决策树-学习笔记

1. 模型思想

决策树（decision tree）是一个树结构。其每个非叶节点表示一个特征属性，每个分支代表这个特征属性在某个值域上的输出，而每个叶节(leaf)点存放一个类别。

使用决策树进行决策的过程就是从根节点开始，递归地选择分裂特征，并根据该特征对训练数据进行分割，并按照其值选择输出分支，直到叶子节点，将叶子节点存放的类别作为决策结果。

优点：

1. 复杂度低；
2. 解释性强；

缺点：

1. 过拟合；
2. 缺失值处理；
3. 忽略特征与特征间的相关性**(?)**

关键步骤：

分裂特征，所谓分裂特征就是在某个节点处按照某一特征属性的不同划分构造不同的分支，其目标是让各个分裂子集尽可能地“纯”。尽可能“纯”就是尽量让一个分裂子集中待分类项属于同一类别。

2. 熵与条件熵

2.1 熵

熵$(Entropy)$，表示随机变量不确定性的度量。熵越大，一个变量的不确定性就越大（也就是可能的取值越多），把它搞清楚所需要的信息量也就越大。熵越小，说明系统越有序，携带的信息就越多。

随机变量X的熵定义为：

$H(X)=-\sum_i^n p_i log(p_i), pi是D中任意元组属于类C_i非零概率$

上式中，**若**$p_i=0$，**则定义** $0log0=0$ 。

有定义可知，熵只依赖于X的分布，而与X的取值无关。故也可：

$H(p)=-\sum_i^n p_i log(p_i)$

若随机变量X只取两个值，例如“0”和“1”，则熵为：

$H(p)=-p\ log_2 p-(1-p)log_2(1-p)$

当$p=0$或$p=1$时$H(p)=0$，随机变量完全没有不确定性。 当$p=0.5$时，$H(p)=1$，熵取值最大，随机变量不确定性最大。

2.2 条件熵

随机(X,Y)，其联合概率分布为：

$P(X=x_i,Y=y_i)=p_{ij},\ i=1,2,\dots,n\ j=1,2,\dots,m$

条件熵 $H(Y|X)$，表示在已知随机变量 X 的条件下随机变量 Y 的不确定性。

随机变量 X 给定的条件下随机变量 Y 的条件熵 $H(Y|X)$，定义为 X 给定条件下 Y 的条件概率分布的熵对 X 的期望：

$H(Y|X)=\sum_{i=1}^{n}P(X=x_i)H(Y|X=x_i)\quad i=1,2,\dots,n$

3. 信息增益、信息增益比、基尼指数

3.1 信息增益

信息增益$(information\ gain)$表示，得知特征 X 的信息而使得类 Y 的信息的不确定性减少的程度。

特征 A 对训练集 D 的信息增益 $g(D,A)$，定义为集合 D 的经验熵 $H(D)$ 与特征 A 给定条件下 D 的经验条件熵 $H(D|A)$之差，即

$g(D,A)=H(D)-H(D|A)$

一般地，熵 $H(Y)$ 与条件熵 $H(Y|X)$ 之差称为互信息（$mutual\ information$），决策树中的信息增益等价于训练数据集中类 Y 与特征 X 的互信息。

决策树中，经验熵 $H(D)$ 表示对数据集 D 进行分类的不确定性。而条件经验熵 $H(D|A)$ 表示在特征 A 给定的条件下对数据集 D 进行分类的不确定性。**那么，它们的差，即信息增益，就表示由于特征A而使得对数据集D进行分类的不确定性的减少的程度。信息增益越大的特征，具有越强的分类能力。**

设训练数据集D，$|D|$为其样本容量，即样本个数。设有K个类C_k，$|C_k|$为属于类C_k的样本个数，$\sum_{k=1}^{K}|C_k|=|D|$。设特征A有n个不同的取值$\{a_1,a_2,\dots,a_n\}$，根据特征A的取值将D划分为n个子集$D_1,D_2,\dots,D_n$，$|D_i|$为D_i的样本个数。

