Machine Learning

Decision Tree

Dr. Shuang LIANG

Recall: KNN

L1(Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

L2(Euclidean) distance

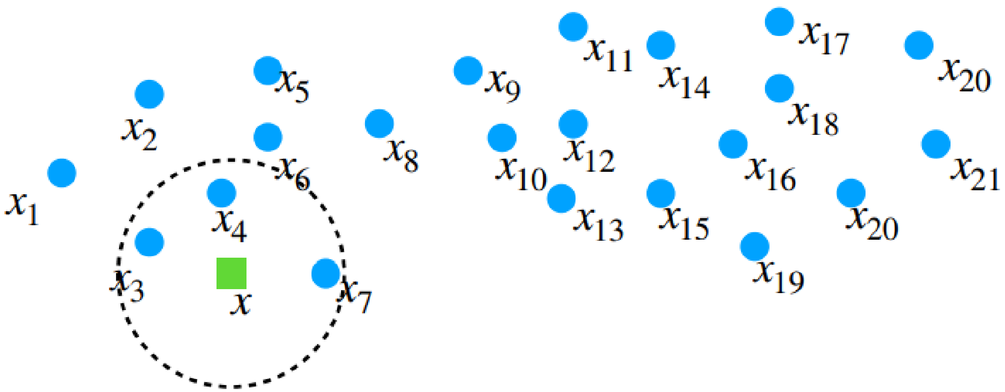
$$d_1(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$

Step1: Find nearest neighbors

$$nbh_{S_{train},k}: \mathcal{X} \rightarrow \mathcal{X}^k$$

$x \longmapsto \{k$ elements of S_{train} which are the closest to $x\}$

How to define the distance?

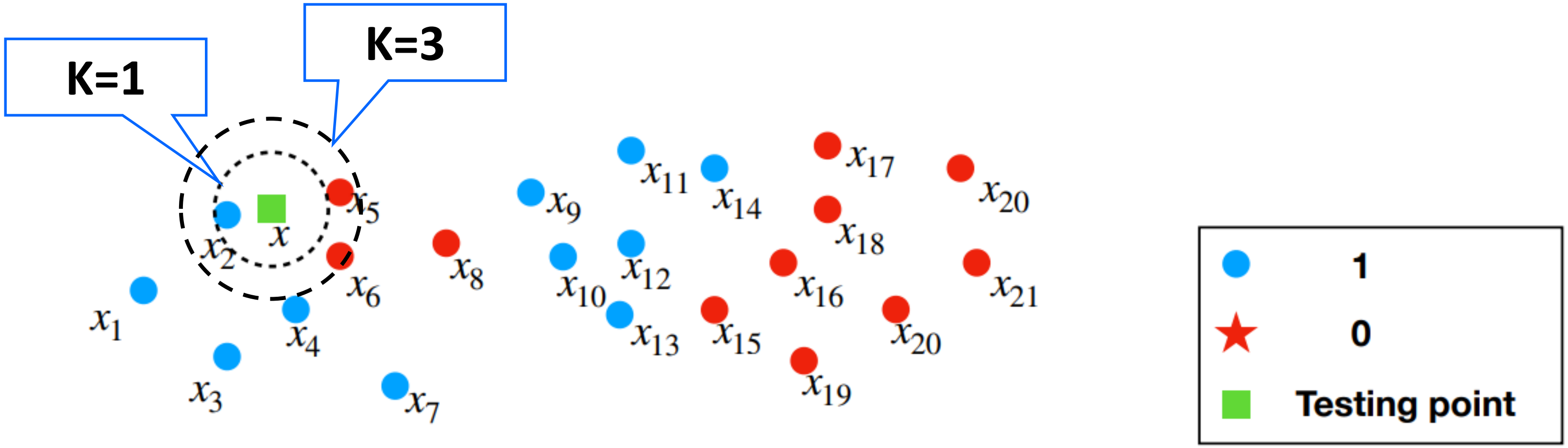


$$nbh_{S_{train},3}(x) = \{x_3, x_4, x_7\}$$

Recall: KNN

Step2: Select Class

$$f_{S_{train},k}(x) = majority\{y_i : x_i \in nbh_{S_{train},k}(x)\}$$



$$f_{S_{train},1}(x) = 1$$
$$f_{S_{train},3}(x) = 0$$

Today's Topics

- Type of classifiers
- Structure of the Decision Tree
- Build A Decision Tree
- Tree Pruning
- Continuous values
- Multivariate decision tree
- Random forest

Today's Topics

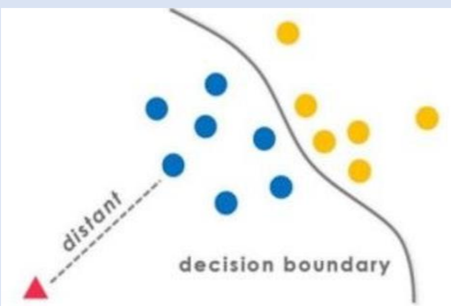
- ***Type of classifiers***
- Structure of the Decision Tree
- Build A Decision Tree
- Tree Pruning
- Continuous values
- Multivariate decision tree
- Random forest

Types of Classifiers

Model-based

Discriminative

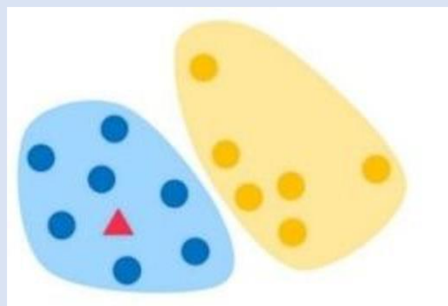
directly estimate a decision rule/boundary



Logistic regression
Decision tree
Neural network
……

Generative

build a generative statistical model



Naïve Bayes
Bayesian Networks
HMM
……

No Model

Instance-based

Use observation directly

KNN

Discriminative

- Only care about estimating the conditional probabilities $P(y|x)$
- Very good when underlying distribution of data is really complicated (e.g. texts, images, movies)

Generative

- Model observations (x, y) first ($P(x, y)$), then infer $P(y|x)$
- Good for missing variables, better diagnostics
- Easy to add prior knowledge about data

Today's Topics

- Type of classifiers
- ***Structure of the Decision Tree***
- Build A Decision Tree
- Tree Pruning
- Continuous values
- Multivariate decision tree
- Random forest

Structure of the Decision Tree

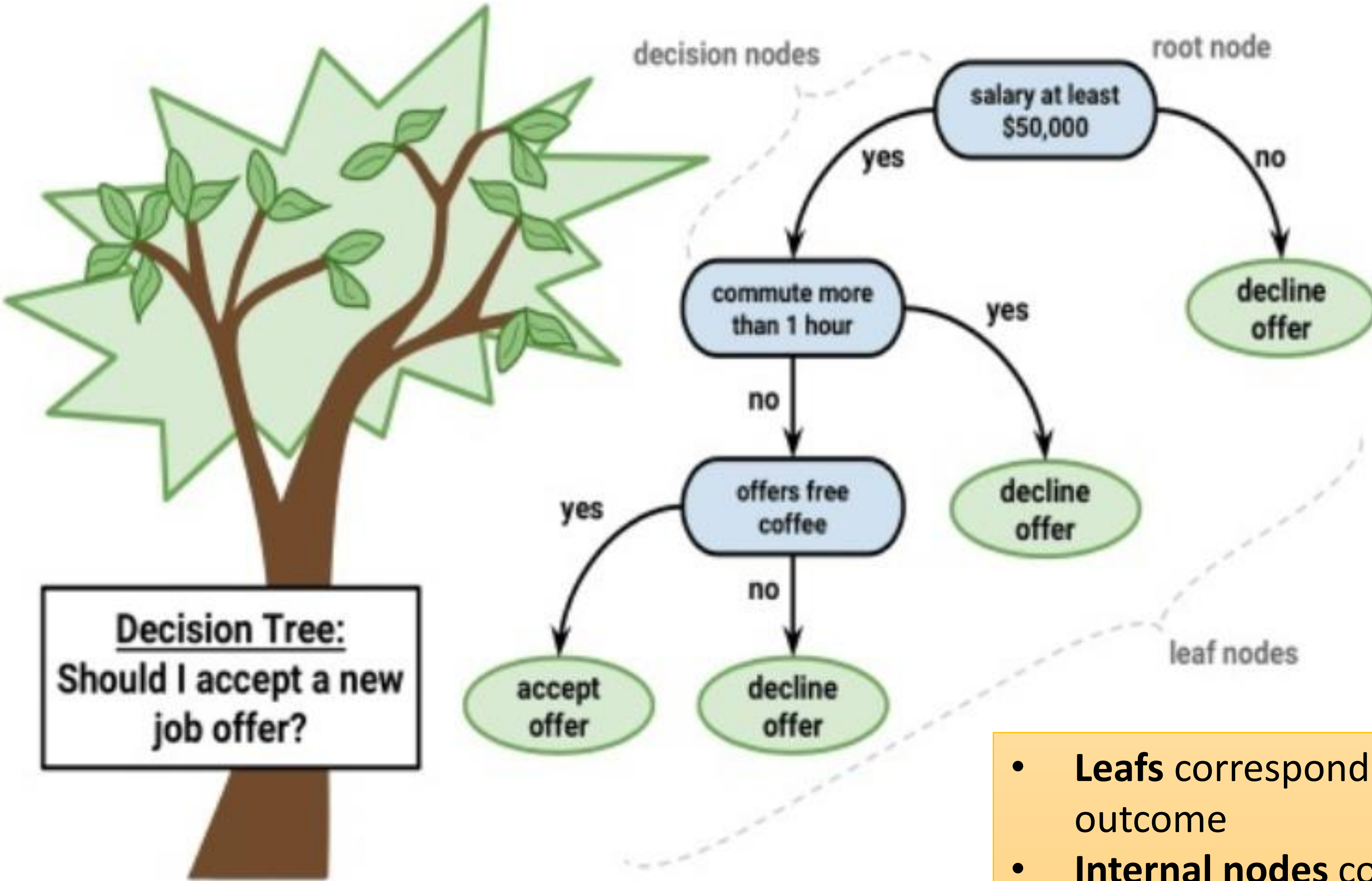
- **Why Decision Tree?**

✓One of the most intuitive classifiers

✓Easy to understand and construct

✓Surprisingly, also works very (very) well

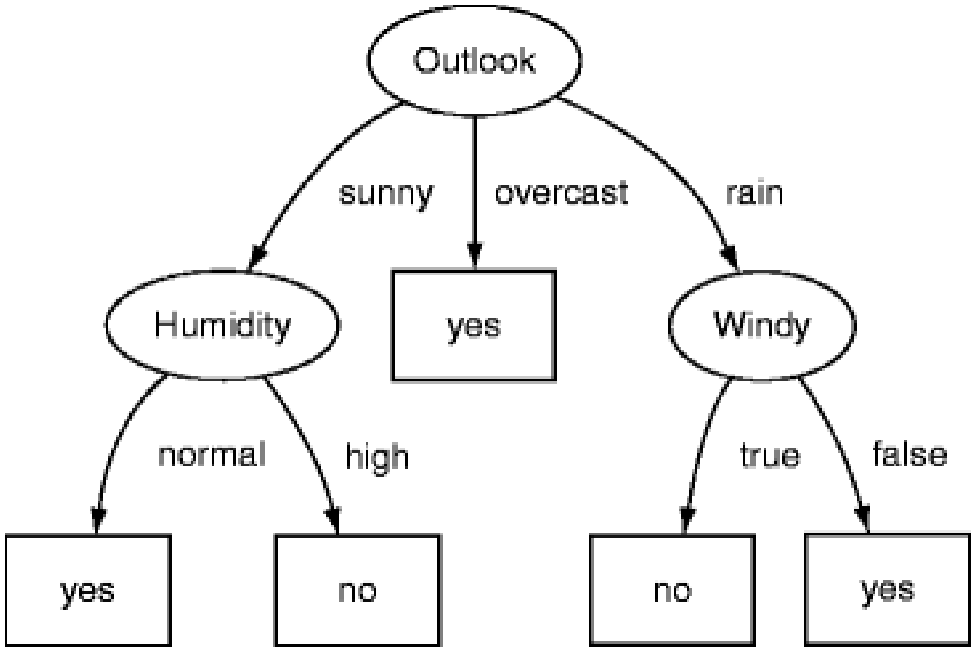
Structure of the Decision Tree



- **Leafs** correspond to classification outcome
- **Internal nodes** correspond to attributes (features)
- **Edges** denote assignment

Structure of the Decision Tree

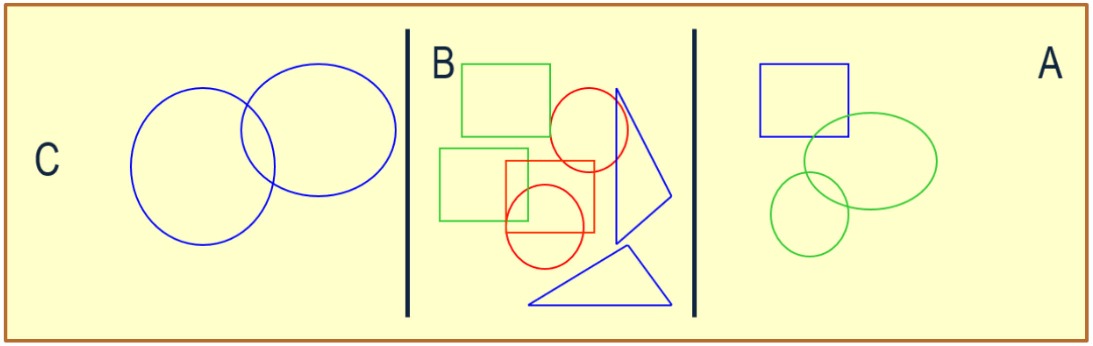
- **Case 1: golf playing**
- Examples: descriptions of weather conditions (Outlook, Humidity, Windy, Temperature)
- The target concept: whether these conditions are suitable for playing golf or not



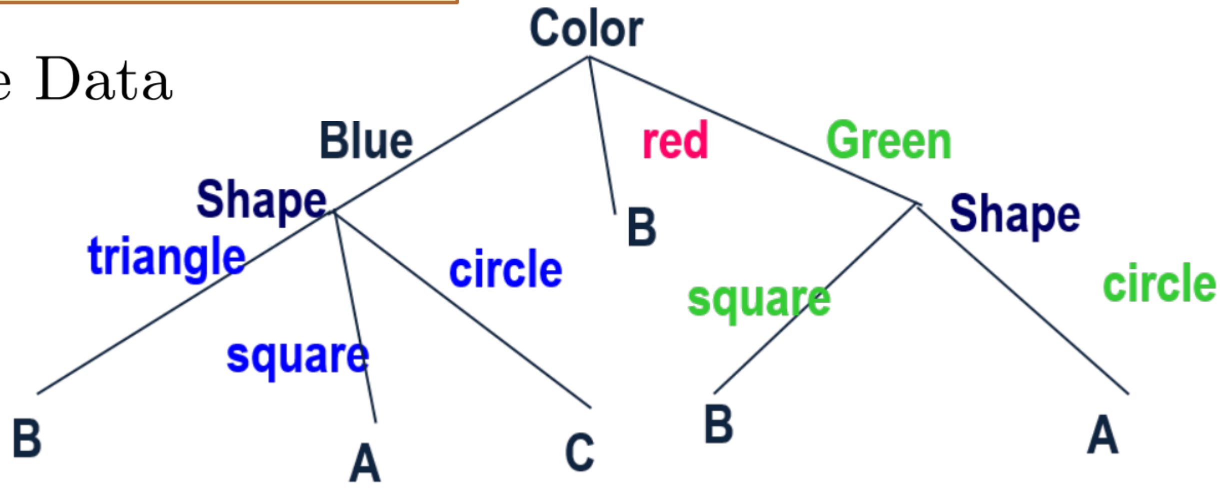
[1] Quinlan J R. Induction of decision trees[J]. Machine learning, 1986, 1(1): 81-106.

