决策树

谭　忠

XIAMEN UNIVERSITY
数学科学学院
SCHOOL OF MATHEMATICS SCIENCE

目录
CONTENTS

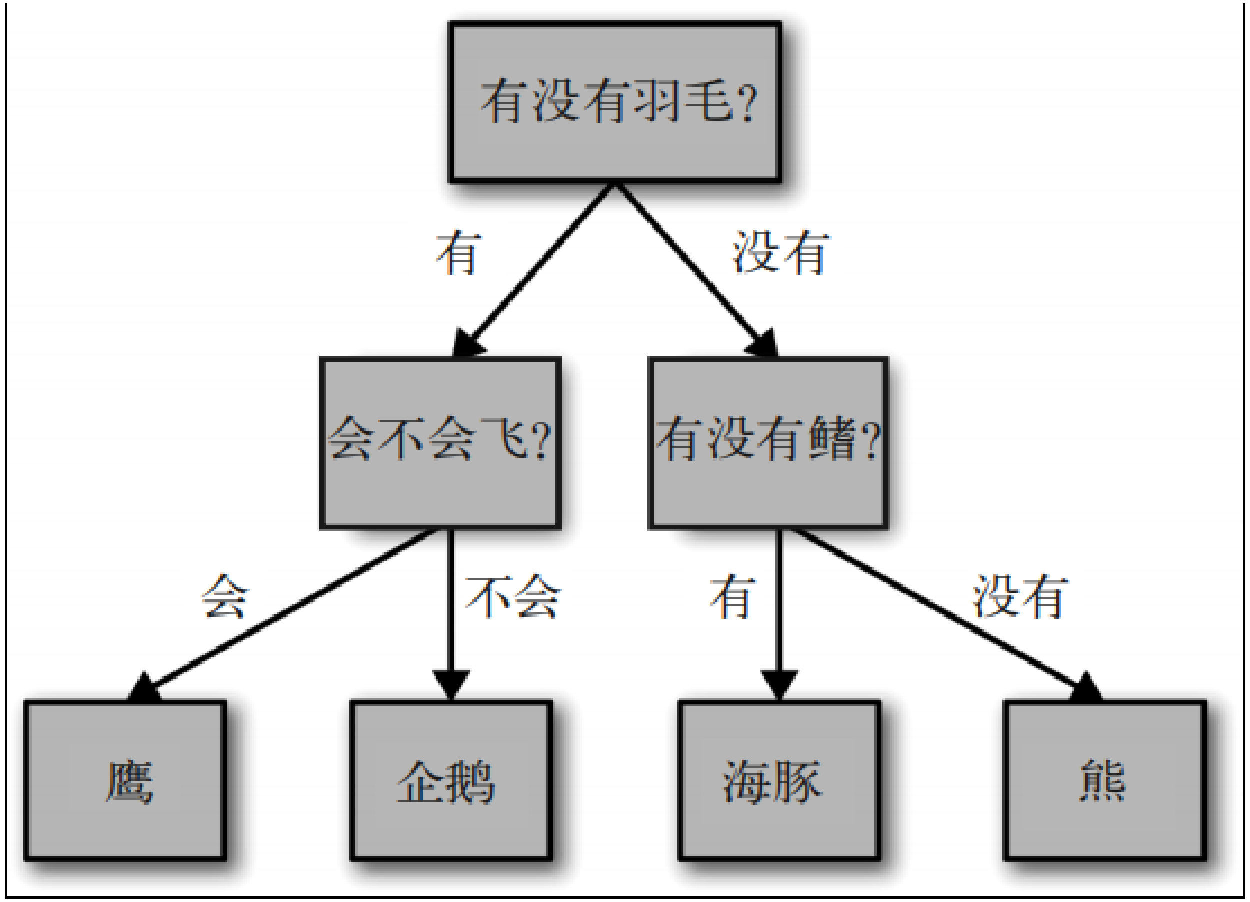
Part 1 当今应用

12.1源头问题与当今应用

决策树(decision tree) 每个决策或事件（即自然状态）都可能引出两个或多个事件，导致不同的结果，为了便于理解，把这种决策分支画成树的形状，故称决策树。

想象一下，你想要区分下面这四种动物：熊、鹰、企鹅和海豚。目标是通过提出尽可能少的问题来得到正确答案。

你可能首先会问：这种动物有没有羽毛，这个问题会将可能的动物减少到只有两种。如果答案是“有”，你可以问下一个问题，帮你区分鹰和企鹅。例如，你可以问这种动物会不会飞。如果这种动物没有羽毛，那么可能是海豚或熊，所以你需要问一个问题来区分这两种动物——比如问这种动物有没有鳍。


有没有羽毛?
有
没有
会不会飞?
有没有鳍?
会
不会
有
没有
鹰
企鹅
海豚
熊

XIAMEN UNIVERSITY

Part2 决策树思想与建模方法

一、决策树算法的基本思想：

选择最优划分划分当前样本集合并把这个属性作为决策树的一个节点

不断重复这个过程构造后继节点

直到满足下面三个条件之一停止:

- 对于当前节点，所有样本属于同一类
- 或者没有属性可以选择了
- 或者没有样本可以划分了

那么我们应该如何选择最优划分属性呢?

