机器学习

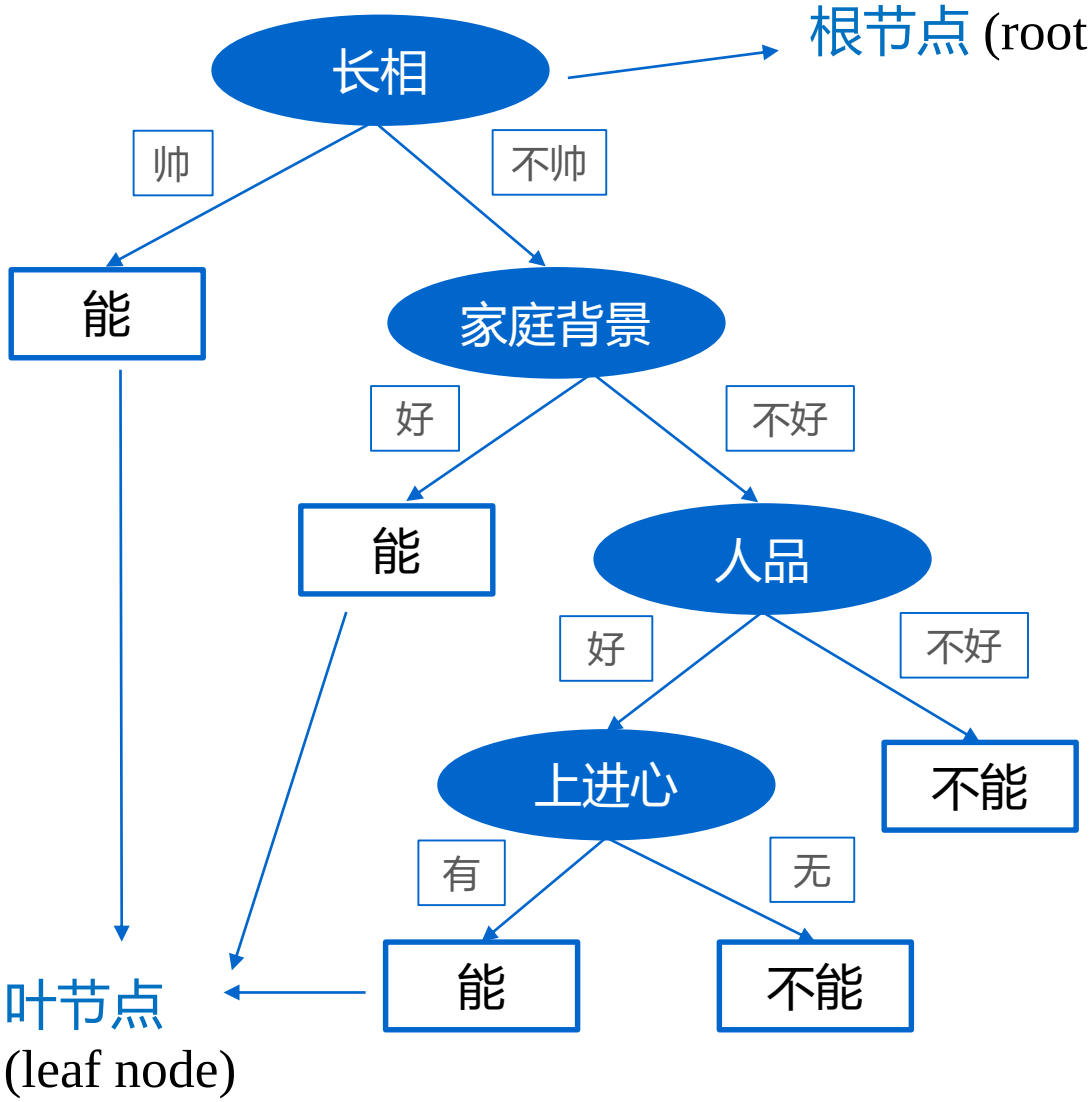
第七章 决策树

鲍鹏
北京交通大学

目录

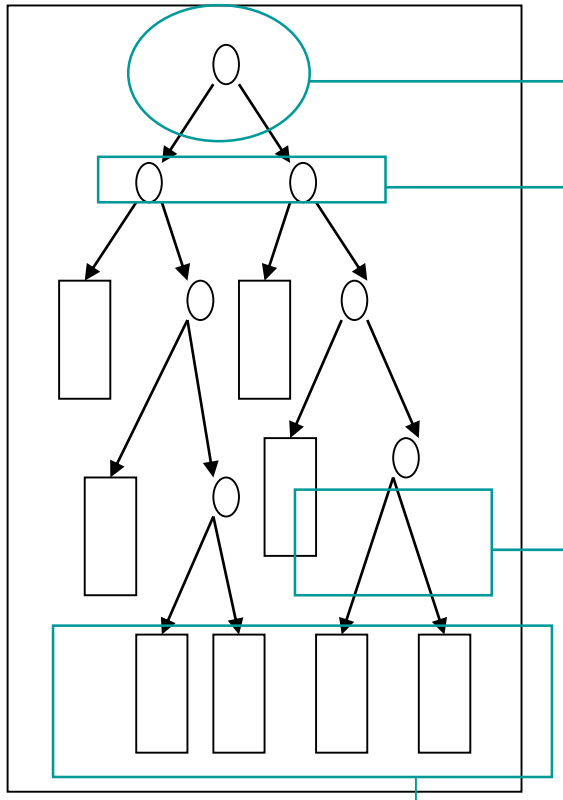
1.决策树原理

1.决策树原理



- 决策树：从训练数据中学习得出一个树状结构的模型。
- 决策树属于判别模型。
- 决策树是一种树状结构，通过做出一系列决策（选择）来对数据进行划分，这类似于针对一系列问题进行选择。
- 决策树的决策过程就是从根节点开始，测试待分类项中对应的特征属性，并按照其值选择输出分支，直到叶子节点，将叶子节点中存放的类别作为决策结果。

1.决策树原理



根节点 (root node)

非叶子节点 (non-leaf node)

(代表测试条件，对数据属性的测试)

分支 (branches) (代表测试结果)

叶节点 (leaf node)

(代表分类后所获得的分类标记)

- 决策树算法是一种归纳分类算法，它通过对训练集的学习，挖掘出有用的规则，用于对新数据进行预测。
- 决策树算法属于监督学习方法。
- 决策树归纳的基本算法是贪心算法，自顶向下来构建决策树。
- 贪心算法：在每一步选择中都采取在当前状态下最好/优的选择。
- 在决策树的生成过程中，分割方法即属性选择的度量是关键。

1.决策树原理

- **决策树-关于分类问题**

x

名称	体温	表皮覆盖	胎生	水生动物	飞行动物	有腿	冬眠	类标号
人类	恒温	毛发	是	否	否	是	否	哺乳动物
海龟	冷血	鳞片	否	半	否	是	否	爬行类
鸽子	恒温	羽毛	否	否	是	是	否	鸟类
鲸	恒温	毛发	是	是	否	否	否	哺乳类

y

分类与回归

- 分类：对于已知信息x，分类目标属性y是离散的。
- 回归：对于已知信息x，回归目标属性y是连续的。

1.决策树原理

- **决策树-解决分类问题的一般方法**

模型构建（归纳）：

- 通过对训练集合的归纳，建立分类模型。

预测应用（推论）：

- 根据建立的分类模型，对测试集合进行测试。

1.决策树原理

- **决策树和归纳算法**

  - 决策树技术发现数据模式和规则的核心是归纳算法。
  - 归纳是从特殊到一般的过程。
  - 归纳推理从若干个事实中表征出的特征、特性和属性中，通过比较、总结、概括而得出一个规律性的结论。
  - 归纳推理试图从对象的一部分或整体的观察中获得一个完备且正确的描述。即从特殊事实到普遍性规律的结论。
  - 归纳对于认识的发展和完善具有重要的意义。人类知识的增长主要来源于归纳学习。