Structure of the Decision Tree

- **Case 2: shape prediction**



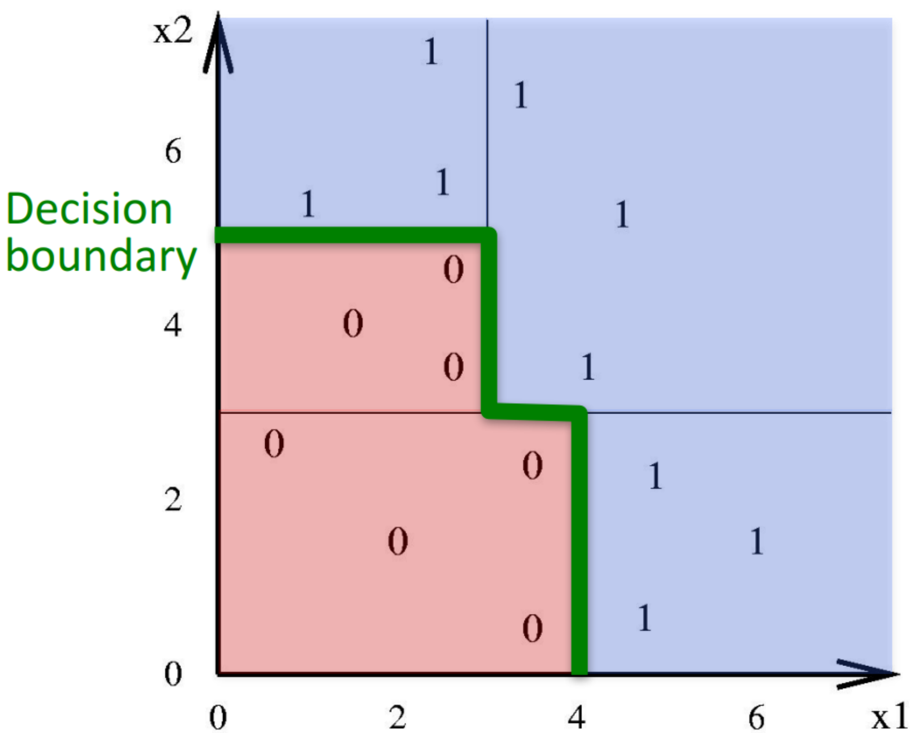
(a) Example Data

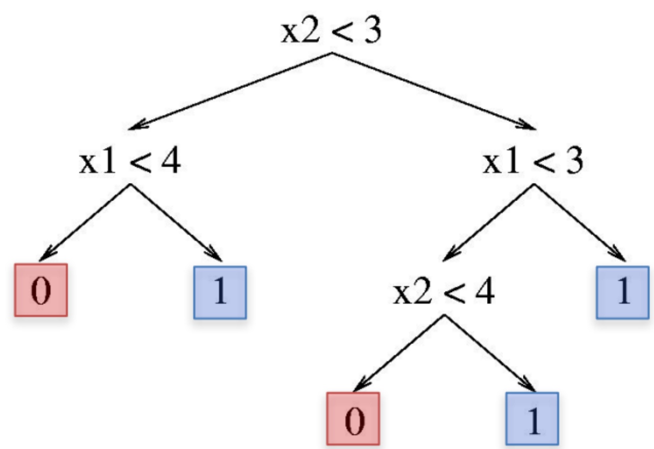


(b) Decision Tree

Structure of the Decision Tree

- **Case 3: binary classification**
- Decision trees divide the feature space into axis-parallel rectangles
- Each rectangular region is labeled with one label (or a probability distribution over labels)





Structure of the Decision Tree

- **Practice**

The following code can be viewed as a decision tree of three leaves. What is the output of the tree for **(income, debt) = (98765, 56789)?**

```
if (income > 100000)
    return true;
else {
    if (debt > 50000)
        return false;
    else return true;
}
```

A. **true**

B. **false**

C. **98765**

D. **56789**

Today's Topics

- Type of classifiers
- Structure of the Decision Tree
- ***Build A Decision Tree***
- Tree Pruning
- Continuous values
- Multivariate decision tree
- Random forest

Pseudo Code for Building A Decision Tree

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: 函数 TreeGenerate(D, A)

生成结点 node;
if D 中样本全属于同一类别 C **then**
    将 node 标记为 C 类叶结点; **return**
end if
if $A = \emptyset$ **OR** D 中样本在 A 上取值相同 **then**
    将 node 标记叶结点，其类别标记为 D 中样本数最多的类; **return**
end if
从 A 中选择最优划分属性 a_*; **The key step**
for a_* 的每一个值 a_*^v **do**
    为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
    if D_v 为空 **then**
        将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; **return**
    else
        以 TreeGenerate$(D_v, A - \{a_*\})$ 为分枝结点
    end if
end for

输出: 以 node 为根结点的一棵决策树

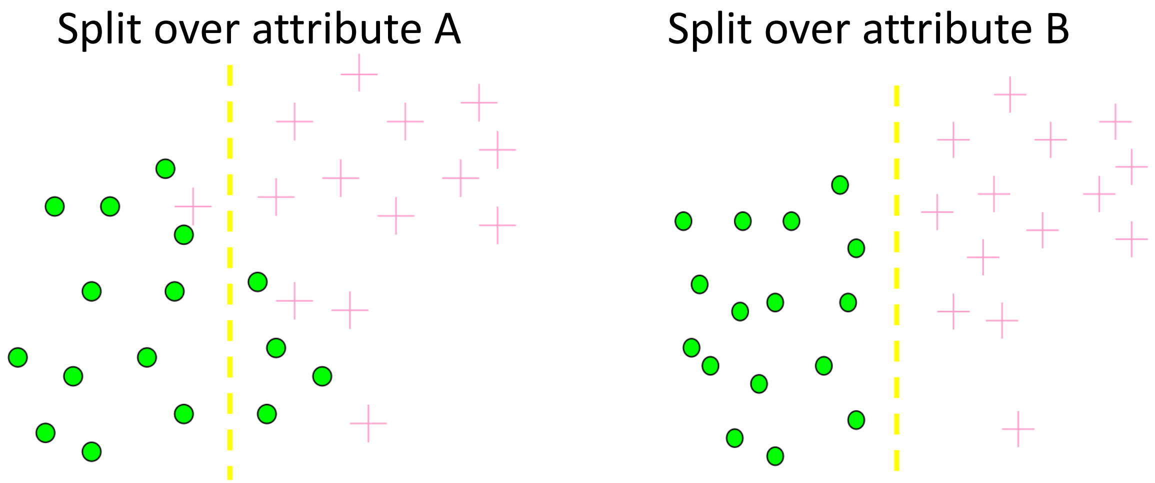
（1）当前结点包含的样本全部属于同一类别

（2）当前属性集为空，或所有样本在所有属性上取值相同

（3）当前结点包含的样本集合为空

Identify the best attribute

- The key to decision tree learning is **how to identify the best attribute**.
- **Our Goal**: The samples contained in the branch nodes of the decision tree belong to the same class as much as possible, that is, the ***"purity"*** of the nodes is getting higher and higher
- Which of the following two splits will result in higher purity?

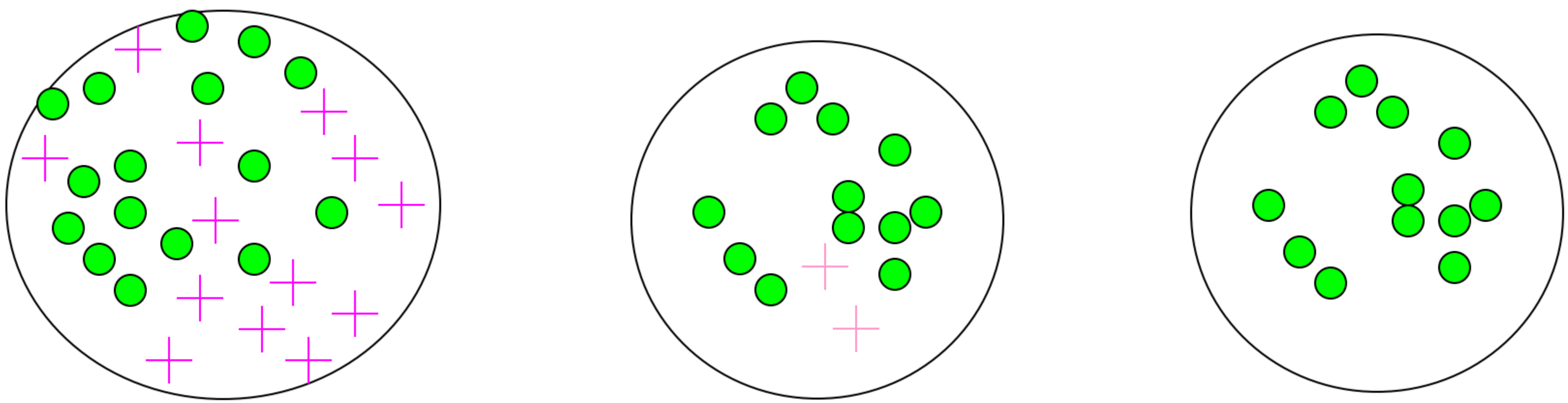


- How to measure which split brings a higher purity?

Identify the best attribute

- How to measure that how much purity a certain split brings?
- We can measure the level of ***impurity*** in a group of examples

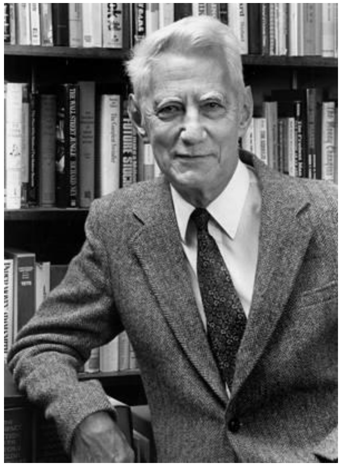
Very impure group **Less impure** **Minimum Impurity**



Entropy

- A common way to measure impurity, comes from information theory.
- Quantifies the amount of uncertainty associated with a specific probability distribution.
- The higher the entropy, the less confident we are in the outcome.
- Definition

$$Entropy\,(X) = \sum_{c} -p(X=c)log_2 p(X=c)$$

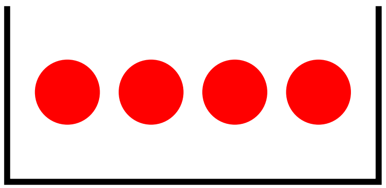


Claude Shannon (1916 – 2001), most of the work was done in Bell labs

Entropy

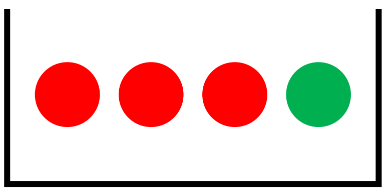
$$Ent(X) = \sum_c -p(X=c)log_2 p(X=c)$$

"+" for red, "-" for green



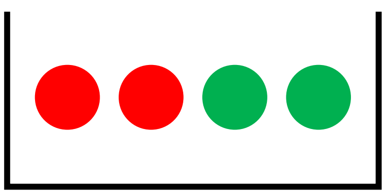
p(+)=1, p(-)=0

$$Ent(4+, 0-) = -(1\log_2 1 + 0\log_2 0) = 0$$



p(+)=3/4, p(-)=1/4

$$Ent(3+, 1-) = -\left(\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}\right) = 0.811$$

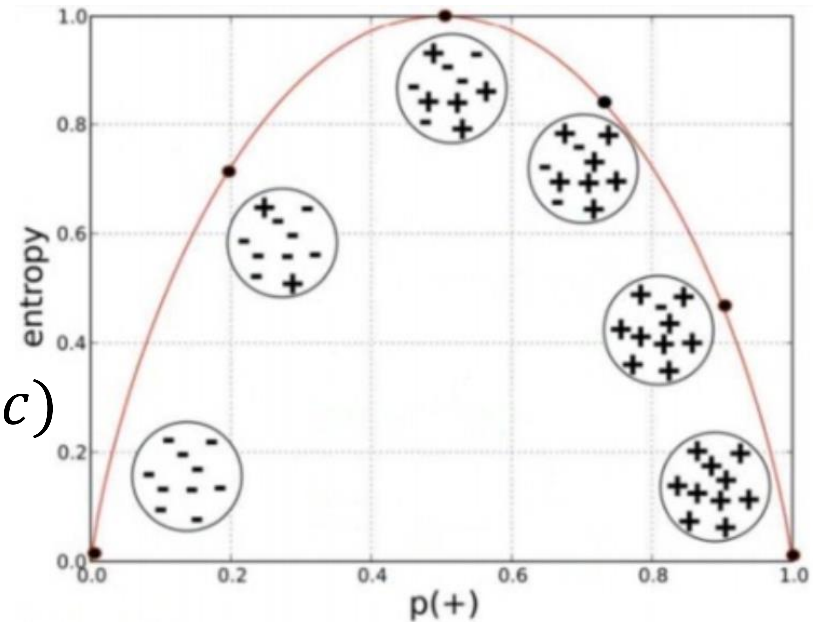


p(+)=1/2, p(-)=1/2

$$Ent(2+, 2-) = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

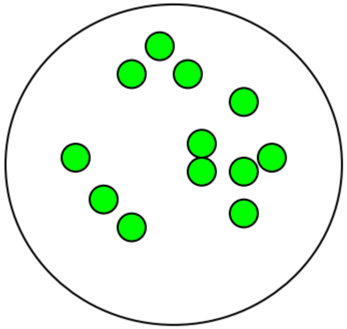
Entropy

$$Ent(X) = \sum_{c} -p(X=c)log_2 p(X=c)$$



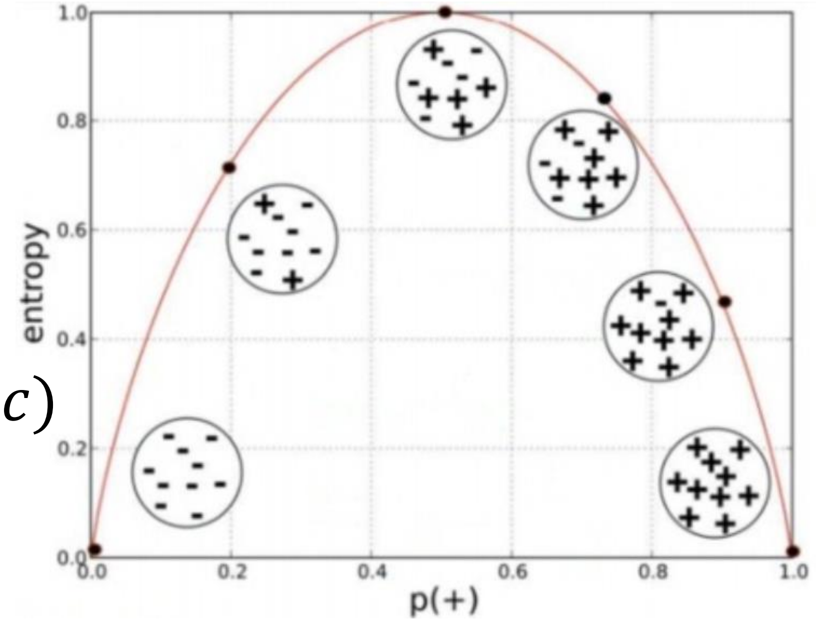
- **What is the entropy of a group in which all examples belong to the same class?**
- $Ent(X) = -p(x=1)\log_2 p(X=1) - p(x=0)\log_2 p(X=0)$
  $= -1\log 1 - 0\log 0 = 0$

