**决策树**

启发式方法，近似求解这一最优化问题，这样得到的决策树是次最优的

决策树学习算法包括特征选择，决策树的生成，决策树的剪枝，决策树的生成对应于模型的局部选择，只考虑局部最优，而决策树的剪枝对应于模型的全局选择，考虑的是全局最优。

可以认为是if--then规则的集合，也可以认为是定义在特征空间与类空间上的条件概率分布

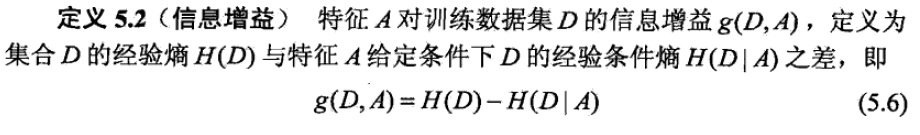
这个条件概率分布可以表示为P(Y|X),X表示特征空间，Y表示类空间，决策树分类时将该结点的实例强行分配到条件概率较大的那类当中去

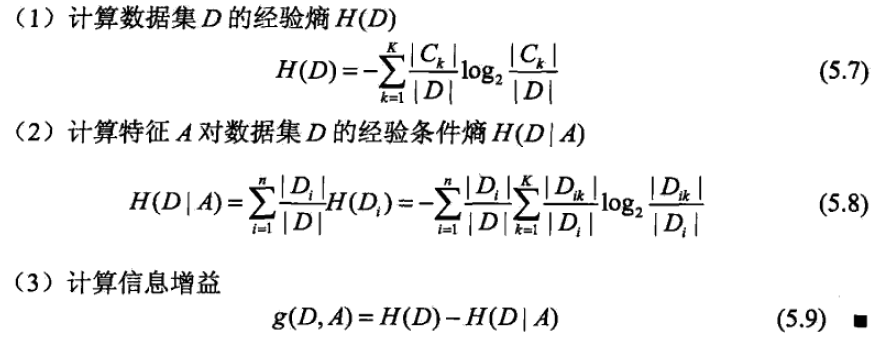
决策树的损失函数(目标函数)：正则化的极大似然函数，因为从所有可能的决策树中选取最优的决策树是NP完全问题，所以现实中决策树算法通常采用