所谓最优划分属性，其目的在于依靠这个属性划分后能得到“纯度”最高的分类样本。

常用的指标有

1. 信息增益
2. 增益率
3. 基尼指数

1、信息增益

信息熵(Information Entropy)是度量样本集合纯度最常用的一种指标。设X是一个取有限个值的离散随机变量，其概率分布为

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

则随机变量X的熵定义为

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

- 假定离散属性 a 有 V 个可能的取值 $\{a^1, a^2, \ldots, a^V\}$, 所谓信息增益就是用属性$a$对样本进行划分并给分支结点赋予权重 $|D^v|/|D|$,所得到的的信息熵的增量
- $\operatorname{Gain}(D, a) = \operatorname{Ent}(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} \operatorname{Ent}(D^v)$

表 4.1 西瓜数据集 2.0

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
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
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
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
10	青绿	硬挺	清脆	清晰	平坦	软粘	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
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

$$\mathrm{Ent}\,(D) = -\sum_{k=1}^{2} p_k \log_2 p_k = -\left(\frac{8}{17}\log_2\frac{8}{17} + \frac{9}{17}\log_2\frac{9}{17}\right) = 0.998$$

以色泽为例计算信息增益，首先计算信息熵

$$\mathrm{Ent}(D^1) = -\left(\frac{3}{6}\log_2\frac{3}{6} + \frac{3}{6}\log_2\frac{3}{6}\right) = 1.000$$

$$\mathrm{Ent}(D^2) = -\left(\frac{4}{6}\log_2\frac{4}{6} + \frac{2}{6}\log_2\frac{2}{6}\right) = 0.918$$

$$\mathrm{Ent}(D^3) = -\left(\frac{1}{5}\log_2\frac{1}{5} + \frac{4}{5}\log_2\frac{4}{5}\right) = 0.722$$

最后计算出信息增益

$$\mathrm{Gain}\,(D, \text{色泽}) = \mathrm{Ent}\,(D) - \sum_{v=1}^{3} \frac{|D^v|}{|D|}\mathrm{Ent}(D^v)$$

$$= 0.998 - \left(\frac{6}{17} \times 1.000 + \frac{6}{17} \times 0.918 + \frac{5}{17} \times 0.722\right)$$

$$= 0.109.$$

类似的, 我们可计算出其他属性的信息增益:

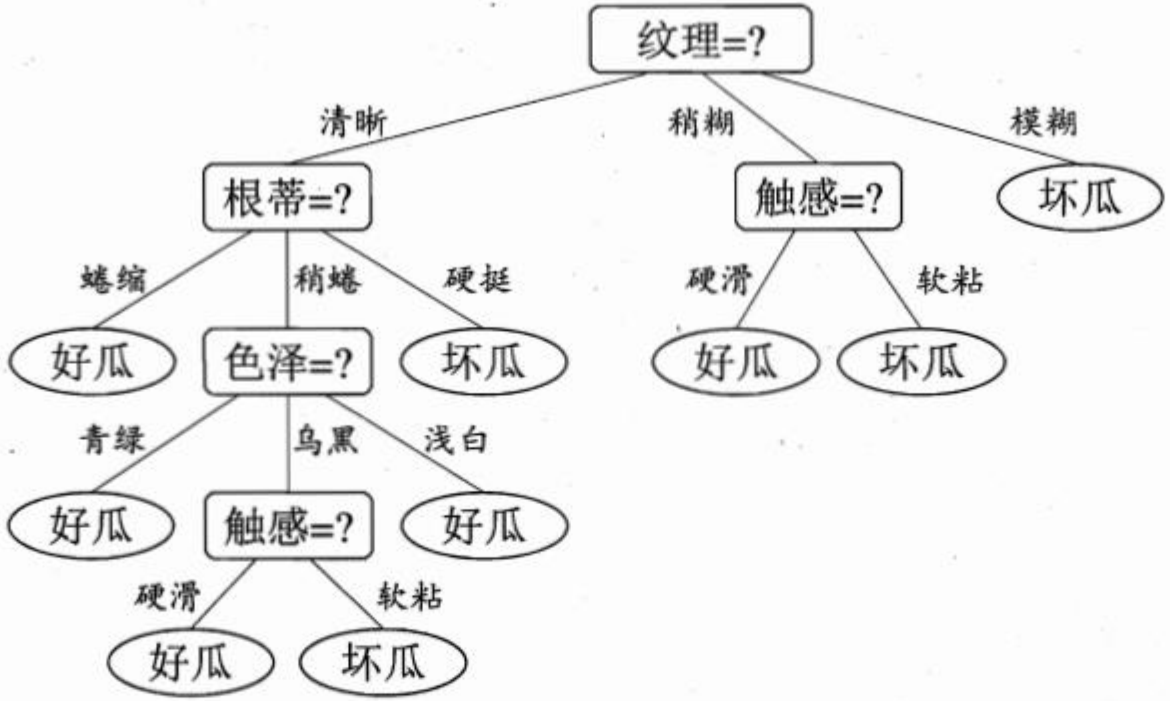
$\text{Gain}(D, \text{根蒂}) = 0.143$; $\text{Gain}(D, \text{敲声}) = 0.141$;

$\text{Gain}(D, \text{纹理}) = 0.381$; $\text{Gain}(D, \text{脐部}) = 0.289$;

$\text{Gain}(D, \text{触感}) = 0.006$.

其中纹理获得了最大的信息增益，将其选择为最优划分。

以此类推，我们就可以得到整颗决策树



信息增益的缺点：

信息增益对可取数目较多的属性有偏好，比如20个样本中有某个属性能取到20个值，即每个样本可以依靠该属性自成一类，那么显然此时按照该属性进行分类纯度将达到最大。

为了解决这个问题，C4.5决策树算法提出使用增益率来代替信息增益。

增益率的定义为：

$$\mathrm{Gainratio}\,(D,a)=\frac{\mathrm{Gain}\,(D,a)}{\mathrm{IV}\,(a)}$$

其中

$$\mathrm{IV}\,(a)=-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\log_2\frac{|D^v|}{|D|}$$