Minimum Impurity



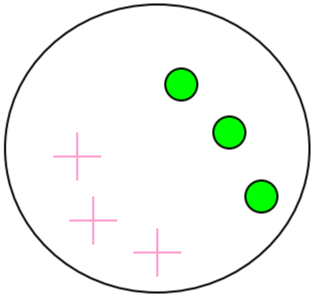
Entropy

$$Ent(X) = \sum_{c} -p(X=c) log_2 p(X=c)$$



- **What is the entropy of a group with 50% in either class?**
- $Ent(X) = ?$ **1**

Maximum Impurity



Information Gain

- How much do we gain (in terms of reduction in entropy) from knowing one of the attributes
- In other words, what is the reduction in entropy from this knowledge.

Information Gain

- Discrete attribute a has v possible values
- Dividing with a will generate v branch nodes
- The v -th branch node contains all samples in D whose value is a^v on attribute a, denoted as D^v
- Then the "information gain" obtained by dividing the sample set D with attribute a can be calculated:

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

It is the branch node weight, and the branch node with more samples has a greater influence.

Information Gain

- In general, the larger the information gain, the larger the ***"purity improvement"*** obtained by using the attribute to divide

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

Case Study

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

该数据集包含17个训练样本，其中正例占 $p_1=\frac{8}{17}$，反例占$p_2=\frac{9}{17}$，计算得到根结点的信息熵为

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

Case Study

- 以属性“色泽”为例，其对应的3个数据子集分别为 D^1(色泽=青绿)，D^2(色泽=乌黑), D^3 (色泽=浅白)
- 子集 D^1 包含编号为$\{1, 4, 6, 10, 13, 17\}$的6个样例，其中正例占 $p_1 = \frac{3}{6}$ ，反例占 $p_2 = \frac{3}{6}$，D^2、D^3 同理，3个结点的信息熵为：

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

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

Case Study

- 以属性“色泽”为例，其对应的3个数据子集分别为 D^1(色泽=青绿)，D^2(色泽=乌黑)，D^3 (色泽=浅白)
- 子集 D^1 包含编号为$\{1, 4, 6, 10, 13, 17\}$的6个样例，其中正例占 $p_1 = \frac{3}{6}$ ，反例占 $p_2 = \frac{3}{6}$ ，D^2、D^3 同理，3个结点的信息熵为：

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

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

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

Case Study

- 以属性“色泽”为例，其对应的3个数据子集分别为 D^1(色泽=青绿)，D^2(色泽=乌黑), D^3 (色泽=浅白)
- 子集 D^1 包含编号为$\{1, 4, 6, 10, 13, 17\}$的6个样例，其中正例占 $p_1 = \frac{3}{6}$ ，反例占 $p_2 = \frac{3}{6}$ ，D^2、D^3 同理，3个结点的信息熵为：

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

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

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

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

Case Study

- 以属性“色泽”为例，其对应的3个数据子集分别为 D^1(色泽=青绿)，D^2(色泽=乌黑)，D^3 (色泽=浅白)
- 子集 D^1 包含编号为$\{1, 4, 6, 10, 13, 17\}$的6个样例，其中正例占 $p_1 = \frac{3}{6}$ ，反例占 $p_2 = \frac{3}{6}$，D^2、D^3 同理，3个结点的信息熵为：

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

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

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

- 属性“色泽”的信息增益为

$$\begin{aligned}\operatorname{Gain}(D, \text{色泽}) &= \operatorname{Ent}(D) - \sum_{v=1}^{3}\frac{|D^v|}{|D|}\operatorname{Ent}(D^v)\\ &= 0.998 - \left(\frac{6}{17}\times 1.000 + \frac{6}{17}\times 0.918 + \frac{5}{17}\times 0.722\right)\\ &= 0.109\end{aligned}$$

Case Study

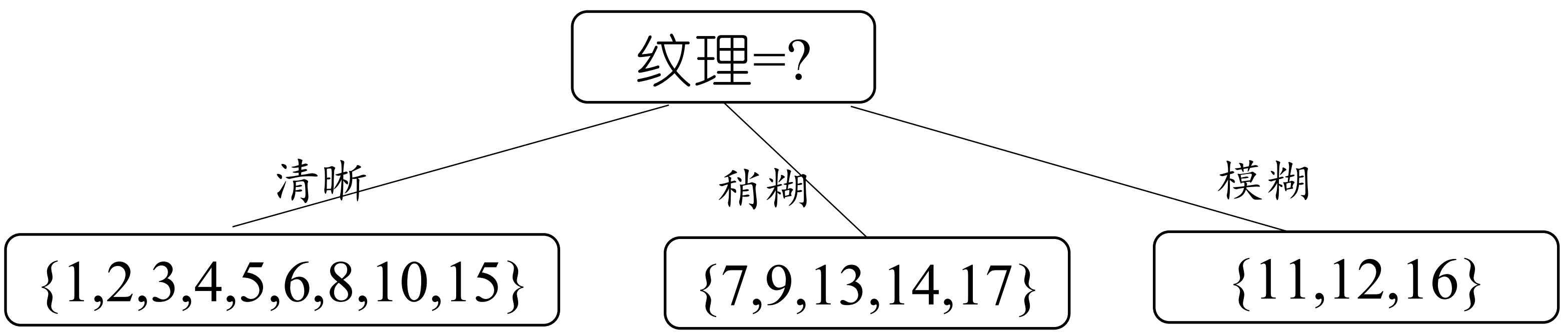
- 类似的，其他属性的信息增益为

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

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

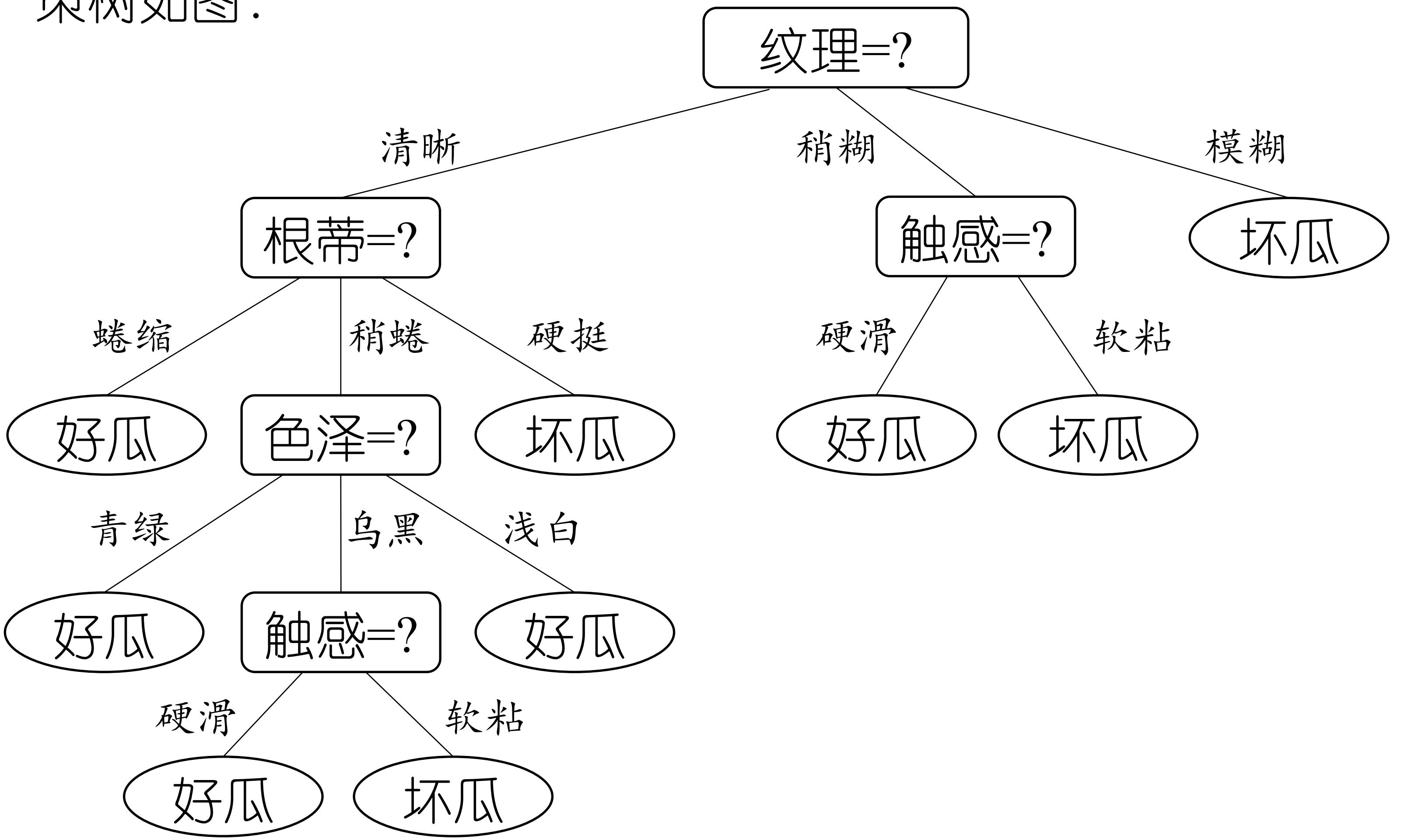
$$\mathrm{Gain}(D,\text{触感})=0.006$$

- 显然，属性“纹理”的信息增益最大，其被选为划分属性



Case Study

- 决策树学习算法将对每个分支结点做进一步划分，最终得到的决策树如图：



Information Gain

- If 编号 is also used as a candidate division attribute, its information gain is generally much greater than other attributes.
- Obviously, such a decision tree does not have the ability to generalize and cannot make effective predictions on new samples

Information gain has a preference for attributes with a larger number of possible values

- Information Gain is used by **ID3** algorithm[1]

[1]Quinlan J R. Induction of decision trees[J]. Machine learning, 1986, 1(1): 81-106.

Other rules

- **Gain Ratio**

$$\text{Gain\_ratio}(D, a) = \frac{\text{Gain}(D,a)}{\text{IV}(a)}$$

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

- $IV(a)$ is called the *intrinsic value* (固有值) of attribute a. The more possible values of attribute a (that is, the larger V), the larger the value of $IV(a)$ is.

Gain Ratio has a preference for properties with a lower number of possible values

- Gain Ratio is used by **C4.5** algorithm[2]

[2] Quinlan J R . C4.5: Programs for Machine Learning[J]. 1993.

Other rules

- **Gini index**
- We also use Gini value to measure the ***"purity"*** of the sample set

$$\mathrm{Gini}(D) = \sum_{k=1}^{|\mathcal{Y}|} \sum_{k' \neq k} p_k p_{k'} \quad = 1 - \sum_{k=1}^{|\mathcal{Y}|} p_k^2$$

Reflects the probability that two samples are randomly drawn from *D* with inconsistent class labels

- $Gini(D) \downarrow$, *Purity* $\uparrow$
- $Gini\_index$ of attribute a is defined as：

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

- The attribute that **minimizes** the $Gini\_index$ after division should be selected as the optimal division attribute, that is

$$a_* = \underset{a \in A}{\arg\min}\ \mathrm{Gini\_index}(D, a)$$

- Gini index is used by **CART** algorithm[3]

[3] Breiman L, Friedman J H, Olshen R, et al. Classification and Regression Trees[J]. 1984.

Summary: How to build a DT?

- **Key Steps**

  -Identify the best attribute

  -Build the tree recursively

- **Identify the best attribute**

  -entropy

$$Ent(X) = \sum_{c} -p(X = c)log_2 p(X = c)$$

  -information gain

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

$$Gain(Temperature) \approx 0.03$$
$$Gain(Humidity) \approx 0.15$$
$$Gain(Windy) \approx 0.05$$

Practice

- Which of the 4 attributes will be selected as the root node?

TID	Outlook	Temperature	Humidity	Windy	Play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

https://blog.csdn.net/qq_28697571

$$Ent(Play) = -\frac{9}{14}log\frac{9}{14} - \frac{5}{14}log\frac{5}{14} \approx 0.94$$
(9 Yes, 5 No)

$$Ent(sunny) = -\frac{2}{5}log\frac{2}{5} - \frac{3}{5}log\frac{3}{5} \approx 0.97$$
(2 Yes, 3 No)

$$Ent(rainy) = -\frac{3}{5}log\frac{3}{5} - \frac{2}{5}log\frac{2}{5} \approx 0.97$$
(3 Yes, 2 No)

$$Ent(overcast) = 0$$
(All Yes)

$$Gain(Outlook) = Ent(Play) - \frac{5}{14}Ent(sunny) - \frac{5}{14}Ent(rainy) - \frac{4}{14}Ent(overcast) \approx 0.25$$

Today’s Topics

- Type of classifiers
- Structure of the Decision Tree
- Build A Decision Tree
- ***Tree Pruning***
- Continuous values
- Multivariate decision tree
- Random forest

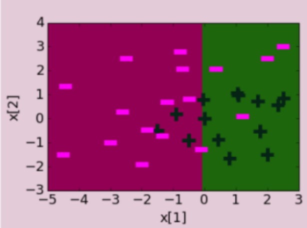
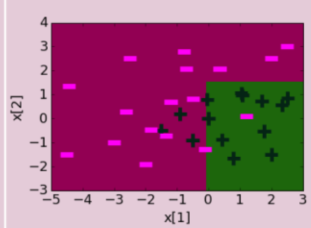
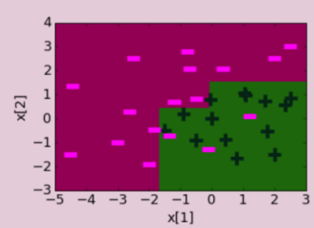
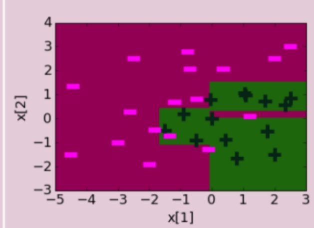
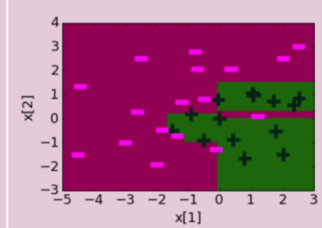
Overfitting in decision trees

- In order to classify the training samples as correctly as possible, the node split process will be repeated, sometimes resulting in ***too many branches in the decision tree***.
- At this time, the training samples may be learned "too well", so that some features of the training set itself are regarded as the general nature of all data, resulting in ***overfitting***.

