Decision Tree

Yanyan Lan

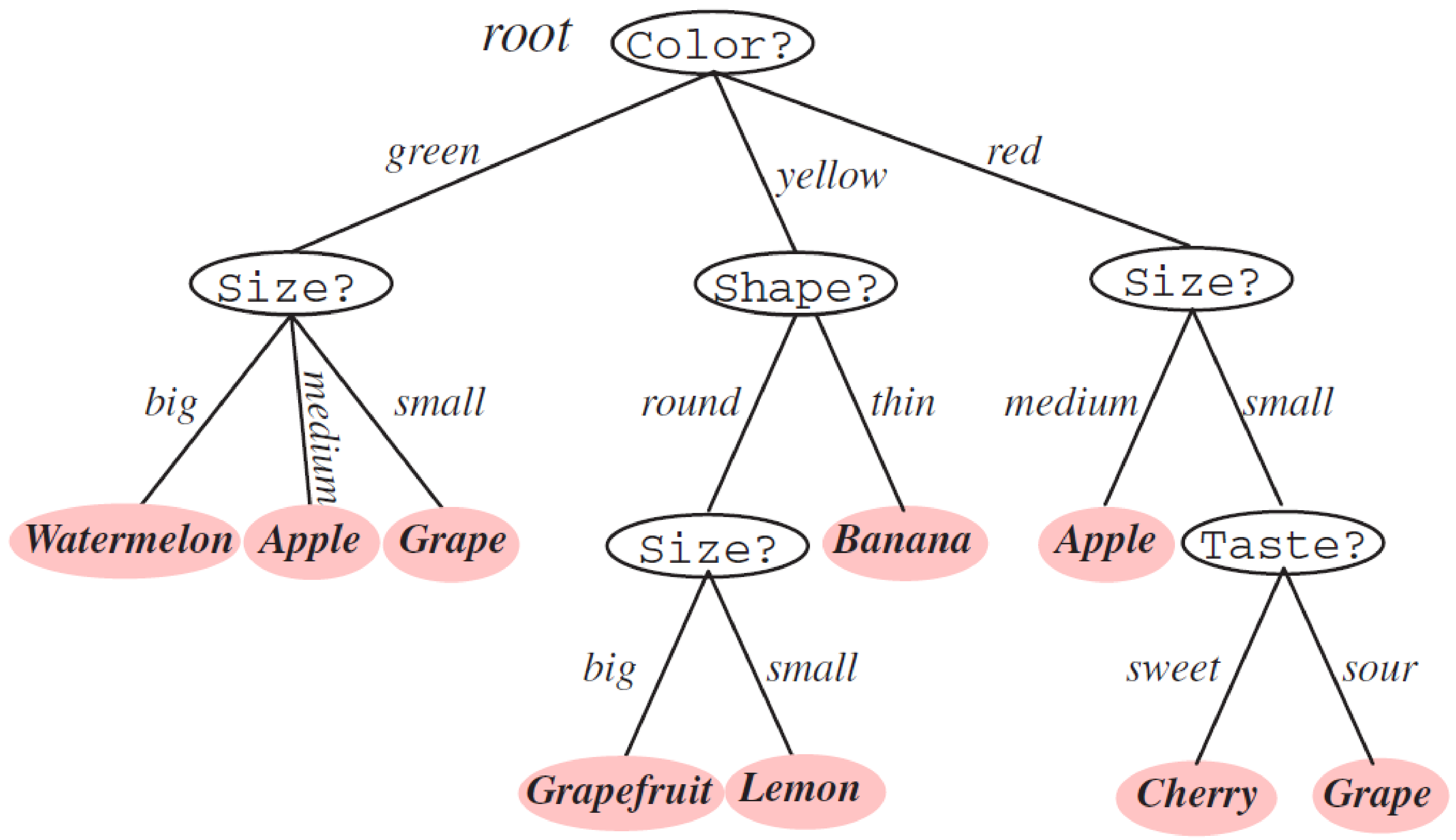
lanyanyan@ict.ac.cn

20 Questions

- I am thinking of a fruit. Ask me up to 20 yes/no questions to determine what this fruit is that I am thinking about. Consider your questions wisely…
- How did you ask the questions?
- What underlying measure led you the questions, if any?
- Most importantly, iterative yes/no questions of this sort require no metric and are well suited for nominal data.

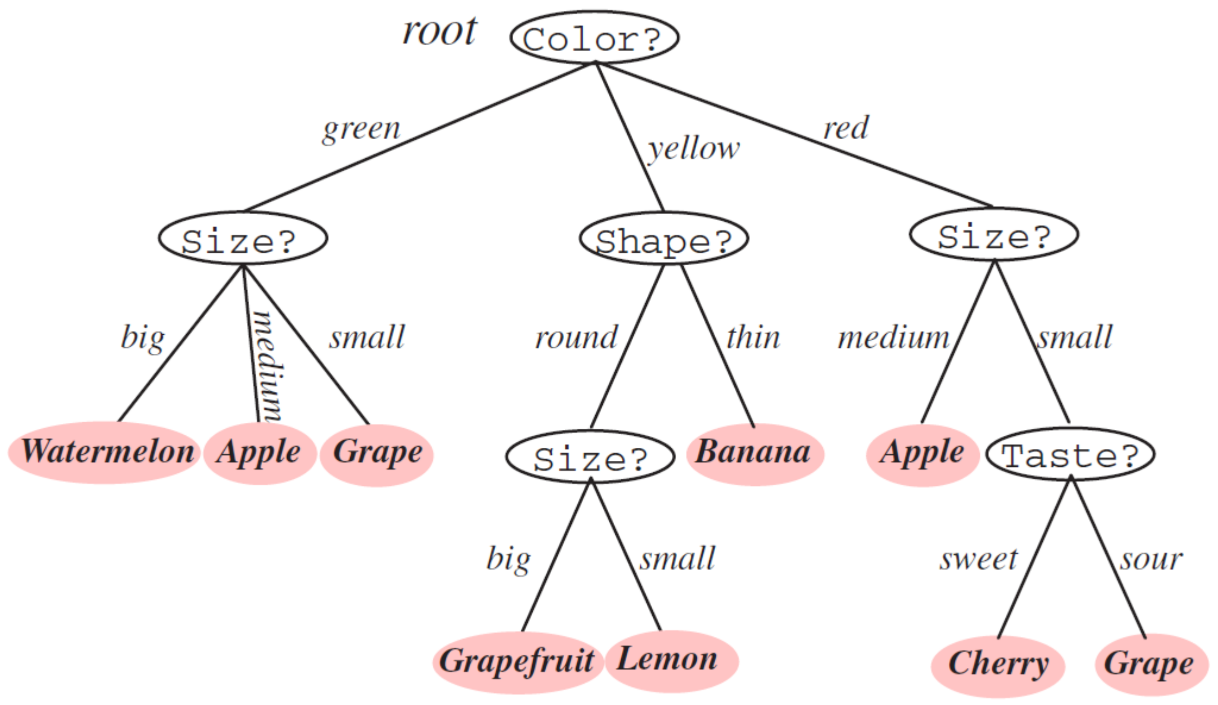
Nominal data (in this chapter)--that is, data that are discrete and without any natural notion of similarity or even ordering.

Decision Trees



These sequence of questions are a decision tree…

Structure of a Decision Tree



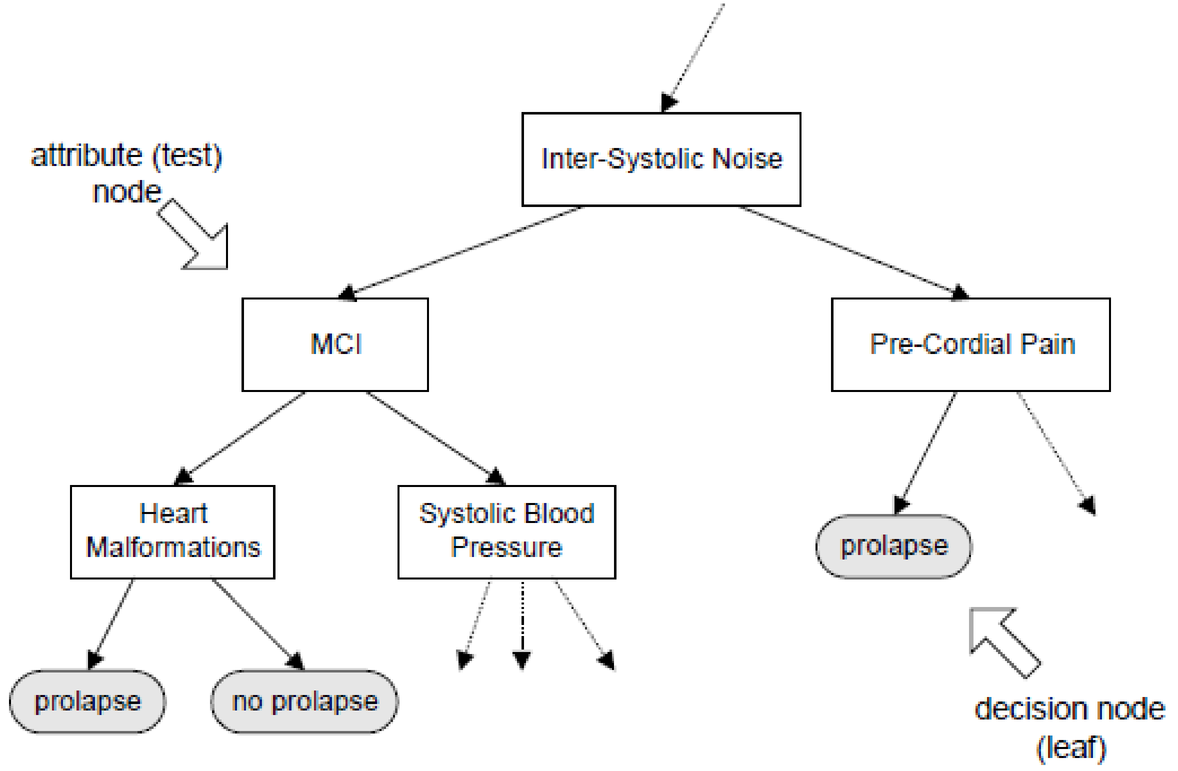
- The root node of the tree, displayed at the top, is connected to successive branches to the other nodes.
- Internal nodes correspond to attributes (features).
- Edges denote attributes assignment
- Leaves correspond to classification outcome

Structure of a Decision Tree

- The classification of a particular pattern begins at the root node, which queries a particular property (selected during tree learning).
- We follow the link corresponding to the appropriate value of the pattern and continue to a new node, at which we check the next property. And so on.
- The connections continue until the leaf nodes are reached, implying a decision.