称为属性的 a 的 “固有值”，当取值属性越多，它通常就会越大。

3、基尼指数

基尼值的定义为：

$$
\begin{aligned}
\text{Gini}(D) &= \sum_{k=1}^{|\mathcal{Y}|} \sum_{k' \neq k} p_k p_{k'} \\
&= 1 - \sum_{k=1}^{|\mathcal{Y}|} p_k^2
\end{aligned}
$$

显然基尼值描述了数据集D的纯度，即任意抽取两个样本，它们不一致的概率。

基尼指数就是基尼指在某个属性上的加权平均：

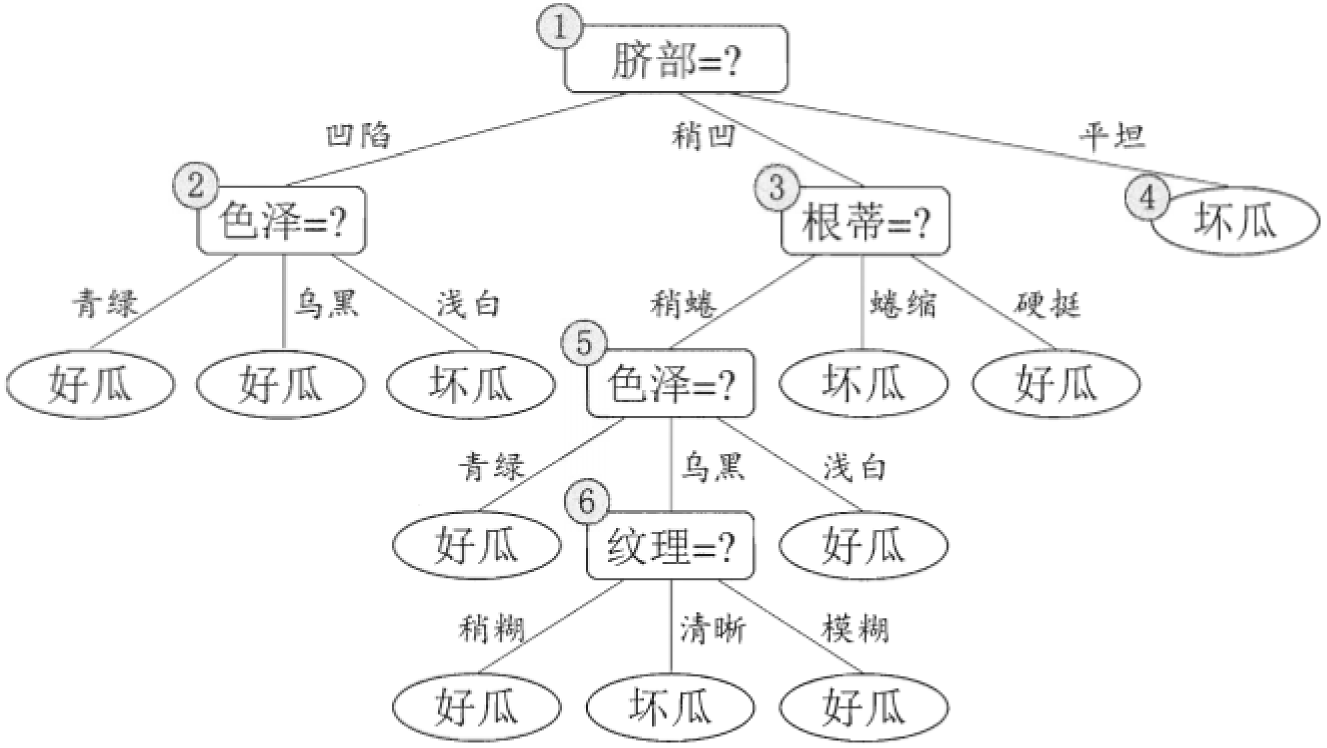
$$\text{Gini\_index}\,(D,a)=\sum_{v=1}^{V}\frac{|D^v|}{|D|}\text{Gini}(D^v).$$

此时最优划分属性为基尼指数最小的属性。

二、剪枝

剪枝是决策树算法中应对过拟合的一种策略，它分为预剪枝和后剪枝。预剪枝为自上而下的剪枝方法，而后剪枝则刚好相反。

1、预剪枝


① 脐部=?
凹陷
稍凹
平坦
② 色泽=?
③ 根蒂=?
④ 坏瓜
青绿
乌黑
浅白
好瓜
好瓜
坏瓜
稍蜷
蜷缩
硬挺
⑤ 色泽=?
坏瓜
好瓜
青绿
乌黑
浅白
好瓜
⑥ 纹理=?
好瓜
稍糊
清晰
模糊
好瓜
坏瓜
好瓜

将西瓜数据分为训练集和验证集

表 4.2 西瓜数据集 2.0 划分出的训练集(双线上部)与验证集(双线下部)

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

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

预剪枝要求我们基于验证集上的精度来判断是否进行剪枝操作。

将脐部这个叶结点标记为 “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 $\frac{3}{7} \times 100\% = 42.9\%$.