> **信息增益算法：**
>
> 1. 计算数据集D的经验熵$H(D)$
>
> $H(D)=-\sum_{k=1}^{K}\frac{|C_k|}{|D|}log_2\frac{|C_k|}{|D|}$
>
> 2. 计算特征A对数据集D的经验条件熵$H(D|A)$
>
> $H(D|A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}(-\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}log_2\frac{|D_{ik}|}{|D_i|})$
>
> 3. 计算信息增益
>
> $g(D,A)=H(D)-H(D|A)$

3.2 信息增益比

信息增益比（$information\ gain\ ratio$），特征A对训练集D的信息增益比$g_R(D,A)$定义为其信息增益$g(D,A)$与训练集D关于特征A的值的熵$H_A(D)$之比，即

$g_R(D,A)=\frac{g(D,A)}{H_A(D)}$

其中，$H_A(D)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}log_2\frac{|D_i|}{D}$，$n$是特征$A$的取值个数。

区别：以信息增益作为选择特征的准则，存在偏向于选择取值较多(n较多)的特征的问题。信息增益比可以校正这一问题。

3.3 基尼指数

假设有K个类别，样本点属于第k类的概率为p_k，则概率分布的基尼指数定义为

$Gini(p) = \sum_{k=1}^{K} p_k(1-p_k) = 1 - \sum_{k=1}^{K} p_k^2$

对于二分类问题，$K=2$，若样本点属于第1个类的概率为p，则概率的基尼指数为

$Gini(p) = 2p(1-p)$

对于给定的样本集合D，其基尼指数为

$Gini(D) = 1 - \sum_{k=1}^{K}(\frac{|D_k|}{|D|})^2$

如果样本集合D根据特征A是否取某一可能值a被分割成D_1和D_2两部分，即

$D_1 = \{(x,y) \in D \mid A(x) = a\}, \quad D_2 = D - D_1$

则在特征A的条件下，集合D的基尼指数定义为

$Gini(D,A) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2)$

基尼指数$Gini(D)$表示集合D的不确定性，**基尼指数$Gini(D,A)$表示经$A=a$分割后集合D的不确定性。**基尼指数值越大，样本集合的不确定性也就越大，这一点与熵相似。

4. 剪枝

决策树剪枝的基本策略有“预剪枝”(prepruning)和“后剪枝”(post-pruning) [Quinlan, 1993]. 预剪枝是指在决策树生成过程中, 对每个结点在划分前先进行估计, 若当前结点的划分不能带来决策树泛化性能提升, 则停止划分并将当前结点标记为叶结点; 后剪枝则是先从训练集生成一棵完整的决策树, 然后自底向上地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.

如何判断决策树泛化性能是否提升呢? 这可使用 2.2 节介绍的性能评估方法. 本节假定采用留出法, 即预留一部分数据用作“验证集”以进行性能评估. 例如对表 4.1 的西瓜数据集 2.0, 我们将其随机划分为两部分, 如

4.1 预剪枝

- 优点：
    1. 降低过拟合；
    2. 节省时间；
- 缺点：
    1. 有些分支当前分支虽可能无法提升泛化能力（验证集表现），但以其分支为基础的后续分支可能会显著提升泛化能力；
    2. 基于“贪心”的本质，禁止展开分支，容易欠拟合；

呢? 预剪枝要对划分前后的泛化性能进行估计.

