**2 Task2 GBDT**

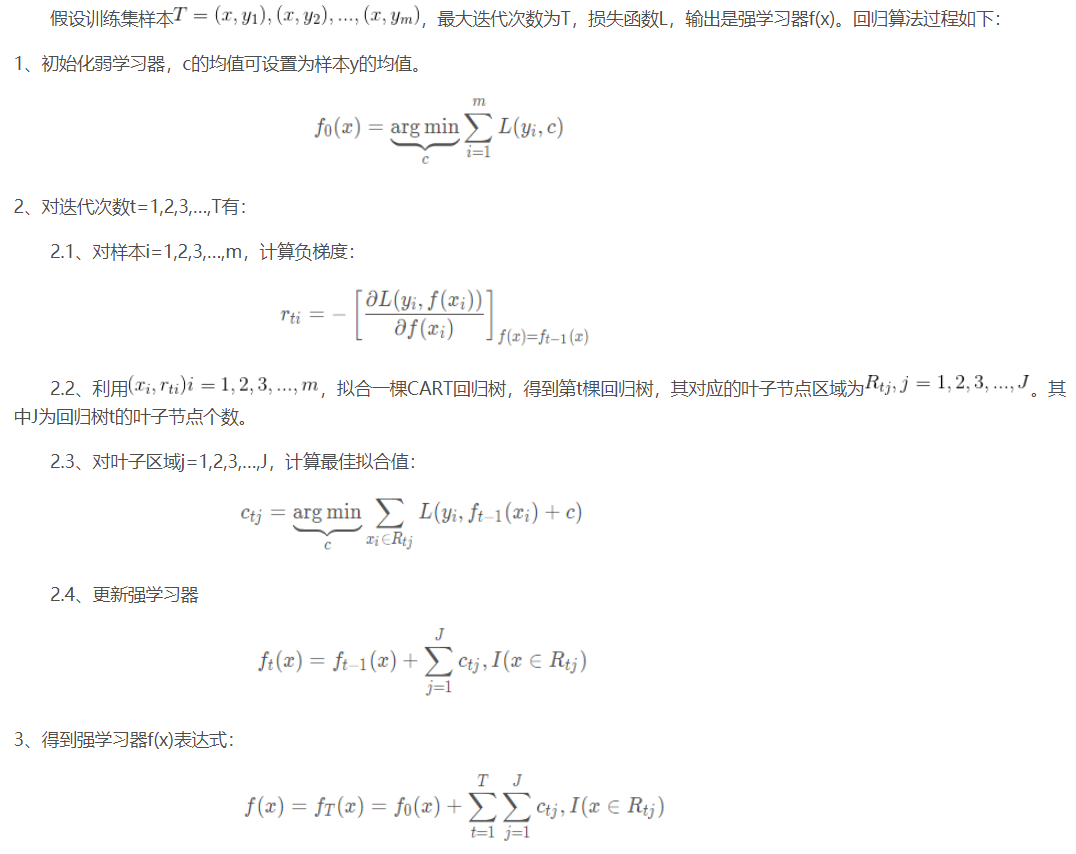
主要参考文献：（原文链接：https://blog.csdn.net/yyy430/article/details/85108797）

**2.1 GBDT算法原理**

GBDT 的全称是 Gradient Boosting Decision Tree，梯度提升决策树。基于梯度提升算法的学习器叫做 GBM(Gradient Boosting Machine)。理论上，GBM 可以选择各种不同的学习算法作为基学习器。GBDT 实际上是 GBM 的一种情况。

GBDT采用的是加法模型与前向分布算法。GBDT与Adboost最主要的区别在于两者如何识别模型的问题。Adaboost用错分数据点来识别问题，通过调整错分数据点的权重来改进模型。GBDT通过负梯度来识别问题，通过计算负梯度来改进模型。