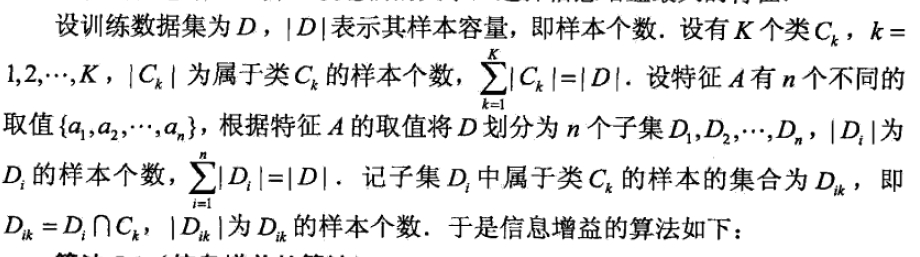
特征选择：

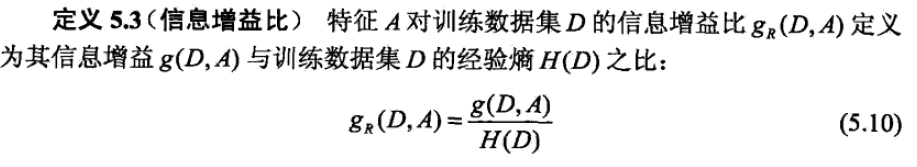
选择准则：信息增益（互信息）和信息增益比，基尼系数

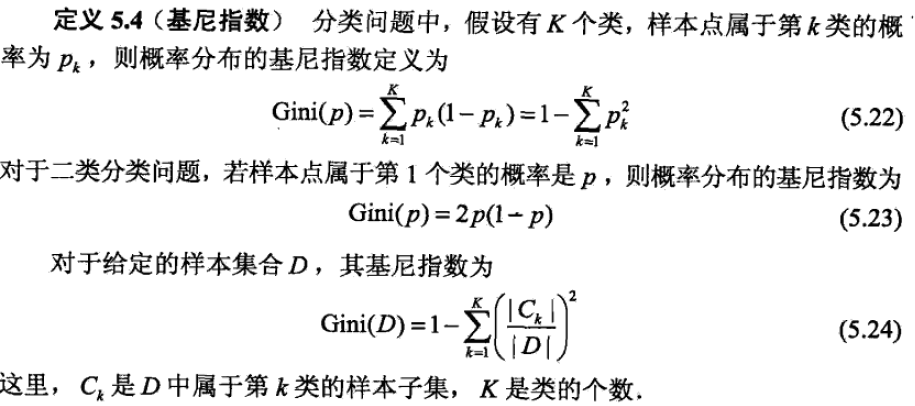
熵就是信息的

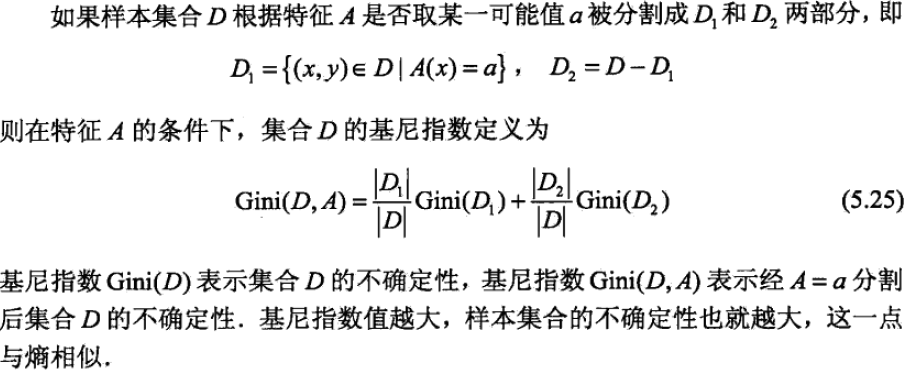












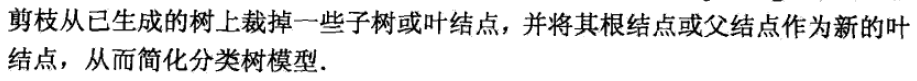
决策树的生成

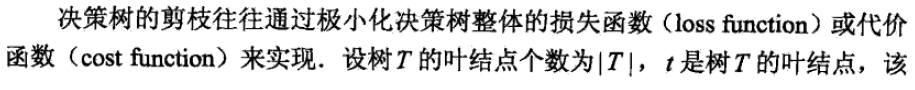
ID3算法：（只可分类）特征选择准则：信息增益，只有树的生成，容易过拟合

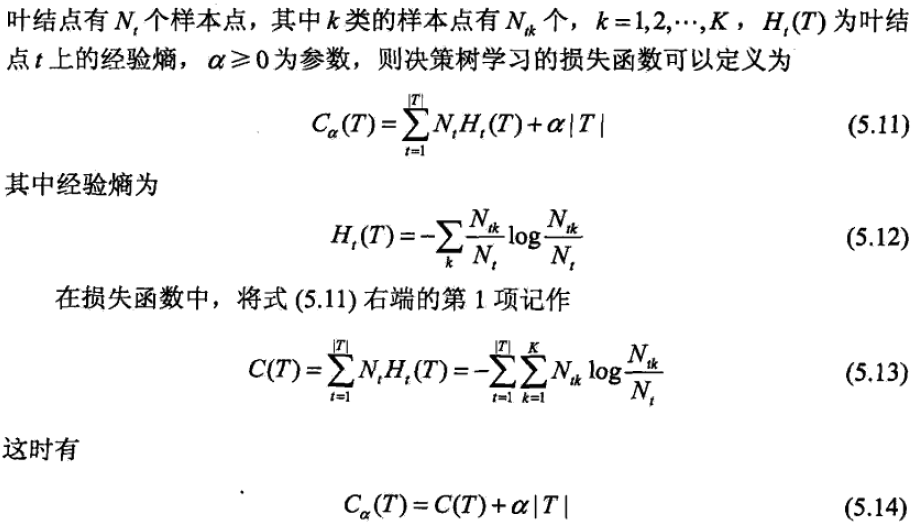
C4.5算法：（只可分类）特征选择准则：信息增益比

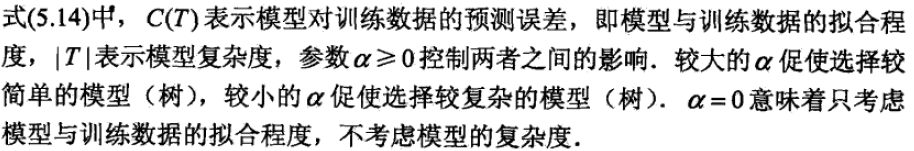
CART算法：生成二叉树，（可分类可回归）分类树的特征选择准则：基尼系数最小化

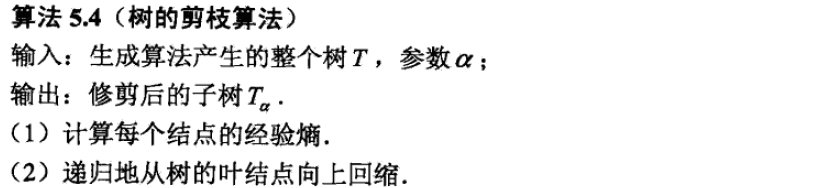
决策树的剪枝

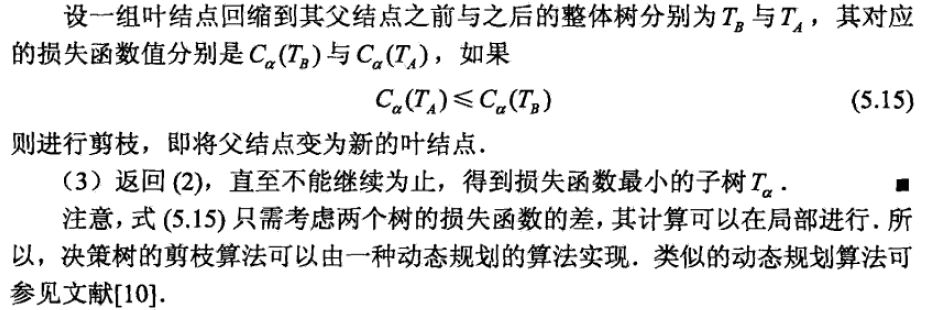




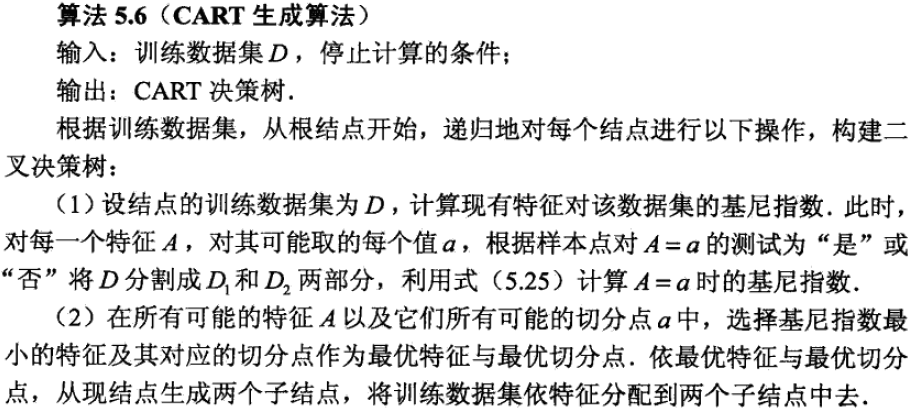


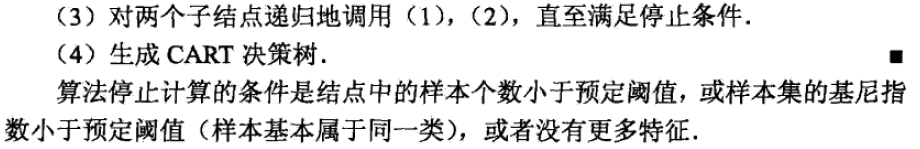


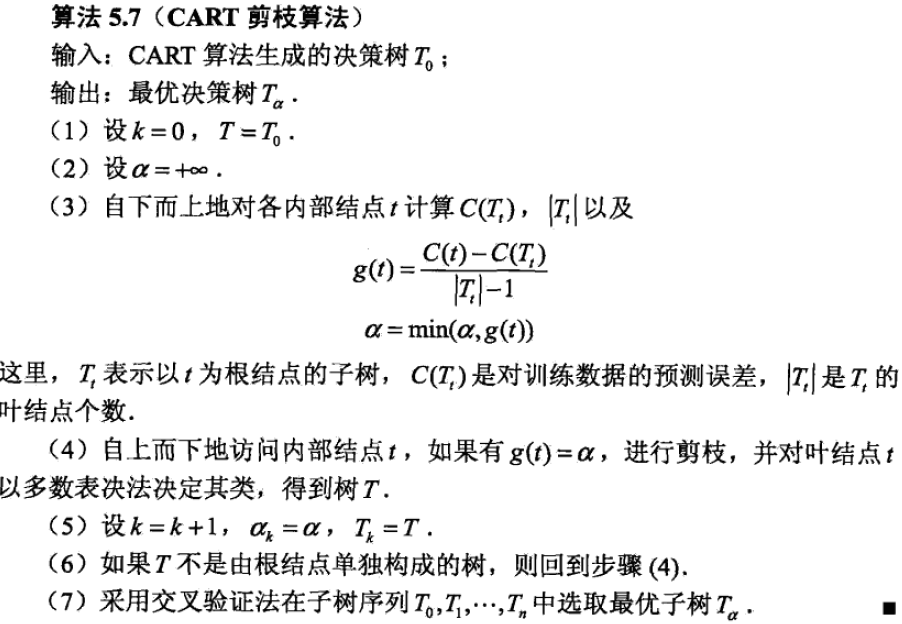




CART树的回归树的生成，连续值还是需要进行离散化处理，然后根据基尼系数找出最优特征和最优切分点







注意事项：