在划分之前, 所有样例集中在根结点. 若不进行划分, 则根据算法 4.2 第 6 行, 该结点将被标记为叶结点, 其类别标记为训练样例数最多的类别, 假设我们

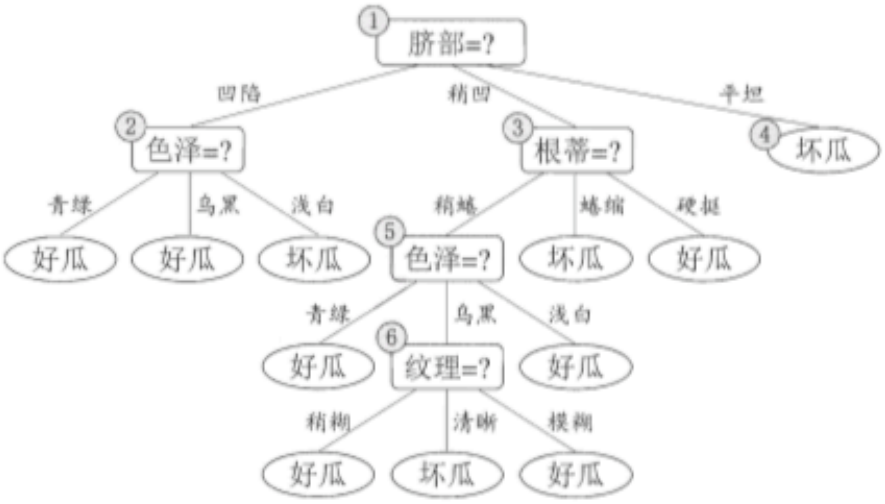


图 4.5 基于表 4.2 生成的未剪枝决策树

图 4.6 基于表 4.2 生成的预剪枝决策树

将这个叶结点标记为"好瓜". 用表 4.2 的验证集对这个单结点决策树进行评估, 则编号为 $\{4,5,8\}$ 的样例被分类正确, 另外 4 个样例分类错误, 于是, 验证集精度为 $\frac{3}{7} \times 100\% = 42.9\%$.

在用属性"脐部"划分之后, 图 4.6 中的结点 ②、③、④ 分别包含编号为 $\{1,2,3,14\}$、$\{6,7,15,17\}$、$\{10,16\}$ 的训练样例, 因此这 3 个结点分别被标记为叶结点"好瓜"、"好瓜"、"坏瓜". 此时, 验证集中编号为 $\{4,5,8,11,12\}$ 的样例被分类正确, 验证集精度为 $\frac{5}{7} \times 100\% = 71.4\% > 42.9\%$. 于是, 用"脐部"进行划分得以确定.

对比图 4.6 和图 4.5 可看出, 预剪枝使得决策树的很多分支都没有“展开”, 这不仅降低了过拟合的风险, 还显著减少了决策树的训练时间开销和测试时间开销. 但另一方面, 有些分支的当前划分虽不能提升泛化性能、甚至可能导致泛化性能暂时下降, 但在其基础上进行的后续划分却有可能导致性能显著提高; 预剪枝基于“贪心”本质禁止这些分支展开, 给预剪枝决策树带来了欠拟合的风险.

4.2 后剪枝

- 判断整体数据集的精度提升；
- 优点：泛化能力优于预剪枝，欠拟合风险小；
- 缺点：自底向上遍历每一个非叶子结点进行评估，时间开销大；

后剪枝先从训练集生成一棵完整决策树, 例如基于表 4.2 的数据我们得到如图 4.5 所示的决策树. 易知, 该决策树的验证集精度为 42.9%.

后剪枝首先考察图 4.5 中的结点⑥. 若将其领衔的分支剪除, 则相当于把⑥ 替换为叶结点. 替换后的叶结点包含编号为 $\{7, 15\}$ 的训练样本, 于是, 该叶结点的类别标记为“好瓜”, 此时决策树的验证集精度提高至 57.1%. 于是, 后剪枝策略决定剪枝, 如图 4.7 所示.

然后考察结点⑤, 若将其领衔的子树替换为叶结点, 则替换后的叶结点包含编号为 $\{6, 7, 15\}$ 的训练样例, 叶结点类别标记为“好瓜”, 此时决策树验证集精度仍为 57.1%. 于是, 可以不进行剪枝.