When to Consider Decision Tree

- Instances are wholly or partly described by attribute-value pairs.
- Target function is discrete valued.
- Examples
  - Medical diagnosis.
  - Credit risk analysis.
  - …



History of Decision Tree

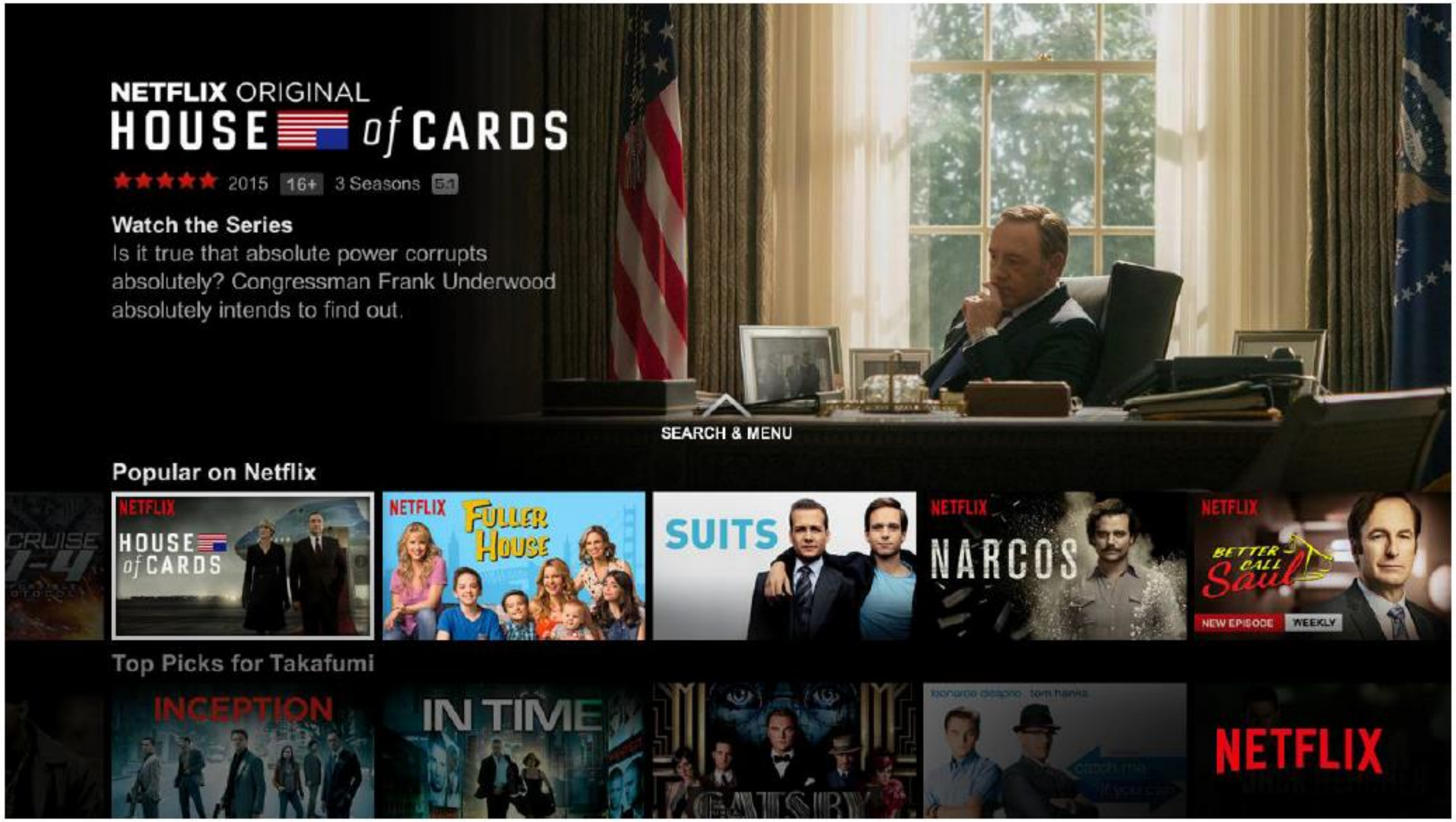
- 1966: First Decision Tree algorithm: CLS (Concept Learning System)
- 1979: ID3 made Decision Tree popular
- 1984: CART (Classification and Regression Tree)
- 1993: C4.5, most commonly used decision tree algorithm
- 2001: Random Forest, powerful algorithm based on a number of decision trees.

Decision Trees

- One of the most intuitive classifiers
- Easy to understand and construct
- Surprisingly, also works (very) well.
- Decision trees have a particularly high degree of interpretability.

Let’s build a decision tree!

Netflix


NETFLIX ORIGINAL
HOUSE of CARDS
2015 16+ 3 Seasons 5.1
Watch the Series
Is it true that absolute power corrupts absolutely? Congressman Frank Underwood absolutely intends to find out.
SEARCH & MENU
Popular on Netflix
HOUSE of CARDS
FULLER HOUSE
SUITS
NARCOS
BETTER CALL Saul
NEW EPISODE WEEKLY
Top Picks for Takafumi
INCEPTION
IN TIME
GATSBY
NETFLIX

Dataset

	Attributes(Features)				Label
Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Basic Idea

Divide-And-Conquer

1. Select a test for root node
   Create branch for each possible outcome of the test
2. Split instances into subsets
   One for each branch extending from the node
3. Repeat recursively for each branch, using only instances that reach the branch
4. Stop recursion for a branch if all its instances have the same class, or no more attribute for selection.

Building a Decision Tree

```
1: procedure TreeGenerate($\mathcal{D}, A$)
2:     Initialize: $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}, A = \{a_1, a_2, \dots, a_d\}$
3:     Generate node
4:     if all samples from $D$ belong to class $C$ then
5:         mark node as leaf node of class $C$; return
6:     end if
7:     if $A = \emptyset$ or samples from $\mathcal{D}$ have same values on $A$ then
8:         mark node as leaf node, class = most common class in $\mathcal{D}$; return
9:     end if
10:    Chose the best split attribute $a_*$ from $A$
11:    for each value $a_*^v$ in $a_*$ do
12:        create a branch for node; $D_v$ denotes subset consists of samples
   from $D$ have $a_*^v$
13:        use TreeGenerate($\mathcal{D}_v, A \backslash \{a_*\}$) as branch node
14:    end for
15: end procedure
```

Identifying “Best Attribute”

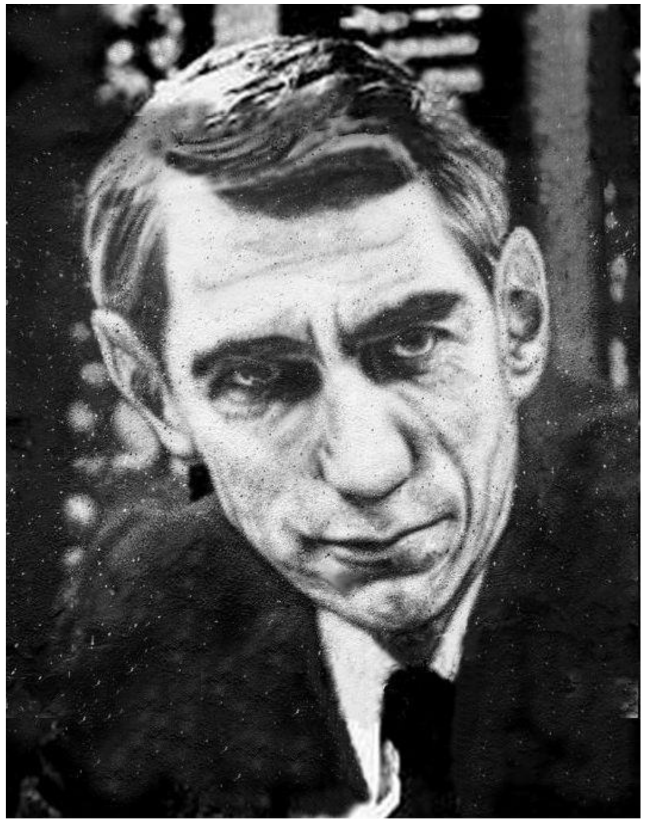
- There are many possible ways to select the best attribute for a given set.
- We want to grow a simple tree.
  A good attribute prefers attributes that split the data so that each successor node is as pure as possible.
- We will discuss one possible way which is based on information theory and generalizes well to non binary variables

Entropy

- Quantifies the amount of uncertainty associated with a specific probability distribution.
- The higher the entropy, the less confident we are in the outcome.

Definition:

$$H(p) = -\sum_{k=1}^{|\mathcal{Y}|} p_k log_2 p_k$$



Claude Shannon

(1916 - 2001)

Entropy

$$H(x) = -\sum_{i} p(x = i) log_2 p(x = i)$$

For Bernoulli distribution,

- If $p(x = 1) = 1$, then
  $$H(x) = -p(x = 1) \log p(x = 1) - p(x = 0) \log p(x = 0) = 0$$
- If $p(x = 1) = 0.5$, then
  $$H(x) = -p(x = 1) \log p(x = 1) - p(x = 0) \log p(x = 0) = 1$$

Interpreting Entropy

Entropy can be interpreted from an information standpoint.

- Assume both sender and receiver know the distribution.
  How many bits, on average, would it take to transmit one value?
- If $p(x = 1) = 1$ then the answer is 0 (we don't need to transmit anything).
- If $p(x = 1) = 0.5$ then the answer is 1 (either values is equally likely).
- If $0 < p(x = 1) < 0.5$ or $0.5 < p(x = 1) < 1$ then the answer is between 0 and 1.
- Why?

