于每一个样本误差来说都是t颗树的预测结果与真实结果的误差的总和。

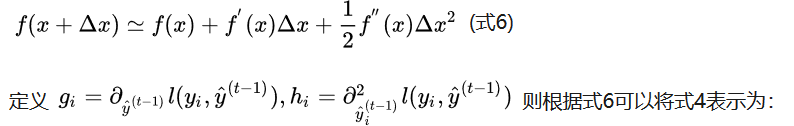
这里进行了关键的一步转换，3式转换为4式

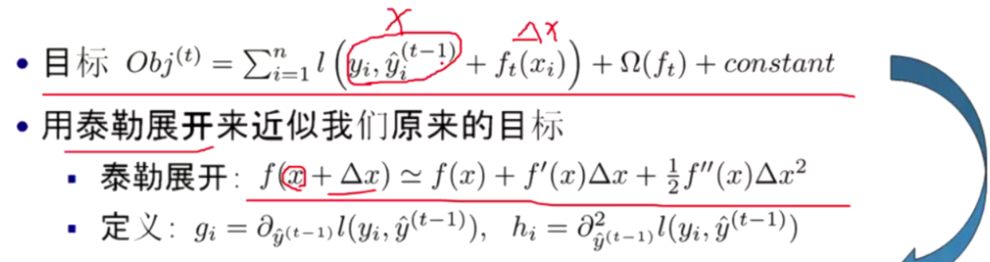


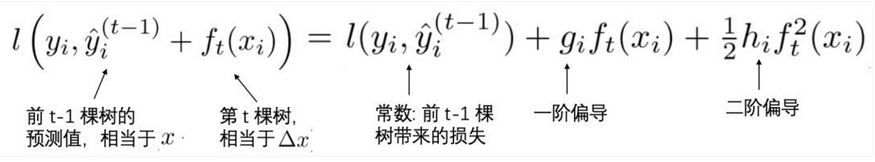


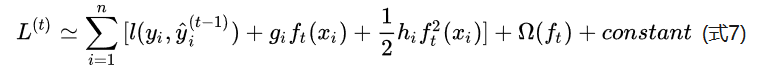
对于样本i，即经过前t-1颗树的预测之后与真实值之间的差距，**就是所谓的残差，在正则项中我们没有发现和和样本i有关的变量，是的在正则项只与树的叶节点个数和叶节点的权值有关，树结构固定，那正则项就是固定的。**

在XGBoost中提出使用二阶泰勒展开式近似表示式5.泰勒展开式的二阶形式为：

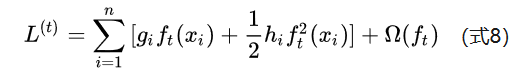








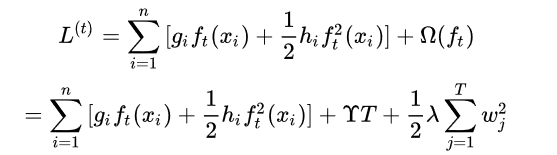
注意式7中部分表示前t-1次迭代的损失函数，在当前第t次迭代来说已经是一个确定的常数，省略常数项则得到下面的式子：



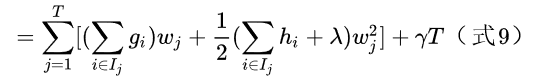
则我们的目标函数变成了公式8.可以看出目标函数只依赖于数据点的一阶和二阶导数。其中



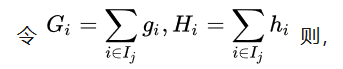
接下来针对公式8进行细分



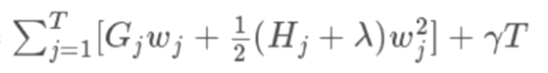
**这里再次进行了一次关键的变换，从样本的变量变成了叶子节点的遍历**



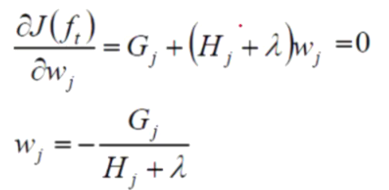
**因为所有的样本都散落在了不同的叶子节点中，故对于叶子节点的遍历相当于对样本进行了全部的遍历，上式将****=w 提了出来，因为每个叶子节点都有一个权值，g、h一阶二阶函数还是按照该叶子节点中所有样本进行了加和。**