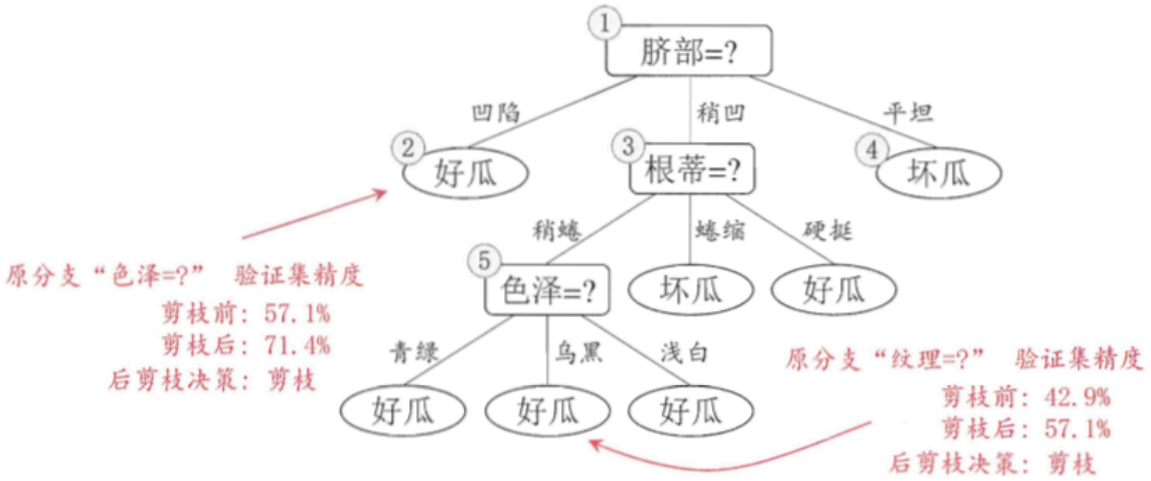


图 4.7 基于表 4.2 生成的后剪枝决策树

最终，基于后剪枝策略从表 4.2 数据所生成的决策树如图 4.7 所示，其验证集精度为 71.4%.

对比图 4.7 和图 4.6 可看出，后剪枝决策树通常比预剪枝决策树保留了更多的分支. 一般情形下，后剪枝决策树的欠拟合风险很小，泛化性能往往优于预剪枝决策树. 但后剪枝过程是在生成完全决策树之后进行的，并且要自底向上地对树中的所有非叶结点进行逐一考察，因此其训练时间开销比未剪枝决策树和预剪枝决策树都要大得多.

通过极小化决策树整体的损失函数来实现。设树T的叶结点个数为|T|，t是树T的叶结点，该叶结点有N_t个样本点，其中k类的样本点有N_{tk}个。$H_t(T)$为叶结点t上的经验熵，$\alpha \geq 0$为参数，则损失函数定义为

$$C_\alpha(T) = \sum_{t=1}^{|T|} N_T H_t(T) + \alpha|T|$$

其中，经验熵为$H_t(T) = -\sum_k \frac{N_{tk}}{N_t} log \frac{N_{tk}}{N_t}$,

此时有

$$C_\alpha(T) = C(T) + \alpha|T|$$

式中，$C(T)$表示预测误差，$|T|$表示复杂度。

剪枝，就是当α确定时，选择损失函数最小的模型，即损失函数最小的子树。决策树生成学习局部的模型，而剪枝学习整体的模型。

5. ID3算法

核心：在决策树各结点上应用信息增益准则选择特征，递归地构建决策树。

特点：

1. 只适用于分类型变量，不适用于连续型变量；

2. 无法处理缺失数据；
3. 只有树的生成，不涉及树的剪枝，所以容易过拟合；

> **ID3 算法：**
>
> 输入：训练集D，特征集A，阈值δ；
>
> 输出：决策树T；
>
> **1）** 若D中所有实例属于同一类C_k，则T为单结点树，并将类C_k作为该结点的类标记，返回T；
>
> **2）** 若特征集$A=\emptyset$，则T为单结点树，并将D中实例数最大的类C_k作为该结点的类标记，返回T；
>
> **3）** 否则，按信息增益算法计算A中各特征对D的信息增益，选择信息增益最大的特征A_g；
>
> **4）** 如果A_g的信息增益小于阈值δ，则置T为单结点树，并将D中实例数最大的类C_k作为该结点的类标记，返回T；
>
> **5）** 否则，对A_g的每一个可能的值a_i，**依$A_g=a_i$将D分割为若干个非空子集D_i，将D_i中实例数最大的类作为标记，构建子结点，**由结点及其子结点构成数T，返回T；
>
> **6）** 对第i个子结点，以D_i为训练集，以$A-\{A_g\}$（**该分支用过的特征不能再用**）为特征集，递归地调用步(1)~步(5)，得到子树T_i，返回T_i；

编程实现中的思考：

- 当特征A某个特征切割后样本为0时，如何计算经验条件熵，此时分母为0?
- 信息增益阈值设定为多少合适?
- 当两个特征的信息增益一样大时，如何选择?
- 当存在缺失值时，如何处理?