Expected Bits Per Symbol

Assume $p(x=1)=0.8$

Then $p(11)=0.64$, $p(10)=p(01)=0.16$, and $p(00)=0.04$

Let's define the following code:

- For 11 we send 0
- For 10 we send 10
- For 01 we send 110
- For 00 we send 1110

So: 01001101110001101110

Can be broken to: 01 00 11 01 11 00 01 10 11 10

Which is: 110 111 0 110 0 1110 110 10 0 10

What is the expected bits/symbol?

$$(0.64*1+0.16*2+0.16*3+0.04*4)/2=0.8$$

Entropy (lower bound) $H(x)=0.7219$

Conditional Entropy

- Entropy measures the uncertainty in a specific distribution.
- What if both sender and receiver know something about the transmission?
- For example, say I want to send the label (liked) when the length is known.
- This becomes a conditional entropy problem:

$$H(like|Length = v)$$

Conditional Entropy

$$
\begin{aligned}
H(Y|X) &= \sum_i p(x=i) H(Y|x=i) \\
&= -\sum_i p(x=i) \sum_j p(y=j|x=i) \log p(y=j|x=i) \\
&= -\sum_i \sum_j p(y=j, x=i) \log p(y=j|x=i)
\end{aligned}
$$

- $H(Y|X) = H(Y)$ if and only if Y and X are independent random variables.
- $H(Y|X) = 0$ if and only if the value of Y is completely determined by the value of X.

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?

Definition:

For data set $\mathcal{D}$ with label $k = 1, \dots, K$

$$IG(\mathcal{D}, a) = H(\mathcal{D}) - H(\mathcal{D}|a)$$

$$H(\mathcal{D}) = -\sum_{k=1}^{K} p_k log_2 p_k$$

$$H(\mathcal{D}|a) = \sum_{v \in value(a)} \frac{|\mathcal{D}^v|}{|\mathcal{D}|} H(\mathcal{D}^v)$$

Where $\mathcal{D}^v$ denotes the samples with value v for attribute a.

Where we are

- We were looking for a good criteria for selecting the best attribute for a node split.
- We defined the entropy, conditional entropy and information gain.
- We will now use information gain as our criteria for a good split.

Best Attribute will return the attribute that maximizes the information gain at each node.

Example: Root Attribute

$$p(Li = Yes) = \frac{6}{9} = \frac{2}{3}$$

$$p(Li = No) = \frac{3}{9} = \frac{1}{3}$$

$$H(Li) = -\left(\frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3}\right) = 0.92$$

$$H(Li|T) \\ = \frac{1}{3}H(Li|T = Comedy) \\ + \frac{1}{3}H(Li|T = Animated) \\ + \frac{1}{3}H(Li|T = Drama)$$

Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Example: Root Attribute

$$H(Li|T) = \frac{1}{3}H(Li|T = Comedy) + \frac{1}{3}H(Li|T = Animated) + \frac{1}{3}H(Li|T = Drama)$$

$$H(Li = Yes|T = Comedy) = \frac{2}{3}$$
$$\Rightarrow H(Li|T = Comedy) = 0.92$$
$$H(Li = Yes|T = Animated) = \frac{1}{3}$$
$$\Rightarrow H(Li|T = Animated) = 0.92$$
$$H(Li = Yes|T = Drama) = 1$$
$$\Rightarrow H(Li|T = Drama) = 0$$

$$H(Li|T) = \frac{1}{3} \times 0.92 + \frac{1}{3} \times 0.92 = 0.61$$
$$IG(Li,T) = H(Li) - H(Li|T) = 0.92 - 0.61 = 0.31$$

Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

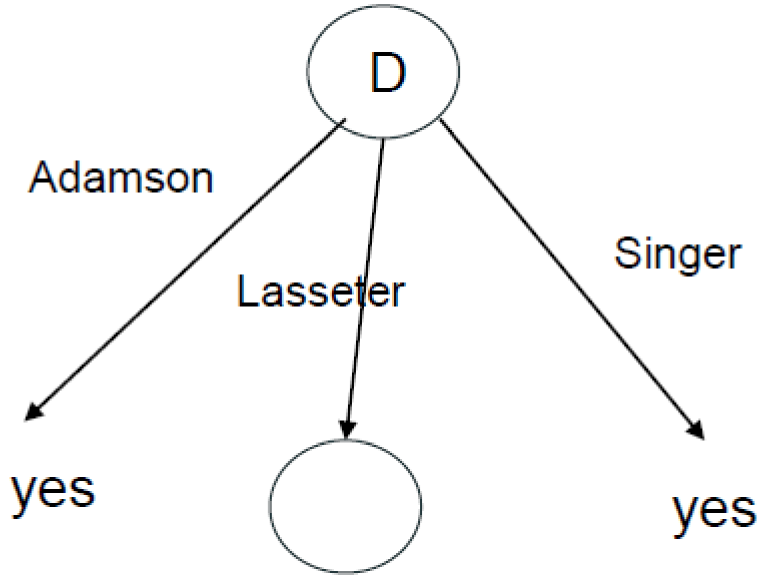
Example: Root Attribute

$$H(Li|T) = 0.61$$
$$H(Li|Le) = 0.61$$
$$H(Li|D) = 0.36$$
$$H(Li|F) = 0.85$$

$$IG(Li,T) = 0.92 - 0.61 = 0.31$$
$$IG(Li,Le) = 0.92 - 0.61 = 0.31$$
$$IG(Li,D) = 0.92 - 0.36 = 0.56$$
$$IG(Li,F) = 0.92 - 0.85 = 0.07$$

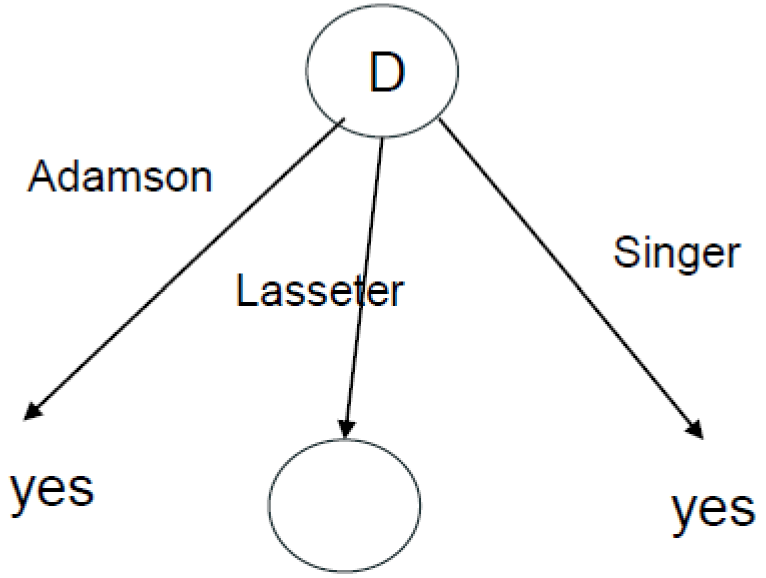
Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Building a Tree



Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

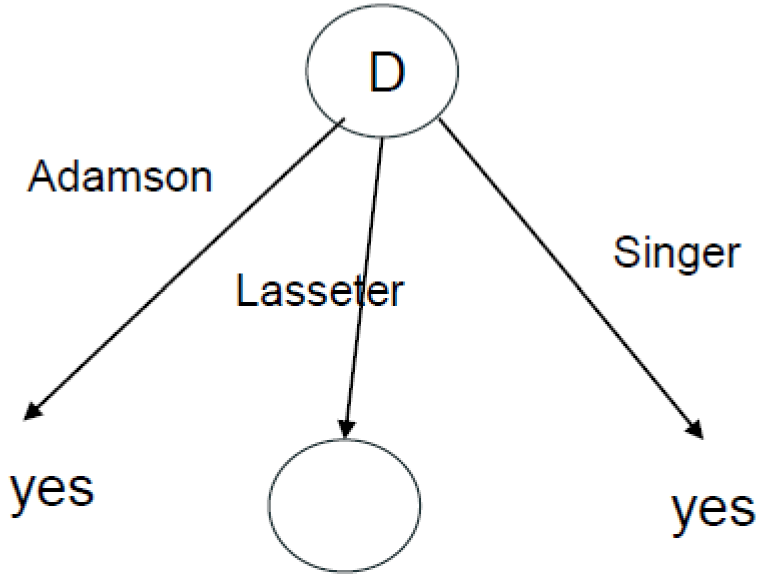
Building a Tree



Movie	Type	Length	Director	Famous actors	Liked
m2	Animated	Short	Lasseter	No	No
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m9	Drama	Medium	Lasseter	No	Yes

We only need to focus on the records (samples) associated with this node

Building a Tree



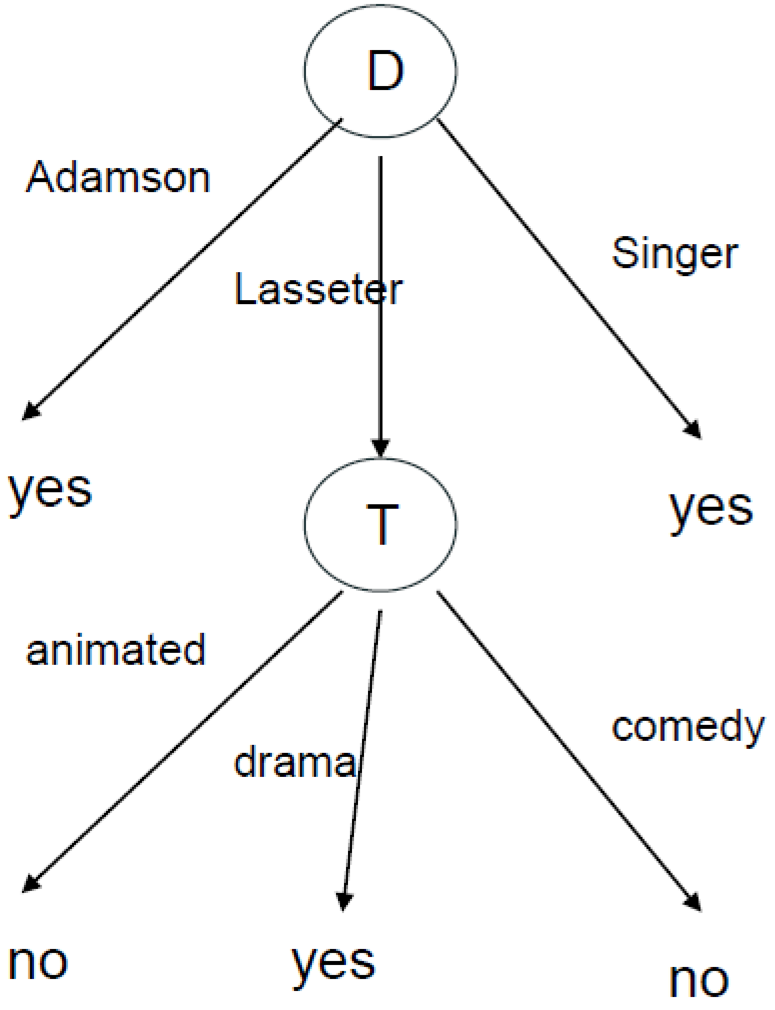
Movie	Type	Length	Famous actors	Liked
m2	Animated	Short	No	No
m4	Animated	long	Yes	No
m5	Comedy	long	Yes	No
m9	Drama	Medium	No	Yes

We eliminated the director attribute.
All samples have the same director.

