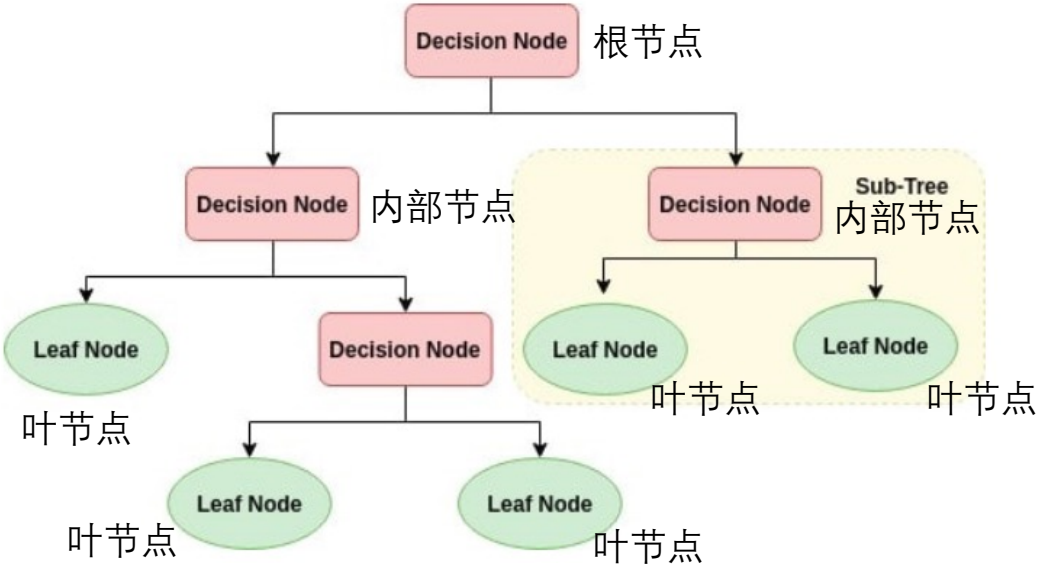
决策树

监督学习：决策树

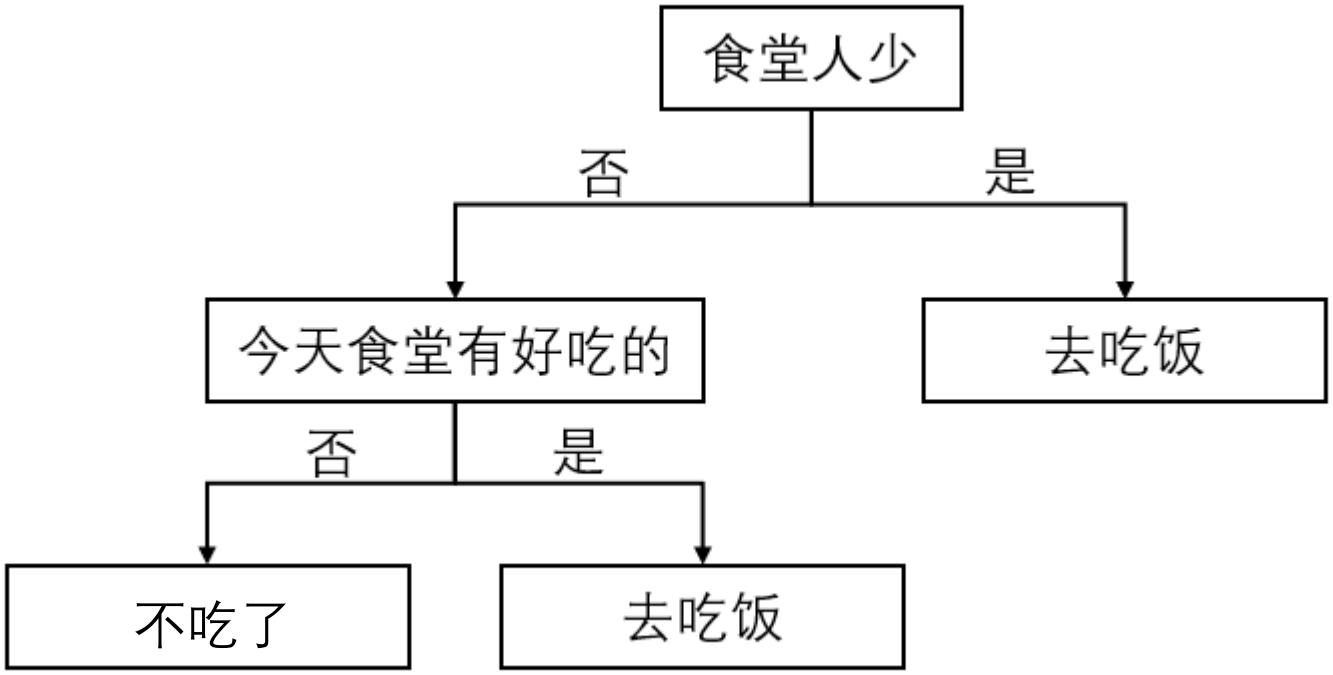
- 决策树是一种监督学习算法（1986年），既适用于分类，也适用于回归任务。
- 它呈现为一个树状结构，包括一个**根节点**、若干**内部节点**和多个**叶节点**。根节点一般代表整个数据集，内部节点代表数据的特征判断，叶节点则对应于最终的输出类别或数值结果。
- 在分类问题中，每个叶节点代表一个类别标签；在回归问题中，叶节点代表一个连续值。



- 常用的决策树有ID3，C4.5和CART（Classification And Regression Tree）

监督学习：决策树

- 决策树作为一种决策制定的策略，在日常生活中也被经常用到



监督学习：决策树

例

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

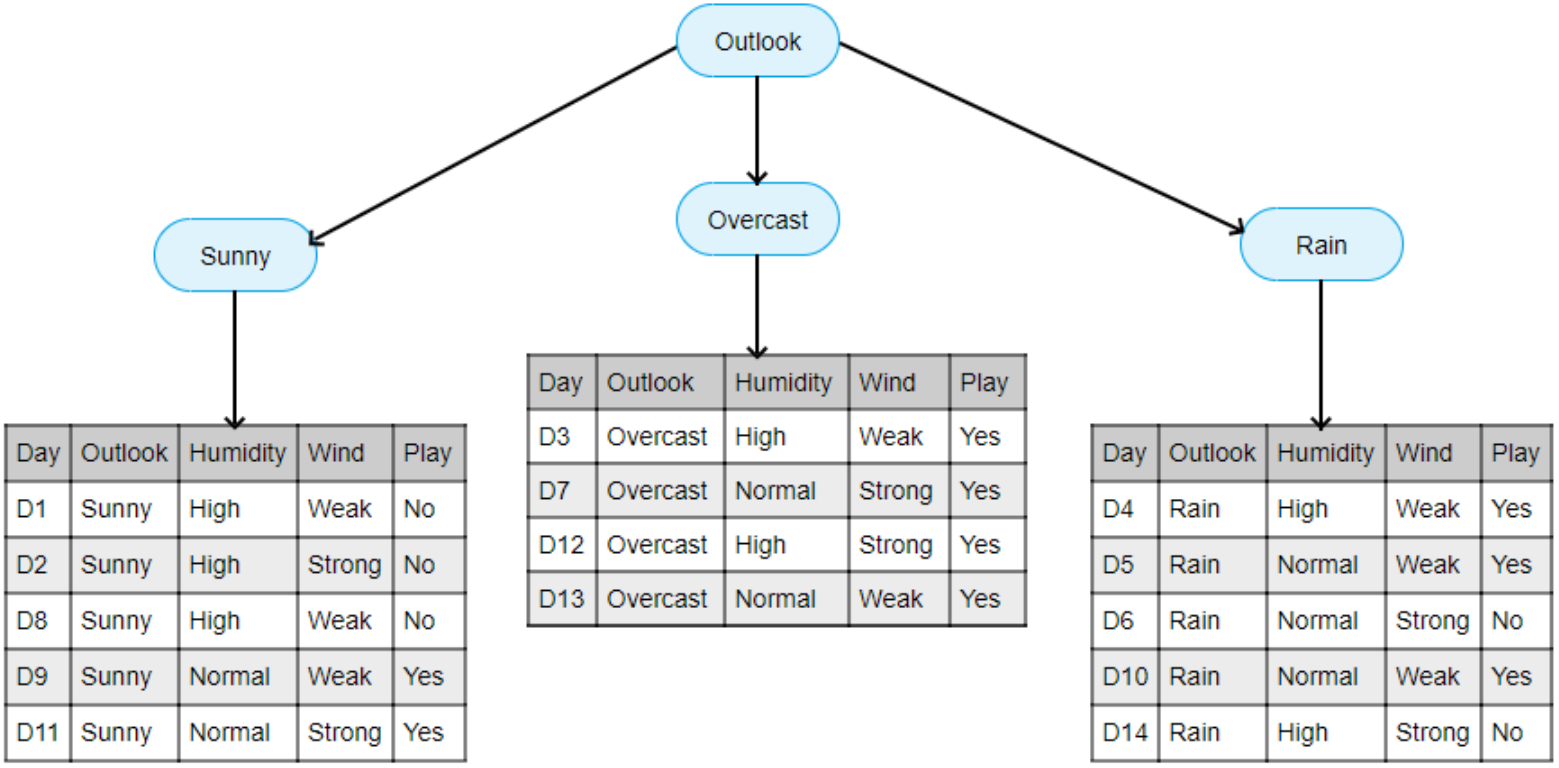
根据天气状况（outlook）、湿度（humidity）和风力（wind）这三个因素来判断当天是否会打网球。