6. C4.5算法

核心：用信息增益比替代信息增益准则进行特征选择。

特点：

1. 适用于分类型和连续型变量，离散化处理。
2. 只有树的生成，不涉及树的剪枝，所以容易过拟合；

6.1 算法

> **C4.5 算法：**
>
> 输入：训练集D，特征集A，阈值δ；
>
> 输出：决策树T；
>
> **1）**若D中所有实例属于同一类C_k，则T为单结点树，并将类C_k作为该结点的类标记，返回T；
>
> **2）** 若特征集$A=\emptyset$，则T为单结点树，并将D中实例数最大的类C_k作为该结点的类标记，返回T；
>
> **3）** 否则，按信息增益比算法$g_R(D,A)=\dfrac{g(D,A)}{H_A(D)}$计算$A$中各特征对$D$的信息增益比，选择信息增益比最大的特征$A_g$；

> **4）** 如果A_g的信息增益比小于阈值δ，则置T为单结点树，并将D中实例数最大的类C_k作为该结点的类标记，返回T；
>
> **5）** 否则，对A_g的每一个可能的值a_i，**依$A_g = a_i$将D分割为若干个非空子集D_i，将D_i中实例数最大的类作为标记，构建子结点，**由结点及其子结点构成数T，返回T；
>
> **6）** 对第i个子结点，以D_i为训练集，以$A - \{A_g\}$（**该分支用过的特征不能再用**）为特征集，递归地调用步(1)~步(5)，得到子树T_i，返回T_i；

6.2 处理连续型特征

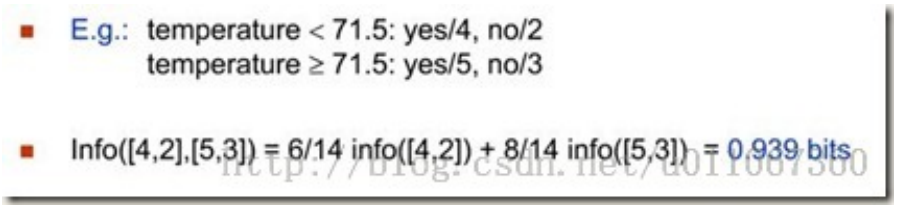
> **重点：**
>
> **1)** 对于连续特征排序，去重，依次在特征值发生改变的地方（两值均值）进行切分；
>
> **2)** 连续特征根据切分点，二元离散化；
>
> **3)** 选特征：信息增益率。选最优切分点：信息增益；

> **C4.5 处理连续型特征：**
>
> 1. 对特征的取值进行升序排序.
> 2. 两个特征取值之间的中点作为可能的分裂点，将数据集分成两部分，计算每个可能的分裂点的信息增益。优化算法就是只计算分类属性发生改变的那些特征取值.
> 3. 选择修正后**信息增益**最大的分裂点作为该特征的**最佳分裂点.**
> 4. 计算最佳分裂点的**信息增益率**作为**特征的信息增益率**。注意，此处需对最佳分裂点的信息增益进行修正：减去$log_2(N-1)/|D|$（N是连续特征的取值个数，D是训练数据数目，此修正的原因在于：**当离散属性和连续属性并存时，C4.5算法倾向于选择连续特征做最佳树分裂点**）？？？