$p(Li = yes | D = Lasseter) = 1/4$
$H(Li | D = Lasseter) = 0.81$

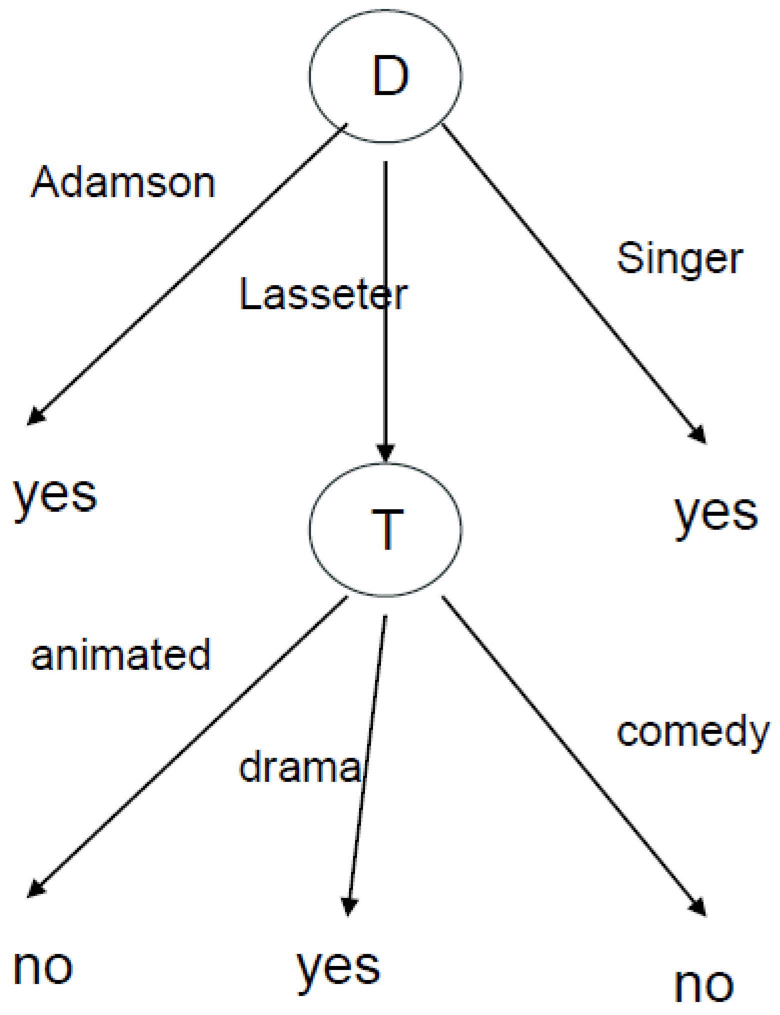
$H(Li | T, D = Lasseter) = 0$ $\qquad$ $IG(Li, T) = 0.81$
$H(Li | Le, D = Lasseter) = 0$ $\qquad$ $IG(Li, Le) = 0.81$
$H(Li | F, D = Lasseter) = 0.5$ $\qquad$ $IG(Li, F) = 0.31$

Building a Tree



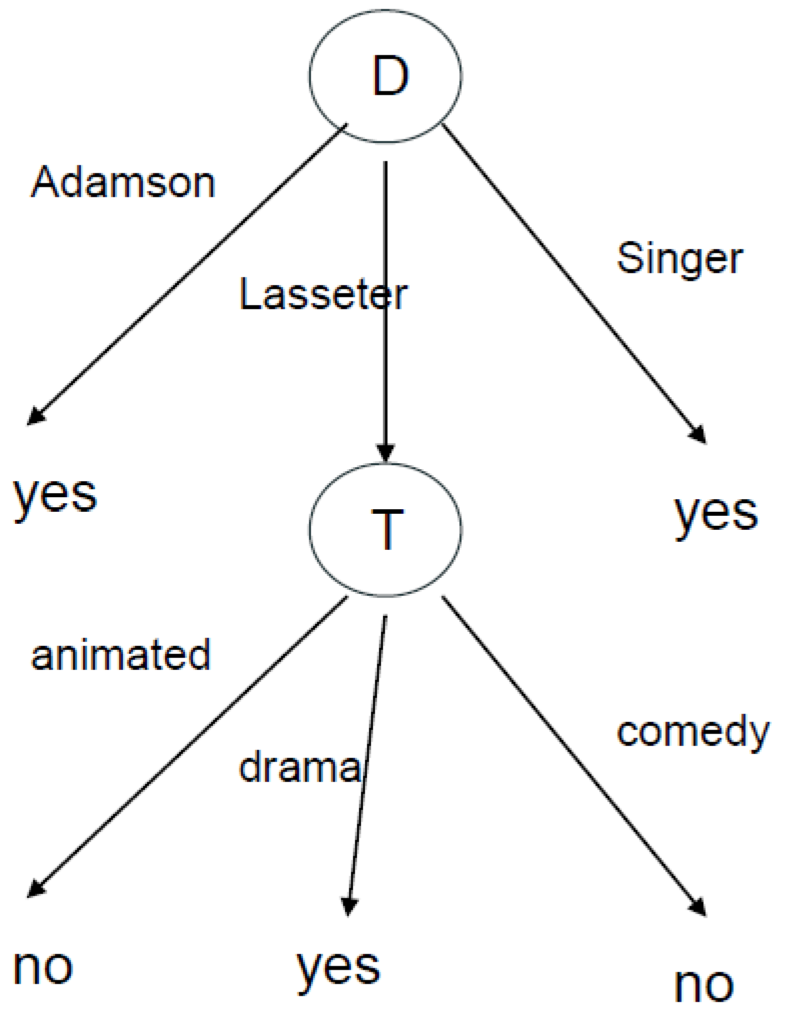
Movie	Type	Length	Famous actors	Liked
m2	Animated	Short	No	No
m4	Animated	long	Yes	No
m5	Comedy	long	Yes	No
m9	Drama	Medium	No	Yes

Final Tree



Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Tree to Rules



- Each path, from the root to a leaf, corresponds to a rule where all of the decisions leading to the leaf define the antecedent to the rule, and the consequent is the classification at the leaf node.
- For example, from the tree in the Director/Type example, we could generate the rule:

If Director = Lasseter & Type = drama, then yes

Building a Decision Tree

```
1:  procedure TreeGenerate($\mathcal{D}, A$)
2:      Initialize: $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}, A = \{a_1, a_2, \ldots, a_d\}$
3:      Generate node
4:      if all samples from $D$ belong to class $C$ then
5:          mark node as leaf node of class $C$; return
6:      end if
7:      if $A = \emptyset$ or samples from $\mathcal{D}$ have same values on $A$ then
8:          mark node as leaf node, class = most common class in $\mathcal{D}$; return
9:      end if
10:     Chose the best split attribute $a_*$ from $A$        IG
11:     for each value $a_*^v$ in $a_*$ do
12:         create a branch for node; $D_v$ denotes subset consists of samples
    from $D$ have $a_*^v$
13:         use TreeGenerate($\mathcal{D}_v, A \backslash \{a_*\}$) as branch node
14:     end for
15: end procedure
```

Attribute with Many Values

Information Gain for attribute a:

For data set $\mathcal{D}$ with label $k = 1, \dots, K$

$$IG(\mathcal{D}, a) = H(\mathcal{D}) - \sum_{v \in value(a)} \frac{|\mathcal{D}^v|}{|\mathcal{D}|} H(\mathcal{D}^v)$$

Problem: Information Gain will prefer the attribute that has many values.

Example:

- Build a decision tree for some data describing the customers of a business.
- Feature a: the customer's credit card number.
- a has a high mutual information (uniquely identifies each customer)
- But we do not want to include a in the decision tree: deciding how to treat a customer based on their credit card number is unlikely to generalize to customers we haven't seen before.

Attribute with Many Values (cont.)

- Many alternative measures to Information Gain.
- Gain Ratio:
  - Used in *e.g.*, C4.5
  - corrects the information gain by taking the intrinsic information of a split into account.

$$GR(\mathcal{D}, \mathrm{a}) = \frac{IG(\mathcal{D}, a)}{IV(\mathcal{D}, a)}$$

$$IV(\mathcal{D}, a) = -\sum_{v \in value(a)} \frac{|\mathcal{D}^v|}{|\mathcal{D}|} log_2 \frac{|\mathcal{D}^v|}{|\mathcal{D}|}$$

- The larger $value(a)$ is, the larger $IV(\mathcal{D}, a)$ is.

Attribute with Many Values (cont.)

Split for attribute "Movie":
{1, 1, 1, 1, 1, 1, 1, 1, 1}

$$IV(\mathcal{D}, M) = 9 * \left(-\frac{1}{9} log_2 \frac{1}{9}\right) = 3.17$$

$$IG(\mathcal{D}, M) = 0.92$$

$$GR(\mathcal{D}, M) = \frac{IG(\mathcal{D}, M)}{IV(\mathcal{D}, M)} = \frac{0.92}{3.17} = 0.29$$

Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Attribute with Many Values (cont.)