输入：天气状况（outlook）、湿度（humidity）和风力（wind）

输出：是否打网球

监督学习：决策树

在决策树中，“pure node”（纯净节点）指的是一个节点（通常是叶节点）当中，所有的数据点都属于同一个类别。



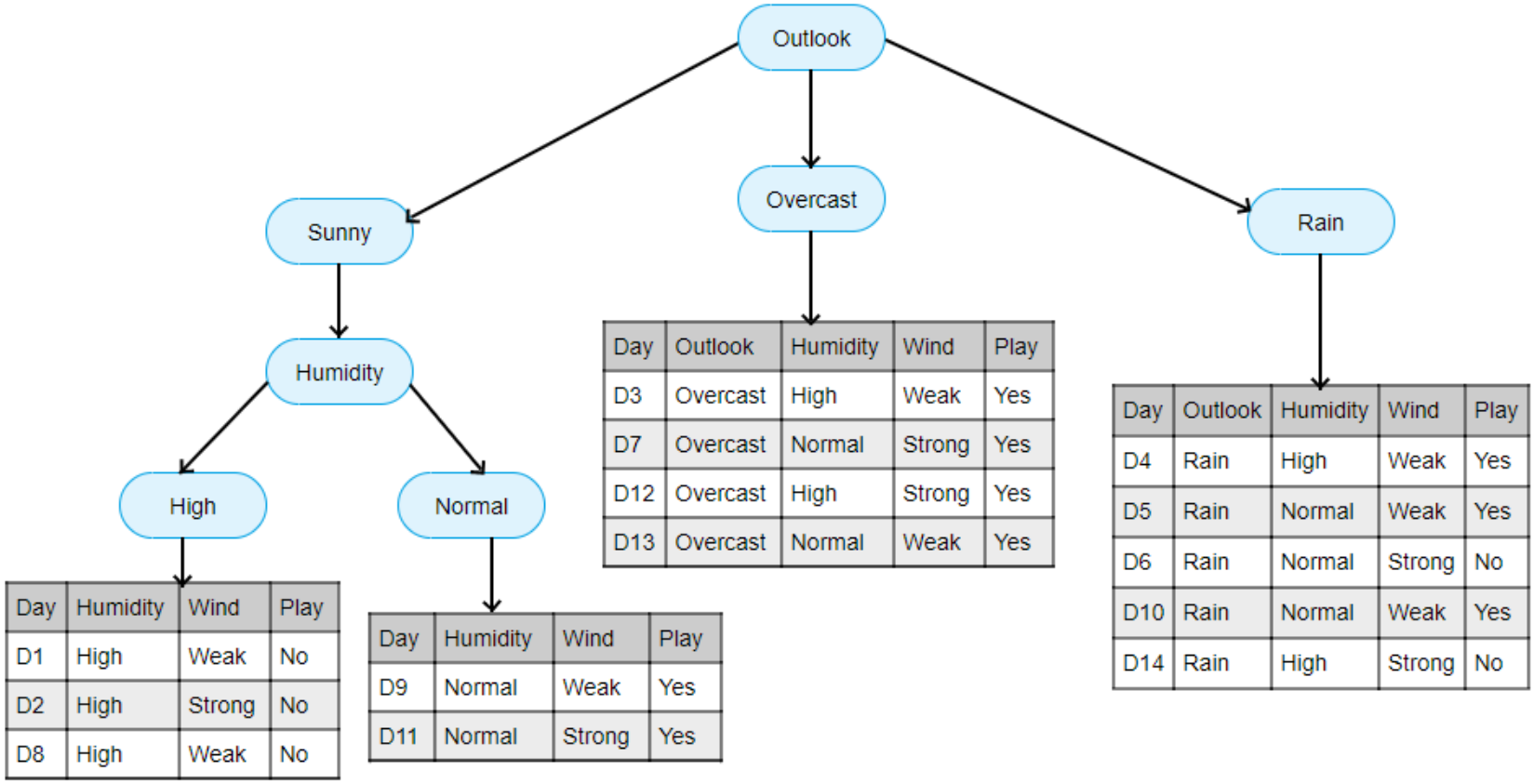
Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes

Day	Outlook	Humidity	Wind	Play
D3	Overcast	High	Weak	Yes
D7	Overcast	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes

Day	Outlook	Humidity	Wind	Play
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D10	Rain	Normal	Weak	Yes
D14	Rain	High	Strong	No

- 多云（Overcast）的情况下，小明肯定会打网球 （pure node）
- 晴天（Sunny）或者雨天(Rain)的情况下，不确定是否会打网球（impure node）