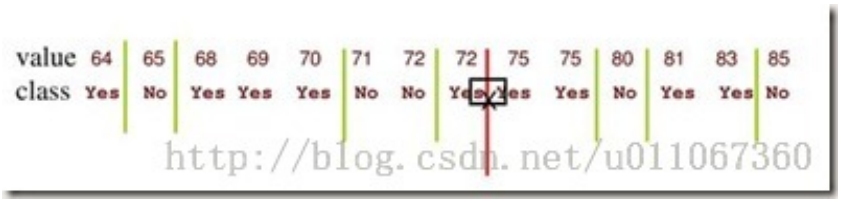
- **先把连续属性转换为离散属性再进行处理；**
- **对于连续属性先进行排序，只有在特征值发生改变的地方才需要切开，这可以显著减少运算量；**

假设该属性对应的不同的属性值一共有N个，那么总共有$N-1$个可能的候选分割阈值点，每个候选的分割阈值点的值为上述排序后的属性值链表中两两前后连续元素的中点，那么我们的任务就是从这个$N-1$个候选分割阈值点中选出一个，使得前面提到的信息论标准最大。举个例子，对play数据集，我们来处理温度属性，来选择合适的阈值。首先按照温度大小对对应样本进行排序如下：

那么可以看到有13个可能的候选阈值点，比如$middle[64,65], middle[65,68] \ldots, middle[83,85]$。那么最优的阈值该选多少呢？应该是$middle[71,72]$，如上图中红线所示。为什么呢？如下计算：



通过上述计算方式，0.939是最大的，因此测试的增益是最小的。（测试的增益和测试后的熵是成反比的，这个从后面的公式可以很清楚的看到）。根据上面的描述，我们需要对每个候选分割阈值进行增益或熵的计算才能得到最优的阈值，我们需要算$N-1$次增益或熵（对应温度这个变量而言就是13次计算）。能否有所改进呢？少算几次，加快速度。答案是可以该进，如下图：



该图中的绿线代表可能的最优分割阈值点，根据信息论知识，像$middle[72,75]$（红线所示）这个分割点，$72,75$属于同一个类，这样的分割点是不可能有信息增益的。（把同一个类分成了不同的类，这样的阈值点显然不会有信息增益，因为这样的分类没能帮上忙，减少可能性）

- **连续型特征的改进：选择分裂特征时才使用增益率，在决定划分点时还是采用增益；**

原因：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，信息增益也最大。算法需要克服这种倾向，我们利用增益率来克服这种倾向。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，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。因此在决定分界点时还是采用增益这个指标，而选择属性时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。

6.3 处理缺失值

> **C4.5还能对缺失值进行处理,处理的方式通常有3种：**
>
> - 赋上该属性最常见的值
> - 丢弃有缺失值的样本
> - 根据节点的样例上该属性值出现的情况赋一个概率,比如该节点上有10个样本,其中属性A的取值有6个为是,4个为否.那么对改节点上缺失的属性A,以0.6的概率设为是,0.4的概率设为否

7. CART算法

特点：

1. 基于基尼指数最小原则，选择分裂特征；
2. 二叉树；
3. 可用于回归和分类；

7.1 树的生成

> **CART 分类算法：**
>
> 输入：训练集D，停止计算条件；
>
> 输出：CART决策树；
>
> 从跟结点开始，递归地对每个结点进行以下操作，构建二叉决策树：

> **1）** 设结点的训练数据集为D，计算现有所有特征**（分裂过的不再用）**对该数据集的基尼指数。此时，对每一个特征A，对其可能的取的每个值a，根据样本点对$A=a$的测试为“是”或“否”将D分割成D_1和D_2两部分，利用$Gini(D,A)=\frac{|D_1|}{|D|}Gini(D_1)+\frac{|D_2|}{|D|}Gini(D_2)$计算$A=a$时的基尼指数。
>
> **2）** **在所有可能的特征A和所有可能的切分点a中，选择基尼指数最小的特征及其对应的切分点作为最优特征与最优切分点。**生成两个子结点，将训练集分配到两个子结点中。
>
> **3）** 对两个子结点递归地调用（1）、（2），直至满足停止条件。
>
> **4）** 生成CART决策树。
>
> **停止条件：**
>
> 1. 结点中的样本个数小于预定阈值；
> 2. 样本集的基尼指数小于预定阈值（样本基本属于同一类）；
> 3. 没有更多的特征；