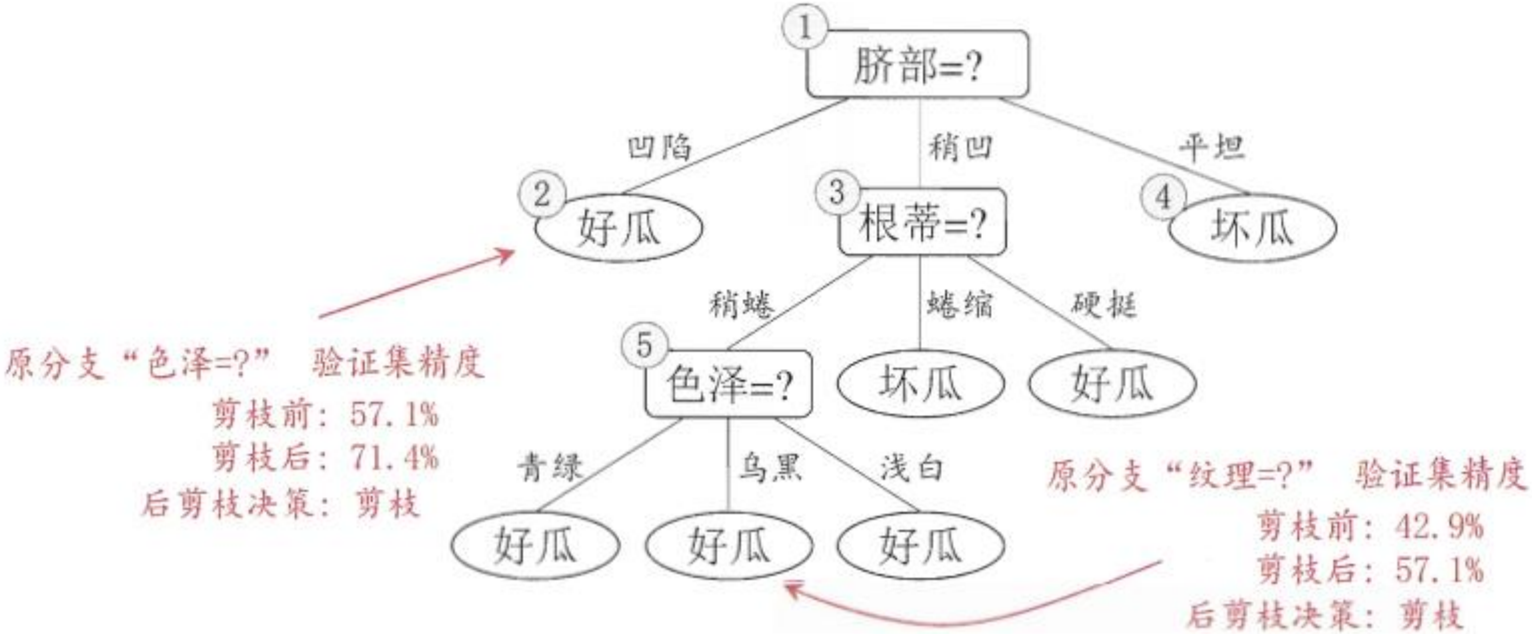
若对脐部进行进一步的划分，三个节点的训练样例分别为$\{1,2,3,14\}$、$\{6,7,15,17\}$、$\{10,16\}$，此时三个节点分别被标记为被标记为叶结点 “好瓜”、“好瓜”、“坏瓜”，验证集中 $\{4,5,8,11,12\}$ 的样例被分类正确, 验证集精度为 $\frac{5}{7}\times 100\% = 71.4\% > 42.9\%$. 于是, 用 “脐部”进行划分得以确定.

再根据信息增益原则对每个节点进一步划分，若划分后验证集的精度下降或者不变，则禁止这个划分。最后得到的仅以脐部进行划分的一棵决策树。

预剪枝的优点在于大量减少了训练时间和验证时间，但会造成欠拟合的风险，即按某个属性展开后导致的验证集精度下降是暂时的，可能完全展开后会得到性能更好的树。

后剪枝先从训练集生成一棵完整决策树，以节点6为例，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。

以此类推考察每个结点，得到



三、连续值与缺失值

1、连续值处理

对于离散值，我们通常用 “” = “” 来描述每个样例的取值，那么很容易想到，对于连续值来说可以用 “<” “>” 来描述该属性的分布。

给定样本集 D 和连续属性 a, 假定 a 在 D 上出现了 n 个不同的取值, 将这 些值从小到大进行排序, 记为 $\{a^1, a^2, \dots, a^n\}$. 基于划分点 t 可将 D 分为子集 D_t^-和 D_t^+, 其中 D_t^-包含那些在属性 a 上取值不大于 t 的样本, 而 D_t^+则包含 那些在属性 a 上取值大于 t 的样本. 因此, 对连续属性 a, 我们 可考察包含 $n-1$ 个元素的候选划分点集合

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} \mid 1 \leqslant i \leqslant n-1 \right\},$$

再根据信息增益再这t-1个点中选择最优划分：

$$\mathrm{Gain}\,(D,a) = \max_{t\in T_a}\mathrm{Gain}\,(D,a,t)$$

$$= \max_{t\in T_a}\mathrm{Ent}\,(D) - \sum_{\lambda\in\{-,+\}}\frac{|D_t^\lambda|}{|D|}\mathrm{Ent}\big(D_t^\lambda\big),$$

在之前的数据中加入密度和含糖率

表 4.3 西瓜数据集 3.0

编号	色泽	根蒂	敲声	纹理	脐部	触感	密度	含糖率	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	0.697	0.460	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	0.774	0.376	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	0.634	0.264	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	0.608	0.318	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	0.556	0.215	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	0.403	0.237	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	0.481	0.149	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	0.437	0.211	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	0.666	0.091	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	0.243	0.267	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	0.245	0.057	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	0.343	0.099	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	0.639	0.161	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	0.657	0.198	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	0.360	0.370	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	0.593	0.042	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	0.719	0.103	否

例如对属性“密度”，在决策树学习开始时, 根结点包含的17个训练样本在该属性上取值均不同. 该属性的候选划分点集合 包含 16 个候选值:

$$T_{密度} = \{0.244,0.294,0.351,0.381,0.420,0.459,0.518,0.574,0.600,0.621,0.636,0.648,0.661,0.681,0.708,0.746\}.$$

可计算出属性“密度” 的信息增益为 0.262, 对应于划分点 0.381.