监督学习：决策树



Day	Humidity	Wind	Play
D1	High	Weak	No
D2	High	Strong	No
D8	High	Weak	No

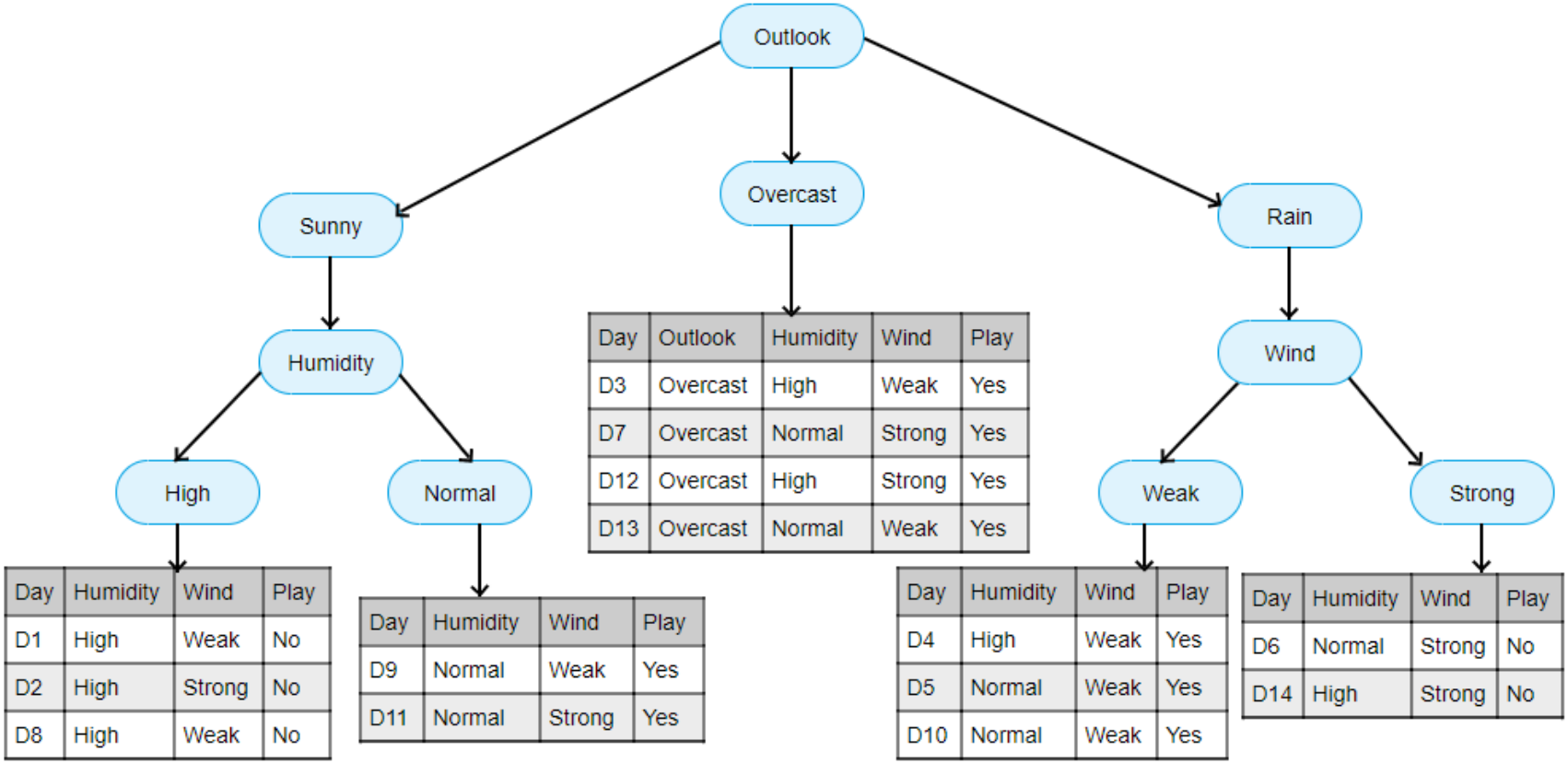
Day	Humidity	Wind	Play
D9	Normal	Weak	Yes
D11	Normal	Strong	Yes

Day	Outlook	Humidity	Wind	Play
D3	Overcast	High	Weak	Yes
D7	Overcast	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes

Day	Outlook	Humidity	Wind	Play
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D10	Rain	Normal	Weak	Yes
D14	Rain	High	Strong	No

