决策树

1. **种类：**
2. ID3：基于信息增益（略）
3. C4.5：基于信息增益比（略）
4. CART：基于基尼系数

CART分类树算法使用基尼系数来代替信息增益比，基尼系数代表了模型的不纯度，基尼系数越小，则不纯度越低，特征越好。这和信息增益(比)是相反的。

CART分类树算法每次仅仅对某个特征的值进行二分，而不是多分，这样CART分类树算法建立起来的是二叉树，而不是多叉树。这样一可以进一步简化基尼系数的计算，二可以建立一个更加优雅的二叉树模型。

对于CART分类树连续值的处理问题，其思想和C4.5是相同的，都是将连续的特征离散化。唯一的区别在于在选择划分点时的度量方式不同，C4.5使用的是信息增益比，则CART分类树使用的是基尼系数。

1. CART分类树建立算法的具体流程

算法输入是训练集D，基尼系数的阈值，样本个数阈值。

输出是决策树T。

我们的算法从根节点开始，用训练集递归的建立CART树。

1) 对于当前节点的数据集为D，如果样本个数小于阈值或者没有特征，则返回决策子树，当前节点停止递归。

2) 计算样本集D的基尼系数，如果基尼系数小于阈值，则返回决策树子树，当前节点停止递归。

3) 计算当前节点现有的各个特征的各个特征值对数据集D的基尼系数，对于离散值和连续值的处理方法和基尼系数的计算见第二节。缺失值的处理方法和上篇的C4.5算法里描述的相同。

缺失值处理方法：

对于第三个缺失值处理的问题，主要需要解决的是两个问题，一是在样本某些特征缺失的情况下选择划分的属性，二是选定了划分属性，对于在该属性上缺失特征的样本的处理。

　　对于第一个子问题，对于某一个有缺失特征值的特征A。C4.5的思路是将数据分成两部分，对每个样本设置一个权重（初始可以都为1），然后划分数据，一部分是有特征值A的数据D1，另一部分是没有特征A的数据D2. 然后对于没有缺失特征A的数据集D1来和对应的A特征的各个特征值一起计算加权重后的信息增益比，最后乘上一个系数，这个系数是无特征A缺失的样本加权后所占加权总样本的比例。

　　对于第二个子问题，可以将缺失特征的样本同时划分入所有的子节点，不过将该样本的权重按各个子节点样本的数量比例来分配。比如缺失特征A的样本a之前权重为1，特征A有3个特征值A1,A2,A3。 3个特征值对应的无缺失A特征的样本个数为2,3,4.则a同时划分入A1，A2，A3。对应权重调节为2/9,3/9, 4/9。

4) 在计算出来的各个特征的各个特征值对数据集D的基尼系数中，选择基尼系数最小的特征A和对应的特征值a。根据这个最优特征和最优特征值，把数据集划分成两部分D1和D2，同时建立当前节点的左右节点，做节点的数据集D为D1，右节点的数据集D为D2.

5) 对左右的子节点递归的调用1-4步，生成决策树。

对于生成的决策树做预测的时候，假如测试集里的样本A落到了某个叶子节点，而节点里有多个训练样本。则对于A的类别预测采用的是这个叶子节点里概率最大的类别。

**三、CART树算法的剪枝**

CART回归树和CART分类树的剪枝策略除了在度量损失的时候一个使用均方差，一个使用基尼系数，算法基本完全一样，这里我们一起来讲。

　　由于决策时算法很容易对训练集过拟合，而导致泛化能力差，为了解决这个问题，我们需要对CART树进行剪枝，即类似于线性回归的正则化，来增加决策树的返回能力。但是，有很多的剪枝方法，我们应该这么选择呢？CART采用的办法是后剪枝法，即先生成决策树，然后产生所有可能的剪枝后的CART树，然后使用交叉验证来检验各种剪枝的效果，选择泛化能力最好的剪枝策略。

也就是说，CART树的剪枝算法可以概括为两步，第一步是从原始决策树生成各种剪枝效果的决策树，第二部是用交叉验证来检验剪枝后的预测能力，选择泛化预测能力最好的剪枝后的数作为最终的CART树。

**四、决策树的优缺点**

首先我们看看决策树算法的优点：

　1）简单直观，生成的决策树很直观。

　2）基本不需要预处理，不需要提前归一化，处理缺失值。

　3）使用决策树预测的代价是*O*(*log*2*m*)O(log2m)。 m为样本数。

　4）既可以处理离散值也可以处理连续值。很多算法只是专注于离散值或者连续值。

　5）可以处理多维度输出的分类问题。

　6）相比于神经网络之类的黑盒分类模型，决策树在逻辑上可以得到很好的解释

　7）可以交叉验证的剪枝来选择模型，从而提高泛化能力。

　8） 对于异常点的容错能力好，健壮性高。

我们再看看决策树算法的缺点:

　1）决策树算法非常容易过拟合，导致泛化能力不强。可以通过设置节点最少样本数量和限制决策树深度来改进。

　2）决策树会因为样本发生一点点的改动，就会导致树结构的剧烈改变。这个可以通过集成学习之类的方法解决。

　3）寻找最优的决策树是一个NP难的问题，我们一般是通过启发式方法，容易陷入局部最优。可以通过集成学习之类的方法来改善。

　4）有些比较复杂的关系，决策树很难学习，比如异或。这个就没有办法了，一般这种关系可以换神经网络分类方法来解决。

　5）如果某些特征的样本比例过大，生成决策树容易偏向于这些特征。这个可以通过调节样本权重来改善。

五、算法比较

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 算法 | 支持模型 | 树结构 | 特征选择 | 连续值处理 | 缺失值处理 | 剪枝 |
| ID3 | 分类 | 多叉树 | 信息增益 | 不支持 | 不支持 | 不支持 |
| C4.5 | 分类 | 多叉树 | 信息增益比 | 支持 | 支持 | 支持 |
| CART | 分类、回归 | 二叉树 | 基尼系数、均方差 | 支持 | 支持 | 支持 |

CART回归树在剪枝的度量损失的时候是基于均方差，而CART分类树是基于基尼系数。