Overfitting in decision trees

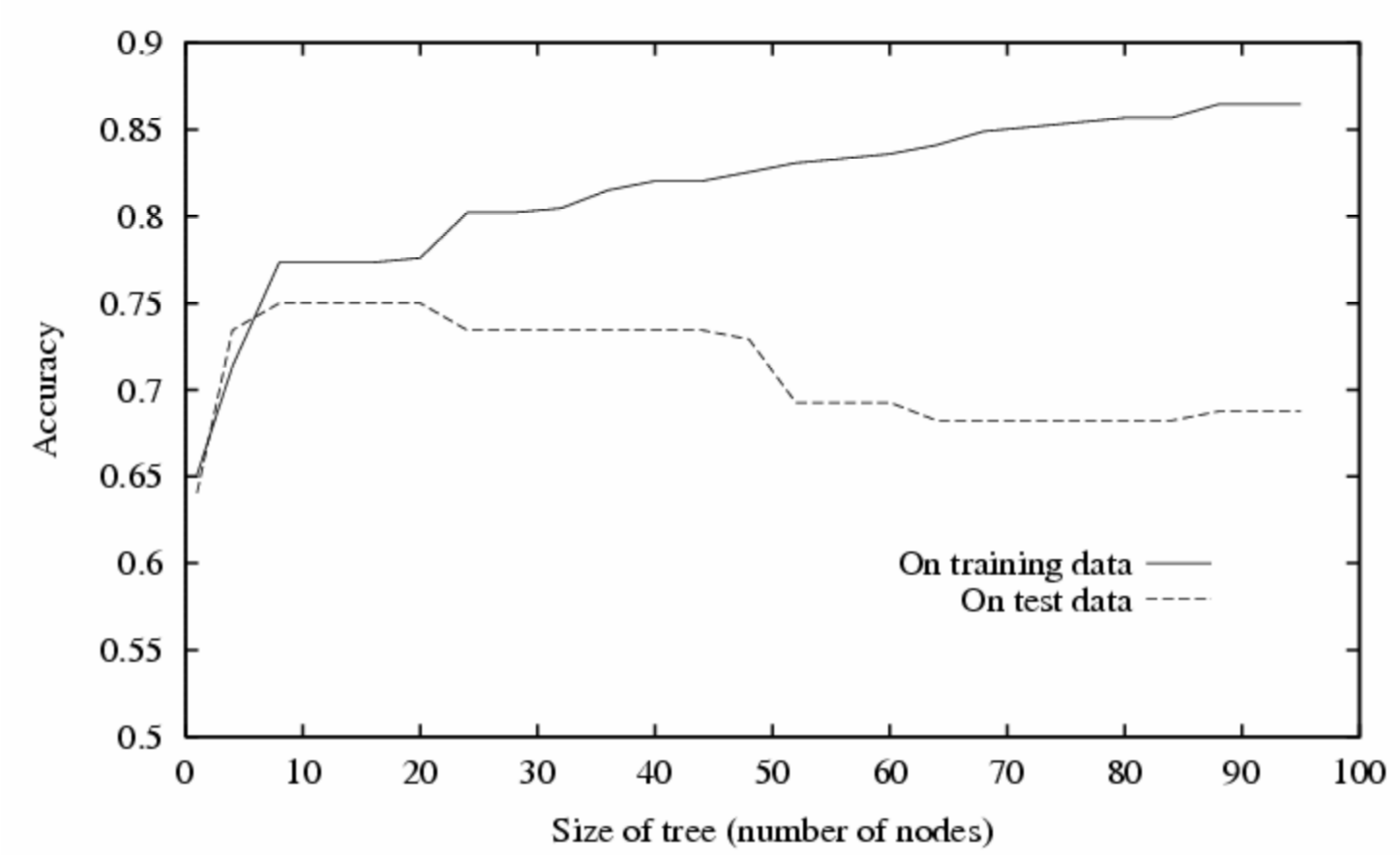
- What happens when we increase depth?

Training error reduces with depth

Tree depth	depth = 1	depth = 2	depth = 3	depth = 5	depth = 10
Training error	0.22	0.13	0.10	0.03	0.00
Decision boundary					

The decision boundary changes from being too simple (underfitting) to being too complex (overfitting)

Overfitting in decision trees



Tree pruning

- The risk of overfitting can be reduced by removing some branches.
- ***Pruning*** is the main method for decision tree learning algorithms to deal with overfitting
- Basic strategies
  - Pre-pruning
  - Post-pruning

Prepruning

- Estimate the generalization performance before dividing the node, and stop if the division cannot bring about an improvement.

✓Reduce risk of overfitting

✓Significantly reduces training time and test time overhead

× Underfitting risk: some branches that may bring performance improvements in the future are prohibited from expanding

A watermelon case study

- Dataset

训练集

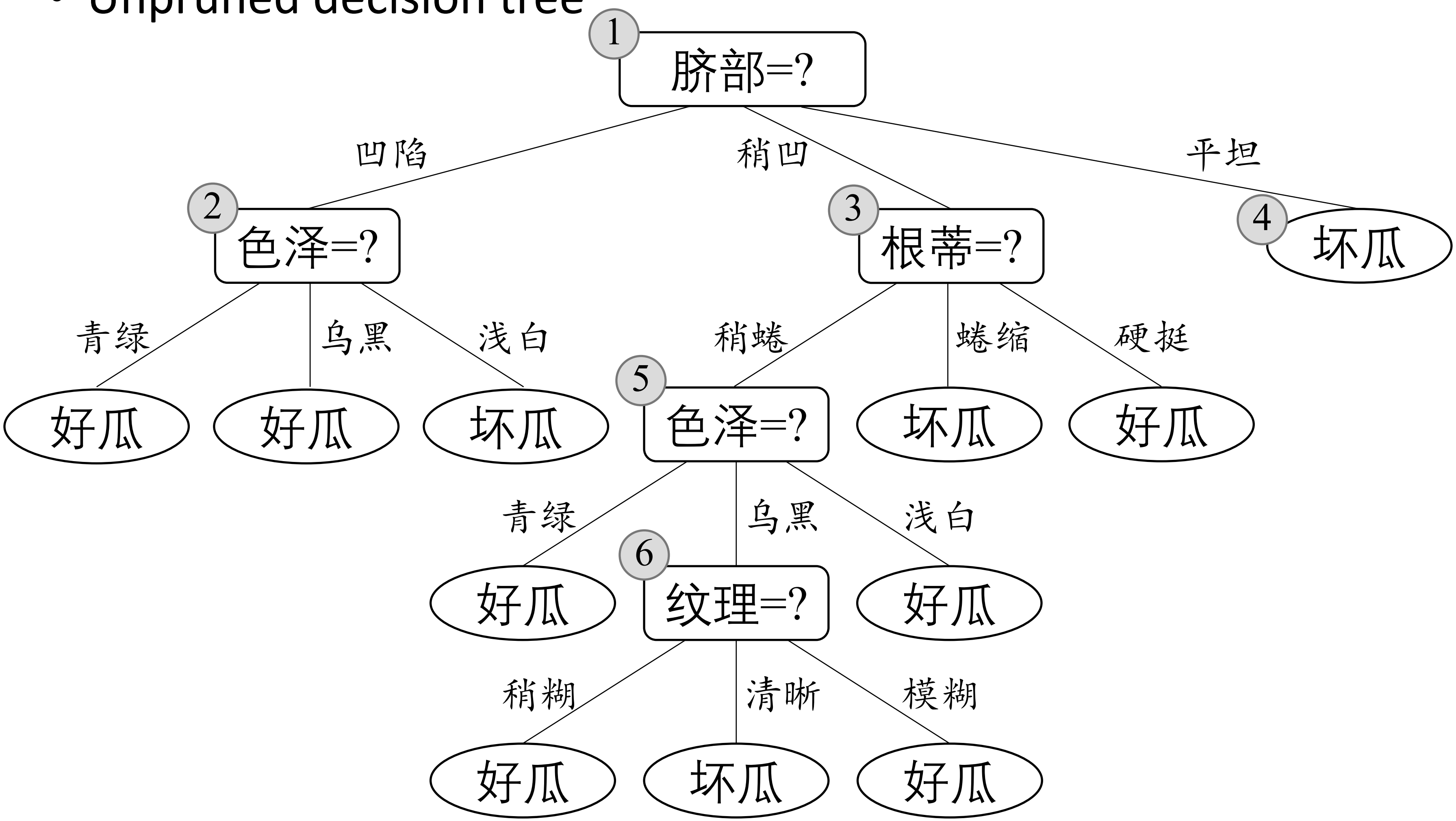
编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

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

A watermelon case study

- Unpruned decision tree



Prepruning: Case Study

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

结点1：若不划分，则将其标记为叶结点，类别标记为训练样例中最多的类别，即好瓜。验证集中，{4,5,8}被分类正确，得到验证集精度为：$\frac{3}{7} \times 100\% = 42.9\%$

1

脐部=?

验证集精度

“脐部=?” 划分前: 42.9%

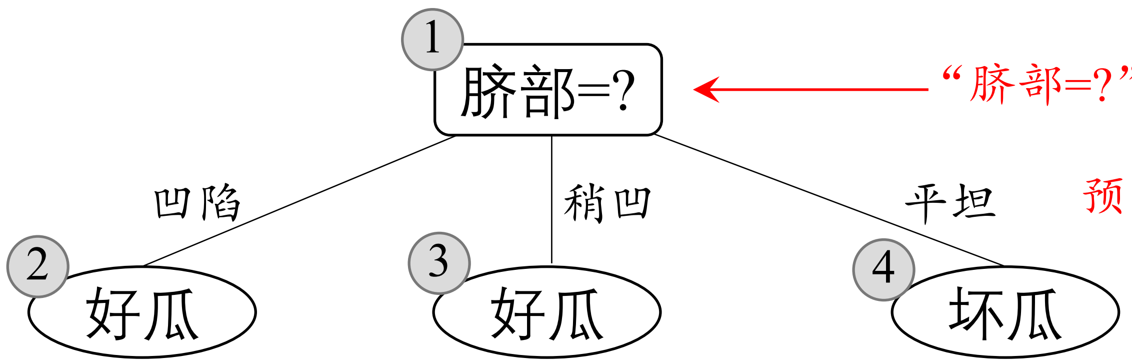
Prepruning: Case Study

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

结点1：若划分，根据结点②③④的训练样例，将这3个结点分别标记为“好瓜”、“好瓜”、“坏瓜”。此时，验证集中编号为 {4,5,8,11,12}的样例被划分正确，验证集精度为

$$\frac{5}{7} \times 100\% = 71.4\%$$



验证集精度

“脐部=?” 划分前: 42.9%

划分后: 71.4%

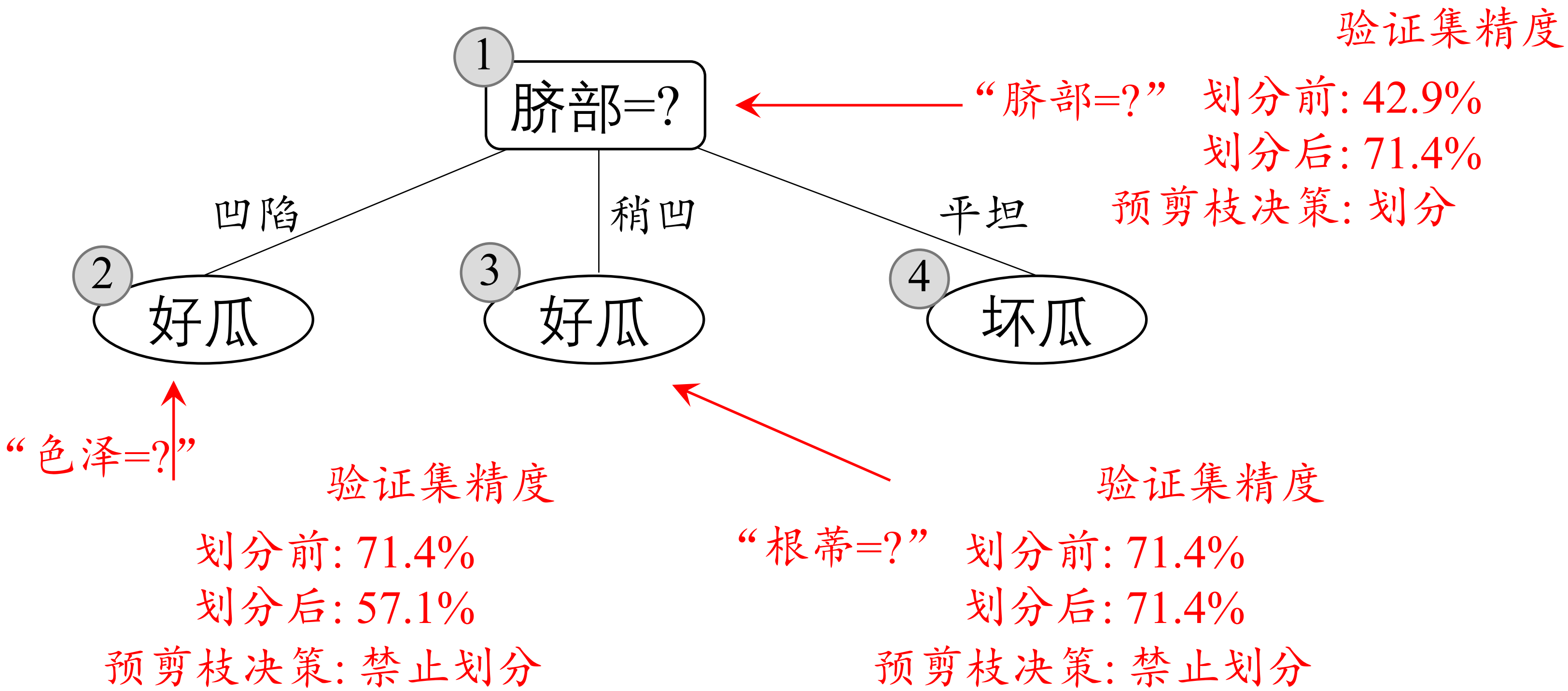
预剪枝决策: 划分

Prepruning: Case Study

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

对结点②③④分别进行剪枝判断，结点②③都禁止划分，结点④本身为叶子结点。最终得到仅有一层划分的决策树，称为“**决策树桩**”



Postpruning

- First generate a complete decision tree, and then examine non-leaf nodes in a bottom-up manner. Replacing the subtree corresponding to the node with a leaf node when the generalization performance can be improved.