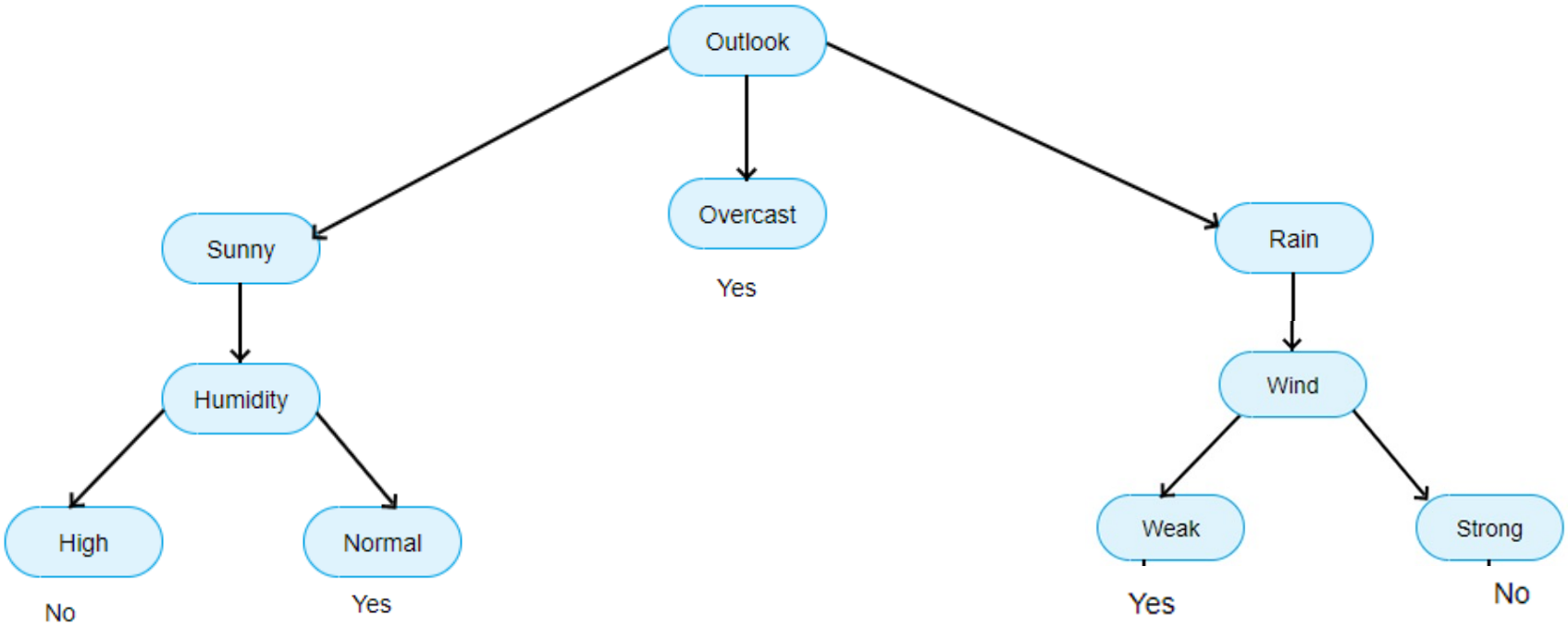
- 晴天情况下，对湿度进行分类可以得到两个pure node

监督学习：决策树



- 雨天情况下，对风的强度进行分类可以得到两个pure node

监督学习：决策树



- 最终决定小明是否会玩网球的模型

特征划分选择

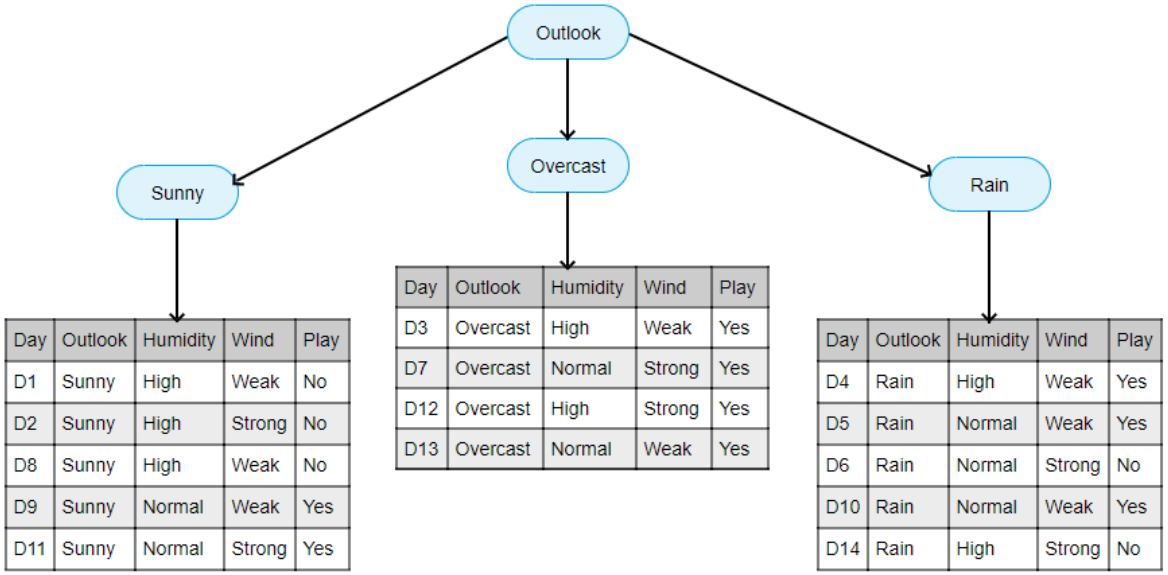
- 为了构建一个决策树，我们需要**确定哪个特征最适合作为根节点**。然后，我们需要为树的每个分支**选择适当的特征作为内部节点**，以便进一步细分数据集直至达到纯净节点（叶节点）。
- 我们可以用**信息熵**作为划分标准。**信息熵**是衡量数据无序程度的指标。初始时数据可能混杂，**熵值**较高。决策树通过其结构对数据进行分类，以达到更有序和一致的状态（pure node），这个时候数据的熵值就会很低。

计算某一数据子集的熵 $\mathrm{H}(T) = \mathrm{I}_E(p_1, p_2, \ldots, p_J) = -\sum_{i=1}^{J} p_i \log_2 p_i$

P_i代表着在一数据子集中，第i个数据类别出现的概率，它们的和等于1

特征划分选择

熵 $$\mathrm{H}(T) = \mathrm{I}_E(p_1, p_2, \ldots, p_J) = -\sum_{i=1}^{J} p_i \log_2 p_i$$



Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes

Day	Outlook	Humidity	Wind	Play
D3	Overcast	High	Weak	Yes
D7	Overcast	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes

Day	Outlook	Humidity	Wind	Play
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D10	Rain	Normal	Weak	Yes
D14	Rain	High	Strong	No

Sunny分支子集的熵: $$I_E([3,2]) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.9709508$$

Overcast分支子集的熵: $$I_E([0,4]) = -\frac{0}{4}log_2\frac{0}{4} - \frac{4}{4}log_2\frac{4}{4} = 0$$

Rain分支子集的熵: $$I_E([3,2]) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.9709508$$

特征划分选择

当Humidity作为根节点时：

High分支的熵：$I_E([3,4]) = -\frac{3}{7}log_2\frac{3}{7} - \frac{4}{7}log_2\frac{4}{7} = 0.98533$

Normal分支的熵：$I_E([1,6]) = -\frac{1}{7}log_2\frac{1}{7} - \frac{6}{7}log_2\frac{6}{7} = 0.59159$

当Wind作为根节点时：

Weak分支的熵：

Strong分支的熵：

特征划分选择

当Humidity作为根节点时：

High分支的熵：$I_E([3,4]) = -\frac{3}{7}log_2\frac{3}{7} - \frac{4}{7}log_2\frac{4}{7} = 0.98533$

Normal分支的熵：$I_E([1,6]) = -\frac{1}{7}log_2\frac{1}{7} - \frac{6}{7}log_2\frac{6}{7} = 0.59159$

当Wind作为根节点时：

Weak分支的熵：$I_E([6,2]) = -\frac{6}{8}log_2\frac{6}{8} - \frac{2}{8}log_2\frac{2}{8} = 0.81128$

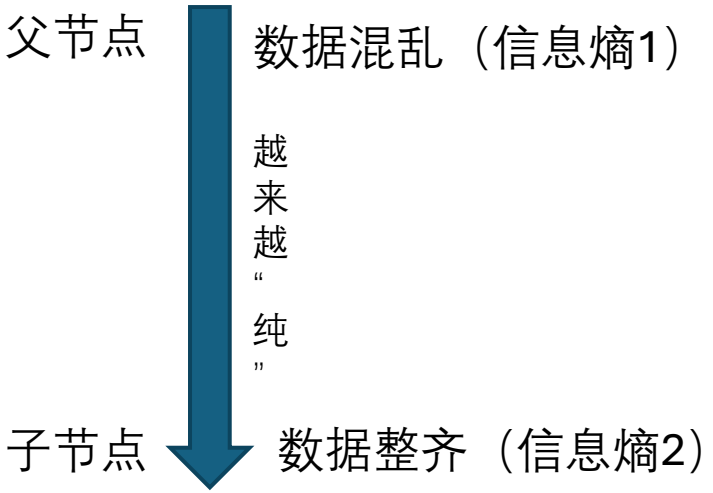
Strong分支的熵：$I_E([3,3]) = -\frac{3}{6}log_2\frac{3}{6} - \frac{3}{6}log_2\frac{3}{6} = 1.0$

特征划分选择

原数据集的信息熵（不论谁做根节点，根节点信息熵都是这个值）：

$$I_E([9,5]) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94029$$

信息增益（Information gain）



信息增益 = 信息熵1 - 信息熵2

特征划分选择

信息增益（Information gain）

用属性a对样本集D进行划分所获得的"信息增益" (information gain)

$$\mathrm{Gain}(D,a)=\mathrm{Ent}(D)-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\mathrm{Ent}(D^v)\ .$$

考虑到不同的分支结点所包含的样本数不同，给分支结点赋予权重。

当Humidity作为父节点时：

High分支的熵：$I_E([3,4])=-\frac{3}{7}log_2\frac{3}{7}-\frac{4}{7}log_2\frac{4}{7}=0.98533$

Normal分支的熵：$I_E([1,6])=-\frac{1}{7}log_2\frac{1}{7}-\frac{6}{7}log_2\frac{6}{7}=0.59159$

$$Gain(humidity)=0.94029-\left(0.98533\times\frac{7}{14}+0.59159\times\frac{7}{14}\right)=0.15183$$

特征划分选择

当Wind作为父节点时：

Weak分支的熵：$I_E([6,2]) = -\frac{6}{8}log_2\frac{6}{8} - \frac{2}{8}log_2\frac{2}{8} = 0.81128$

Strong分支的熵：$I_E([3,3]) = -\frac{3}{6}log_2\frac{3}{6} - \frac{3}{6}log_2\frac{3}{6} = 1.0$

$$Gain(wind) = 0.94029 - \left(0.81128\times\frac{8}{14} + 1.0\times\frac{6}{14}\right) = 0.04813$$

当Outlook作为父节点时：

Sunny分支的熵：$I_E([3,2]) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.9709508$

Overcast分支的熵：$I_E([0,4]) = -\frac{0}{4}log_2\frac{0}{4} - \frac{4}{4}log_2\frac{4}{4} = 0$

Rain分支的熵：$I_E([3,2]) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.9709508$

$Gain(Outlook) = ?$

特征划分选择

当Wind作为父节点时：

Weak分支的熵：$I_E([6,2]) = -\frac{6}{8}log_2\frac{6}{8} - \frac{2}{8}log_2\frac{2}{8} = 0.81128$