1.决策树原理

- **决策树的特点**

优点：

- 推理过程容易理解，计算简单，可解释性强。
- 比较适合处理有缺失属性的样本。
- 可自动忽略目标变量没有贡献的属性变量，也为判断属性变量的重要性，减少变量的数目提供参考。

缺点：

- 容易造成过拟合，需要采用剪枝操作。
- 忽略了数据之间的相关性。
- 对于各类别样本数量不一致的数据，信息增益会偏向于那些更多数值的特征。

1.决策树原理

- **决策树算法的发展过程**
  - 1966年，CLS (Concept Learning System) 学习系统中就已经提出决策树算法的概念。
  - 1979年, J.R. Quinlan提出ID3算法，并在1983年和1986年对ID3 进行了总结和简化，使其成为决策树学习算法的典型。
  - Schlimmer 和Fisher 于1986年对ID3进行改进，在每个可能的决策树节点创建缓冲区，使决策树可以递增式生成，得到ID4算法。
  - 1988年，Utgoff 在ID4基础上提出了ID5学习算法，进一步提高了效率。
  - 1993年，Quinlan 进一步发展了ID3算法，改进成C4.5算法。
  - 另一类决策树算法为CART，与C4.5不同的是，CART的决策树由二元逻辑问题生成，每个树节点只有两个分支。

2.CLS算法

2.CLS算法

- **CLS（Concept Learning System）算法**
  - CLS算法是早期的决策树学习算法。它是许多决策树学习算法的基础。
  - CLS基本思想
    - 从一棵空决策树开始，选择某一分类属性作为测试属性。该测试属性对应决策树中的决策结点。根据该属性值的不同，可将训练样本分成相应的子集：

      - 如果该子集为空，或该子集中样本属于同一个类，则该子集为叶节点；
      - 否则该子集对应于决策树的内部节点，即测试节点，需要选择一个新的分类属性对该子集进行划分，直到所有的子集都为空或者属于同一类。

2.CLS算法

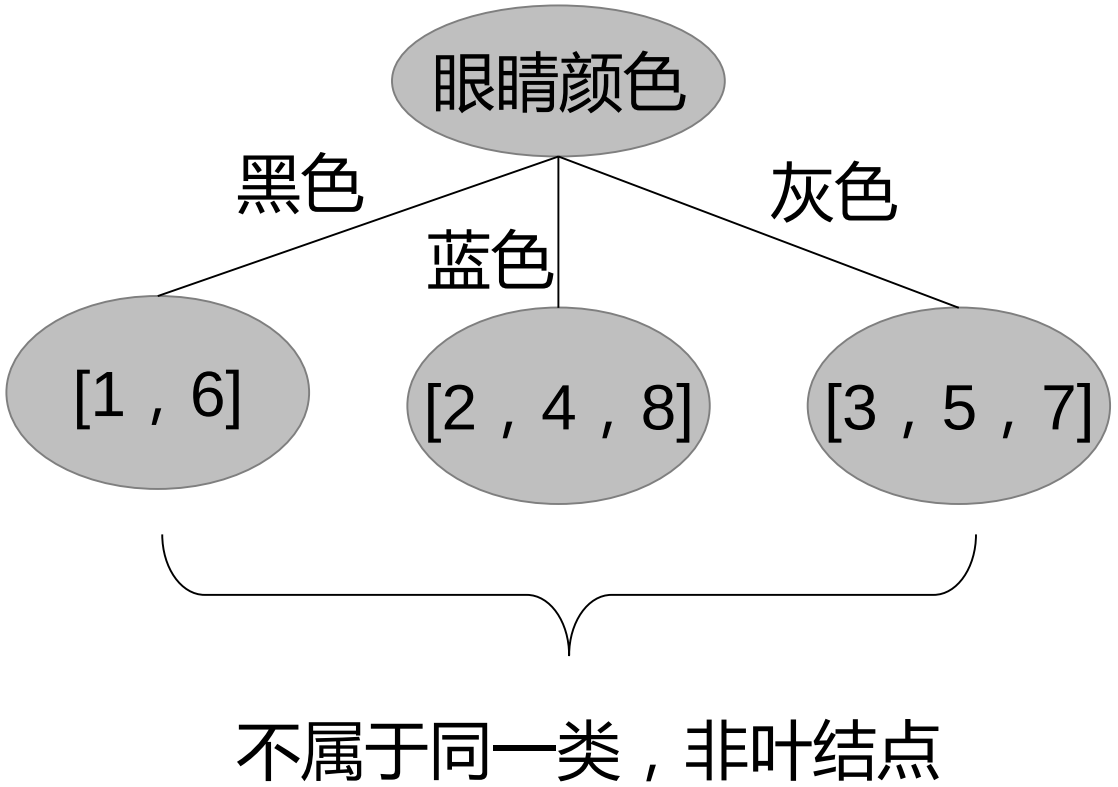
- **CLS算法**

人员	眼睛颜色	头发颜色	所属人种
1	黑色	黑色	黄种人
2	蓝色	金色	白种人
3	灰色	金色	白种人
4	蓝色	红色	白种人
5	灰色	红色	白种人
6	黑色	金色	混血
7	灰色	黑色	混血
8	蓝色	黑色	混血

2.CLS算法

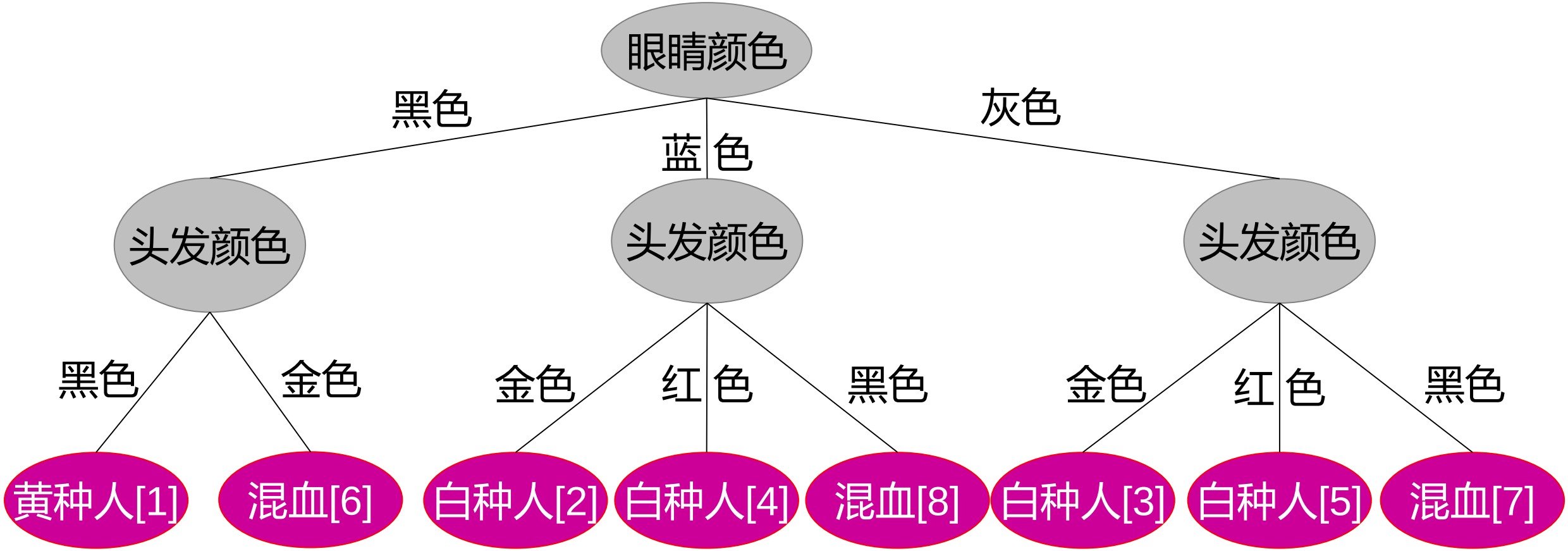
- **CLS算法-决策树的构建**

人员	眼睛颜色	头发颜色	所属人种
1	黑色	黑色	黄种人
2	蓝色	金色	白种人
3	灰色	金色	白种人
4	蓝色	红色	白种人
5	灰色	红色	白种人
6	黑色	金色	混血
7	灰色	黑色	混血
8	蓝色	黑色	混血