✓ Low underfitting risk

✓ The generalization performance is often better than that of pre-pruned decision trees

✕ High training time

A watermelon case study

- Dataset

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

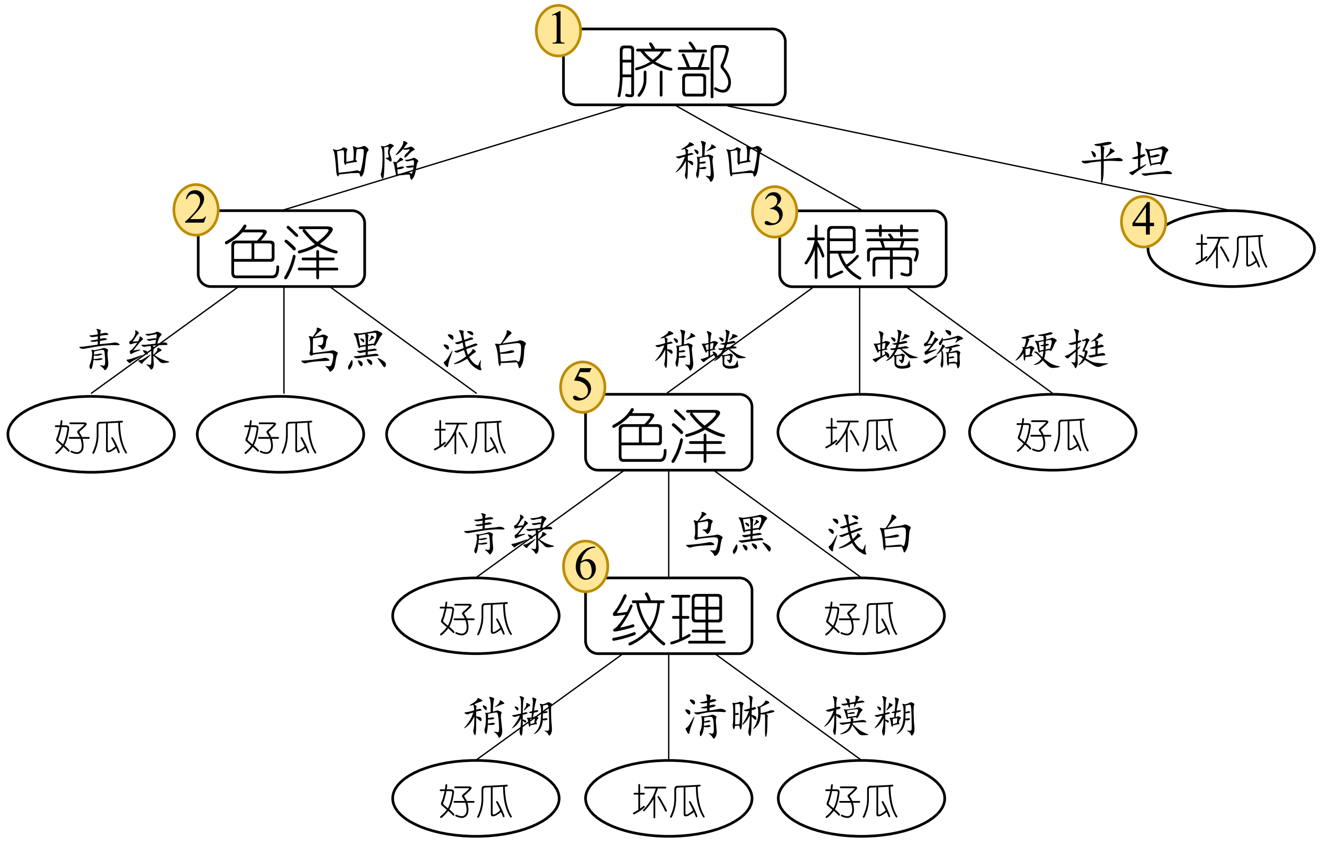
验证集

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

Postpruning: Case Study

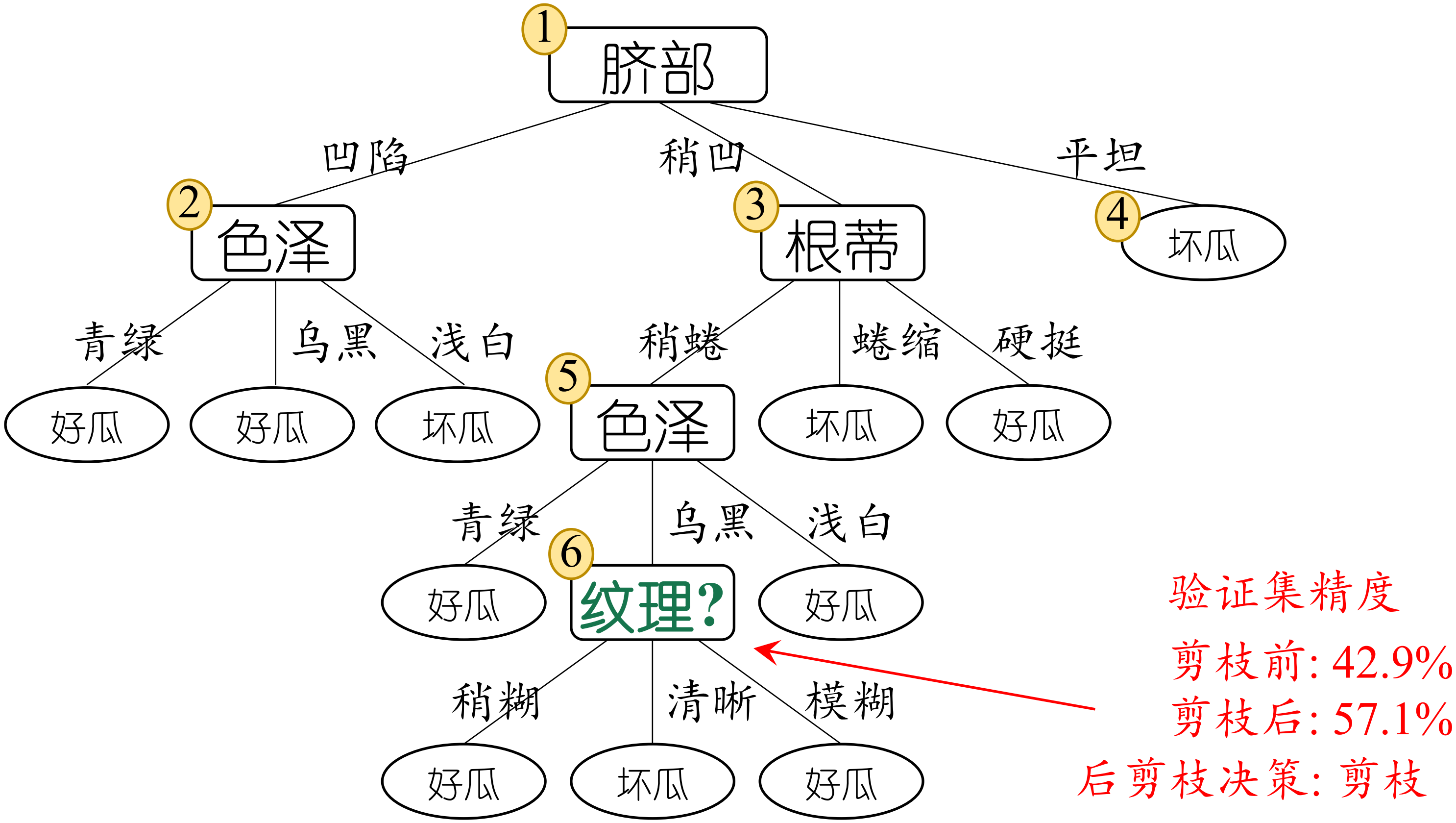
- 先从训练集生成一棵完整的决策树，然后自底向上地对非叶结点进行考察，若将该结点对应的子树替换为叶结点能带来决策树泛化性能提升，则将该子树替换为叶结点