Strong分支的熵：$I_E([3,3]) = -\frac{3}{6}log_2\frac{3}{6} - \frac{3}{6}log_2\frac{3}{6} = 1.0$

$$Gain(wind) = 0.94029 - \left(0.81128\times\frac{8}{14} + 1.0\times\frac{6}{14}\right) = 0.04813$$

当Outlook作为父节点时：

Sunny分支的熵：$I_E([3,2]) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.9709508$

Overcast分支的熵：$I_E([0,4]) = -\frac{0}{4}log_2\frac{0}{4} - \frac{4}{4}log_2\frac{4}{4} = 0$

Rain分支的熵：$I_E([3,2]) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.9709508$

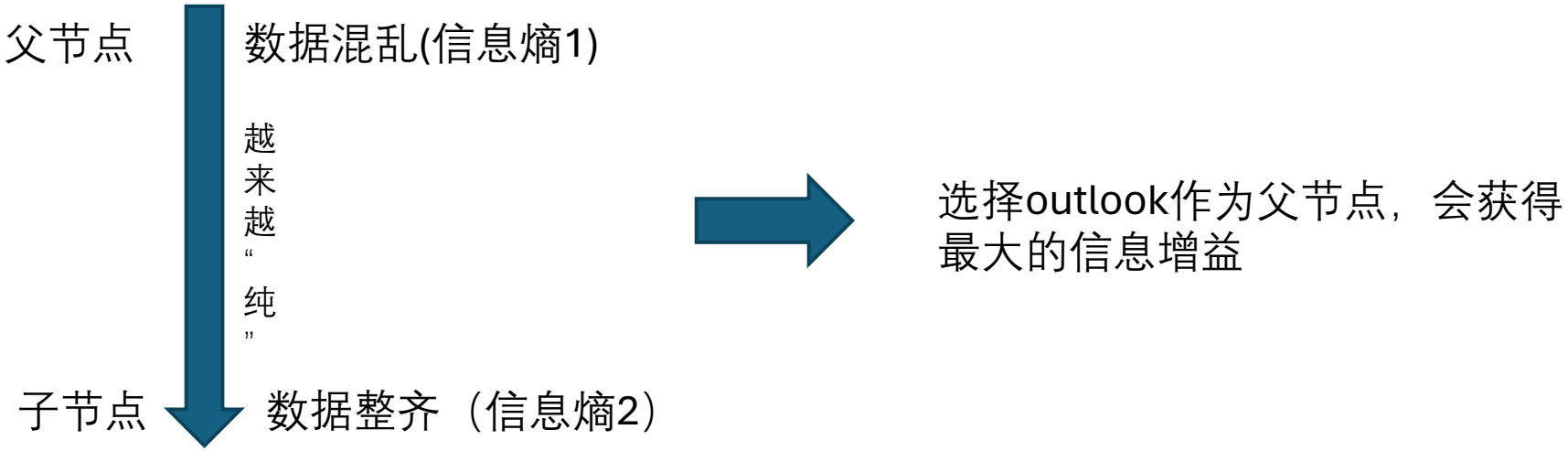
$$Gain(Outlook) = 0.94029 - \left(0.97095\times\frac{5}{14} + 0\times\frac{4}{14} + 0.97095\times\frac{5}{14}\right) = 0.24675$$

特征划分选择

$$Gain(humidity) = 0.94029 - \left(0.98533 \times \frac{7}{14} + 0.59159 \times \frac{7}{14}\right) = 0.15183$$

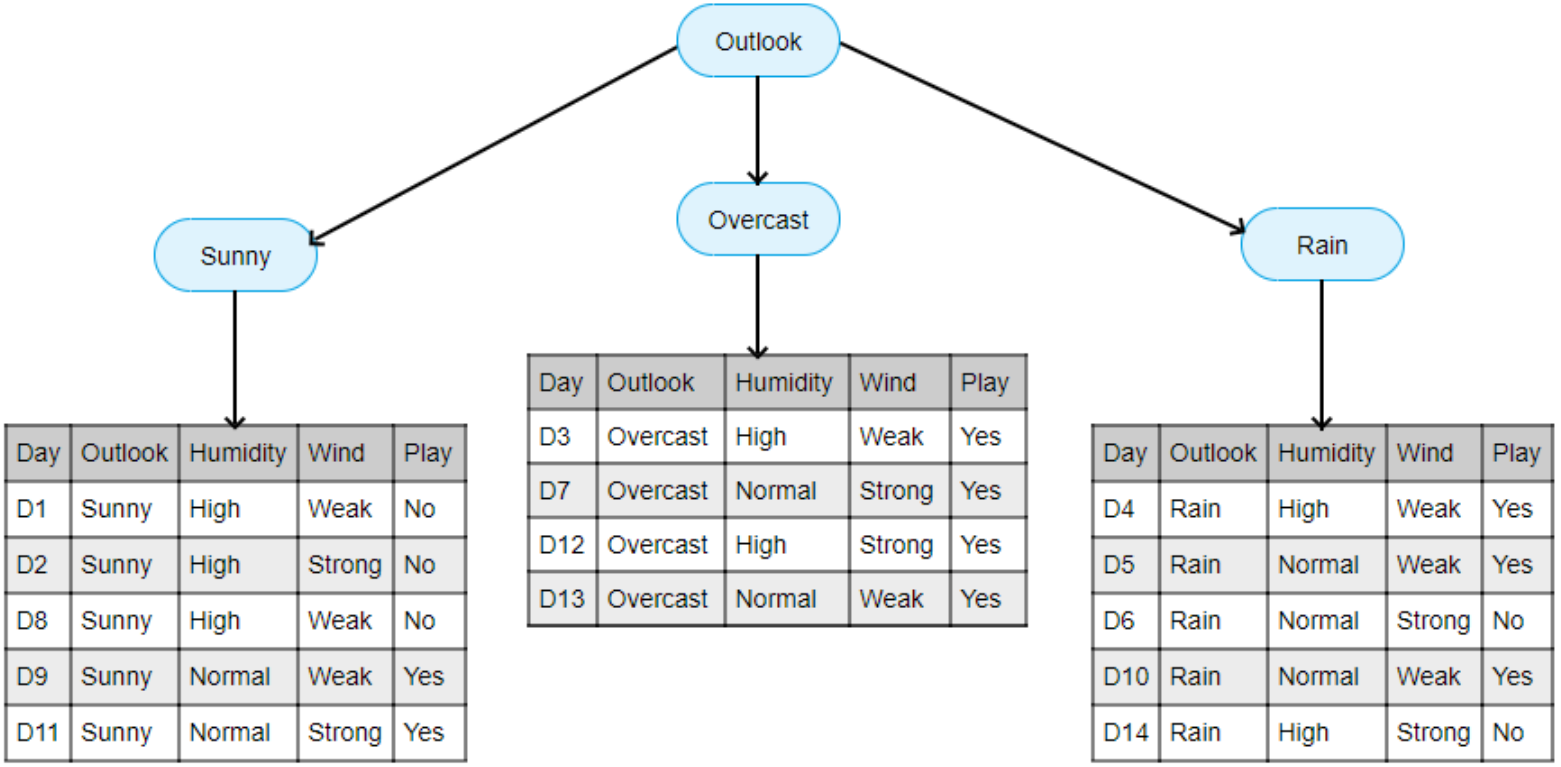
$$Gain(wind) = 0.94029 - \left(0.81128 \times \frac{8}{14} + 1.0 \times \frac{6}{14}\right) = 0.04813$$

$$Gain(Outlook) = 0.94029 - \left(0.97095 \times \frac{5}{14} + 0 \times \frac{4}{14} + 0.97095 \times \frac{5}{14}\right) = 0.24675$$



特征划分选择

计算到这里，我们才确定了根节点为outlook



Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes

Day	Outlook	Humidity	Wind	Play
D3	Overcast	High	Weak	Yes
D7	Overcast	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes

Day	Outlook	Humidity	Wind	Play
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D10	Rain	Normal	Weak	Yes
D14	Rain	High	Strong	No

Sunny的子节点是Humidity还是wind还要经过相似的计算。

这种用信息增益为准则来划分属性（特征）的方法就是**ID3决策树**学习算法

ID3决策树

ID3代表“Iterative Dichotomiser 3”，其中：

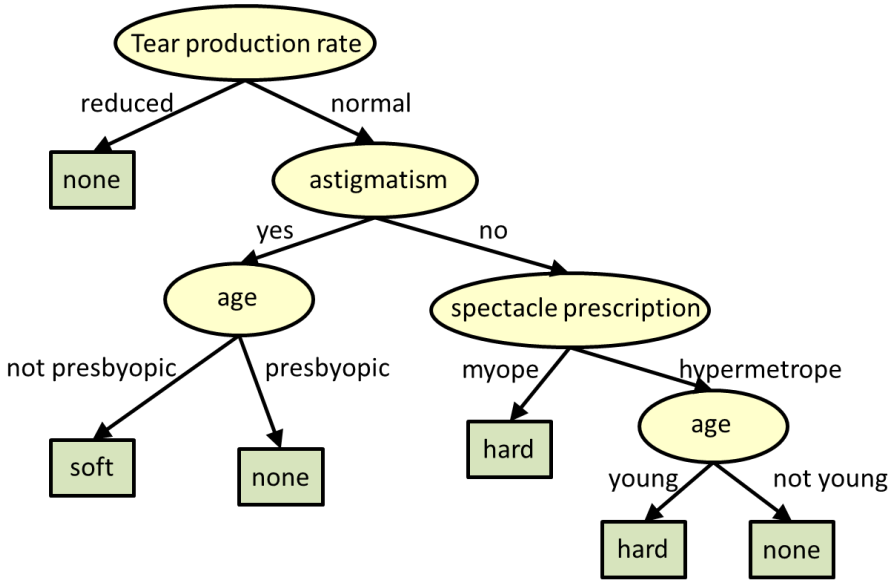
- **Iterative（迭代的）**：指的是这个算法通过迭代过程构造决策树。每次迭代选择最好的属性，然后在该属性的每个分支上递归重复这个过程，直到满足停止条件。

- **Dichotomiser（分裂法）**：这个词指的是算法在每次选择属性时，都是基于该属性的不同值来分裂数据集的。

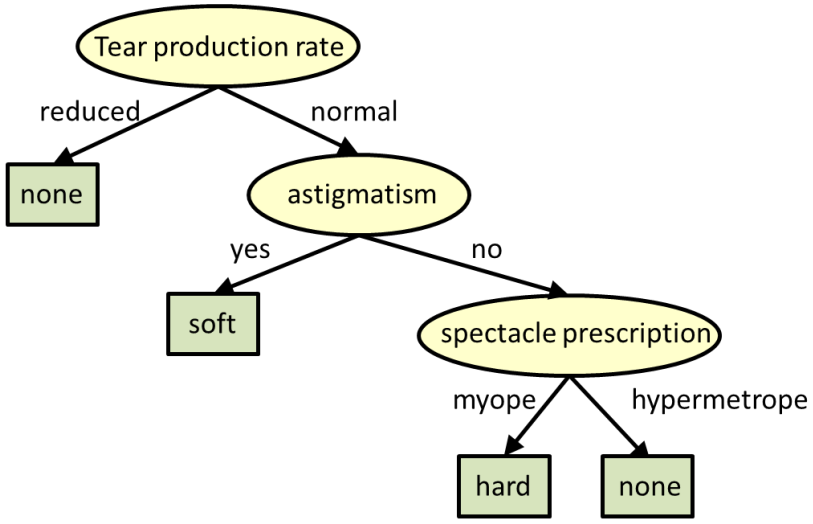
- **数字3**：指的是Iterative Dichotomiser的第三个版本。

剪枝

- **决策树剪枝**（pruning）是决策树算法中的一个操作，主要是减少决策树的大小，通过减小模型的复杂程度来避免过拟合。
- 决策树剪枝主要有两种类型：**预剪枝**和**后剪枝**



剪枝前



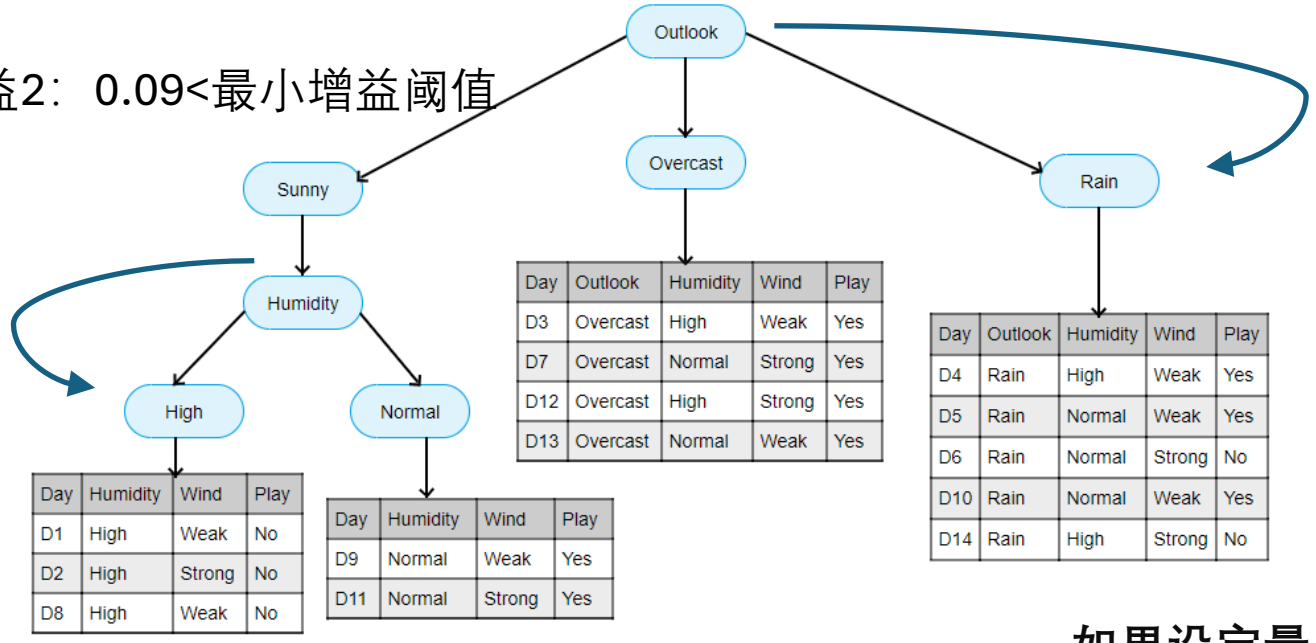
剪枝后

预剪枝（以信息增益为判断标准）

- 预剪枝中在每个节点上**检查信息增益是否大于最小增益阈值**。如果扩展某个节点对于改进模型的预测能力的贡献（即信息增益）小于某个预先设定的最小值，那么这个节点就不会进一步分支，而是成为一个叶节点。

信息增益1：0.25

信息增益2：0.09<最小增益阈值



Day	Outlook	Humidity	Wind	Play
D3	Overcast	High	Weak	Yes
D7	Overcast	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes

Day	Outlook	Humidity	Wind	Play
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D10	Rain	Normal	Weak	Yes
D14	Rain	High	Strong	No