再比较其他属性的信息增益，我们最终可以得到如下的决策树。

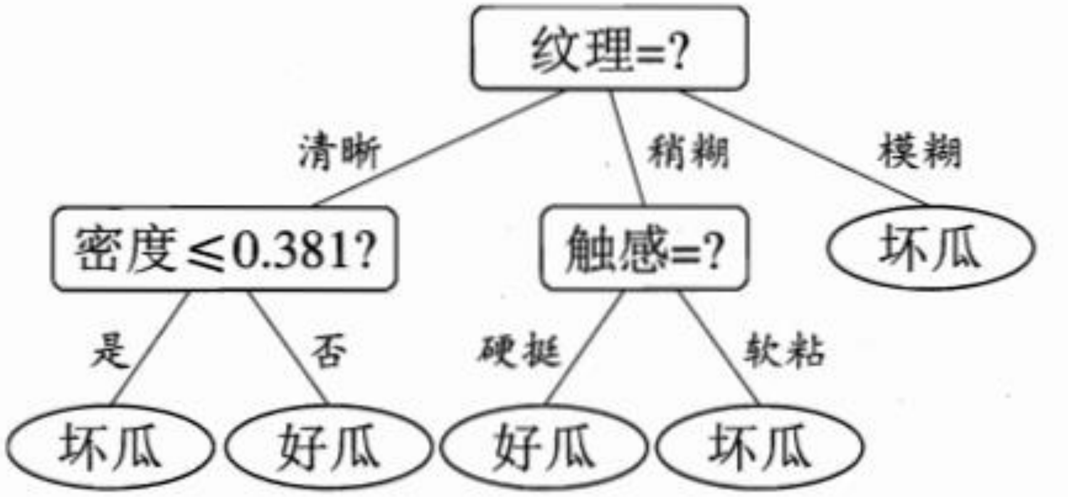


图 4.8 在西瓜数据集 3.0 上基于信息增益生成的决策树

2、缺失值的处理

由于一系列原因，我们得到的数据往往是不完整的，如：

表 4.4 西瓜数据集 2.0α

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

XIAMEN UNIVERSITY

我们需解决两个问题:

(1) 如何在属性值缺失的情况下进行划分属性选择?

(2) 给定划分属性, 若样本在该属性上的值缺失, 如何对样本进行划分?

给定训练集 D 和属性 a, 令 $\tilde{D}$ 表示 D 中在属性 a 上没有缺失值的样本子集. 对问题(1), 显然我们仅可根据 $\tilde{D}$ 来判断属性 a 的优劣. 假定属性 a 有 V 个 可取值 $\{a^1, a^2, \dots, a^V\}$, 令 $\tilde{D}^v$ 表示 $\tilde{D}$ 中在属性 a 上取值为 a^v 的样本子集, $\tilde{D}_k$ 表示 $\tilde{D}$ 中属于第 k 类 $(k = 1,2, \dots, |\mathcal{Y}|)$ 的样本子集, 则显然有 $\tilde{D} = \bigcup_{k=1}^{|\mathcal{Y}|} \tilde{D}_k$, $\tilde{D} = \bigcup_{v=1}^{V} \tilde{D}^v$.

假定我们为每个样本 $\boldsymbol{x}$ 赋予一个权重 $w_{\boldsymbol{x}}$, 并定义

$$\rho = \frac{\sum_{\boldsymbol{x} \in \tilde{D}} w_{\boldsymbol{x}}}{\sum_{\boldsymbol{x} \in D} w_{\boldsymbol{x}}}$$

$$\tilde{p}_k = \frac{\sum_{\boldsymbol{x} \in \tilde{D}_k} w_{\boldsymbol{x}}}{\sum_{\boldsymbol{x} \in \tilde{D}} w_{\boldsymbol{x}}} \quad (1 \leqslant k \leqslant |\mathcal{Y}|),$$

$$\tilde{r}_v = \frac{\sum_{\boldsymbol{x} \in \tilde{D}^v} w_{\boldsymbol{x}}}{\sum_{\boldsymbol{x} \in \tilde{D}} w_{\boldsymbol{x}}} \quad (1 \leqslant v \leqslant V).$$

基于上述定义，我们可将信息增益的计算推广为

$$
\begin{aligned}
\operatorname{Gain}(D, a) &= \rho \times \operatorname{Gain}(\tilde{D}, a) \\
&= \rho \times \left(\operatorname{Ent}(\tilde{D}) - \sum_{v=1}^{V} \tilde{r}_v \operatorname{Ent}\left(\tilde{D}^v\right) \right),
\end{aligned}
$$

其中

$$
\operatorname{Ent}(\tilde{D}) = -\sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k \log_2 \tilde{p}_k .
$$

对于问题2，我们可以将属性取值未知的数据以不同概率划入所有子结点中，即将样本权值在与属性值 a^v 对应的子结点 中调整为 $\tilde{r}_v \cdot w_x$

在学习开始时, 根结点包含样本集D中全部 17 个样例, 各样例的权值均为1 . 以属性“色泽”为例, 该属性上无缺失值的样例子集 $\tilde{D}$ 包含14个样例. 显然, $\tilde{D}$ 的信息熵为

$$\begin{aligned}\operatorname{Ent}(\tilde{D}) &= -\sum_{k=1}^{2} \tilde{p}_k \log_2 \tilde{p}_k \\ &= -\left(\frac{6}{14}\log_2\frac{6}{14}+\frac{8}{14}\log_2\frac{8}{14}\right) = 0.985.\end{aligned}$$