2.CLS算法

- **CLS算法-决策树的构建**



2.CLS算法

- **CLS算法步骤**

1. 生成一棵空决策树和一张训练样本属性集;
2. 若训练样本集T中所有的样本都属于同一类, 则生成结点T , 并终止学习算法;
3. 否则， 根据某种策略从训练样本属性表中选择属性A作为测试属性， 生成测试节点A ；
4. 若A的取值为v_1，v_2，…，v_m，则根据A的取值的不同,将T划分成 m个子集T_1，T_2，…，T_m;
5. 从训练样本属性表中删除属性A;
6. 转步骤2, 对每个子集递归调用CLS。

3.ID3算法

3.ID3算法

- **ID3算法**

  - ID3 算法最早是由罗斯昆（J. Ross Quinlan）于1975年提出的一种决策树构建算法，算法的核心是“信息熵”，期望信息越小，信息熵越大，样本纯度越低。
  - ID3 算法是以信息论为基础，以信息增益为衡量标准，从而实现对数据的归纳分类。
  - ID3 算法计算每个属性的信息增益，并选取具有最高增益的属性作为给定的测试属性。

3.ID3算法

- **熵**
  - Shannon 1948年提出的信息论理论。
  - 熵(entropy)：信息量大小的度量，是表示随机变量不确定性的度量。
  - 熵的通俗解释：事件a_i的信息量$I(a_i)$可表示为：

$$I(a_i) = p(a_i) log_2 \frac{1}{p(a_i)}$$

  - 其中$p(a_i)$表示事件a_i发生的概率。

3.ID3算法

- **熵的理论解释**
  - 设X是一个取有限个值的离散随机变量，其概率分布为：

$$P(X = x_i) = p_i, \qquad i = 1,2,\ldots,n$$

  - 则随机变量X的熵定义为：

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i$$

  - 熵越大，随机变量的不确定性越大。

3.ID3算法

- **条件熵**
  - 设有随机变量(X, Y),其联合概率分布为：

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

  - 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，定义为给定X条件下，Y的条件概率分布的熵对X的数学期望：

$$H(Y|X) = \sum_{i=1}^{n} p_i \, H(Y|X = x_i)$$

3.ID3算法

- **信息增益**

  - 定义：特征A对训练数据集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)
  - 决策树学习中的信息增益等价于训练数据集中类与特征的互信息。

3.ID3算法

- **信息增益的算法**

  - 输入：训练数据集D和特征A
  - 输出：特征A对训练数据集D的信息增益$g(D,A)$
  - 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$H(D|A)$
  - 3. 计算信息增益$g(D,A) = H(D) - H(D|A)$

3.ID3算法

- **ID3算法**

其大致步骤为：

1. 初始化特征集合和数据集合；
2. 计算数据集合信息熵和所有特征的条件熵，选择信息增益最大的特征作为当前决策节点；
3. 更新数据集合和特征集合（删除上一步使用的特征，并按照特征值来划分不同分支的数据集合）；
4. 重复 2，3 两步，若子集值包含单一特征，则为分支叶子节点。

3.ID3算法

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

K是类别，D是数据集，C_k是类别K下的数据集

右边数据中：

数量	是	否	信息熵
15	9	6	0.971

$$H(D)=-\sum_{k=1}^{K}\frac{|C_k|}{|D|}log_2\frac{|C_k|}{|D|}=-\frac{9}{15}log_2\frac{9}{15}-\frac{6}{15}log_2\frac{6}{15}$$
$$=0.971$$

	年龄	有工作	有房子	信用	类别
0	青年	否	否	一般	否
1	青年	否	否	好	否
2	青年	是	否	好	是
3	青年	是	是	一般	是
4	青年	否	否	一般	否
5	中年	否	否	一般	否
6	中年	否	否	好	否
7	中年	是	是	好	是
8	中年	否	是	非常好	是
9	中年	否	是	非常好	是
10	老年	否	是	非常好	是
11	老年	否	是	好	是
12	老年	是	否	好	是
13	老年	是	否	非常好	是
14	老年	否	否	一般	否

3.ID3算法

- **按年龄划分**

年龄	数量	是	否	信息熵
青年	5	2	3	0.9710
中年	5	3	2	0.9710
老年	5	4	1	0.7219

A_1	年龄
A_2	有工作
A_3	有房子
A_4	信用

$$H(D|A_1 = \text{青年}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.971$$

$$H(D|A_1 = \text{中年}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.971$$

$$H(D|A_1 = \text{老年}) = -\frac{4}{5}log_2\frac{4}{5} - \frac{1}{5}log_2\frac{1}{5} = 0.7219$$

	年龄	有工作	有房子	信用	类别
0	青年	否	否	一般	否
1	青年	否	否	好	否
2	青年	是	否	好	是
3	青年	是	是	一般	是
4	青年	否	否	一般	否
5	中年	否	否	一般	否
6	中年	否	否	好	否
7	中年	是	是	好	是
8	中年	否	是	非常好	是
9	中年	否	是	非常好	是
10	老年	否	是	非常好	是
11	老年	否	是	好	是
12	老年	是	否	好	是
13	老年	是	否	非常好	是
14	老年	否	否	一般	否

3.ID3算法

- **条件熵** $H(D|A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)$

其中，D是数据集，A是特征。

$$H(D|\text{年龄})=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)$$

$$=\frac{5}{15}\times 0.971+\frac{5}{15}\times 0.971+\frac{5}{15}\times 0.7219$$

$$=0.8897$$

	年龄	有工作	有房子	信用	类别
0	青年	否	否	一般	否
1	青年	否	否	好	否
2	青年	是	否	好	是
3	青年	是	是	一般	是
4	青年	否	否	一般	否
5	中年	否	否	一般	否
6	中年	否	否	好	否
7	中年	是	是	好	是
8	中年	否	是	非常好	是
9	中年	否	是	非常好	是
10	老年	否	是	非常好	是
11	老年	否	是	好	是
12	老年	是	否	好	是
13	老年	是	否	非常好	是
14	老年	否	否	一般	否

3.ID3算法

- **信息增益** $g(D,A)=H(D)-H(D|A)$

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

$g(D,A_1=\text{老年})$

$=H(D)-H(D|A_1=\text{老年})$

=0.971-0.7219 = 0.2491

	年龄	有工作	有房子	信用	类别
0	青年	否	否	一般	否
1	青年	否	否	好	否
2	青年	是	否	好	是
3	青年	是	是	一般	是
4	青年	否	否	一般	否
5	中年	否	否	一般	否
6	中年	否	否	好	否
7	中年	是	是	好	是
8	中年	否	是	非常好	是
9	中年	否	是	非常好	是
10	老年	否	是	非常好	是
11	老年	否	是	好	是
12	老年	是	否	好	是
13	老年	是	否	非常好	是
14	老年	否	否	一般	否

3.ID3算法

- **ID3算法-小结**

  - ID3算法的基本思想是：以信息熵为度量，用于决策树节点的属性选择，每次优先选取信息量最多的属性，亦即能使熵值变为最小的属性，以构造一棵熵值下降最快的决策树，到叶子节点处的熵值为0。此时，每个叶子节点对应的实例集中的实例属于同一类。

3.ID3算法

- **缺点**
  - ID3 没有剪枝策略，容易过拟合；
  - 信息增益准则对可取值数目较多的特征有所偏好，类似“编号”的特征其信息增益接近于 1；
  - 只能用于处理离散分布的特征；
  - 没有考虑缺失值。