Day	Humidity	Wind	Play
D1	High	Weak	No
D2	High	Strong	No
D8	High	Weak	No

Day	Humidity	Wind	Play
D9	Normal	Weak	Yes
D11	Normal	Strong	Yes

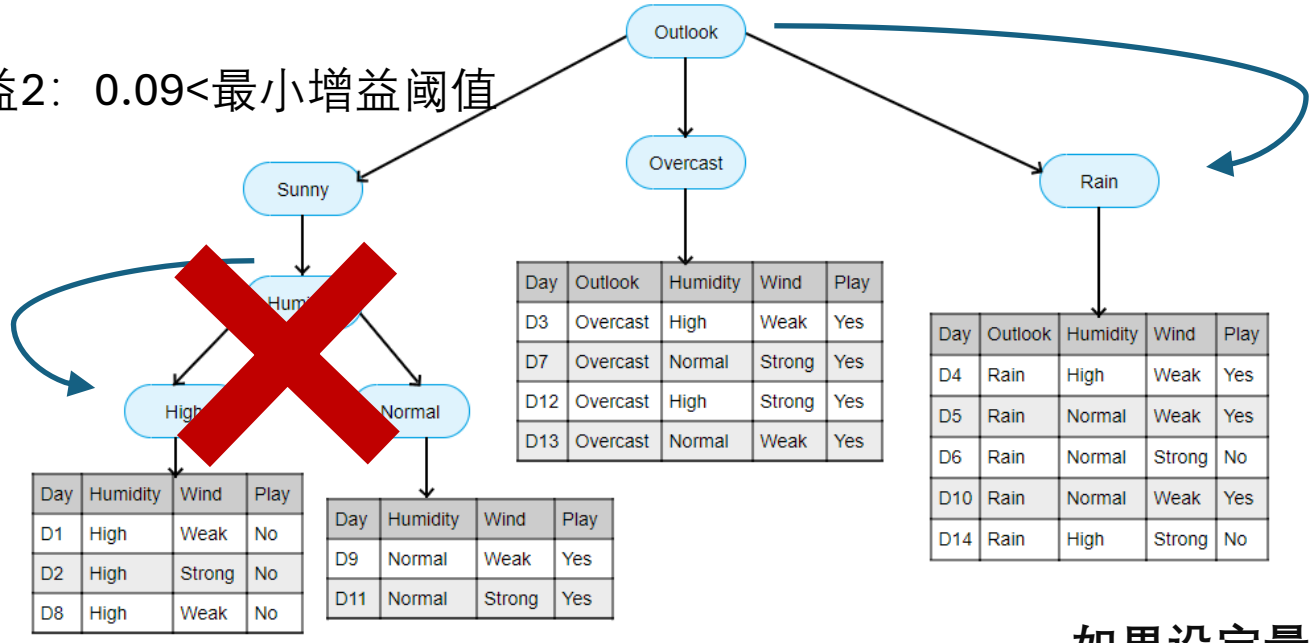
如果设定最小增益阈值：0.1

预剪枝（以信息增益为判断标准）

- 预剪枝中在每个节点上**检查信息增益是否大于最小增益阈值**。如果扩展某个节点对于改进模型的预测能力的贡献（即信息增益）小于某个预先设定的最小值，那么这个节点就不会进一步分支，而是成为一个叶节点。

信息增益1：0.25

信息增益2：0.09<最小增益阈值



Day	Humidity	Wind	Play
D1	High	Weak	No
D2	High	Strong	No
D8	High	Weak	No

Day	Humidity	Wind	Play
D9	Normal	Weak	Yes
D11	Normal	Strong	Yes

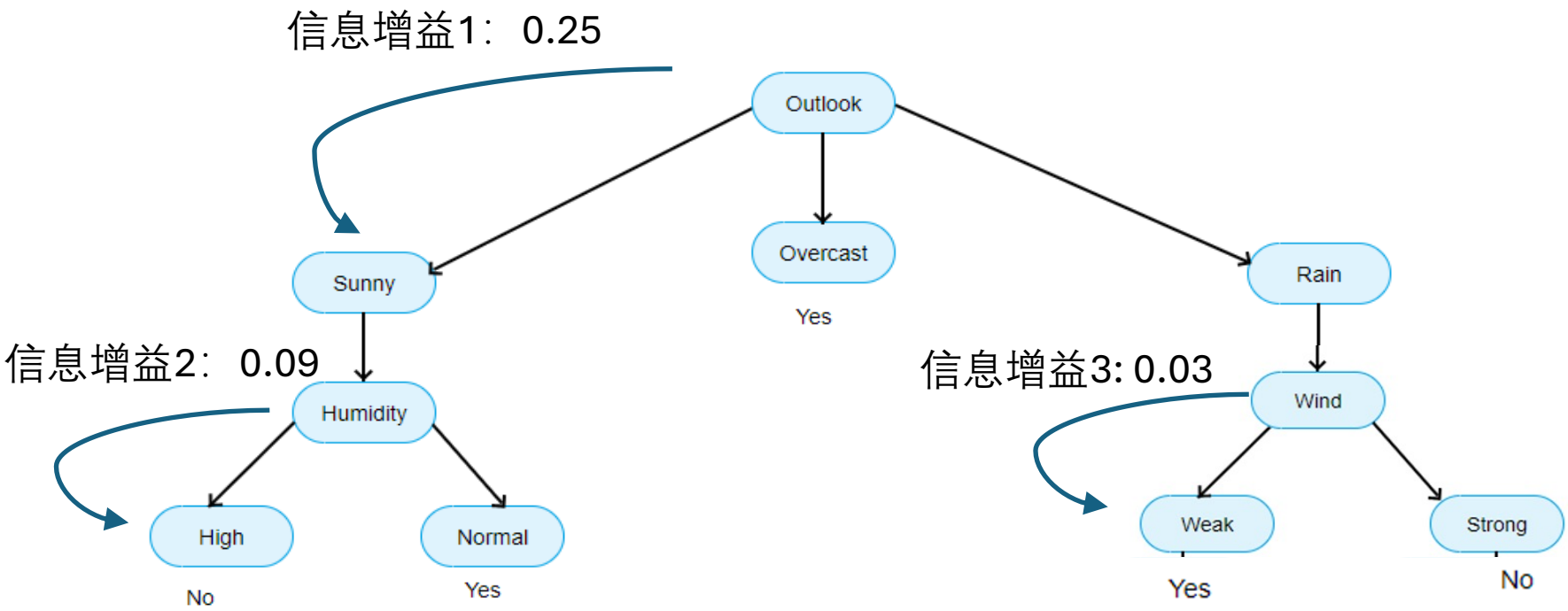
Day	Outlook	Humidity	Wind	Play
D3	Overcast	High	Weak	Yes
D7	Overcast	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes

Day	Outlook	Humidity	Wind	Play
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D10	Rain	Normal	Weak	Yes
D14	Rain	High	Strong	No