令 $\tilde{D}^1, \tilde{D}^2$ 与 $\tilde{D}^3$ 分别表示在属性“色泽”上取值为“青绿”“乌黑”以及“浅白”的样本子集, 有

$$\mathrm{Ent}(\tilde{D}^1) = -\left(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right) = 1.000$$

$$\mathrm{Ent}(\tilde{D}^2) = -\left(\frac{4}{6}\log_2\frac{4}{6} + \frac{2}{6}\log_2\frac{2}{6}\right) = 0.918$$

$$\mathrm{Ent}(\tilde{D}^3) = -\left(\frac{0}{4}\log_2\frac{0}{4} + \frac{4}{4}\log_2\frac{4}{4}\right) = 0.000$$

因此, 样本子集 $\tilde{D}$ 上属性 “色泽” 的信息增益为

$$\mathrm{Gain}(\tilde{D},\ 色泽\ ) = \mathrm{Ent}\ (\tilde{D}) - \sum_{v=1}^{3} \tilde{r}_v \mathrm{Ent}\left(\tilde{D}^v\right)$$

$$= 0.985 - \left(\frac{4}{14} \times 1.000 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0.000\right) = 0.306$$

于是, 样本集 D 上属性 “色泽”的信息增益为

$$\mathrm{Gain}\,(D,\ \text{色泽}\,) = \rho \times \mathrm{Gain}\,(\tilde{D},\ \text{色泽}\,) = \frac{14}{17} \times 0.306$$

$$= 0.252.$$

类似地可计算出所有属性在 D 上的信息增益:

$\mathrm{Gain}(D,\text{色泽}) = 0.252$; $\mathrm{Gain}(D,\text{根蒂}) = 0.171$

$\mathrm{Gain}(D,\text{敲声}) = 0.145$; $\mathrm{Gain}(D,\text{纹理}) = 0.424$

$\mathrm{Gain}(D,\text{脐部}) = 0.289$; $\mathrm{Gain}(D,\text{触感}) = 0.006$

因此选择纹理为最优划分属性，其中编号为8的样本在纹理上缺失，固以$\frac{7}{15}$、$\frac{5}{15}$ 和 $\frac{3}{15}$的权重划入所有子节点。编号10类似。

四、多变量决策树

若将每一个样本看做是高维空间中的一个点，决策树算法的每次划分都对应这一条与轴平行的直线，如图

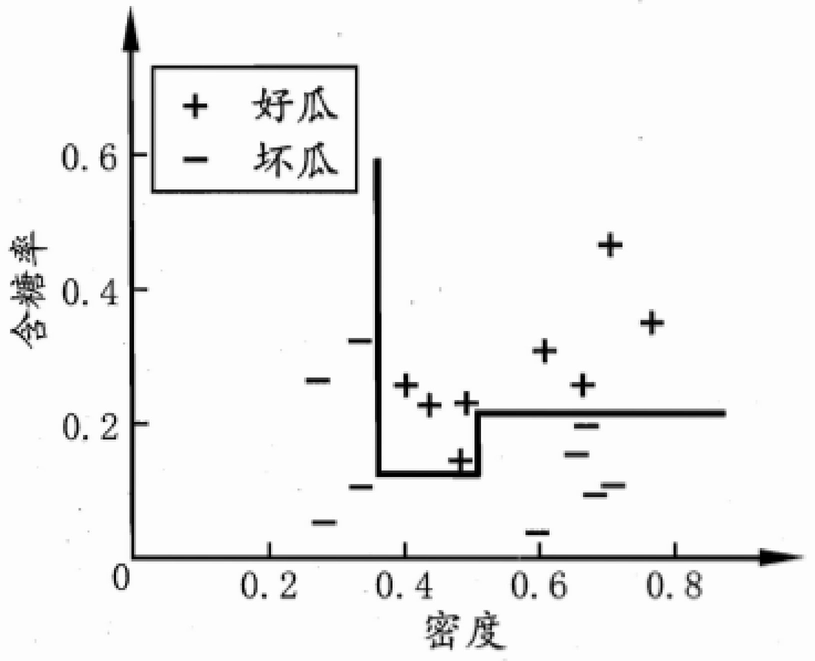


图 4.11 图 4.10 决策树对应的分类边界

如果能使用斜的划分边界，时间成本将大大减少

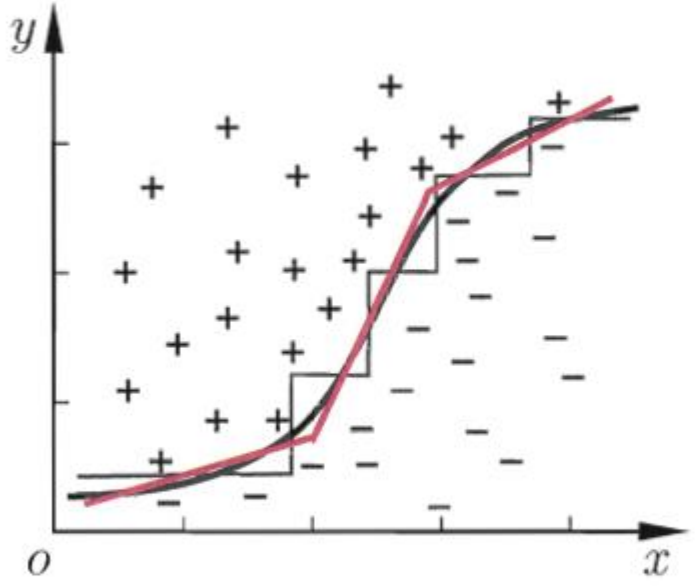


图 4.12　决策树对复杂分类边界的分段近似

“多变量决策树” 就是能实现这样的“斜划分” 甚至更复杂划分的决策树. 以实现斜划分的多变量决策树为例, 每个非叶结点是一个形如 $\sum_{i=1}^{d} w_i a_i = t$ 的线性分类器, 其中 w_i 是属性 a_i 的权重, w_i 和 t 可在该结点所含的样本集和属性集上学得. 于是, 与传统的“单 变量决策树”不同, 在多变量决策树的学习过程中, 不是为每个非叶结点寻找一个最优划分属性, 而是试图建立一个合适的线性分类器。