首先生成一棵完整的决策树，该决策树的验证集精度为 42.9%



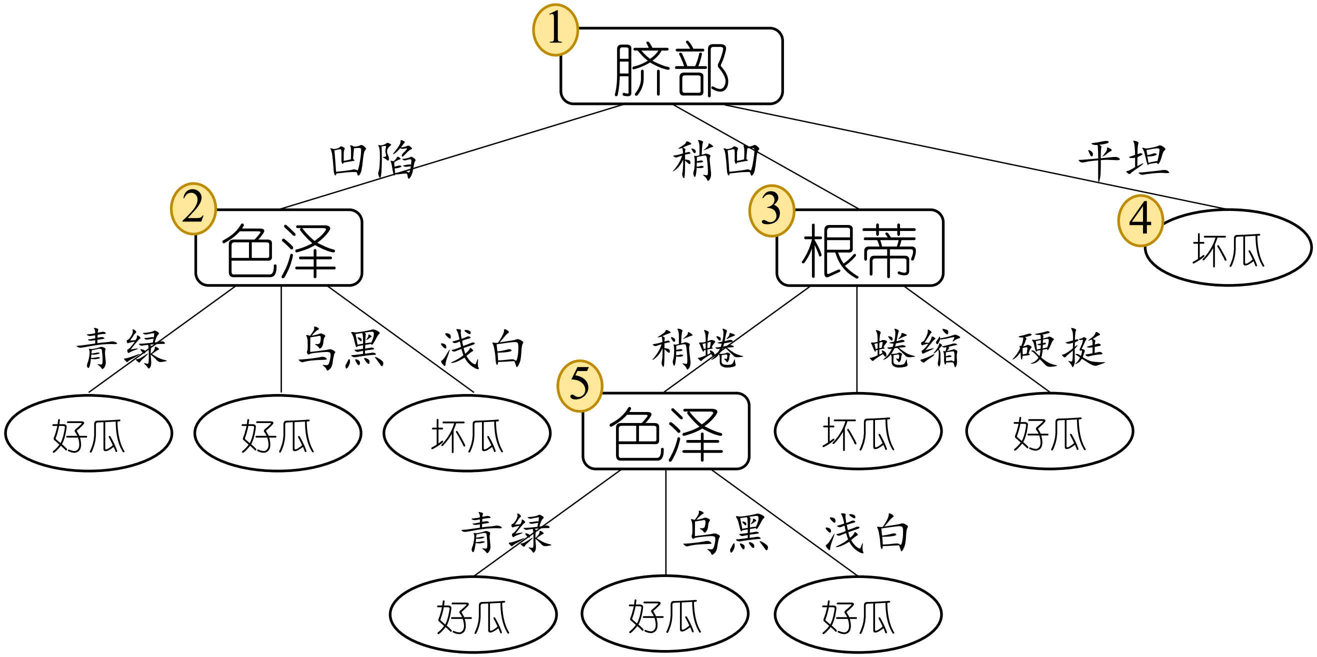
Postpruning: Case Study

- 首先考虑结点⑥，若将其替换为叶结点，根据落在其上的训练样本$\{7, 15\}$将其标记为“好瓜”，得到验证集精度提高至57.1%，则决定剪枝



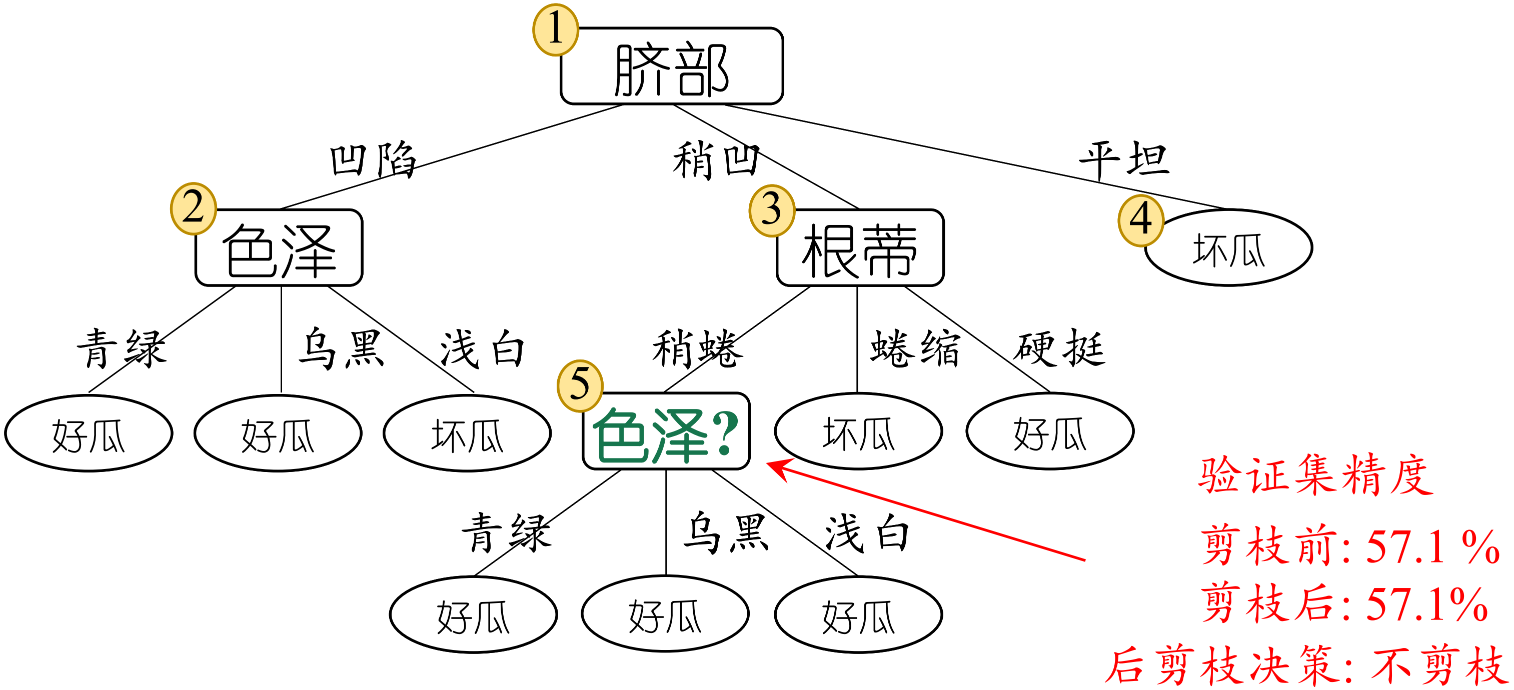
Postpruning: Case Study

- 首先考虑结点⑥，若将其替换为叶结点，根据落在其上的训练样本$\{7, 15\}$将其标记为“好瓜”，得到验证集精度提高至57.1%，则决定剪枝



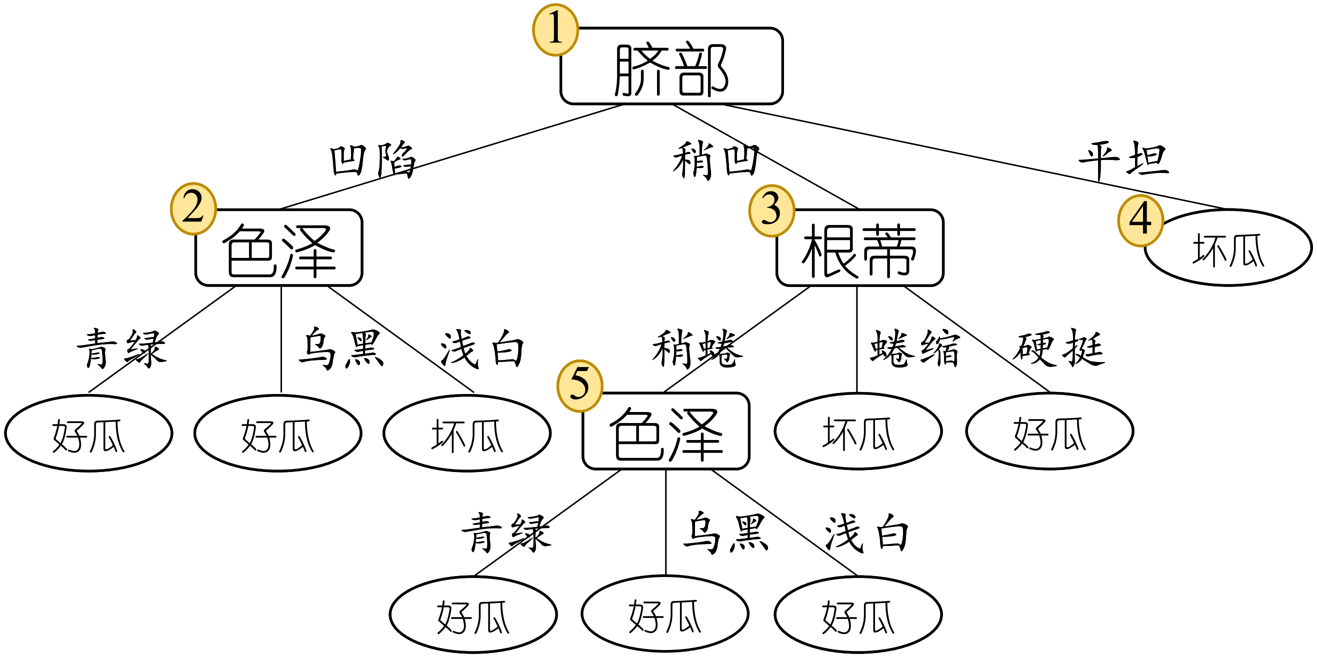
Postpruning: Case Study

- 然后考虑结点⑤，若将其替换为叶结点，根据落在其上的训练样本$\{6,7,15\}$将其标记为“好瓜”，得到验证集精度仍为57.1%，可以不进行剪枝



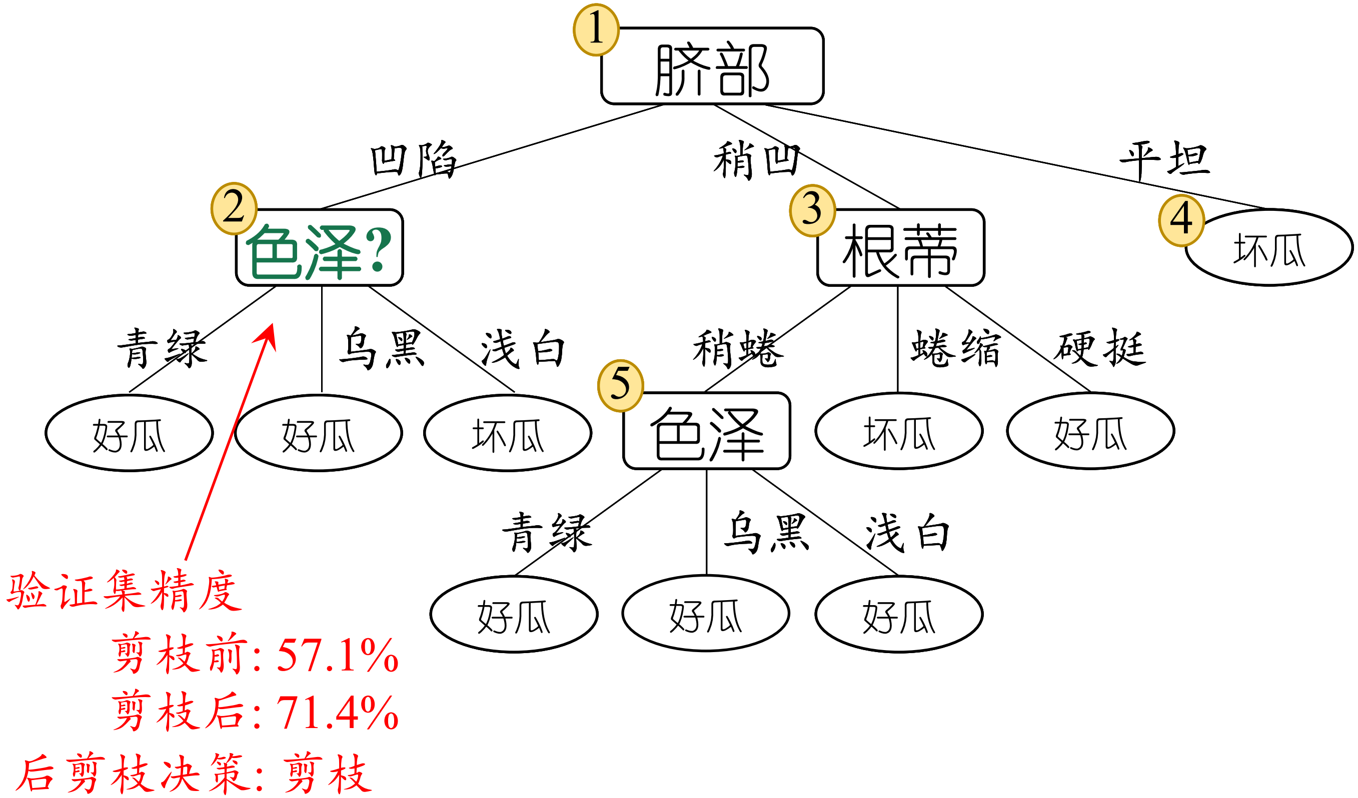
Postpruning: Case Study

- 然后考虑结点⑤，若将其替换为叶结点，根据落在其上的训练样本$\{6,7,15\}$将其标记为“好瓜”，得到验证集精度仍为57.1%，可以不进行剪枝



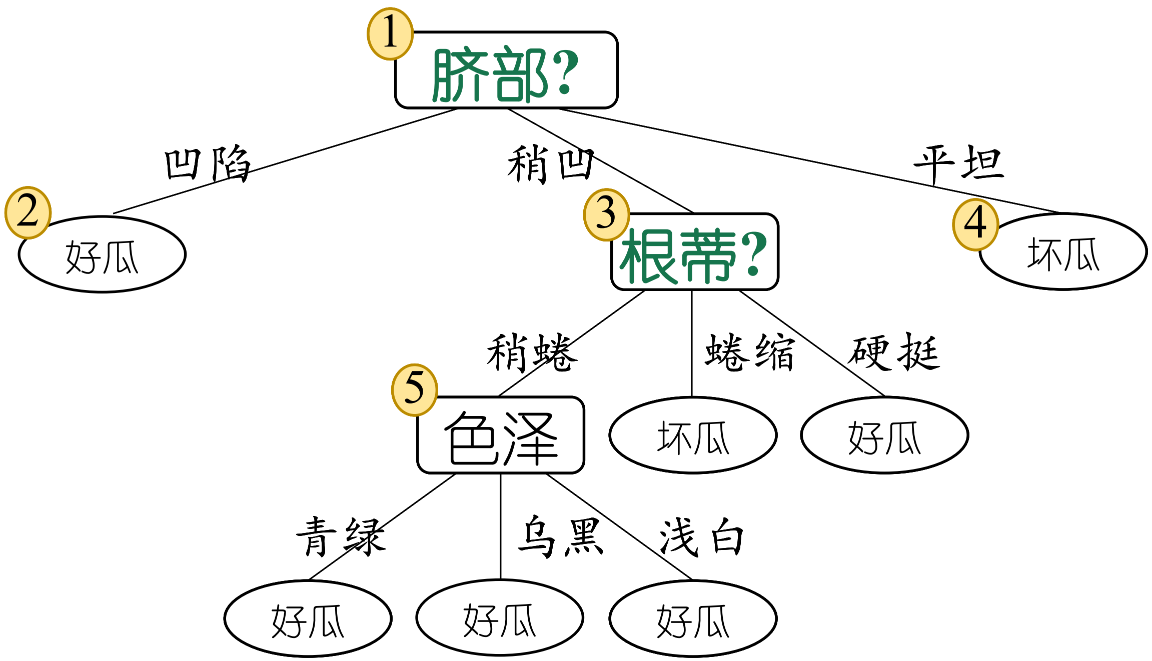
Postpruning: Case Study

- 对结点②，若将其替换为叶结点，根据落在其上的训练样本$\{1, 2, 3, 14\}$，将其标记为“好瓜”，得到验证集精度提升至 71.4%，则决定剪枝



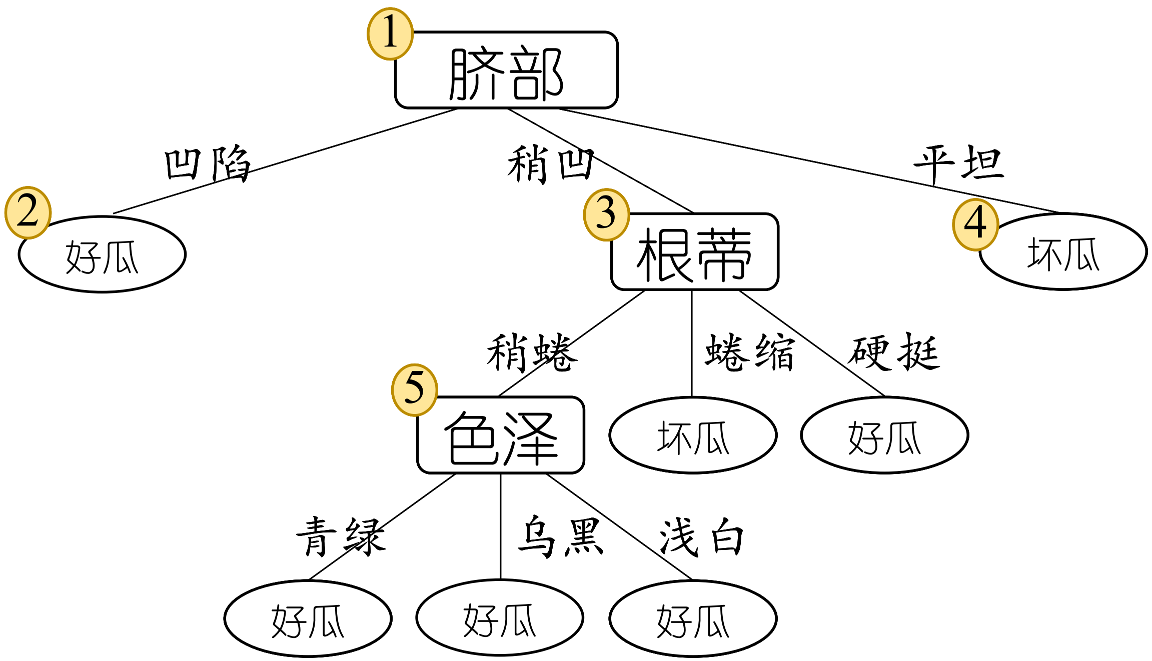
Postpruning: Case Study

- 对结点③和①，先后替换为叶结点，验证集精度均未提升，则分支得到保留



Postpruning: Case Study

- 最终基于后剪枝策略得到的决策树如图所示



Today’s Topics

- Type of classifiers
- Structure of the Decision Tree
- Build A Decision Tree
- Tree Pruning
- ***Continuous values***
- Multivariate decision tree
- Random forest

Continuous values

- Since the number of possible values of continuous attributes is no longer limited, the nodes cannot be divided directly according to the values of continuous attributes.
- Solution: ***Continuous attribute discretization***（连续属性离散化）
- Core idea: Divide the range of continuous values into multiple intervals, and each interval is regarded as an attribute value
- E.g. bi-partition（二分法） used by C4.5 algorithm

Bi-partition

- **Step 1**: Sort the *n* values of continuous attributes *a* on the sample set
- **Step 2**: The median point of adjacent values is used as a candidate division point to obtain a set of candidate division points

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

- **Step 3**: Identify the best attribute according to the method of discrete attributes (e.g. information gain)

A watermelon case study

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

对属性“密度”，其候选划分点集合包含16 个候选值：
$T_{密度} = \{0.244, 0.294, 0.351, 0.381, 0.420, 0.459, 0.518, 0.574, 0.600, 0.621, 0.636, 0.648, 0.661, 0.681, 0.708, 0.746\}$

可计算其信息增益为0.262，对应划分点为0.381

对属性“含糖量”进行同样处理

Different from discrete attributes, if the current node's division attribute is a continuous attribute, this attribute can also be used as the division attribute of its descendant nodes.