Split for attribute “Director”:
{3, 4, 2}

$$IV(\mathcal{D}, D) = -\frac{3}{9} log_2 \frac{3}{9} - \frac{4}{9} log_2 \frac{4}{9} - \frac{2}{9} log_2 \frac{2}{9} = 1.53$$

$$IG(\mathcal{D}, D) = 0.56$$

$$GR(\mathcal{D}, D) = \frac{IG(\mathcal{D}, D)}{IV(\mathcal{D}, D)} = \frac{0.56}{1.53} = 0.37$$

Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Attribute with Many Values (cont.)

$$IG(\mathcal{D}, D) = 0.56$$

$$GR(\mathcal{D}, D) = \frac{IG(\mathcal{D}, D)}{IV(\mathcal{D}, D)} = \frac{0.56}{1.53}$$
$$= 0.37$$

$$IG(\mathcal{D}, M) = 0.92$$

$$GR(\mathcal{D}, M) = \frac{IG(\mathcal{D}, M)}{IV(\mathcal{D}, M)} = \frac{0.92}{3.17}$$
$$= 0.29$$

$$GR(\mathcal{D}, D) > GR(\mathcal{D}, M)$$

So we choose attribute `Director`

Movie	Type	Length	Director	Famous actors	Liked
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	Animated	long	Lasseter	Yes	No
m5	Comedy	long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
m7	Animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Attribute with Many Values (cont.)

- Most popular alternative: Gini Index.
  - Used in *e.g.*, CART
  - impurity measure (instead of entropy)

$$Gini(\mathcal{D}) = \sum_k p_k(1 - p_k) = 1 - \sum_k p_k^2$$

where p_k means the probability of label k.
Gini impurity is a measure of how often the labels of two random samples from $\mathcal{D}$ are different.

  - Gini Index for attribute a:

$$GI(\mathcal{D}, a) = \sum_{v \in value(a)} \frac{|\mathcal{D}^v|}{|\mathcal{D}|} Gini(\mathcal{D}^v)$$

- Select the attribute with the minimum Gini Index

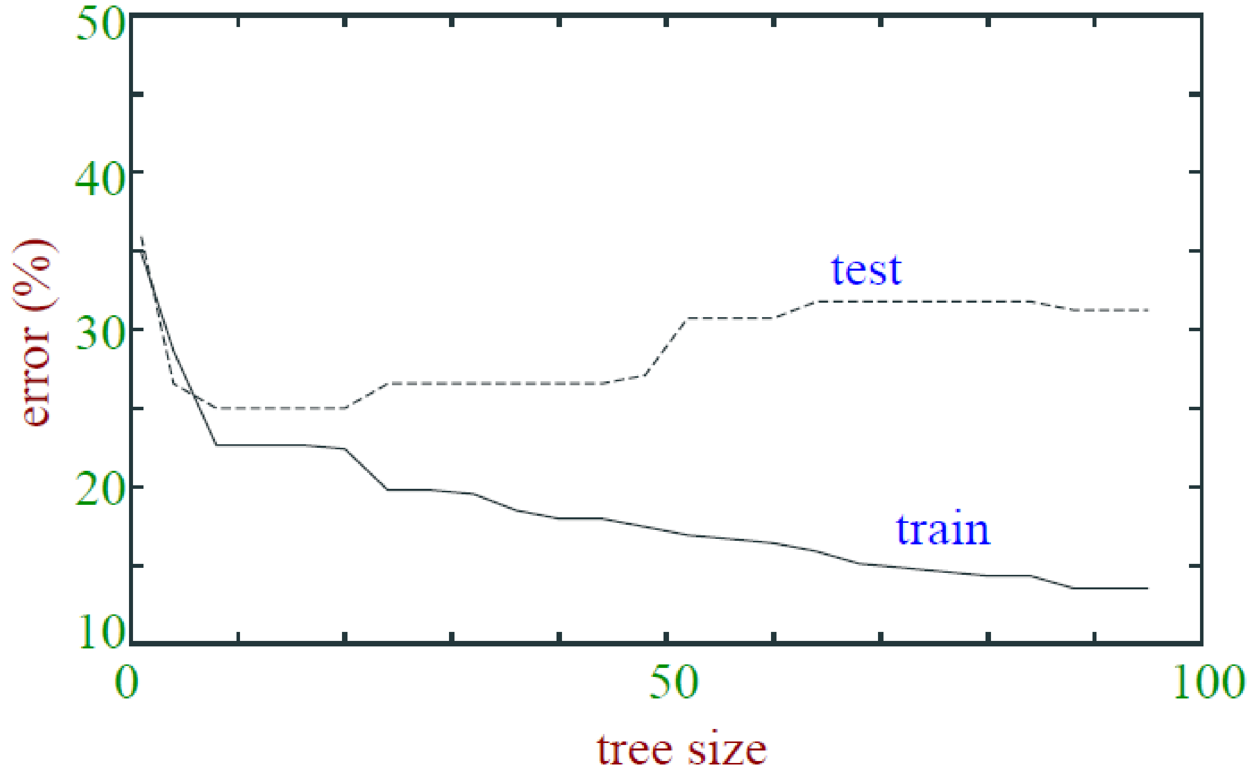
Building a Decision Tree

```
1: procedure TreeGenerate($\mathcal{D}, A$)
2:     Initialize: $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}, A = \{a_1, a_2, \ldots, a_d\}$
3:     Generate node
4:     if all samples from $D$ belong to class $C$ then
5:         mark node as leaf node of class $C$; return
6:     end if
7:     if $A = \emptyset$ or samples from $\mathcal{D}$ have same values on $A$ then
8:         mark node as leaf node, class = most common class in $\mathcal{D}$; return
9:     end if
10:    Chose the best split attribute $a_*$ from $A$
11:    for each value $a_*^v$ in $a_*$ do
12:        create a branch for node; $D_v$ denotes subset consists of samples
   from $D$ have $a_*^v$
13:        use TreeGenerate($\mathcal{D}_v, A \backslash \{a_*\}$) as branch node
14:    end for
15: end procedure
```

Stop Conditions

- The algorithm we gave reaches homogenous nodes (or runs out of attributes).
- This is dangerous: For datasets with many (non relevant) attributes the algorithm will continue to split nodes.
- This will lead to overfitting!

Overfitting in Decision Trees



Overfitting Illustration

- Trees must be big enough to fit training data.
- BUT: trees that are too big may overfit.

Avoiding overfitting: Tree Pruning

- Pre-Pruning
  - Stop growing a branch when information becomes unreliable.
  - Pre-pruning can “stop early”.
- Post-Pruning
  - Grow a decision tree that correctly classifies all training data
  - Simplify it later by replacing some nodes with leafs.
  - Post-pruning preferred in practice

Pre-pruning

- Divide data set into training set and validation set. Try to select split attribute to split nodes using the training set, but decide whether to split using the validation set.

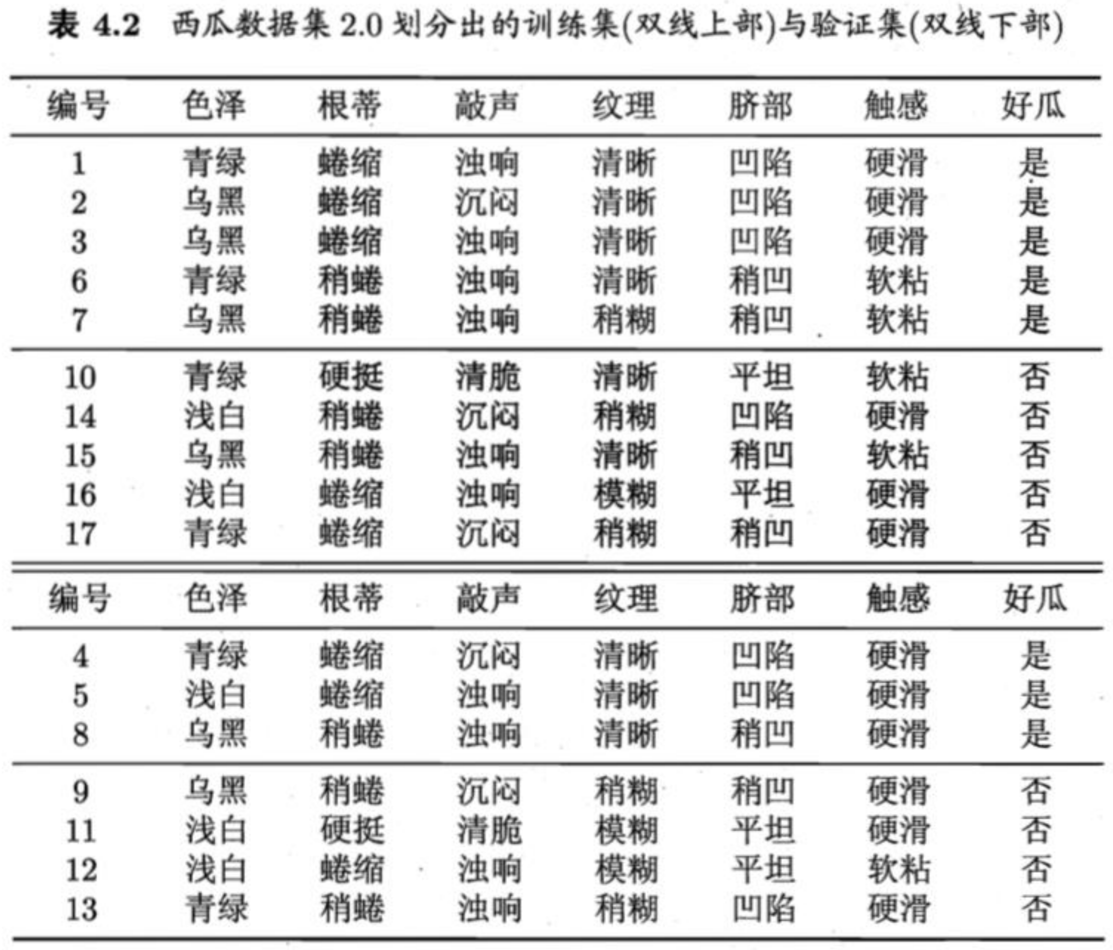


表 4.2 西瓜数据集 2.0 划分出的训练集(双线上部)与验证集(双线下部)

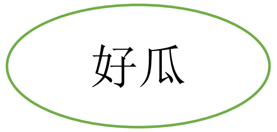
Training set

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

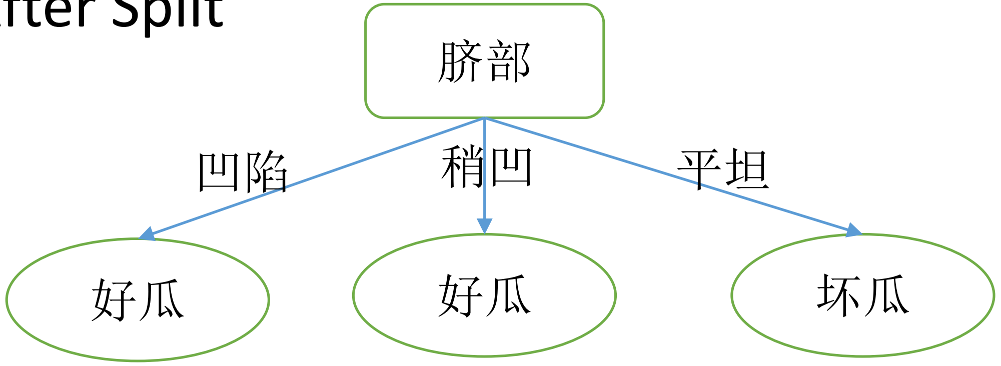
Validation set

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

Pre-pruning



- Before split, Only root
  - Validation accuracy $= \frac{3}{7} \times 100\% = 42.9\%.$
- After Split



  - Validation accuracy $= \frac{2+1+2}{7} \times 100\% = 71.4\%.$
- So split the tree.