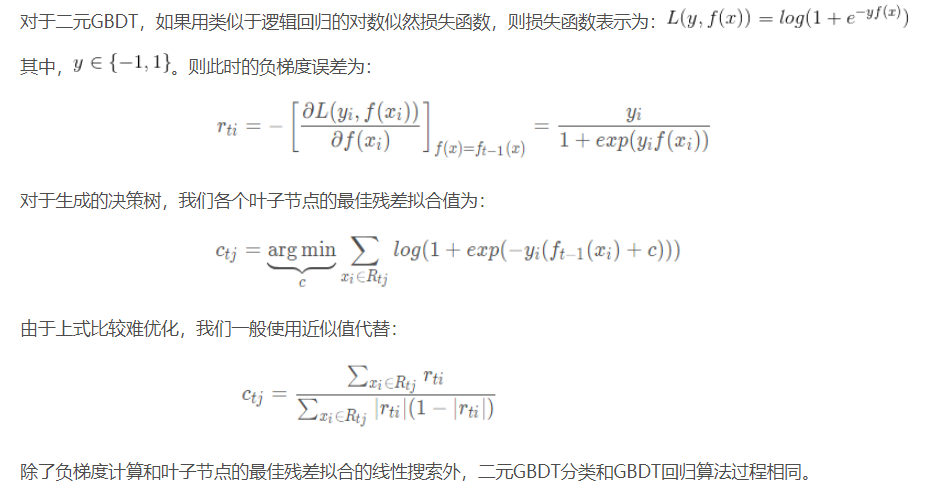
GBDT回归算法：



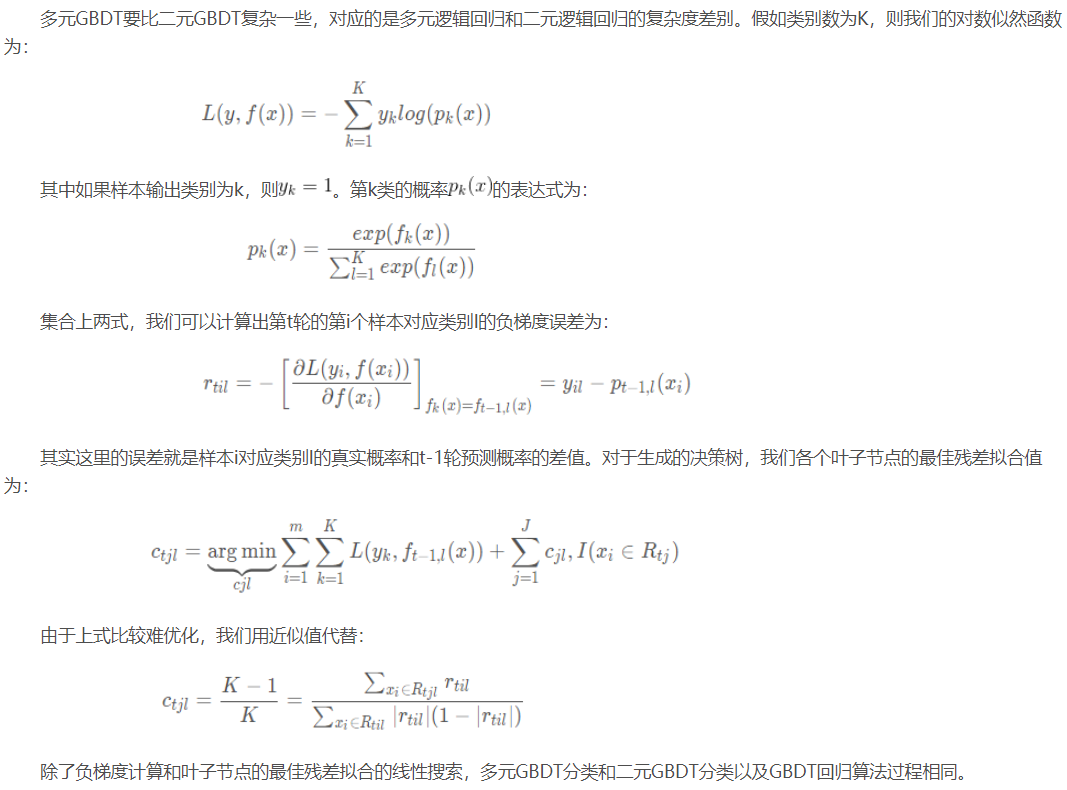
**2.2 GBDT分类算法**

GBDT分类算法在思想上和回归算法没有区别，但是由于样本输出不是连续的值，而是离散的类别，导致我们无法直接从输出类别去拟合类别输出的误差。为解决此问题，我们尝试用类似于逻辑回归的对数似然损失函数的方法，也就是说我们用的是类别的预测概率值和真实概率值来拟合损失函数。对于对数似然损失函数，我们有二元分类和多元分类的区别。

**2.2.1 二分法**



**2.2.2 多分类**

****

**2.3 正则化**

为防止过拟合需要进行正则化，GBDT的正则化主要有3种方式：learning\_rate（学习率）；subsample（子采样比例）；min\_samples\_split（叶子结点包含的最小样本数）。

（1）learning\_rate。学习率是正则化的一部分，它可以降低模型更新的速度（需要更多的迭代）。通常我们用learning\_rate和最大迭代次数一起来决定算法的拟合效果。



经验表明：一个小的学习率 (v<0.1) 可以显著提高模型的泛化能力（相比较于v=1) 。

如果学习率较大会导致预测性能出现较大波动。

（2）subsample，子采样比例。Freidman 从bagging 策略受到启发，采用随机梯度提升来修改了原始的梯度提升树算法。

* 每一轮迭代中，新的决策树拟合的是原始训练集的一个子集（而并不是原始训练集）的残差。这个子集是通过对原始训练集的无放回随机采样而来。
* 无放回抽样的子采样比例（subsample），取值为(0,1]。
* 如果取值为1，则与原始的梯度提升树算法相同，即使用全部样本。如果取值小于1，则使用部分样本去做决策树拟合。