4.C4.5算法

4.C4.5算法

- **C4.5算法**

  - C4.5 算法是用于生成决策树的一种经典算法，是对 ID3 算法的延伸和优化。
  - 用信息增益率来选择属性，克服了用信息增益选择的不足。
  - 在决策树构造过程中进行剪枝。
  - 能够处理离散型和连续型的属性类型。
  - 能够处理具有缺失属性值的训练数据。

4.C4.5算法

- **信息增益率**
  - 以信息增益作为划分训练数据集的特征，存在偏向于选择取值较多的特征的问题，使用信息增益率可以对这一问题进行校正。
  - 定义：信息增益与训练数据集D关于特征A的值的熵之比，即

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

4.C4.5算法

- **信息增益率** $g_R(D,A)=\dfrac{g(D,A)}{H_A(D)}$

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

$$g(D,A_1=\text{老年})$$
$$=H(D)-H(D|A_1=\text{老年})$$

=0.971-0.7219 = 0.2491

$$g_R(D,A_1=\text{老年})=\frac{g(D,A_1=\text{老年})}{H_A(D)}=\frac{0.2491}{-\sum_{i=1}^{n}\frac{|D_i|}{|D|}log_2\frac{|D_i|}{|D|}}$$

$$=\frac{0.2491}{-\frac{9}{15}log_2\frac{9}{15}-\frac{6}{15}log_2\frac{6}{15}}=0.2565$$

备注：信息增益 $g(D,A)=H(D)-H(D|A)$

	年龄	有工作	有房子	信用	类别
0	青年	否	否	一般	否
1	青年	否	否	好	否
2	青年	是	否	好	是
3	青年	是	是	一般	是
4	青年	否	否	一般	否
5	中年	否	否	一般	否
6	中年	否	否	好	否
7	中年	是	是	好	是
8	中年	否	是	非常好	是
9	中年	否	是	非常好	是
10	老年	否	是	非常好	是
11	老年	否	是	好	是
12	老年	是	否	好	是
13	老年	是	否	非常好	是
14	老年	否	否	一般	否

4.C4.5算法

- **C4.5的剪枝**

  ➢ 过拟合的原因：

  为了尽可能正确分类训练样本，节点的划分过程会不断重复直到不能再分，这样就可能对训练样本学习的“太好”，把训练样本的一些特点当做所有数据都具有的一般性质，从而导致过拟合。

  剪枝的基本策略有“预剪枝”（prepruning）和“后剪枝”（post-pruning）

4.C4.5算法

- **C4.5的剪枝**
  - ➢ 预剪枝（prepruning）

预剪枝不仅可以降低过拟合的风险而且还可以减少训练时间，但另一方面，它是基于“贪心”策略，会带来欠拟合风险。

训练集

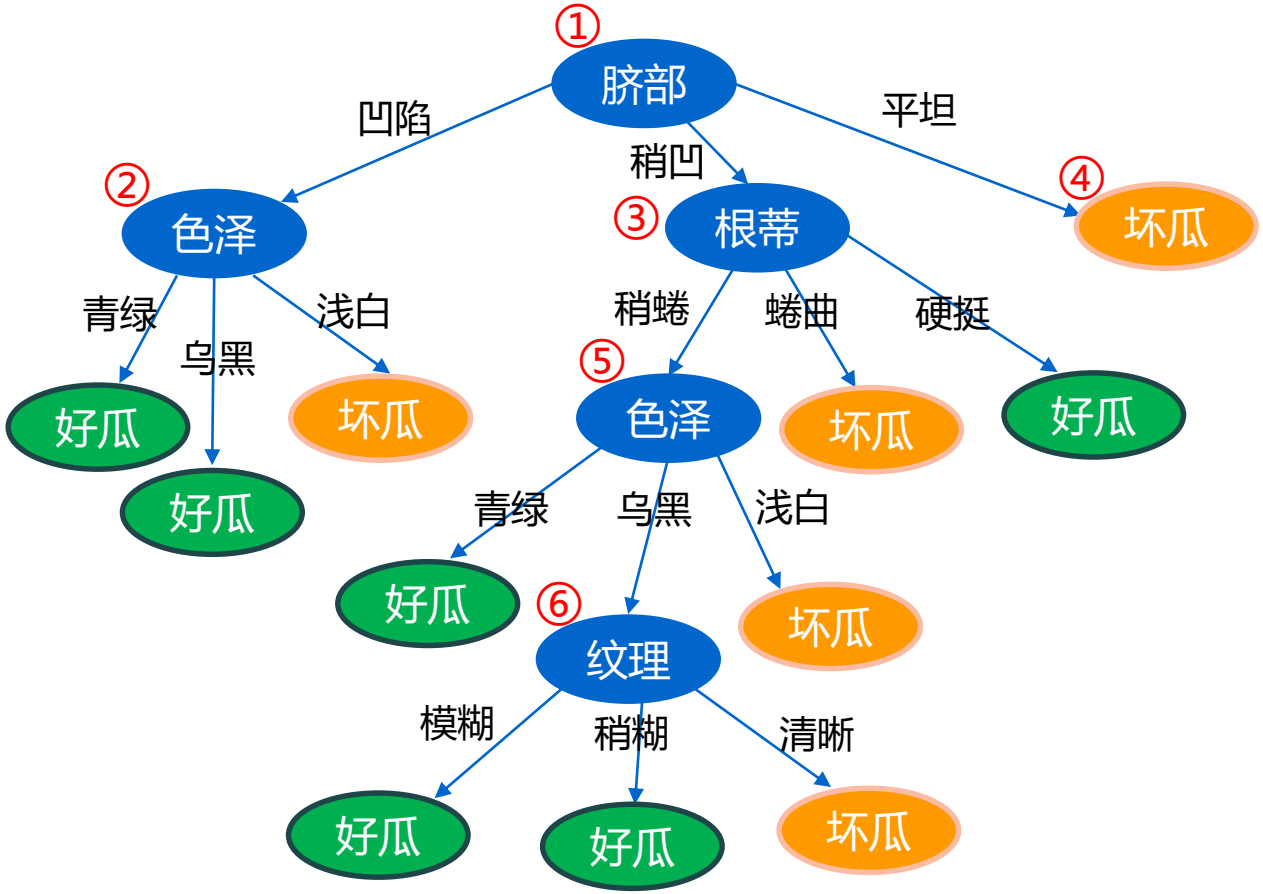
编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