**连续值属性的改进**

相对于那些离散值属性，分类树算法倾向于选择那些连续值属性，因为连续值属性会有更多的分支，熵增益也最大。算法需要克服这种倾向。还记得前面讲得如何克服分类树算法倾向于离散值较多的离散属性么？对了，我们利用增益率来克服这种倾向。增益率也可以用来克服连续值属性倾向。增益率作为选择属性的依据克服连续值属性倾向，这是没有问题的。但是如果利用增益率来选择连续值属性的分界点，会导致一些副作用。分界点将样本分成两个部分，这两个部分的样本个数之比也会影响增益率。根据增益率公式，我们可以发现，当分界点能够把样本分成数量相等的两个子集时（我们称此时的分界点为等分分界点），增益率的抑制会被最大化，因此等分分界点被过分抑制了。子集样本个数能够影响分界点，显然不合理。因此在决定分界点是还是采用增益这个指标，而选择属性的时候才使用增益率这个指标。这个改进能够很好得抑制连续值属性的倾向。当然还有其它方法也可以抑制这种倾向，比如MDL，有兴趣的读者可以自己阅读相关文章

**处理缺失属性**

i)当开始决定选择哪个属性用来进行分支时，如果有些训练样本缺失了某些属性值时该怎么办？

对于问题i),计算属性a的增益或者增益率时，如果有些样本没有属性a，那么可以有这么几种处理方式：

(1)忽略这些缺失属性a的样本。

(2)给缺失属性a的样本赋予属性a一个均值或者最常用的的值。

(3)计算增益或者增益率时根据缺失属性样本个数所占的比率对增益/增益率进行相应的“打折”。

(4)根据其他未知的属性想办法把这些样本缺失的属性补全。

ii)一个属性已被选择，那么在决定分支的时候如果有些样本缺失了该属性该如何处理？

对于问题ii)，当属性a已经被选择，该对样本进行分支的时候，如果有些样本缺失了属性a,那么：

(1)忽略这些样本。

(2)把这些样本的属性a赋予一个均值或者最常出现的值，然后再对他们进行处理。

(3)把这些属性缺失样本，按照具有属性a的样本被划分成的子集样本个数的相对比率，分配到各个子集中去。至于哪些缺失的样本被划分到子集1，哪些被划分到子集2，这个没有一定的准则，可以随机而动。

(4)把属性缺失样本分配给所有的子集，也就是说每个子集都有这些属性缺失样本。

(5)单独为属性缺失的样本划分一个分支子集。

(6)对于缺失属性a的样本，尝试着根据其他属性给他分配一个属性a的值，然后继续处理将其划分到相应的子集。