以西瓜数据为例

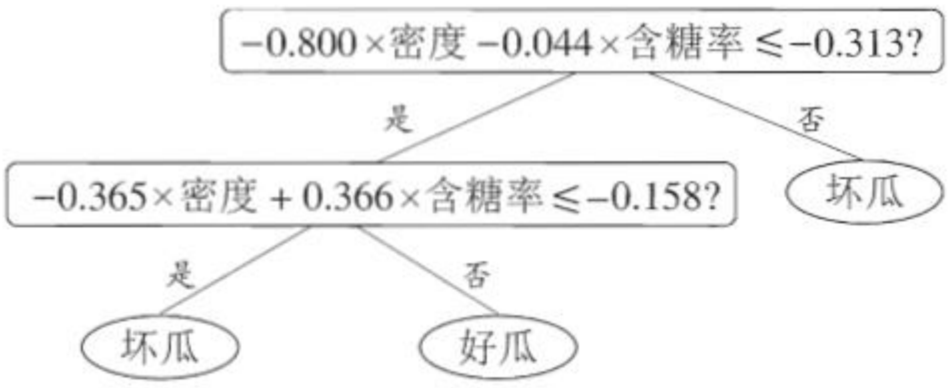


图 4.13 在西瓜数据集 3.0α 上生成的多变量决策树

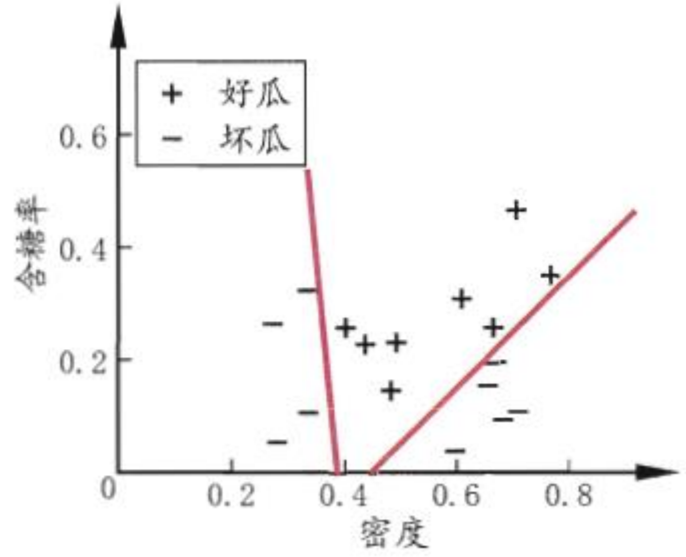


图 4.14 图 4.13 多变量决策树对应的分类边界

Part3 案例分析

我们使用威斯康星州乳腺癌数据集，该数据集可以从sklearn中导入。

In[4]:

```
from sklearn.datasets import load_breast_cancer
cancer = load_breast_cancer()
print("cancer.keys(): \n{}".format(cancer.keys()))
```

Out[4]:

```
cancer.keys():
dict_keys(['feature_names', 'data', 'DESCR', 'target',
'target_names'])
```

In[58]:

```
from sklearn.model_selection import train_test_split
from sklearn import tree
cancer = load_breast_cancer()
X_train, X_test, y_train, y_test = train_test_split(
 cancer.data, cancer.target, stratify=cancer.target, random_state=42)
a = DecisionTreeClassifier(random_state=0)
a.fit(X_train, y_train)
print("Accuracy on training set: {:.3f}".format(a.score(X_train, y_train)))
print("Accuracy on test set: {:.3f}".format(a.score(X_test, y_test)))
```

Out[58]:

```
Accuracy on training set: 1.000
Accuracy on test set: 0.937
```

不出所料，训练集上的精度是 100%，这是因为叶结点都是纯的，树的深度很大，足以完美地记住训练数据的所有标签。但是这常常会导致过拟合。对新数据的泛化性能不佳。现在我们将预剪枝应用在决策树上，这可以在完美拟合训练数据之前阻止树的展开。一种选择是在到达一定深度后停止树的展开。这里我们设置 max_depth=4，这意味着只可以连续问4个问题。限制树的深度可以减少过拟合。这会降低训练集的精度，但可以提高测试集的精度：

In[59]:

```
a= DecisionTreeClassifier(max_depth=4, random_state=0)
a.fit(X_train, y_train)
print("Accuracy on training set: {:.3f}".format(a.score(X_train, y_train)))
print("Accuracy on test set: {:.3f}".format(a.score(X_test, y_test)))
```