验证集

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否

4.C4.5算法

- **预剪枝（prepruning）**



基于表生成未剪枝的决策树

剪枝策略

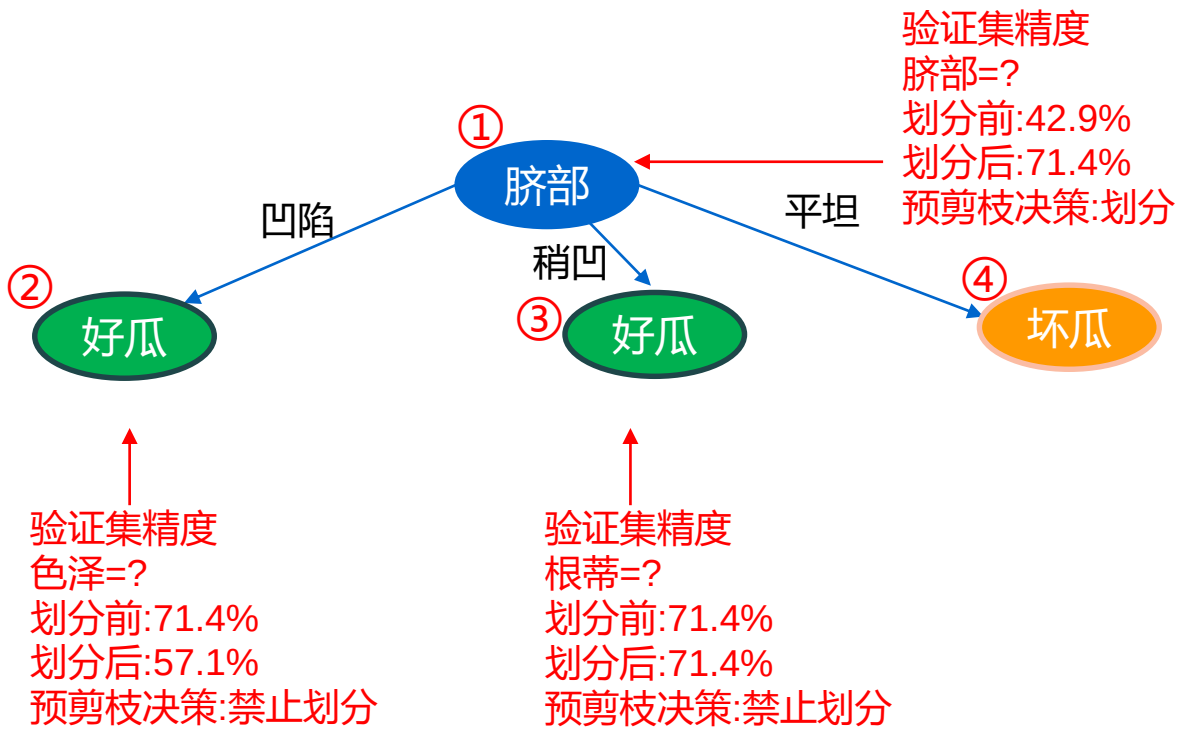
在节点划分前确定是否继续增长，及早停止增长

主要方法有：

- 节点内数据样本低于某一阈值；
- 所有节点特征都已分裂；
- 节点划分前准确率比划分后准确率高。

4.C4.5算法

- **预剪枝（prepruning）**



预剪枝的决策树

剪枝策略

在节点划分前确定是否继续增长，及早停止增长

主要方法有：

- 节点内数据样本低于某一阈值；
- 所有节点特征都已分裂；
- 节点划分前准确率比划分后准确率高。

4.C4.5算法

- **后剪枝**
  - 在已经生成的决策树上进行剪枝，从而得到简化版的剪枝决策树。
  - 后剪枝决策树通常比预剪枝决策树保留了更多的分支。
  - 一般情况下，后剪枝的欠拟合风险更小，泛化性能往往优于预剪枝决策树。

训练集

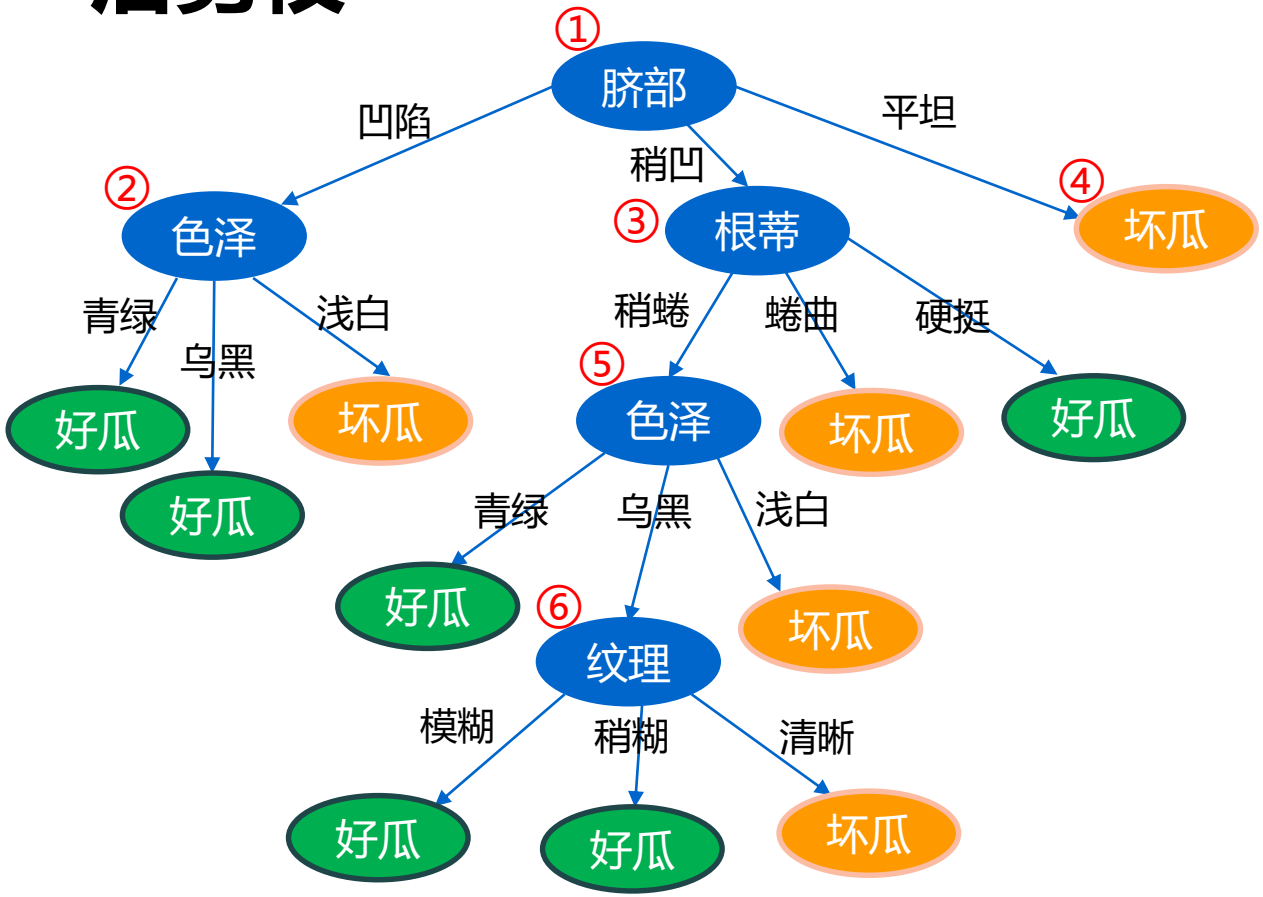
编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