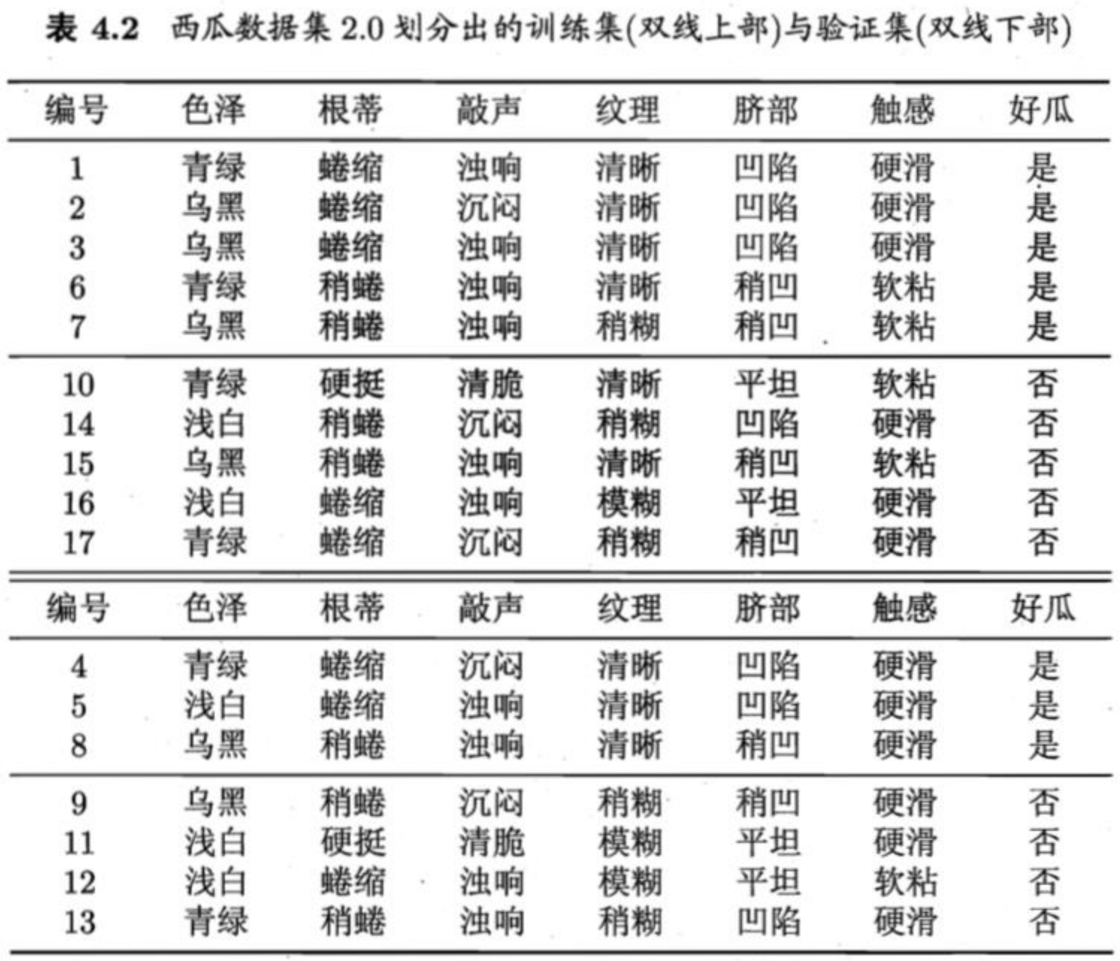


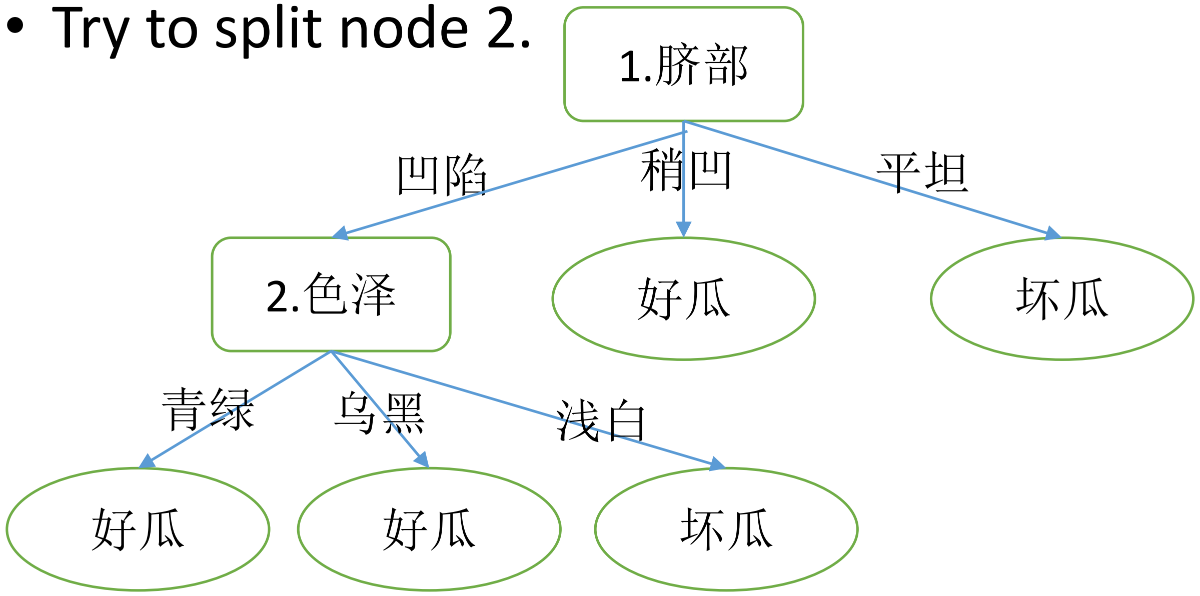
表 4.2 西瓜数据集 2.0 划分出的训练集(双线上部)与验证集(双线下部)

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

Pre-pruning

- Try to split node 2.



  - Validation accuracy $= \frac{4}{7} \times 100\% = 57.1\%$
- So don’t split node 2.
- Repeat…
- Problem: under-fitting

- Try to split node 2.
  - Validation accuracy $= \frac{4}{7} \times 100\% = 57.1\%$
- So don’t split node 2.
- Repeat…
- Problem: under-fitting

Post-pruning

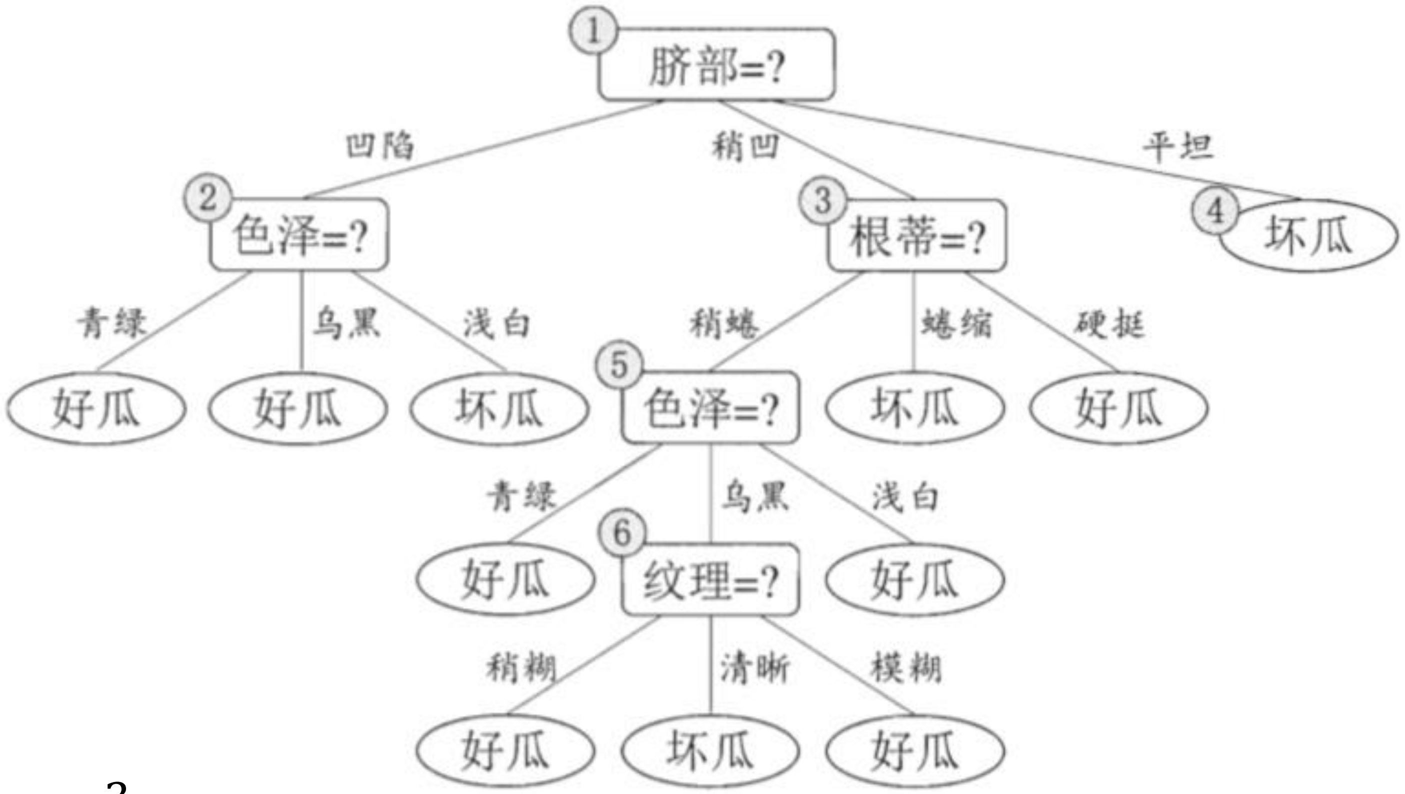
Split data into train and validation set.