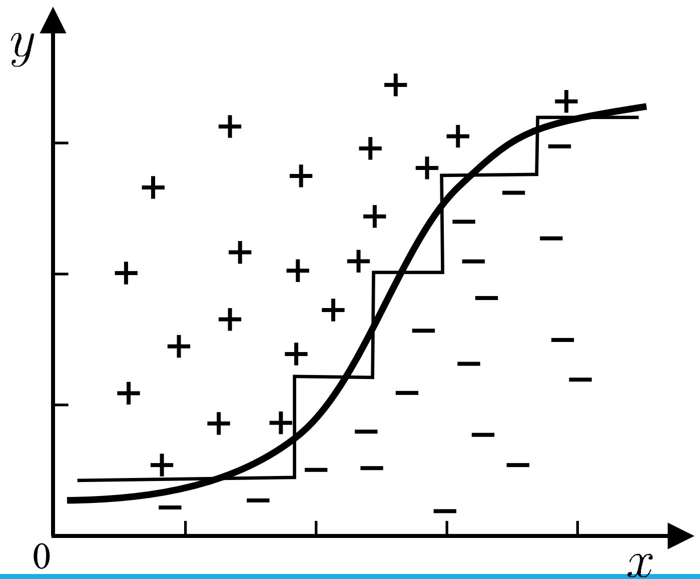
E.g. 父结点使用密度≤0.381，不会禁止子结点上使用密度≤0.294

Today’s Topics

- Type of classifiers
- Structure of the Decision Tree
- Build A Decision Tree
- Tree Pruning
- Continuous values
- ***Multivariate decision tree***
- Random forest

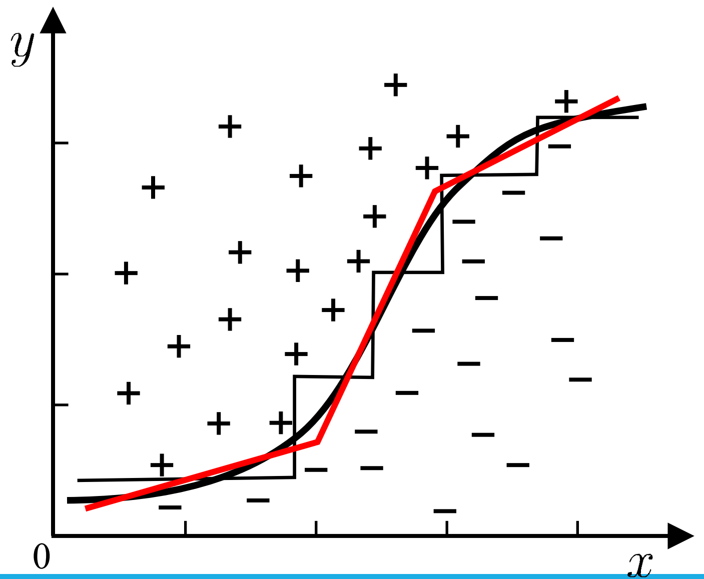
Univariate decision tree

- The decision boundary of the decision tree is axis-parallel
- For complex classification boundaries, many segments of splits must be used to obtain a good approximation
- The prediction time overhead can be significant due to the large number of attribute tests to be performed.



Multivariate decision tree

- We can use oblique boundaries to simplify the model
- Non-leaf nodes are no longer only for a certain attribute, but a linear combination of attributes
- Learning: build a suitable classifier for each non-leaf node (instead of find the best attribute)



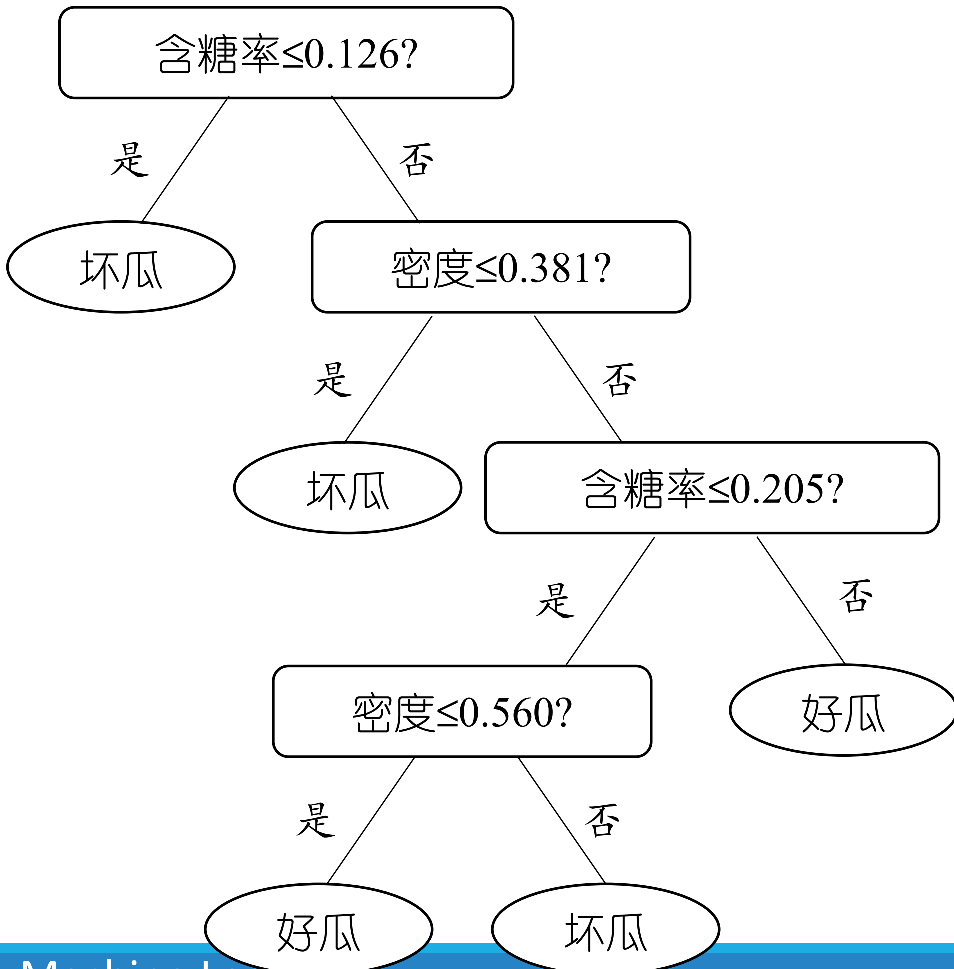
A watermelon case study

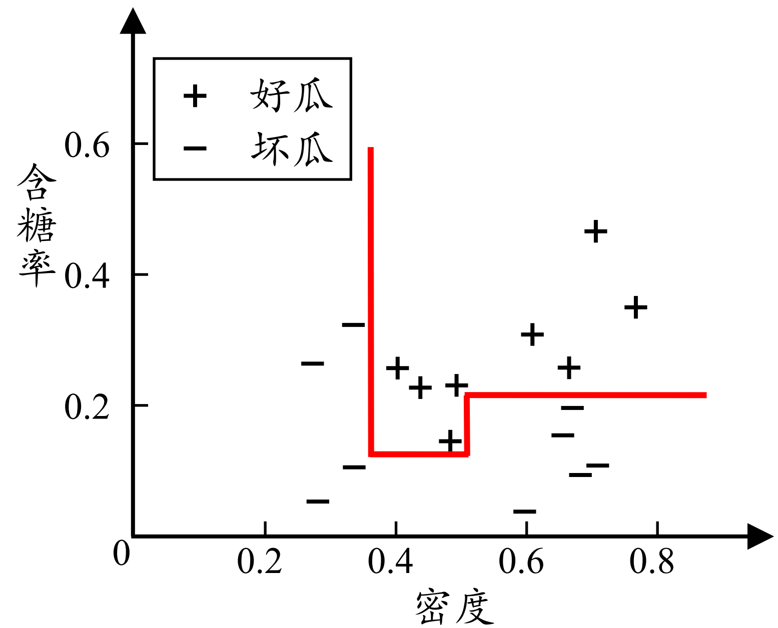
- Dataset

编号	密度	含糖率	好瓜
1	0.697	0.460	是
2	0.774	0.376	是
3	0.634	0.264	是
4	0.608	0.318	是
5	0.556	0.215	是
6	0.403	0.237	是
7	0.481	0.149	是
8	0.437	0.211	是
9	0.666	0.091	否
10	0.243	0.267	否
11	0.245	0.057	否
12	0.343	0.099	否
13	0.639	0.161	否
14	0.657	0.198	否
15	0.360	0.370	否
16	0.593	0.042	否
17	0.719	0.103	否

A watermelon case study

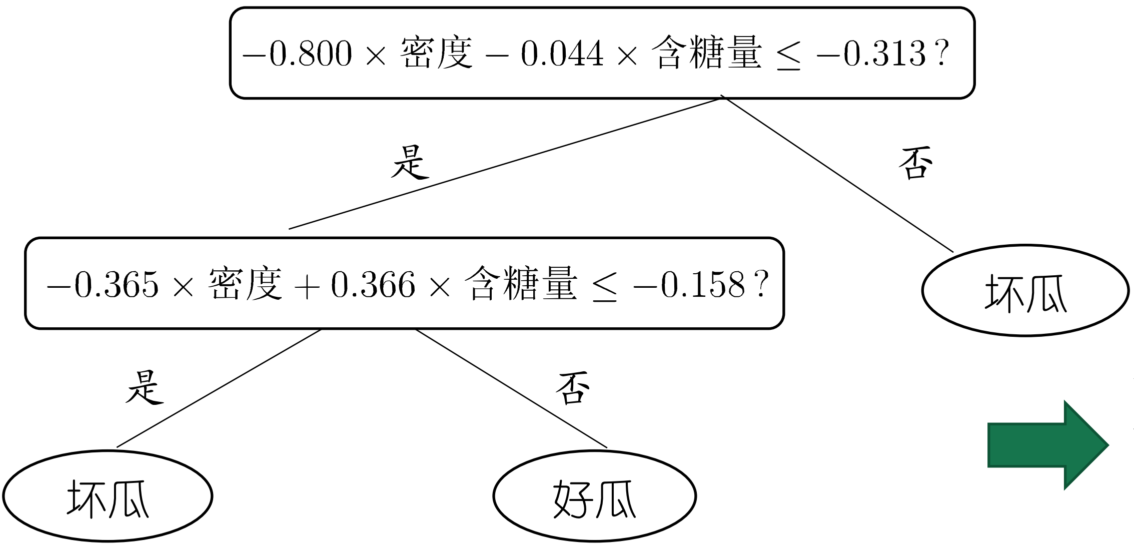
- Univariate decision tree

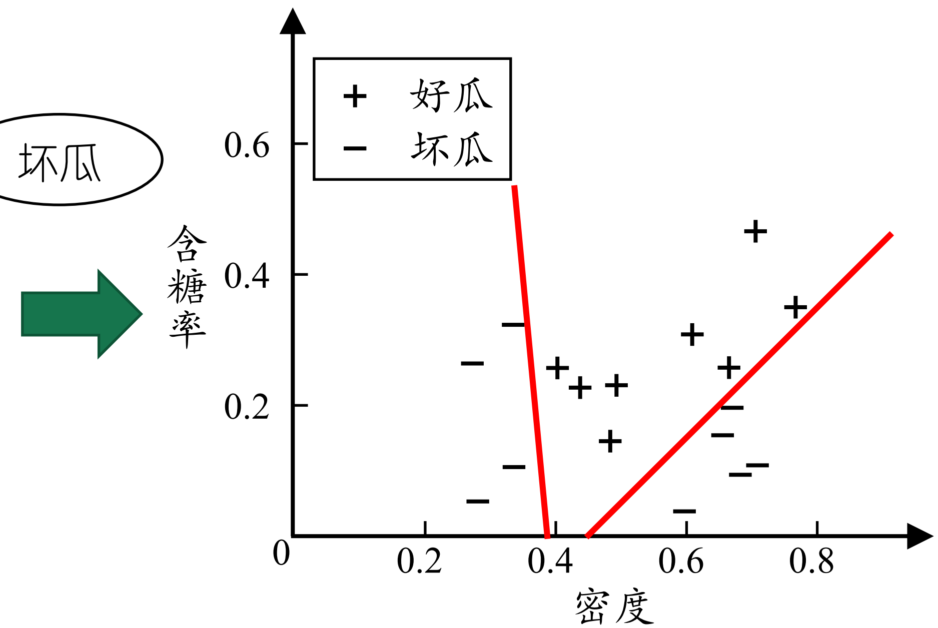




A watermelon case study

- Multivariate decision tree

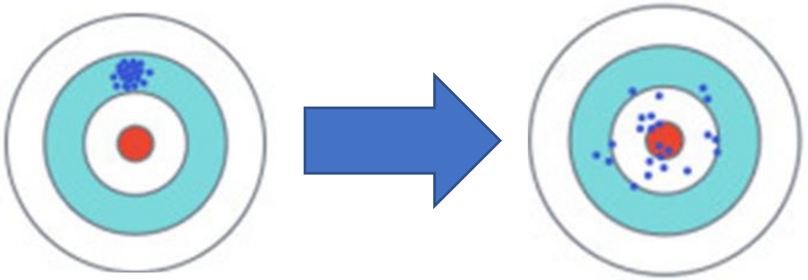


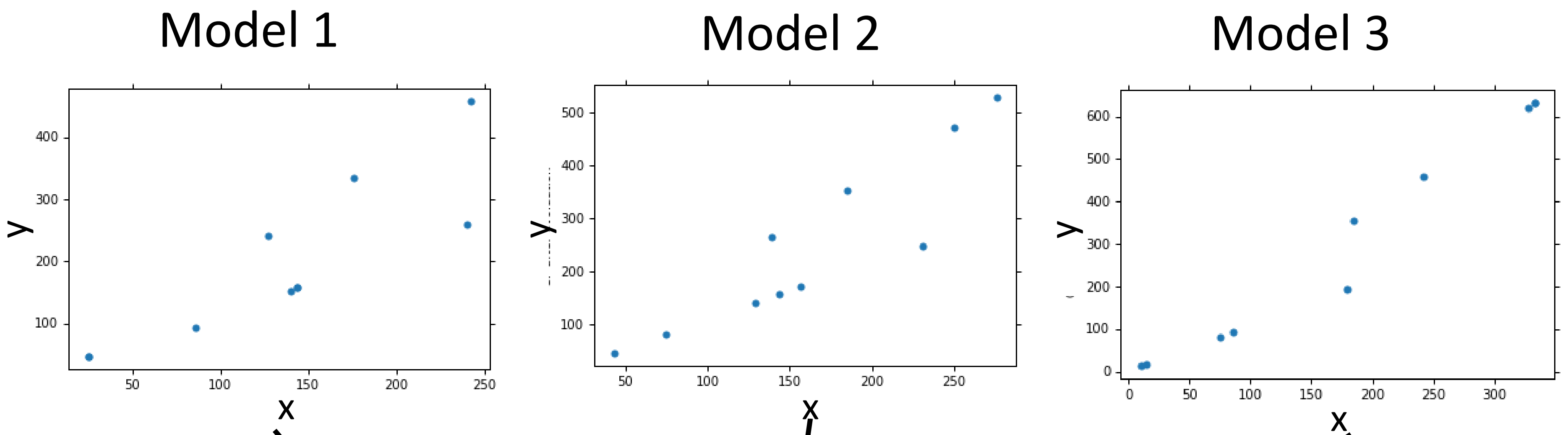


Today's Topics

- Type of classifiers
- Structure of the Decision Tree
- Build A Decision Tree
- Tree Pruning
- Continuous values
- Multivariate decision tree
- ***Random forest***

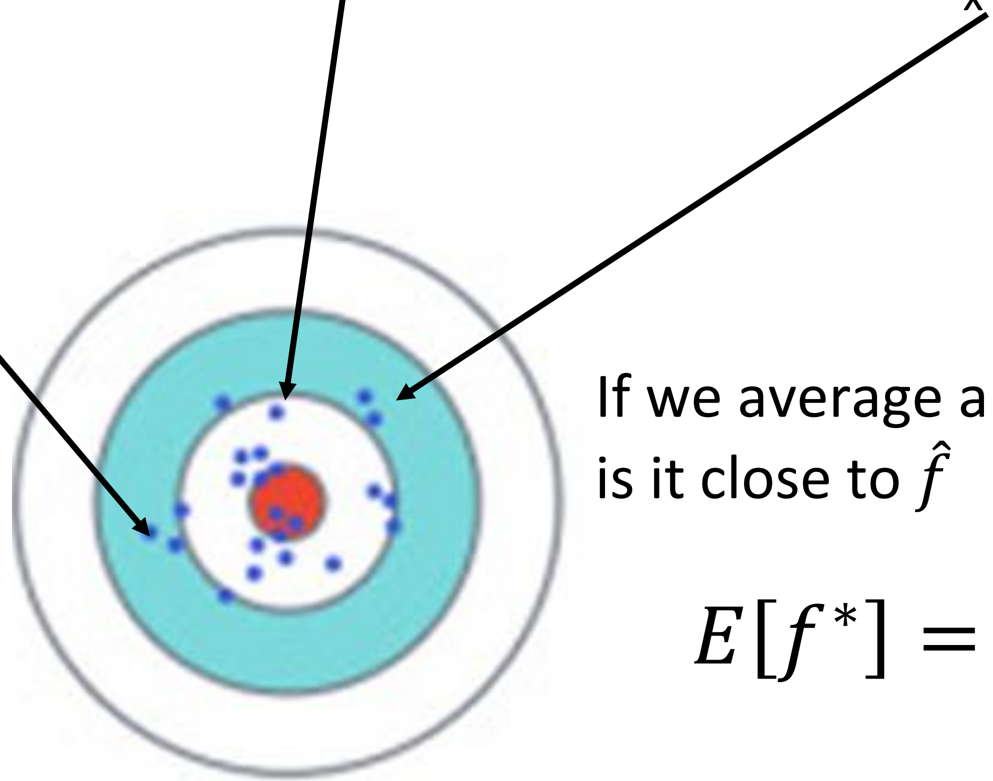
Recall: Bias & Variance





A complex model will have large variance.