验证集

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否

4.C4.5算法

- **后剪枝**



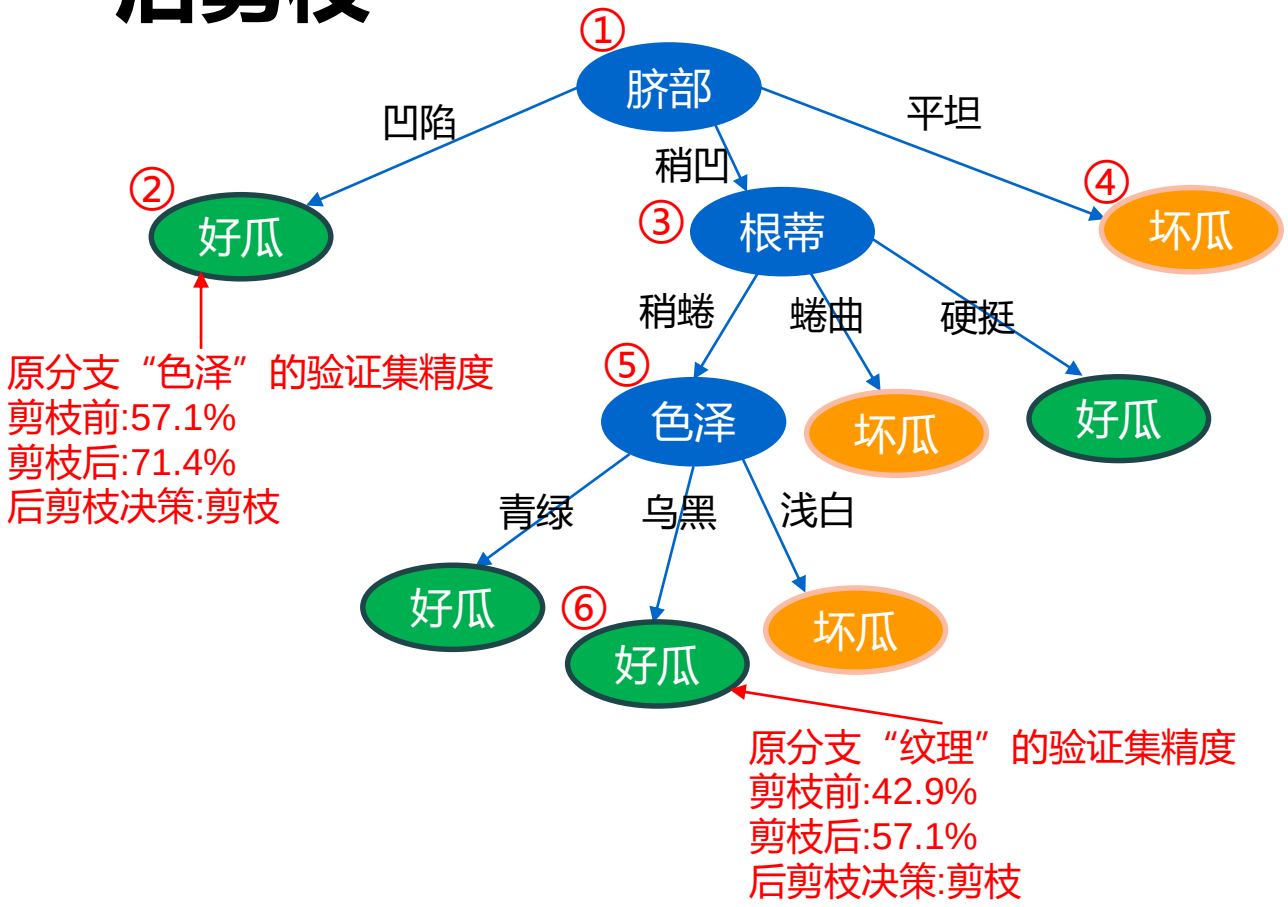
基于表生成未剪枝的决策树

剪枝方法

- C4.5 采用的悲观剪枝方法，用递归的方式自底向上针对每一个非叶子节点，评估用一个最佳叶子节点去代替这课子树是否有益。如果剪枝后与剪枝前相比其错误率是保持或者下降，则这棵子树就可以被替换掉。C4.5 通过训练数据集上的错误分类数量来估算未知样本上的错误率。
- 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。

4.C4.5算法

- **后剪枝**



后剪枝的决策树

剪枝方法

- C4.5 采用的悲观剪枝方法，用递归的方式自底向上针对每一个非叶子节点，评估用一个最佳叶子节点去代替这课子树是否有益。如果剪枝后与剪枝前相比其错误率是保持或者下降，则这棵子树就可以被替换掉。C4.5 通过训练数据集上的错误分类数量来估算未知样本上的错误率。
- 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。

4.C4.5算法

- **C4.5算法的缺点**
  - 剪枝策略可以再优化；
  - C4.5 用的是多叉树，用二叉树效率更高；
  - C4.5 只能用于分类；
  - C4.5 使用的熵模型拥有大量耗时的对数运算，连续值还有排序运算；
  - C4.5 在构造树的过程中，对数值属性值需要按照其大小进行排序，从中选择一个分割点，所以只适合于能够驻留于内存的数据集，当训练集大得无法在内存容纳时，程序无法运行。

5.CART算法

5.CART算法

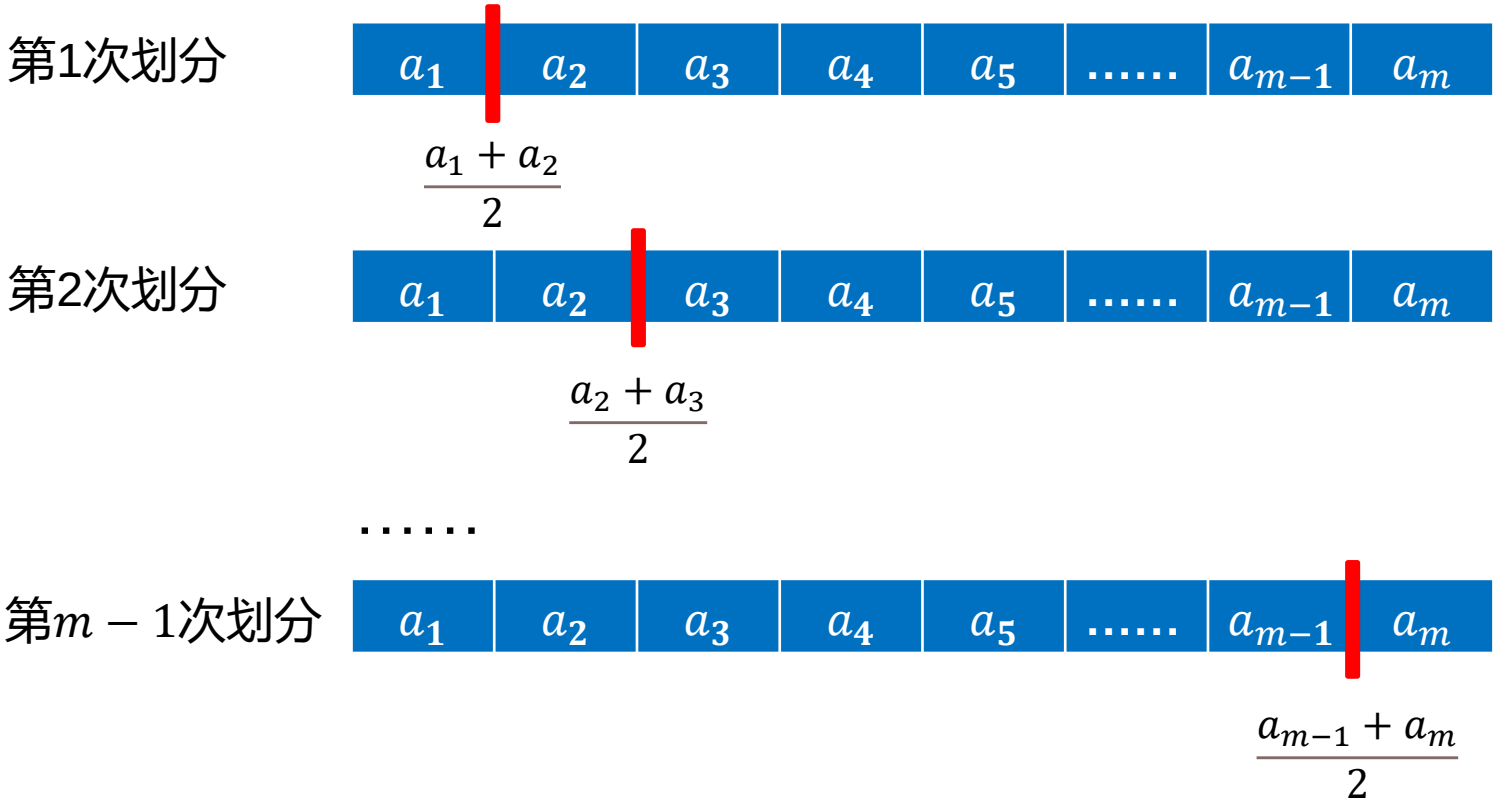
- **CART算法**
  - Classification and Regression Tree (CART) 是决策树的一种。
  - 用基尼指数来选择属性（分类），或用均方差来选择属性（回归）。
  - 顾名思义，CART算法既可以用于创建分类树，也可以用于创建回归树，两者在构建的过程中稍有差异。
  - 如果目标变量是离散的，称为分类树。
  - 如果目标变量是连续的，称为回归树。

5.CART算法

- **CART算法-分类**

  - **连续特征处理**

具体思路：m个样本的连续特征A有m个，$a_1, a_2, a_3, \ldots\ldots, a_{m-1}, a_m$从小到大排列，取相邻两样本值的平均数做划分点，一共取$m-1$个，其中第m个划分点T_m表示为：$T_m = \frac{a_{m-1}+a_m}{2}$。分别计算以这$m-1$个点作为二元分类点时的基尼系数。选择基尼指数最小的点为该连续特征的二元离散分类点。