如果设定最小增益阈值：0.1

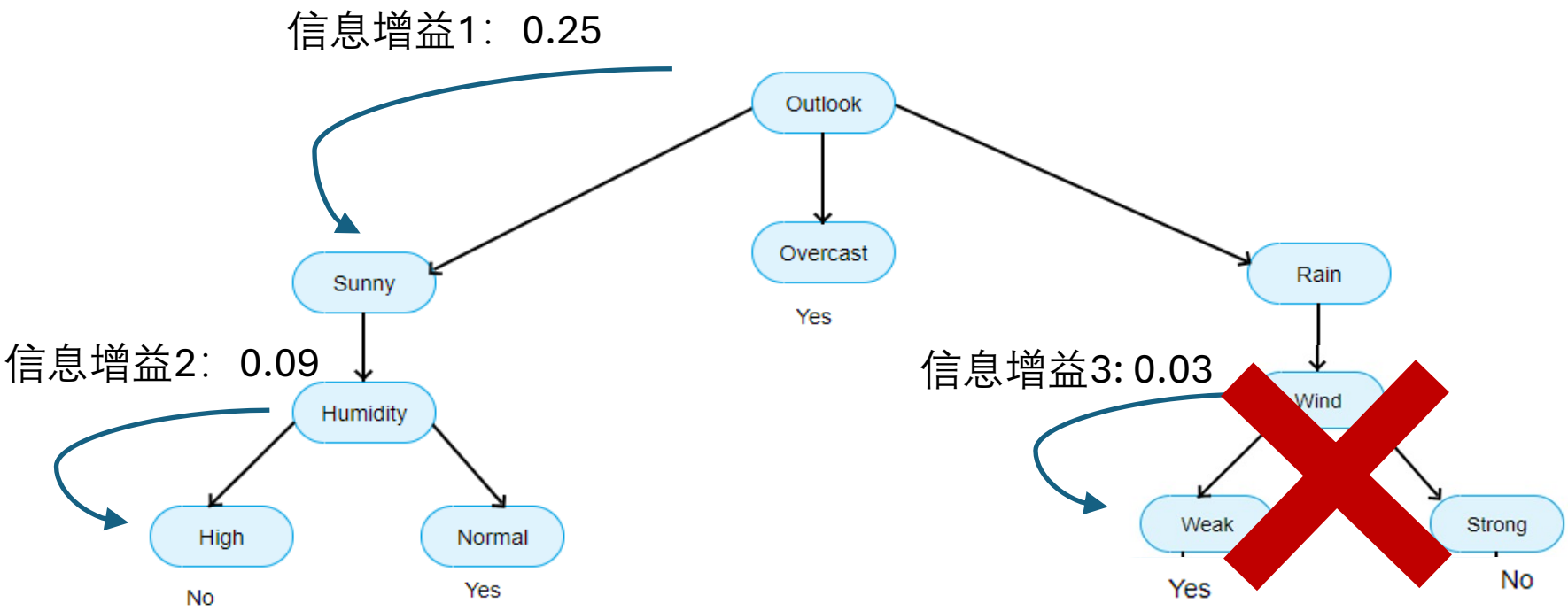
后剪枝（以信息增益为判断标准）

- **后剪枝**从一个构建完成的决策树开始，然后逐步删除那些提供最少信息增益的子树。直到达到了一个期望的叶节点数量，或者剪枝导致的性能损失超过了某个阈值。



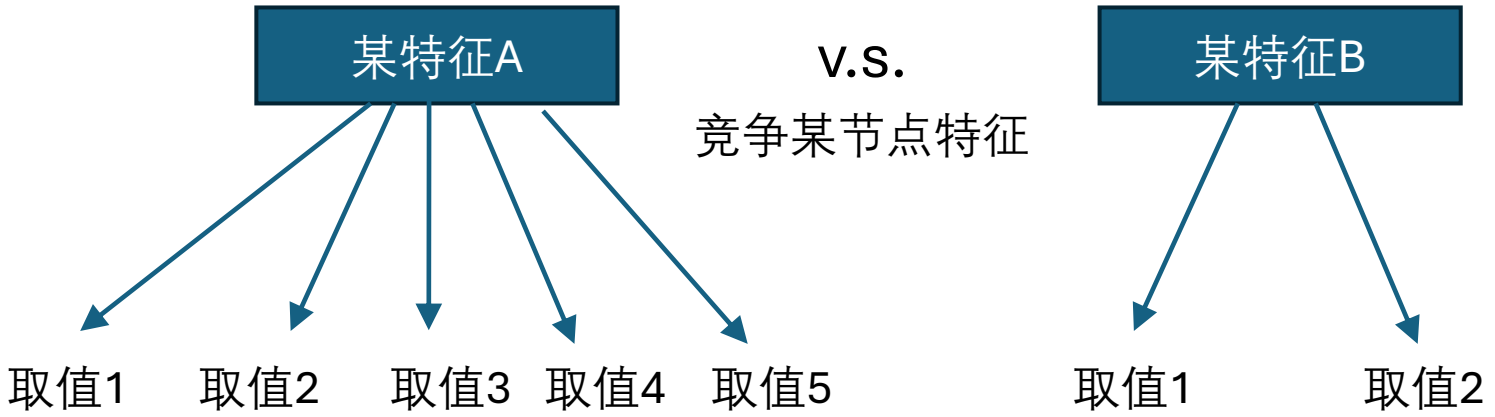
后剪枝（以信息增益为判断标准）

- **后剪枝**从一个构建完成的决策树开始，然后逐步删除那些提供最少信息增益的子树。直到达到了一个期望的叶节点数量，或者剪枝导致的性能损失超过了某个阈值。



信息增益的局限性

- 信息增益对于拥有更多取值的特征拥有偏向性。



- 当一个特征（如特征A）有很多可能的取值时，会导致数据集被分割成许多较小的子数据集，这些小数据集往往因为样本量较少且同质化较高而具有较低的熵。因此，计算出的信息增益会相对较大，使得这个特征看起来像是一个很好的选择用于分割数据。

信息增益率

- 为了避免信息增益准则的偏向性，引入信息增益率。
- 信息增益率（Gain Ratio）是C4.5决策树算法中使用的特征选择方法，它是基于信息增益（Information Gain）算法ID3的一个改进。

$$\text{信息增益率} = \frac{\text{信息增益}}{\text{分割信息}}$$

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

这种方式确保了当某特征拥有大量取值但每个取值的样本量很小时，其对应的增益率不会过高，从而避免了偏向性。

基尼不纯度

基尼不纯度（Gini Impurity）是CART（Classification and Regression Trees）决策树算法中用于度量数据集纯度的一个标准。它是一个衡量数据集中类别混杂程度的指标。基尼不纯度越低，表示数据集的纯度越高。

计算基尼不纯度

对于一个包含多个类别的数据集，基尼不纯度可以用以下公式计算：

$$Gini(p) = 1 - \sum_{i=1}^{n} p_i^2$$

其中：

- p_i表示数据集中第i个类别的概率。
- n是类别的总数。

基尼不纯度

$$Gini(p) = 1 - \sum_{i=1}^{n} p_i^2$$

- 如果数据集中的所有元素都属于同一个类别（即完全纯净），那么基尼不纯度为 0。如果数据集中的元素均匀地分布在各个类别中（即最不纯净），那么基尼不纯度达到最大值。

- CART决策树算法会寻找那些能最大程度降低不纯度的特征作为分裂的节点特征。

三种决策树比较

特性/算法	ID3	C4.5	CART
创造者	Ross Quinlan	Ross Quinlan	Leo Breiman, Jerome Friedman, Richard Olshen, Charles Stone
年份	1986年	1993年	1984年
处理类型	**分类问题**	**分类问题**	**分类和回归问题**
分割策略	多路分割	多路分割	二元分割
属性选择	**信息增益**	**增益率**	**Gini指数或平方误差最小化**
连续属性	不直接处理，需离散化	可以直接处理	可以直接处理
缺失数据	不直接处理	有策略处理	有策略处理
剪枝策略	无内建剪枝，需要外部方法	错误率降低剪枝（Reduced Error Pruning）	成本复杂性剪枝（Cost-Complexity Pruning）
过拟合防止	依赖外部剪枝	使用剪枝和规则集合	使用剪枝
输出	**分类标签**	**分类标签或概率**	**分类标签或数值预测**
软件实现	较少，大多已过时	C4.5是独立的程序，但现在更多使用改进版本，如C5.0	广泛应用，如R语言和Python的scikit-learn库