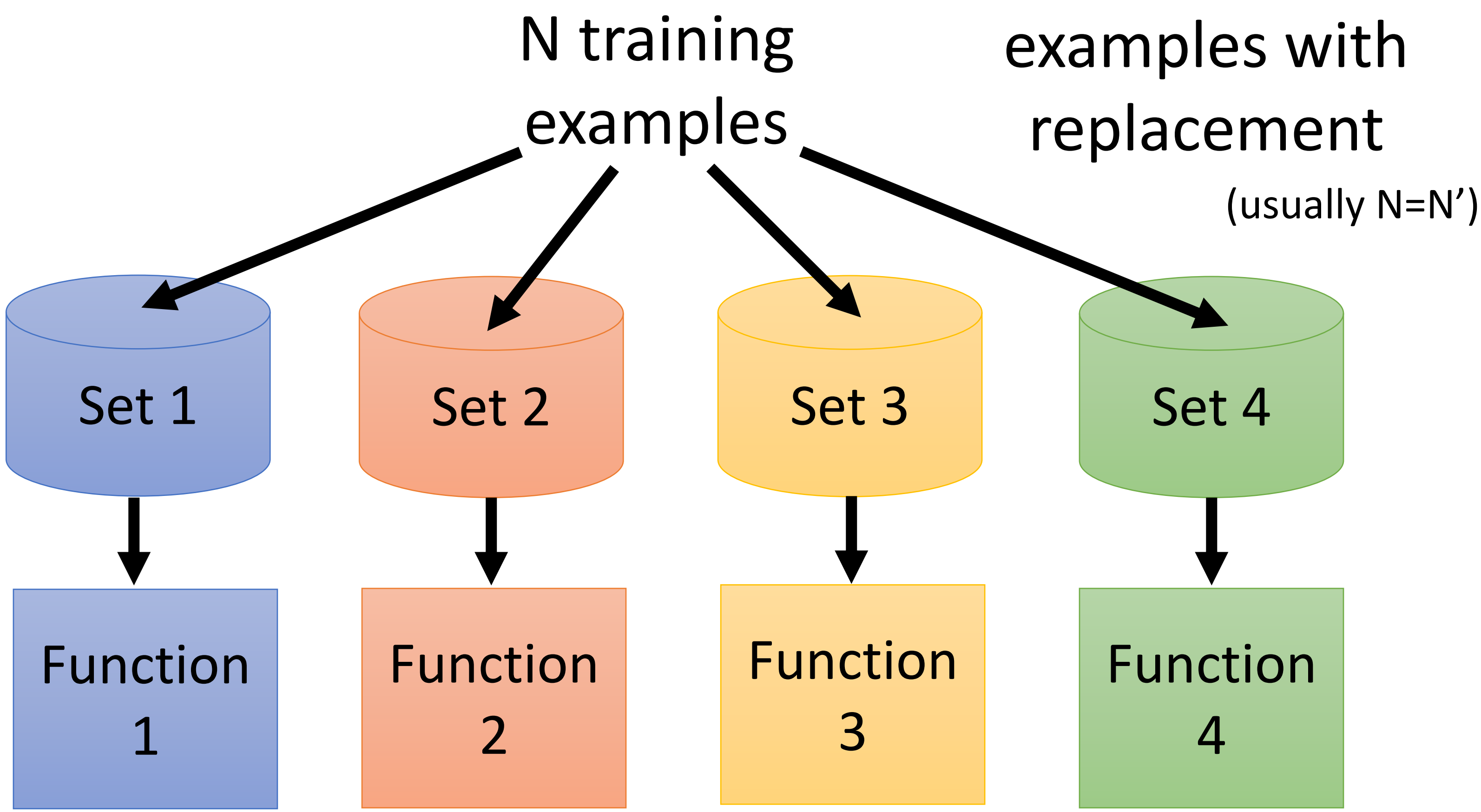
We can average complex models to reduce variance.



If we average all the f^*, is it close to $\hat{f}$

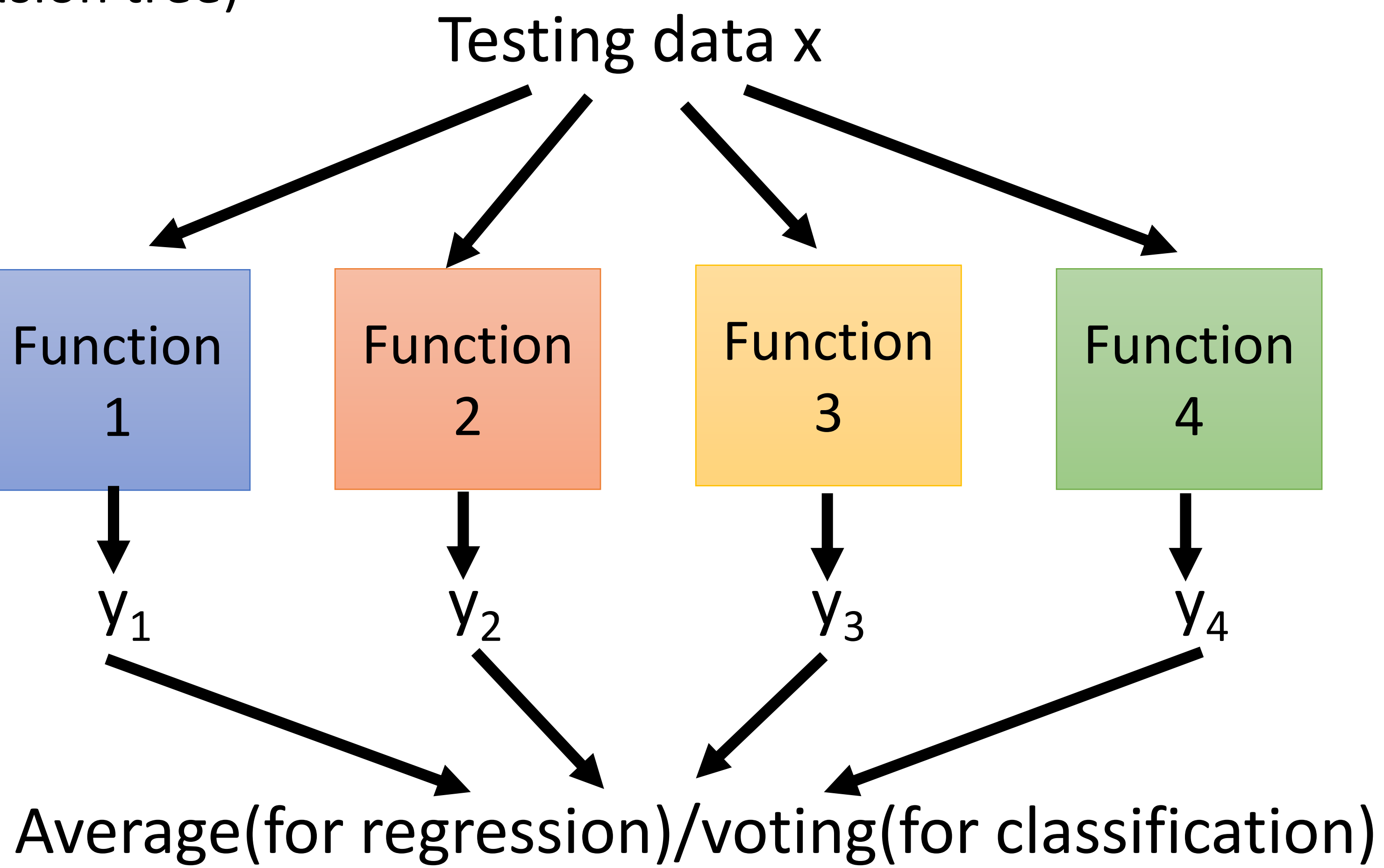
$$E[f^*] = \hat{f}$$

Bagging


N training examples
Sampling N' examples with replacement
(usually N=N')
Set 1
Set 2
Set 3
Set 4
Function 1
Function 2
Function 3
Function 4

Bagging

- Helpful when your model is ***complex, easy to overfit*** (e.g. decision tree)



Random Forest

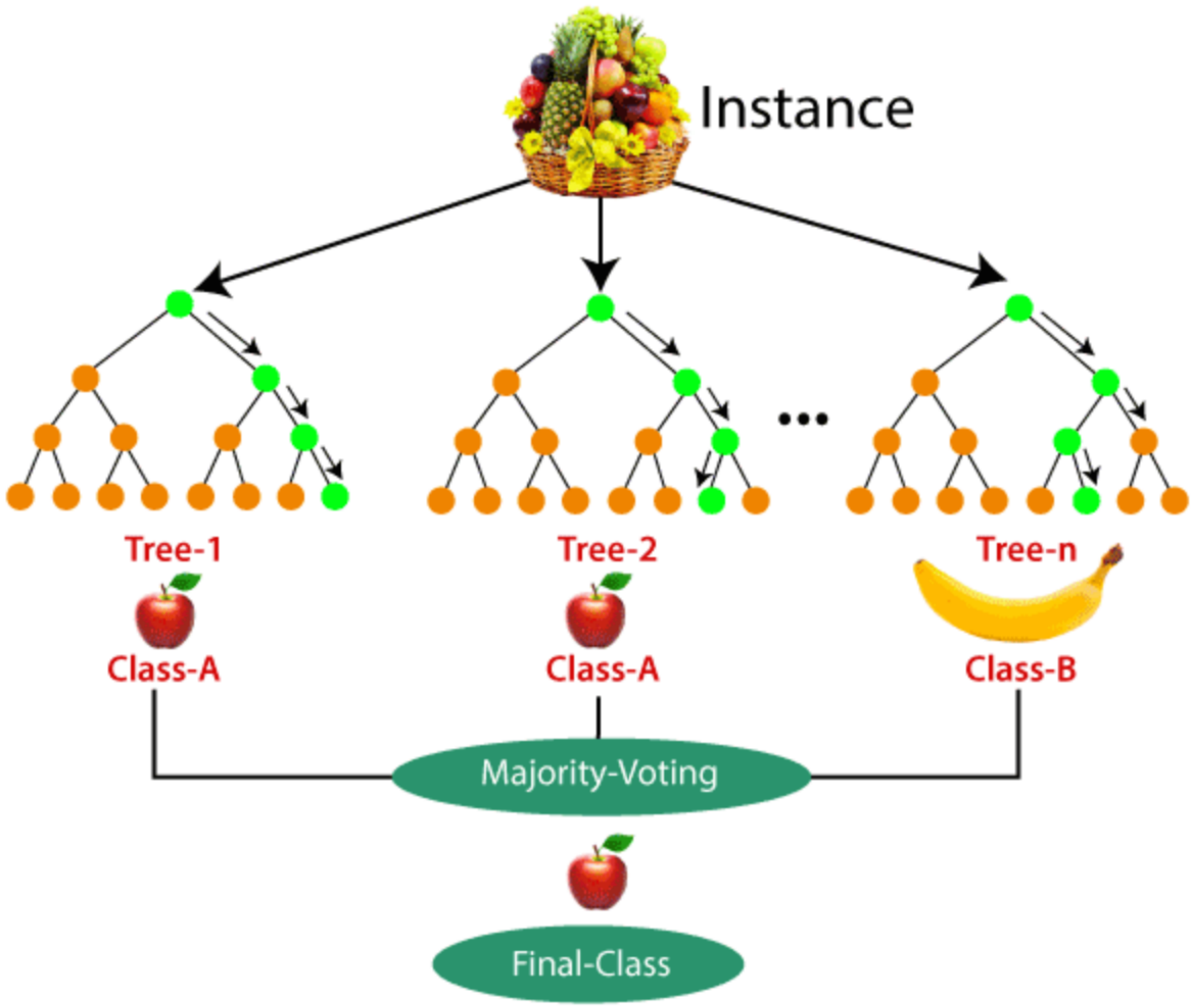
train	f_1	f_2	f_3	f_4
x^1	O	X	O	X
x^2	O	X	X	O
x^3	X	O	O	X
x^4	X	O	X	O

- Decision tree:
  - Easy to achieve 0% error rate on training data
    - If each training example has its own leaf ……
- Random forest: ***Bagging of decision tree***
  - Resampling training data is not sufficient
  - Randomly restrict the features/questions used in each split
- Out-of-bag validation for bagging
  - Using RF = f_2+f_4 to test x^1
  - Using RF = f_2+f_3 to test x^2
  - Using RF = f_1+f_4 to test x^3
  - Using RF = f_1+f_3 to test x^4

Out-of-bag (OOB) error

Good error estimation of testing set

Random Forest



Summary

- **Decision Tree**
  - Structure
  - Identify the best attribute (entropy, information gain)
  - Tree pruning
- **Strength and weakness of Decision Tree**
  - ✓ Fast and simple to implement
  - ✓ Can convert to rules
  - ✕ Ignore dependencies between attributes

Question

- If the decision tree is overfitted, will reducing the depth solve the problem?