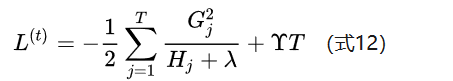
**Gi该叶子节点内所有样本一阶导之和**



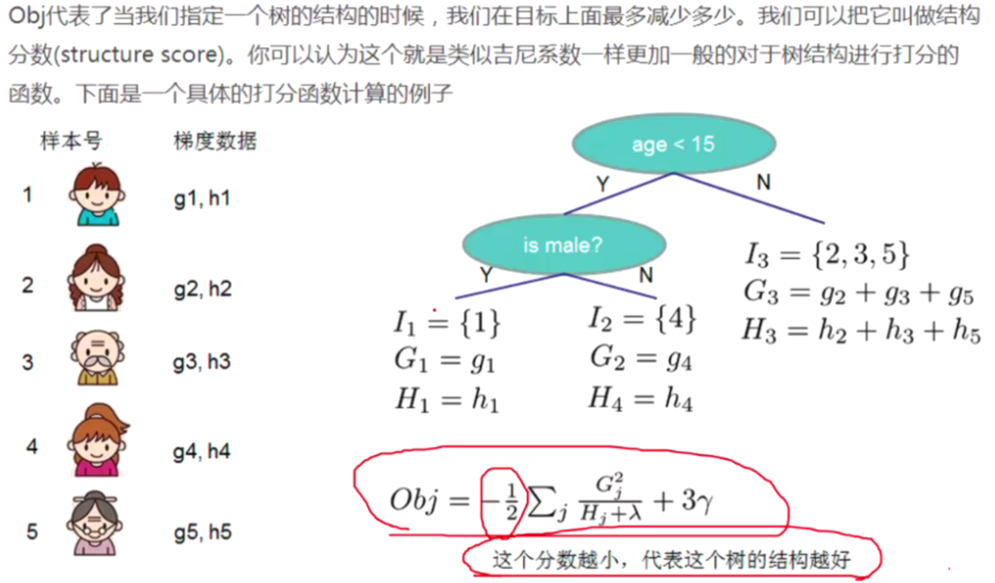




**权值可以确定了，最终目标函数得**

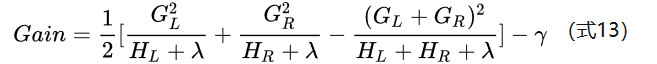


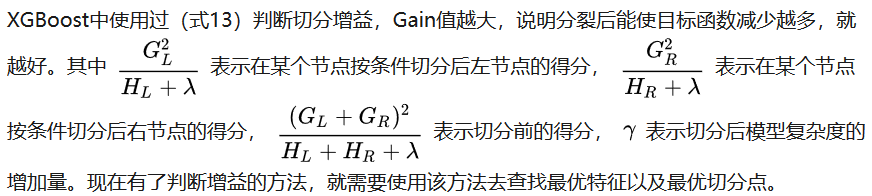
**到这里，只要我们指定具体的损失函数，那G、H都可以快速求出来。**



## 树的生成

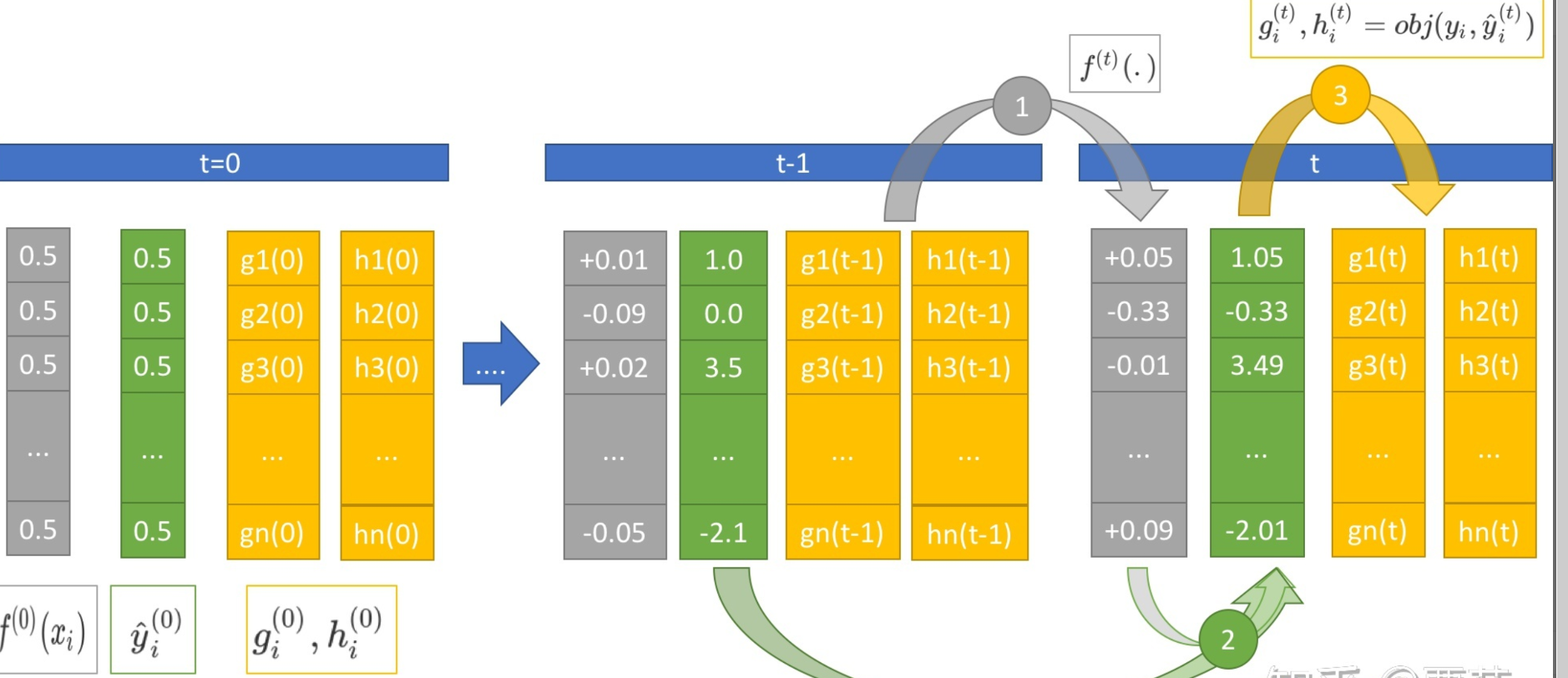
有了损失函数，我们可以评价当前这个树的好坏，损失可以理解成误差，误差越小越好，在决策树的生成中，我们用ID3、C4.5、Gini指数等指标去选择最优分裂特征、切分点(CART时)，XGBoost同样定义了特征选择和切分点选择的指标。目标函数前面有个负号，说明，分裂之后的Gain越大越好