Build tree using training set.

- For all internal nodes.
  - Remove sub tree rooted at node.
  - Assign class to be the most common among training set.
  - check test data error
    - If error is lower, keep change.
    - Otherwise restore subtree, repeat for all nodes in subtree.

Post-pruning

- Build the tree using training set.

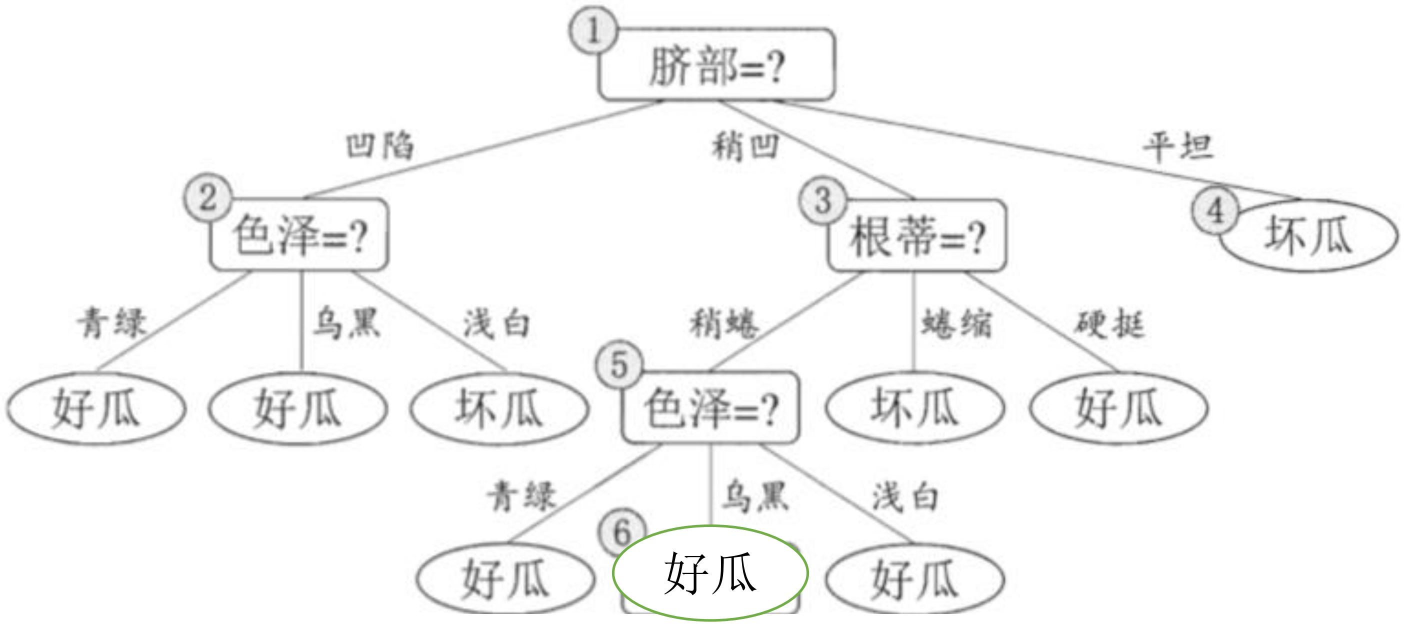


$$Validation\ accuracy = \frac{3}{7} \times 100\% = 42.9\%$$

图 4.5 基于表 4.2 生成的未剪枝决策树

Post-pruning

- First consider node 6. If we remove node 6, and label it as 好瓜.



$$Validation\ accuracy = \frac{4}{7} \times 100\% = 57.1\%$$

- So we prune the node 6.

Post-pruning

- Finally

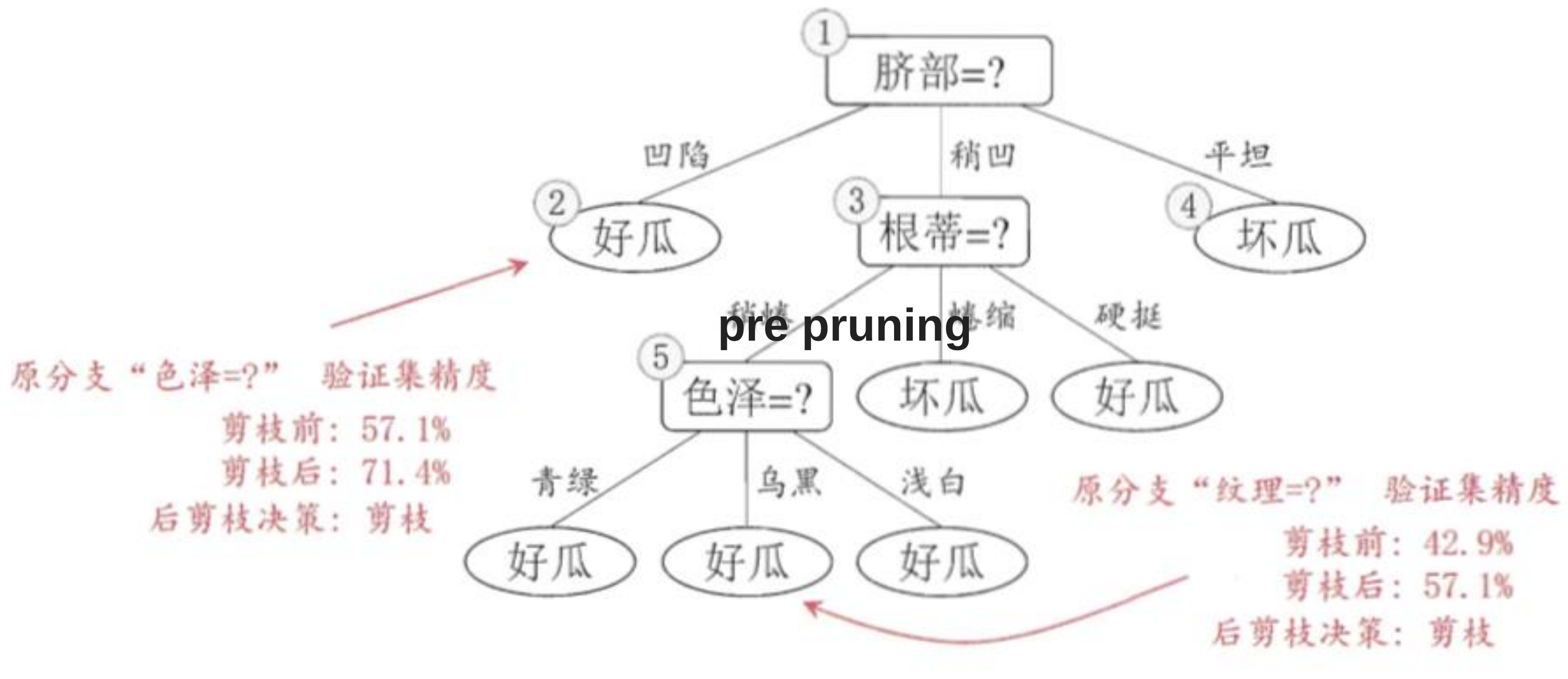


图 4.7 基于表 4.2 生成的后剪枝决策树

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

Pre-pruning & Post-pruning

- **Pre-pruning**

  stop growing the tree earlier, before it perfectly classifies the training set.
- **Post-pruning**

  allows the tree to perfectly classify the training set, and then post prune the tree.

The post-pruning usually retains more branches than the pre-pruning, and the under-fitting risk is small, so the generalization performance of the post-pruning is often better than that of the pre-pruning.

But the post-pruning process starts from the bottom up, so it takes more time to train than the pre-pruning.

Building a Decision Tree for Continuous Data

```
1:  procedure TreeGenerate($\mathcal{D}, A$)
2:      Initialize: $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}, A = \{a_1, a_2, \ldots, a_d\}$
3:      Generate node
4:      if all samples from $D$ belong to class $C$ then
5:          mark node as leaf node of class $C$; return
6:      end if
7:      if $A = \emptyset$ or samples from $\mathcal{D}$ have same values on $A$ then
8:          mark node as leaf node, class = most common class in $\mathcal{D}$; return
9:      end if
10:     Chose the best split attribute $a_*$ from $A$
11:     for each value $a_*^v$ in $a_*$ do
12:         create a branch for node; $D_v$ denotes subset consists of samples
    from $D$ have $a_*^v$
13:         use TreeGenerate($\mathcal{D}_v, A \setminus \{a_*\}$) as branch node
14:     end for
15: end procedure
```

Continuous Values

- Create a discrete attribute to test continuous
- Temperature = 82.5
- (Temperature > 72) = T, F
  - Either use threshold to turn into binary or discretize.
  - It is possible to compute information gain for all possible thresholds (there are a finite number of training samples)
  - Harder if we wish to assign more than two values (can be done recursively).

Temperature	**40**	**64**	**71**	**81**
Hot	F	F	F	T

Missing Values

What if some examples are missing values of attribute a?

- How to select partition attribute in case of missing attribute values?
- Given a partition attribute, if the value of the sample is missing, how do we divide the sample?