iii)当决策树已经生成，但待分类的样本缺失了某些属性，这些属性该如何处理？

(1)如果有单独的确实分支，依据此分支。

(2)把待分类的样本的属性a值分配一个最常出现的a的属性值，然后进行分支预测。

(3)根据其他属性为该待分类样本填充一个属性a值，然后进行分支处理。(F)在决策树中属性a节点的分支上，遍历属性a节点的所有分支，探索可能所有的分类结果，然后把这些分类结果结合起来一起考虑，按照概率决定一个分类。

(4)待分类样本在到达属性a节点时就终止分类，然后根据此时a节点所覆盖的叶子节点类别状况为其分配一个发生概率最高的类。

决策树的优点

* 使用一个白匣子模型，易于理解和解释，树可视化
* 只需要很少的数据预处理，但是不支持缺省值
* 决策树的数据训练和预测的代价是log级
* 可以处理数值型和连续型
* 可以处理多输出问题
* 效果好

决策树的缺点

* 过拟合问题，需要进行剪枝
* 决策树不稳定，数据很小的变化都会导致产生完全不同的树，受噪声影响较大，可以通过决策树的增强方法。如随机森林，GBDT等
* 由于采用启发式方法，得到的决策树不是全局最优的，可以通过决策树的增强方法。如随机森林，特征和样本都是随机的
* There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.
* Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

影响决策树的参数：

**DecisionTreeClassifier**(*criterion='gini'*, *splitter='best'*, *max\_depth=None*, *min\_samples\_split=2*,*min\_samples\_leaf=1*, *min\_weight\_fraction\_leaf=0.0*, *max\_features=None*, *random\_state=None*,*max\_leaf\_nodes=None*, *class\_weight=None*, *presort=False*)