第四章：决策树

(不全，因个人原因稍后会上传更全面的)

【引言】决策树是基于树结构来进行决策的，可以类比于常见的 if 条件语句。一般对于二分类，其判断过程就被称为“决策”或“判定”的过程。而前一轮的决策结果便是下一轮的必然前提，最终的决策结论对应了我们所希望的判定结果。

1、基本流程

【概念】：一颗决策树包含一个根结点、若干个内部结点和若干个叶结点。叶结点对应于决策结果，其他每个结点则对应于一个属性测试；每个节点包含的样本集合根据属性测试的结果被划分到子节点中；根节点包含样本全集。从根结点到每个叶节点的路径对应了一个判定测试序列。

决策树学习的目的是为了产生一颗泛化性能强，即处理未见示例能力强的决策树，其基本流程遵循简单且直观的“分而治之”原则。

生成决策树的过程是递归过程。其逻辑如下：

（1）生成结点 node，对所有样本进行属性判别，找到划分能力最强的属性特征（类别标记样本数最多的属性），以该属性作为 node 生成根结点；
（2）此时所有样本应该已被分为两类（属于该属性的和不属于改属性的）。此时生成 node2，分别对两类重复（1）中的划分过程。
（3）以此类推，最后直到只能划分出单一类别为止。
在决策树的基本算法中，又三种情形会导致递归返回：
（1）当前结点包含的样本全属于同一个类别，无需划分；
（2）当前属性集为空，或是所有样本在所有属性上取值相同，无法划分；
（3）当前结点包含的样本集合为空，不能划分。

纵观以上三个过程，都遵循了少数服从多数原则，无论是（2）或是（3），最终都以多数类或父类作为最后的分类依据。

2、划分选择

很显然，决策树算法中最关键的部分是选出划分能力最强的属性作为结点的决策准则。我们希望决策树的分支结点所包含的样本尽可能属于同一类别，即结点的“纯度”越来越高。

2.1 信息增益

信息熵：是度量样本集合纯度最常用的一种指标。假设当前样本集合 D 中第 k 类样本所占的比例为 pk(k=1, 2, …|Y|)，则 D 的信息熵定义为（其中，pk 取不到 1，因为此时一定在上一个结点时已经分类完毕，继承父结点所属的类别）：

由于决策树是根据离散属性 a 来进行划分的，因此有多少的离散属性一般就会产生多少个分支结点，而可以计算每一个分支结点（划分属性）上的信息上 Dv，其中 V 表示离散属性 a 可能的取值{a1, a2, …, aV}。

可以看出，该式的含义在于：信息增益=样本空间的信息熵 - （离散属性 a 的 v 各分支对应的信息熵 * 对应样本空间中权重）之和 。

2.2 增益率

【引言】我们自动忽略了编号这一列，如果将编号也算作是特征属性的一种，那么就不是上述的 6 类特征，而是 7 类了。再对编号这一类单独求信息熵，计算出来的信息增益为 0.998 远大于其他属性划分出所得到得信息增益，这说明以编号划分得分支结点纯度达到最大，不具备泛化能力，即每一个编号都对应一个类别。因此，信息增益准则可能会因为偏好的问题产生不利的影响。

因此，在使用信息增益为理论基础的 ID3 算法之上，C4.5 决策树算法采用了”增益率“来选择最优划分属性。定义为：
称为属性 a 的固有值。从公式中不难看出，属性 a 的可能取值数目越多（V 越大），则 IV(a) 的值通常会越大。

增益率准则对课取值数目较少的属性有所偏好。因此，C4.5 算法不是直接选择增一律最大的候选划分属性，而是使用了启发式规则：先从候选划分属性中找出信息增益高于平均水平的属性，再从中选择增益率最高的。

• 启发式算法：在一个随机的群体寻优过程中，个体能够利用自身或全局的经验来制定各自的搜索过程。（一人经商赚钱，全村下海经商）

2.3 基尼指数

CART 决策树就是采用基尼指数来选择划分属性。数据集 D 的纯度用基尼值来度量：
Gini(D)反映了从数据集 D 中随机抽取两个样本，其类别标记不一致的概率。因此，Gini(D) 越小，数据集 D 的纯度越高。

因此在属性集合 A 种选择划分后基尼指数最小的属性作为最有划分属性。

【小结】:
注：信息熵永远是找最小的，因为越小纯度越高
1
ID3——信息增益<找最大>
C4.5——增益率<找最高>

CART（适用于回归和分类）——基尼指数<找最小>

3、剪枝处理

剪枝是决策树学习算法对付过拟合的主要手段，目的是为了尽可能正确分类训练样本。因为划分结点的过程有时会造成决策树分支过多产生过拟合（划分过于精细）。以至于把训练集自身的一些特点当作所有数据都具有的一般性质（如前述的“编号”问题）。

【预剪枝】（降低过拟合，容易欠拟合；时间、计算开销小）：对每个结点在划分前先进性估计，若当前结点的划分不能带来决策树泛化性能的提升，则停止划分，并将当前结点标记为叶结点。预剪枝可以降低过拟合风险，减少决策树的训练时间开销和测试时间开销。但其基于贪心策略的本质禁止分支展开，从而导致了预剪枝容易出现欠拟合的风险。（详见书上 81 页例子已详细标记）

【后剪枝】（降低欠拟合，泛化性能强；时间、计算开销大）：先从训练集生成一棵完整的决策树，然后自底向上对非叶结点进行考察，若将该结点对应的子树替换为叶结点能带来决策树泛化性能提升，则将该子树替换为叶结点。后剪枝通常比预剪枝决策树保留了更多的分支，而后剪枝决策树欠拟合风险小，泛化性能往往优于预剪枝决策树。但是时间开销会大许多（因为需要生成一颗完全决策树之后再剪）。