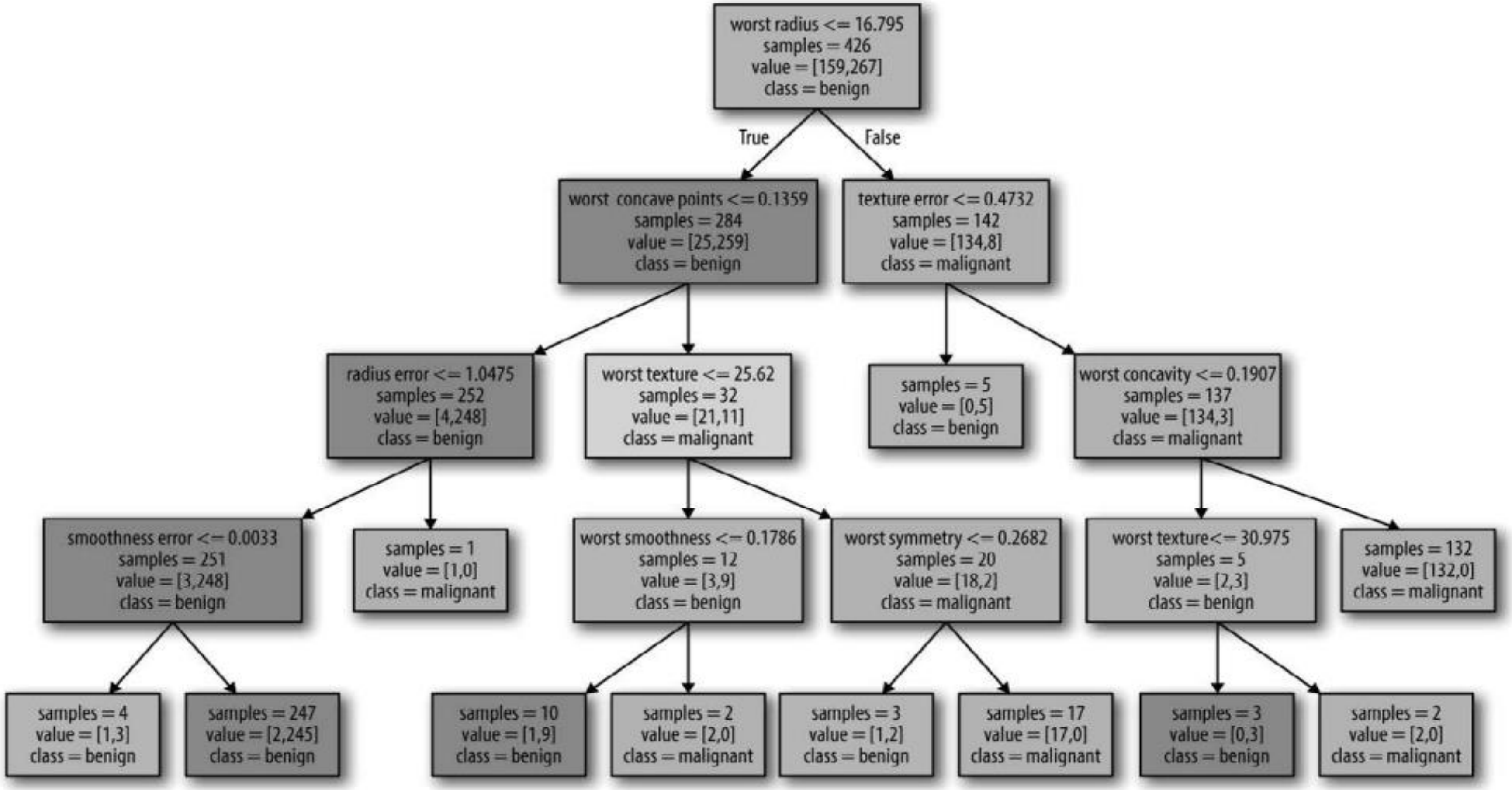
Out[59]:

```
Accuracy on training set: 0.988
Accuracy on test set: 0.951
```

最后我们将数据可视化

In[60]:

```
a=a.fit(X_train, y_train)
tree.plot_tree(a)
```


worst radius <= 16.795
samples = 426
value = [159,267]
class = benign
True
False
worst concave points <= 0.1359
samples = 284
value = [25,259]
class = benign
texture error <= 0.4732
samples = 142
value = [134,8]
class = malignant
radius error <= 1.0475
samples = 252
value = [4,248]
class = benign
worst texture <= 25.62
samples = 32
value = [21,11]
class = malignant
samples = 5
value = [0,5]
class = benign
worst concavity <= 0.1907
samples = 137
value = [134,3]
class = malignant
smoothness error <= 0.0033
samples = 251
value = [3,248]
class = benign
samples = 1
value = [1,0]
class = malignant
worst smoothness <= 0.1786
samples = 12
value = [3,9]
class = benign
worst symmetry <= 0.2682
samples = 20
value = [18,2]
class = malignant
worst texture <= 30.975
samples = 5
value = [2,3]
class = benign
samples = 132
value = [132,0]
class = malignant
samples = 4
value = [1,3]
class = benign
samples = 247
value = [2,245]
class = benign
samples = 10
value = [1,9]
class = benign
samples = 2
value = [2,0]
class = malignant
samples = 3
value = [1,2]
class = benign
samples = 17
value = [17,0]
class = malignant
samples = 3
value = [0,3]
class = benign
samples = 2
value = [2,0]
class = malignant

- 决策树的一个主要缺点在于经常对训练数据过拟合。随机森林是解决这个问题的一种方法。随机森林本质上是许多决策树的集合，其中每棵树都和其他树略有不同。随机森林背后的思想是，每棵树的预测可能都相对较好，但可能对部分数据过拟合。如果构造很多树，并且每棵树的预测都很好，但都以不同的方式过拟合，那么我们可以对这些树的结果取平均值来降低过拟合。既能减少过拟合又能保持树的预测能力，这可以在数学上严格证明。

- 随机森林中的每棵树对数据自助采样，并且在指定的特征个数下对特征进行随机选择。这样保证了森林中的每棵树都不相同。
- 对乳腺癌的包含了100棵树的随机森林预测，可以看到在测试集上的准确率要高于单棵决策树。

- **In[70]:**
- from sklearn.ensemble import RandomForestClassifier
- X_train, X_test, y_train, y_test = train_test_split(
- cancer.data, cancer.target, random_state=0)
- forest = RandomForestClassifier(n_estimators=100, random_state=0)
- forest.fit(X_train, y_train)
- print("Accuracy on training set: {:.3f}".format(forest.score(X_train, y_train)))
- print("Accuracy on test set: {:.3f}".format(forest.score(X_test, y_test)))
- **Out[70]:**
- Accuracy on training set: 1.000
- Accuracy on test set: 0.972

THANK YOU