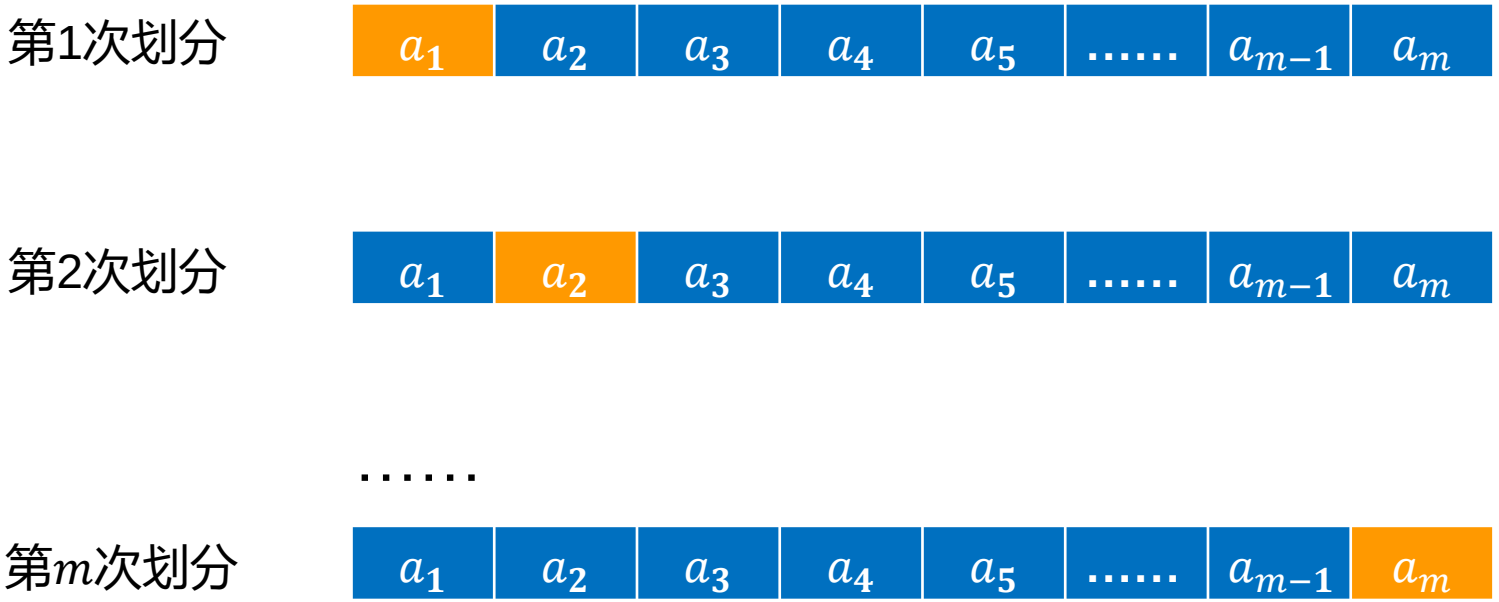
比如取到的基尼指数最小的点为a_t，则小于a_t的值为类别1，大于a_t的值为类别2，这样就做到了连续特征的离散化，接着采用基尼指数的大小来度量特征的各个划分点。

5.CART算法

- **CART算法-分类**

➢ **离散特征处理**

具体思路：假设特征a有m个离散值。分类标准是：每一次将其中一个特征分为一类，其他非该特征分为另一类。依照这个标准遍历所有分类情况，计算每个分类下的基尼指数，最后选择最小的作为最终的特征划分。



比如第1次取$\{a_1\}$为类别1，那么剩下的特征$\{a_2, a_3, \ldots\ldots, a_{m-1}, a_m\}$为类别2，由此遍历，第$m$次取$\{a_m\}$为类别1，那么剩下的特征$\{a_1, a_2, a_3, \ldots\ldots, a_{m-1}\}$为类别2。

CART的特征会多次参与节点的建立，而在ID3或C4.5的一颗子树中，离散特征只会参与一次节点的建立。

5.CART算法

- **基尼指数** - 分类时用基尼指数来选择属性

$Gini(D,A)$表示经过$A=a$分割后集合D的不确定性。

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

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

$$Gini(D,A_1=\text{青年})=\frac{5}{15}\times\left(2\times\frac{2}{5}\times\left(1-\frac{2}{5}\right)\right)+\frac{10}{15}\times\left(2\times\frac{7}{10}\times\left(1-\frac{7}{10}\right)\right)=0.4$$

$Gini(D,A_1=\text{中年})=0.48$

$Gini(D,A_1=\text{老年})=0.44$

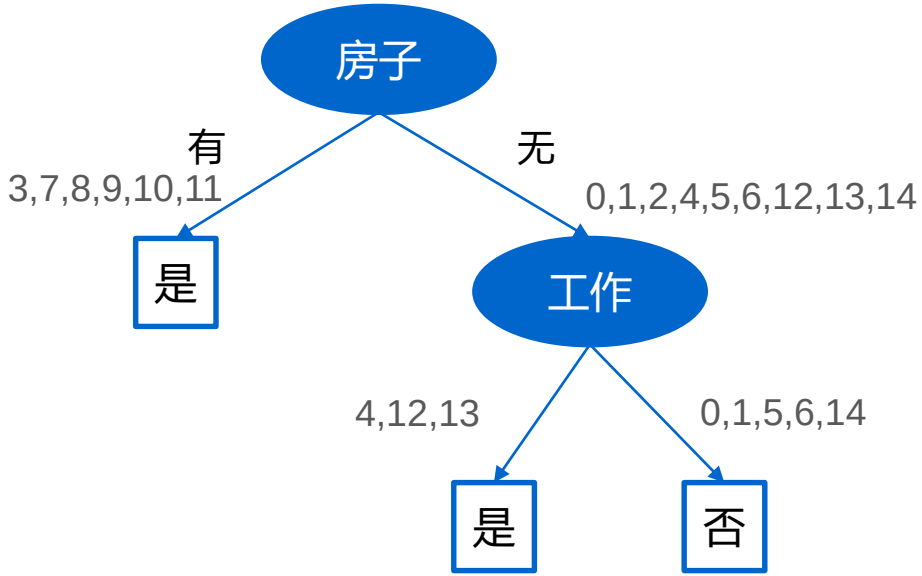
$Gini(D,A_2=\text{是})=0.32$

$Gini(D,A_3=\text{是})=0.27$

$Gini(D,A_4=\text{非常好})=0.36$

$Gini(D,A_4=\text{好})=0.47$

$Gini(D,A_4=\text{一般})=0.32$



	年龄	有工作	有房子	信用	类别
0	青年	否	否	一般	否
1	青年	否	否	好	否
2	青年	是	否	好	是
3	青年	是	是	一般	是
4	青年	否	否	一般	否
5	中年	否	否	一般	否
6	中年	否	否	好	否
7	中年	是	是	好	是
8	中年	否	是	非常好	是
9	中年	否	是	非常好	是
10	老年	否	是	非常好	是
11	老年	否	是	好	是
12	老年	是	否	好	是
13	老年	是	否	非常好	是
14	老年	否	否	一般	否

5.CART算法

- **CART算法-回归**
  - **用均方差来选择属性**

对于连续值的处理，CART分类树采用基尼系数的大小来度量特征的各个划分点。对于任意划分特征A、任意划分点s两边划分成的数据集a，求出使D_1和D_2各自集合的均方差最小，同时D_1和D_2的均方差之和最小所对应的特征和特征值划分点。表达式为：

$$\min_{a,s}[\min_{c_1}\sum_{x_i\in D_1}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in D_2}(y_i-c_2)^2]$$