7.2 树的剪枝

CART的剪枝算法与ID3&C4.5有所不同，CART剪枝算法由两部组成：

1. 从生成的决策树T_0底端开始不断剪枝，直到T_0的跟结点，形成一个子树序列$\{T_0,T_1,\ldots,T_n\}$；
2. 通过交叉验证法在独立的验证数据集上，对子树序列进行测试，选取最优子树；

1. 剪枝，形成一个子树序列

子树损失函数：

$$C_\alpha(T)=C(T)+\alpha|T|$$

其中，T为任意子树，$C(T)$为预测误差（**如基尼指数**）。$C_a(T)$是参数为α时的子树T的整体损失函数，参数α权衡训练数据的拟合程度与模型的复杂度。

对固定的α，一定存在使损失函数$C_\alpha(T)$最小的树，即最优子树T_α。用递归的方式进行剪枝，将α从小增大，$0<\alpha_0<\alpha_1<\ldots<\alpha_n<+\infty$,产生一系列的区间$[\alpha_i,\alpha_{i+1}],i=0,1,\ldots,n$;剪枝得到的子树序列对应着区间$\alpha\in[\alpha_i,\alpha_{i+1}],i=0,1,\ldots,n$;的最优子树序列$\{T_0,T_1,\ldots,T_n\}$，序列中的子树是嵌套的。

从整体数T_0开始剪枝，对T_0的任意内部节点t：

- 以 t 为**单结点树**的损失函数为 $C_\alpha(t)=C(t)+\alpha$；
- 以 t 为**根结点**的子树 T_t 损失函数为 $C_\alpha(T_t)=C(T_t)+\alpha|T_t|$；

随时α不断增大，存在某一 α 使得： $C_\alpha(t)=C_\alpha(T_t)$ 。此时， $\alpha=\frac{C(t)-C(T_t)}{|T_t|-1}$ ，两子树 T_t 和 t 有相同的损失函数，而单结点树t结点更少，因此更可取，对 T_t 进行剪枝。

为此，对T_0中每一内部结点t，计算

$$g(t)=\frac{C(t)-C(T_t)}{|T_t|-1}$$

它表示剪枝后整体损失函数减少的程度。在T_0中剪去$g(t)$最小的以t为根结点的子树T_t，得到子树T_1，同时将最小的$g(t)$设为α_1。T_1为$[\alpha_1,\alpha_2)$的最优子树。如此剪枝下去，直至根结点，在这一工程中，不断增加α的值，产生新的区间。

2. 在剪枝得到的子树序列中，通过交叉验证选取最优子树T_α

利用**独立的**验证数据集，测试子树序列$\{T_0, T_1, \ldots, T_n\}$各棵子树的平方误差（回归）或基尼指数（分类），最小的为最优的决策树。在子树序列中，每棵子树$T_0, T_1, \ldots, T_n$都对应着一个参数$\alpha_1, \alpha_2, \ldots, \alpha_n$。所以，当最优子树 T_k 确定时，对应的 α_k 也确定了，即得到最优决策树 T_α 。

> **CART 剪枝算法：**
>
> 输入：CART算法生成的决策树T_0；
>
> 输出：最优决策树T_α；
>
> **1）** 设$k = 0, T = T_0$.
>
> **2）** 设$\alpha = +\infty$.
>
> **3）** **自下而上**对各内部结点t计算：
>
> - $C(T_t)$
> - $|T_t|$
> - $g(t) = \dfrac{C(t) - C(T_t)}{|T_t| - 1}$
> - $\alpha = min(\alpha, g(t))$
>
> $C(.)$是对训练数据的预测误差.
>
> **4）** 自上而下地遍历内部结点t，如果有$g(t) = \alpha$，进行剪枝，并对叶结点t以多数表决法决定其类，得到树T.
>
> **5）** 设$k = k + 1, \alpha_k = \alpha, T_k = T$.
>
> **6）** 如果T不是由根结点单独构成的树，则回到步骤（4）.
>
> **7）** 采用交叉验证法在子树序列 $\{T_0, T_1, \ldots, T_n\}$ 中选取最优子树 T_α .