4、连续与缺失值

4.1 连续值处理

前述的许多算法都是基于离散属性来生成决策树，但生活中常常会遇到连续属性，因此需要使用连续属性离散化技术如二分法，这也正是 C4.5 算法所采用的机制。

给定样本空间 D，其中连续属性 a 有 n 个不同的取值 a={a1, a2, …, an}，设置阈值 t，将 a 取值不大于 t 的样本放置在集合中，而将 a 取值大于 t 的样本放置在集合中。

【连续属性离散化】：显然对于相邻属性取值 ai 和 ai+1，在区间[ai，ai+1)间的任意值所产生

的划分结果相同，因此我们在相邻划分区域中考察 n-1 个元素，取中位点。可见，Ta 是存放中位点（划分点）的一个集合。这样的过程就是连续属性离散化技术。
我们规定，选择 Gain 最大化的划分点。

step0:求数据集本身的信息熵 Ent(D)
step1:将特征 a 下属的所有值按从小到大顺序排；
step2:两两相邻，求中位值；
step3:用程序求解每个中位值为划分的所有信息增益，取最大的。
划分的过程递归进行，若当前结点划分属性为连续属性，那么该属性还可作为其后代结点继续划分属性。

4.2 缺失值处理

现实任务种常会遇到不完整样本，包括样本的某些属性值缺失，尤其在属性数目较多的情况下，往往会有大量样本出现缺失值。

因此赋予了每个样本权重
若给定划分属性，样本在该属性上的值缺省：
①若划分属性 a 上的取值已知，则将 x 划入与其取值对应的子结点，且样本权值在子结点中保持为
②若划分属性 a 上的取值未知，则将 x 同时划入所有子结点，且样本权值在于属性值 a^v 对应的子结点中调整为
很显然，这就是让同一个样本以不同的概率划入到不同的子结点中去。

选择取得最大信息增益的属性用于对根结点进行划分。权重的计算按照无缺省值的子集数目来确定，而非原来的样本子集。

①【注】关于有缺省值的样本：一般的样本（无缺省），会根据划分条件，进入所对应的划分选项（如：与色泽——青绿对应的就会进入青绿这一个分支）。而对于有缺省值的样本，它会进入所有分支（例如：色泽有三个分支：青绿、浅白、乌黑，则该缺省值的样本会在这三个分支中都走一遍）。

②【注】关于样本的权重：以｛纹理：清晰<7>、稍糊<5>、模糊<3>｝为例。对于无缺省值的子集而言，清晰的权重
。显然，有缺省的样本进入并不会影响权重本身的变化，其只能顺应不同分支下的权重大小而已。

5、多变量决策树

5.1 引言

将属性想象成坐标轴，那么对于多个属性的单个样本，就是多维空间中的一个数据点。而对样本进行分类就意味着在坐标空间中寻找不同类别样本之间的分类边界，决策树所形成的分类边界有一个明显的特点就是轴平行，即它的分类边界由若干个于坐标轴平行的分段组成。这样的分类边界使得学习结果有较好的可解释性，因为每一段划分都直接对应了某个属性取值（就像在某一个阶段使用了控制变量法）。

但是对于多属性划分时，分类任务往往会非常复杂，这样就需要进行大量的属性测试，并且会产生很大的时间开销。

多变量决策树便是为了解决这一问题而提出的方法。类比于传统的阶梯型决策边界，多变量决策树可以对划分实现斜划分，其原理在于非叶结点不再仅对单个属性，而是对属性的线性组合进行测试。

试想，原来的分支结点的作用是根据属性的不同取值（如“色泽”：青绿、浅白、乌黑）将所有样本对应分到不同的分支中去，而对于多变量决策树，分支结点本身就是一个线性回归的判别函数：
很显然，对于属性而言，不能再是离散的属性值，而应该是连续的属性值。

6、小结

尝试了解 C4.5Rule 算法，它是一个将 C4.5 决策树转化为符号规则的新方法。其决策树的每一个分支可以容易地重写为一条规则，但 C4.5Rule 算法在转化过程中会进行规则前件合并、删减等操作，因此最终规则集的泛化性能甚至远优于原决策树。

【本章回顾及关键字】:
信息熵：Ent(D)：熵越小，纯度高
信息增益：Gain(D, a)：增益越大，划分指标
增益率：Gain_ratio（D, a）：增率越高，划分越好
基尼指数：Gini（D）：基尼越小纯度越高
剪枝处理（预剪枝、后剪枝）：先预、剪后构树；先构数后算、剪
连续属性离散化：相邻离散值取中位点
缺省值处理：无缺值子集、权重
多变量决策树：条件语句函数化，离散属性变量代入线性计算。

往往剪枝方法和程度对决策树泛化性能的影响相当显著，甚至在某些带有噪声的数据中能将泛化性能提升 25%。

6.1 多变量决策树算法

主要有 OC1 算法以及 Brodley and Utgoff 提出的一些算法。

【基本原理】

OC1：先贪心地寻找每个属性地最优权值，在局部优化的基础上对分类边界进行随机扰动寻找更好的边界。

B&U 相关算法：直接引入了线性分类器学习的最小二乘法

叶结点上的神经网络：某些算法试图在叶结点熵嵌入神经网络，从而结合两种学习机制的优势（e.g 感知机树——在决策树每个叶结点熵训练一个感知机）

6.2 增量学习

即在收到新样本之后对已学得的模型进行调整，而不用完全重新学习。主要机制是通过调整分支路径熵的划分属性此许来对树进行部分重构，如 ID4 算法、ID5R、ITI 等。

增量学习可以有效地降低每次接收到新样本后地训练时间开销，但经过多次步骤增量学习之后的模型会与基于全部数据训练而得的模型有较大差别。