其中，c_1为D_1数据集的样本输出均值，c_2为D_2 数据集的样本输出均值。

5.CART算法

- **CART算法-回归**
  - **预测方式**

对于决策树建立后做预测的方式，CART 分类树采用叶子节点里概率最大的类别作为当前节点的预测类别。

而回归树输出不是类别，它采用的是用最终叶子的均值或者中位数来预测输出结果。

5.CART算法

- **CART的生成算法**

➢ 输入：训练数据集D，停止计算条件

➢ 输出：CART决策树

从根节点开始：

1. 设数据集为D，对每个特征A，对其每个值a，根据样本点对$A = a$的测试为是或否，将D分为D_1，D_2，计算$A = a$的基尼指数；

2. 在所有的特征A以及所有可能的切分点a中，选择基尼指数最小的特征和切分点，将数据集分配到两个子节点中。

3. 对两个子节点递归调用1，2步骤

4. 生成CART树

5.CART算法

- **CART的剪枝**

  - CART算法采用一种“基于代价复杂度的剪枝”方法进行**后剪枝**，这种方法会生成一系列树，每个树都是通过将前面的树的某个或某些子树替换成一个叶节点而得到的，这一系列树中的最后一棵树仅含一个用来预测类别的叶节点。然后用一种成本复杂度的度量准则来判断哪棵子树应该被一个预测类别值的叶节点所代替。
  - 这种方法需要使用一个单独的测试数据集来评估所有的树，根据它们在测试数据集熵的分类性能选出最佳的树。

5.CART算法

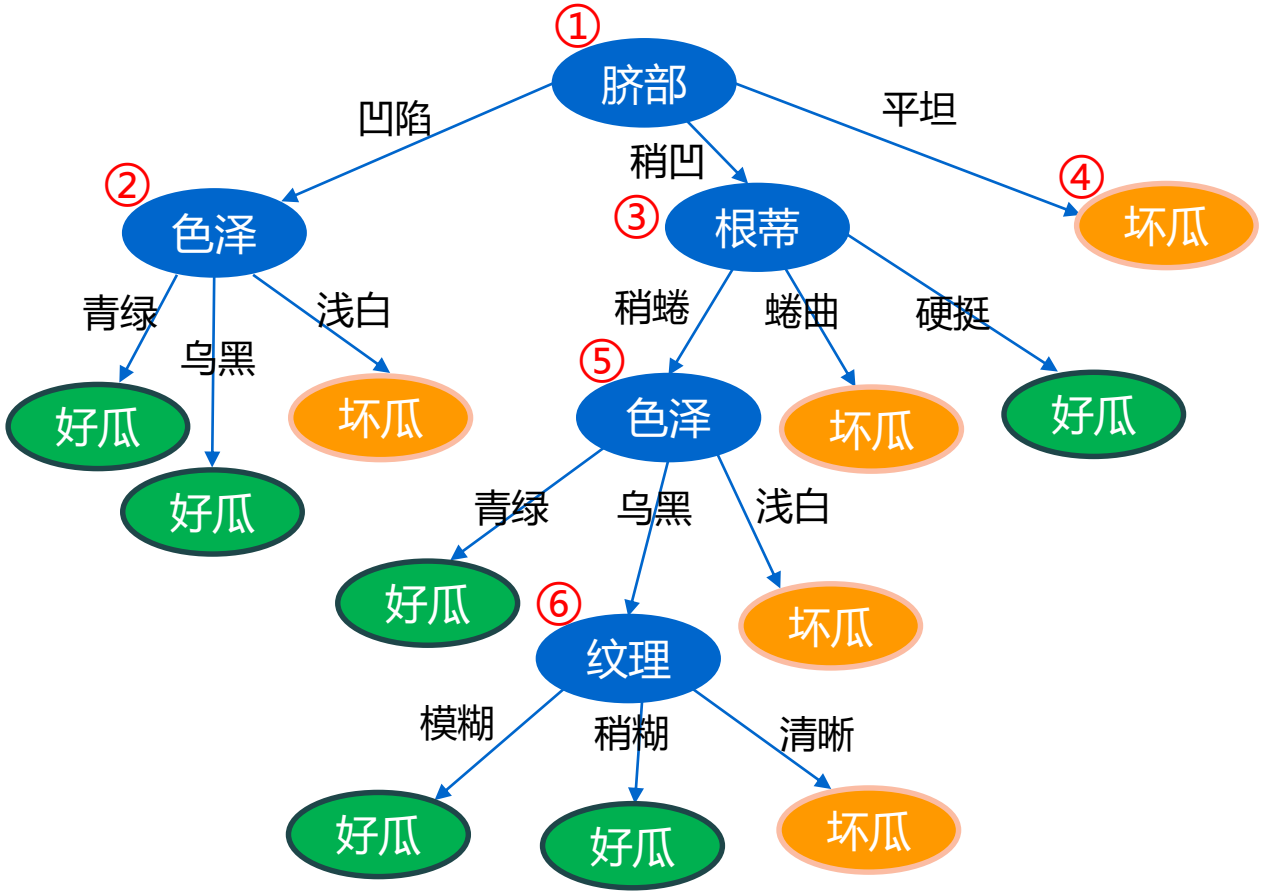
- **CART的剪枝**

➢ 具体流程:

(1)计算每一个结点的条件熵;

(2)递归的从叶子节点开始往上遍历,减掉叶子节点,然后判断损失函数的值是否减少,如果减少,则将父节点作为新的叶子节点;

(3)重复(2),直到完全不能剪枝。



总结

- **决策树的三种基本类型**

➢ 建立决策树的关键，即在当前状态下选择哪个属性作为分类依据。根据不同的目标函数，建立决策树主要有一下三种算法： ID3(Iterative Dichotomiser)、C4.5、CART(Classification And Regression Tree)。

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	特征属性多次使用
ID3	分类	多叉树	信息增益	不支持	不支持	不支持	不支持
C4.5	分类	多叉树	信息增益率	支持	支持	支持	不支持
CART	分类 回归	二叉树	基尼指数 均方差	支持	支持	支持	支持

总结

- **划分标准的差异**：ID3 使用信息增益偏向特征值多的特征，C4.5 使用信息增益率克服信息增益的缺点，偏向于特征值小的特征，CART 使用基尼指数克服 C4.5 需要求 log 的巨大计算量，偏向于特征值较多的特征。
- **使用场景的差异**：ID3 和 C4.5 都只能用于分类问题，CART 可以用于分类和回归问题；ID3 和 C4.5 是多叉树，速度较慢，CART 是二叉树，计算速度很快；
- **样本数据的差异**：ID3 只能处理离散数据且缺失值敏感，C4.5 和 CART 可以处理连续性数据且有多种方式处理缺失值；从样本量考虑的话，小样本建议 C4.5、大样本建议 CART。C4.5 处理过程中需对数据集进行多次扫描排序，处理成本耗时较高，而 CART 本身是一种大样本的统计方法，小样本处理下泛化误差较大 ；
- **样本特征的差异**：ID3 和 C4.5 层级之间只使用一次特征，CART 可多次重复使用特征；
- **剪枝策略的差异**：ID3 没有剪枝策略，C4.5 是通过悲观剪枝策略来修正树的准确性，而 CART 是通过代价复杂度剪枝。

参考文献

[1] QUINLAN J R . Introduction of decision trees[J]. Machine Learning, 1986, 1(1):81-106.

[2] QUINLAN J R. C4.5: programs for machine learning[M]. Elsevier.1993.

[3] BREIMAN L, FRIEDMAN J H, OLSHEN R A, et al. Classification and regression trees[M]. Routledge.1984

[4] 李航. 统计学习方法[M]. 清华大学出版社,2019.

[5] 周志华. 机器学习[M]. 清华大学出版社,2016.

[6] Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning[M]. Springer, New York, NY, 2001.

[7] Peter Harrington.机器学习实战[M]. 人民邮电出版社,2013.

[8] CHRISTOPHER M. BISHOP. Pattern Recognition and Machine Learning[M]. Springer,2006.

谢 谢！