8. 算法比较

算法	特征筛选	特征类型	缺失值	剪枝	异常点	问题
$ID3$	信息增益最大	离散型	敏感	无剪枝	敏感	分类
$C4.5$	信息增益率最大	离散型与连续型	可以处理	有剪枝	敏感	分类
$CART$	基尼指数最小	离散型与连续型	可以处理	有剪枝	可以处理	分类与回归

8.1 C4.5 相对于 ID3：

- 能够处理连续型特征。思路是连续型特征离散化，通过排序得到划分点（转为二元划分），计算各划分点信息增益（比？），**将会耗费大量时间** 。与离散属性不同的是，如果当前节点为连续属性，则该属性后面还可以参与子节点的产生选择过程（？）。
- 改用信息增益比。避免信息增益偏向于取类别更多的特征作为分裂特征。

- 能够处理缺失值。
- 引入剪枝。一定程度缓解了过拟合。

8.2 CART与C4.5：

9. 缺失值处理（Todo）

如果有些训练样本或者待分类样本缺失了一些属性值，那么该如何处理？要解决这个问题，需要考虑3个问题：

1. 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?
2. 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?
3. 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?

针对这三个问题，昆兰提出了一系列解决的思路和方法。

问题1，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。

问题2，几种处理方式：

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. 单独为属性缺失的样本划分一个分支子集。
5. 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。

问题3，几种处理方式：

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。
4. 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。
5. 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。

参考资料

1. 《统计学习方法》，李航；
2. 机器学习经典算法详解及Python实现--决策树（Decision Tree）；

3. 数据挖掘学习笔记--决策树C4.5;
4. 决策树-ID3、C4.5;
5. 剪枝的详细操作：http://www.carefree0910.com/posts/1a7aa546/

附录

Day	Temperatrue	Outlook	Humidity	Windy	PlayGolf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	sunny	high	true	no
07-26	cool	sunny	normal	true	no
07-30	mild	sunny	high	false	yes

1. **信息增益：**
    1. 计算数据集D的经验熵$H(D)$

    $$H(D) = -\sum_{k=1}^{K}\frac{|C_k|}{|D|}log_2\frac{|C_k|}{|D|}$$

    $$H(D) = -\frac{5}{14}log_2\frac{5}{14} - \frac{9}{14}log_2\frac{9}{14} = 0.9403$$

    2. 计算特征A(Outlook)对数据集D的经验条件熵$H(D|A)$

    $$H(D|A) = \sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i) = \sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}log_2\frac{|D_{ik}|}{|D_i|}$$

    $$n = 3, sunny, rain, overcast$$

$$
\begin{aligned}
H(D|A) &= \frac{|D_1|}{|D|}H(D_1) + \frac{|D_2|}{|D|}H(D_2) + \frac{|D_2|}{|D|}H(D_2) \\
&= \frac{8}{14}H(D_1 sunny) + \frac{2}{14}H(D_2 rain) + \frac{4}{14}H(D_3 overcast) \\
&= \frac{8}{14}(-\sum_{k=1}^{K}\frac{|C_k|}{|D_1|}log_2\frac{|C_k|}{|D_1|}) + \frac{2}{14}(-\sum_{k=1}^{K}\frac{|C_k|}{|D_2|}log_2\frac{|C_k|}{|D_2|}) + \frac{4}{14}(-\sum_{k=1}^{K}\frac{|C_k|}{|D_3|}log_2\frac{|C_k|}{|D_3|}) \\
&= \frac{8}{14}(-\frac{5}{8}log_2\frac{5}{8} - \frac{3}{8}log_2\frac{3}{8}) + \frac{2}{14}(-\frac{2}{2}log_2\frac{2}{2} - \frac{0}{2}log_2\frac{0}{2}) + \frac{4}{14}(-\frac{4}{4}log_2\frac{4}{4} - \frac{0}{4}log_2\frac{0}{4}) \\
&= 0.5454
\end{aligned}
$$

3. 计算信息增益

$$
\begin{aligned}
g(D, A) &= H(D) - H(D|A) \\
&= 0.9403 - 0.5454 \\
&= 0.3949
\end{aligned}
$$