该式给出了构造决策树的过程中，每一次划分的评价标准，类似于信息增益，但是却是用推导出来的目标函数做评价的。

下图给出了第0轮到t轮的生成过程



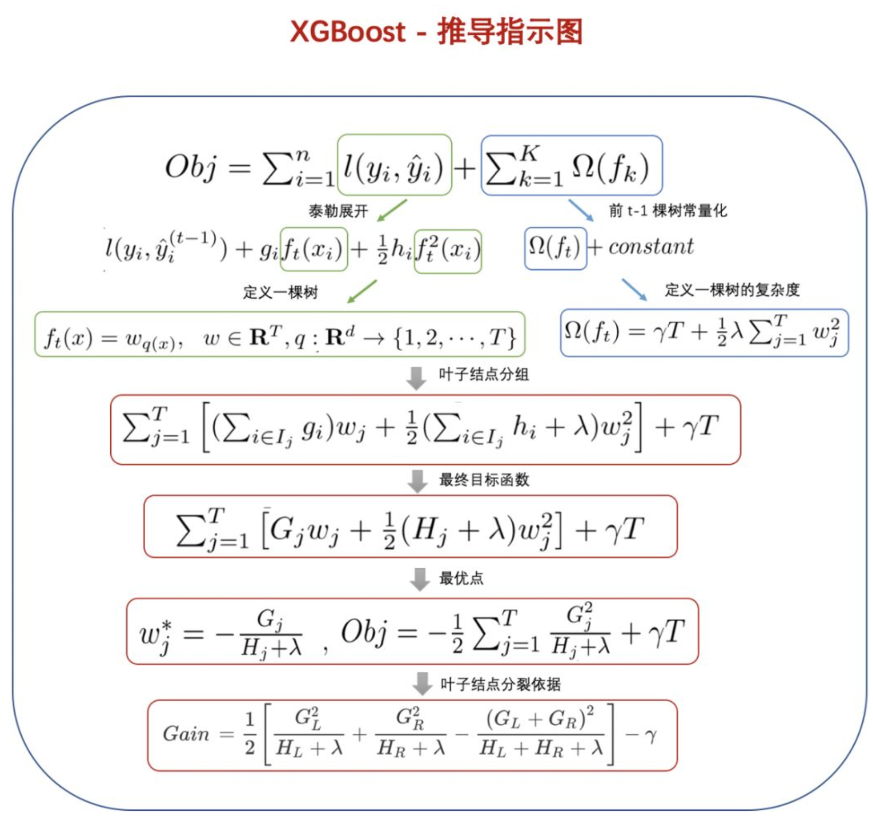
在二分类问题中，0轮默认预测结果都是0.5

### ****寻找最佳分裂点****

在所有特征中找出最好的分裂点（分裂后增益最大的特征及特征值）是一种贪心的方法，每次进行分裂尝试都要遍历一遍全部候选分割点，也叫做全局扫描法。但当数据量过大导致内存无法一次载入或者在分布式情况下，贪心算法的效率就会变得很低，全局扫描法不再适用。

### ****停止生长****

当树达到最大深度时，停止建树，因为树的深度太深容易出现过拟合，这里需要设置一个超参数max\_depth。



XGB面试题

[https://mp.weixin.qq.com/s?\_\_biz=MzI1MzY0MzE4Mg==&mid=2247485159&idx=1&sn=d429aac8370ca5127e1e786995d4e8ec&chksm=e9d01626dea79f30043ab80652c4a859760c1ebc0d602e58e13490bf525ad7608a9610495b3d&scene=21#wechat\_redirect](https://mp.weixin.qq.com/s?__biz=MzI1MzY0MzE4Mg==&mid=2247485159&idx=1&sn=d429aac8370ca5127e1e786995d4e8ec&chksm=e9d01626dea79f30043ab80652c4a859760c1ebc0d602e58e13490bf525ad7608a9610495b3d&scene=21" \l "wechat_redirect)