* 较小的取值会引入随机性，有助于改善过拟合，因此可以视作一定程度上的正则化；但是会增加样本拟合的偏差，因此取值不能太低。推荐在[0.5, 0.8]之间。
* 这种方法除了改善过拟合之外，另一个好处是：未被采样的另一部分子集可以用来计算包外估计误差。因此可以避免额外给出一个独立的验证集。

（3）min\_samples\_split，叶子结点包含的最小样本数。梯度提升树会限制每棵树的叶子结点包含的样本数量至少包含m个样本，其中m为超参数。在训练过程中，一旦划分结点会导致子结点的样本数少于m，则终止划分。

**2.4 RF与GBDT之间的区别**

相同点：

* 都是由多棵树组成；
* 最终的结果都由多棵树共同决定；

不同点：

* 组成随机森林的树可以分类树也可以是回归树，而GBDT只由回归树组成；
* 组成随机森林的树可以并行生成（Bagging）；GBDT 只能串行生成（Boosting）；这两种模型都用到了Bootstrap的思想；
* 随机森林的结果是多数表决的，而GBDT则是多棵树加权累加之和；
* 随机森林对异常值不敏感，而GBDT对异常值比较敏感；
* 随机森林是减少模型的方差，而GBDT是减少模型的偏差；
* 随机森林不需要进行特征归一化。而GBDT则需要进行特征归一化；
* 随机森林对训练集一视同仁权值一样，GBDT是基于权值的弱分类器的集成；

**2.5 GBDT优缺点**

优点：

* 可以灵活处理各种类型的数据，包括连续值和离散值。
* 使用了一些健壮的损失函数，对异常值的鲁棒性非常强。比如 Huber损失函数和Quantile损失函数。
* 充分考虑的每个分类器的权重。

缺点：

* 由于弱学习器之间存在依赖关系，难以并行训练数据。不过可以通过自采样的SGBT来达到部分并行。

**2.6 应用场景**

* GBDT 可以适用于回归问题（线性和非线性）；
* GBDT 也可用于二分类问题（设定阈值，大于为正，否则为负）和多分类问题。