Missing Values

- Given training set D and attribute a, let $\widetilde{D}$ be the subset of D with no missing values on attribute a. Suppose that there are $|V|$ values for attribute a ($\{a^1, a^2, \dots, a^V\}$). Let $\widetilde{D}^v$ be the subset of $\widetilde{D}$ with $a = a^v$, $\widetilde{D}_k$ be the subset of $\widetilde{D}$ with class k. Then $\widetilde{D} = \bigcup_{k=1}^{K} \widetilde{D}_k$ and $\widetilde{D} = \bigcup_{v=1}^{V} \widetilde{D}^v$. We give a weight w_x for each sample x, and define

$$\rho = \frac{\sum_{x \in \widetilde{D}} w_x}{\sum_{x \in D} w_x}$$

$$\tilde{p}_k = \frac{\sum_{x \in \widetilde{D}_k} w_x}{\sum_{x \in \widetilde{D}} w_x}$$

$$\tilde{r}_v = \frac{\sum_{x \in \widetilde{D}^v} w_x}{\sum_{x \in \widetilde{D}} w_x}$$

$$Gain(D, a) = \rho \times Gain\left(\widetilde{D}, a\right) = \rho \times (H\left(\widetilde{D}\right) - \sum_{v=1}^{V} H(\widetilde{D}^v))$$

Missing Values

- If the value on attribute a of sample x is known, then divide sample x into the sub-node with the corresponding value and keep weight w_x. Otherwise, divide sample x into each sub-node with attribute $a = a^v$ and adjust weight to $\tilde{r}_v \cdot w_x$.

Example

- Take attribute 色泽 as an example. $\widetilde{D} = \{2,3,4,6,7,8,9,10,11,12,14,15,16,17\}$, Set $w_k = 1$.
- $H(\widetilde{D}) = -\left(\frac{6}{14}\log_2\frac{6}{14} + \frac{8}{14}\log_2\frac{8}{14}\right) = 0.985$
- Let $\widetilde{D}^1, \widetilde{D}^2\ and\ \widetilde{D}^3$ are the subsets of $\widetilde{D}$ with attribute 色泽 are 青绿, 乌黑 and 浅白 respectively.
- $H(\widetilde{D}^1) = -\left(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right) = 1$
- $H(\widetilde{D}^2) = -\left(\frac{4}{6}\log_2\frac{4}{6} + \frac{2}{6}\log_2\frac{2}{6}\right) = 0.918$
- $H(\widetilde{D}^3) = -\left(\frac{0}{4}\log_2\frac{0}{4} + \frac{4}{4}\log_2\frac{4}{4}\right) = 0$

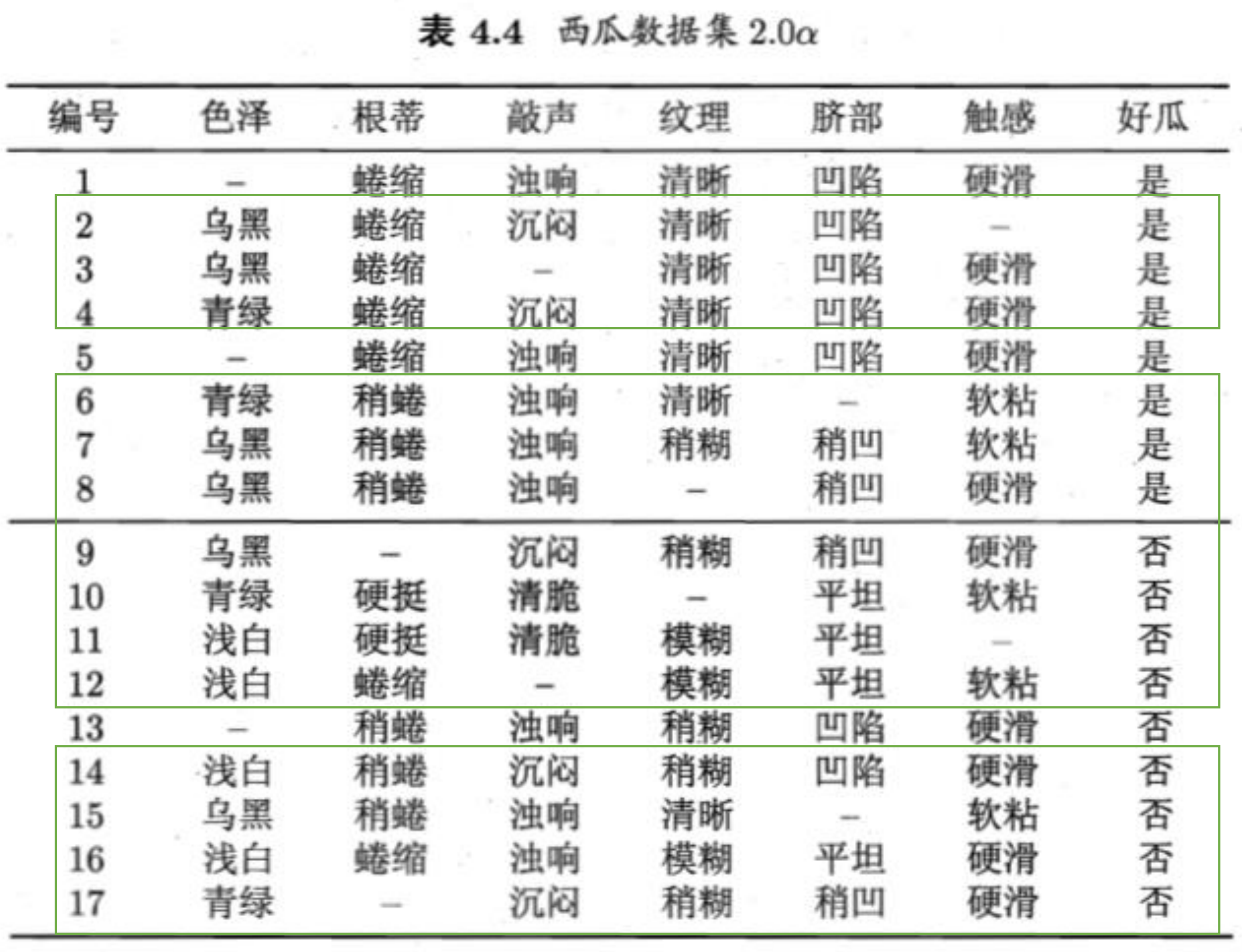


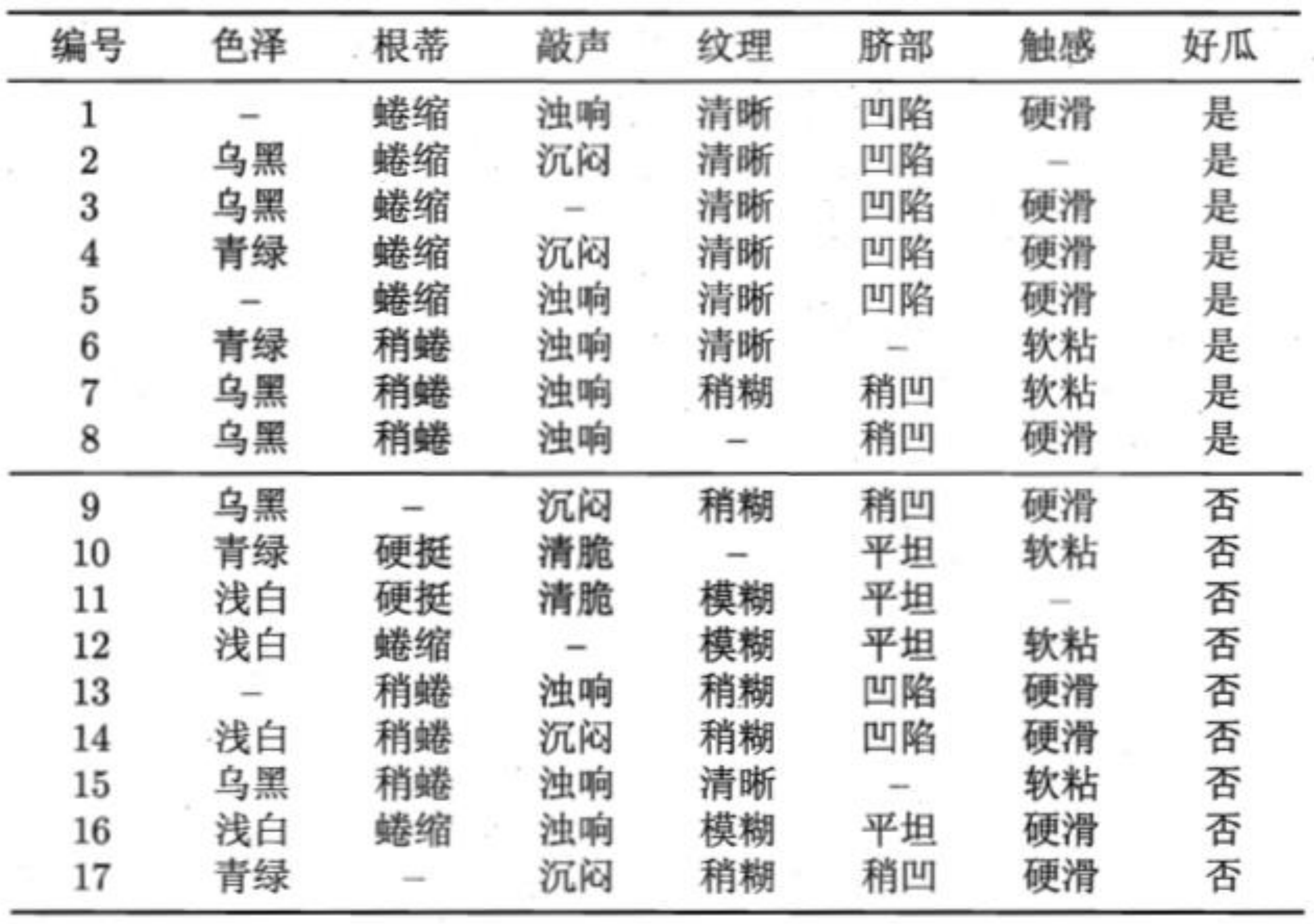
表 4.4 西瓜数据集 2.0α

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

Example

- So the Information Gain on attribute 色泽 of data set $\widetilde{D}$ is
- $Gain(\widetilde{D}, 色泽) = H(\widetilde{D}) - \sum_{v=1}^{3} \tilde{r} H(\widetilde{D}^v) = 0.985 - (\frac{4}{14} \times 1 +$

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

Example

- So we select 纹理 as the partition attribute. Samples {1,2,3,4,5,6,15} are divided into branch with 纹理=清晰, samples {7,9,13,14,17} into 纹理=稍糊, samples {11,12,16} into 纹理=模糊.
- Notice that sample {8} miss the value on attribute 纹理, so it's divided into all three branches with weights 7/15, 5/15 and 3/15 respectively. And sample {10} do the same thing.

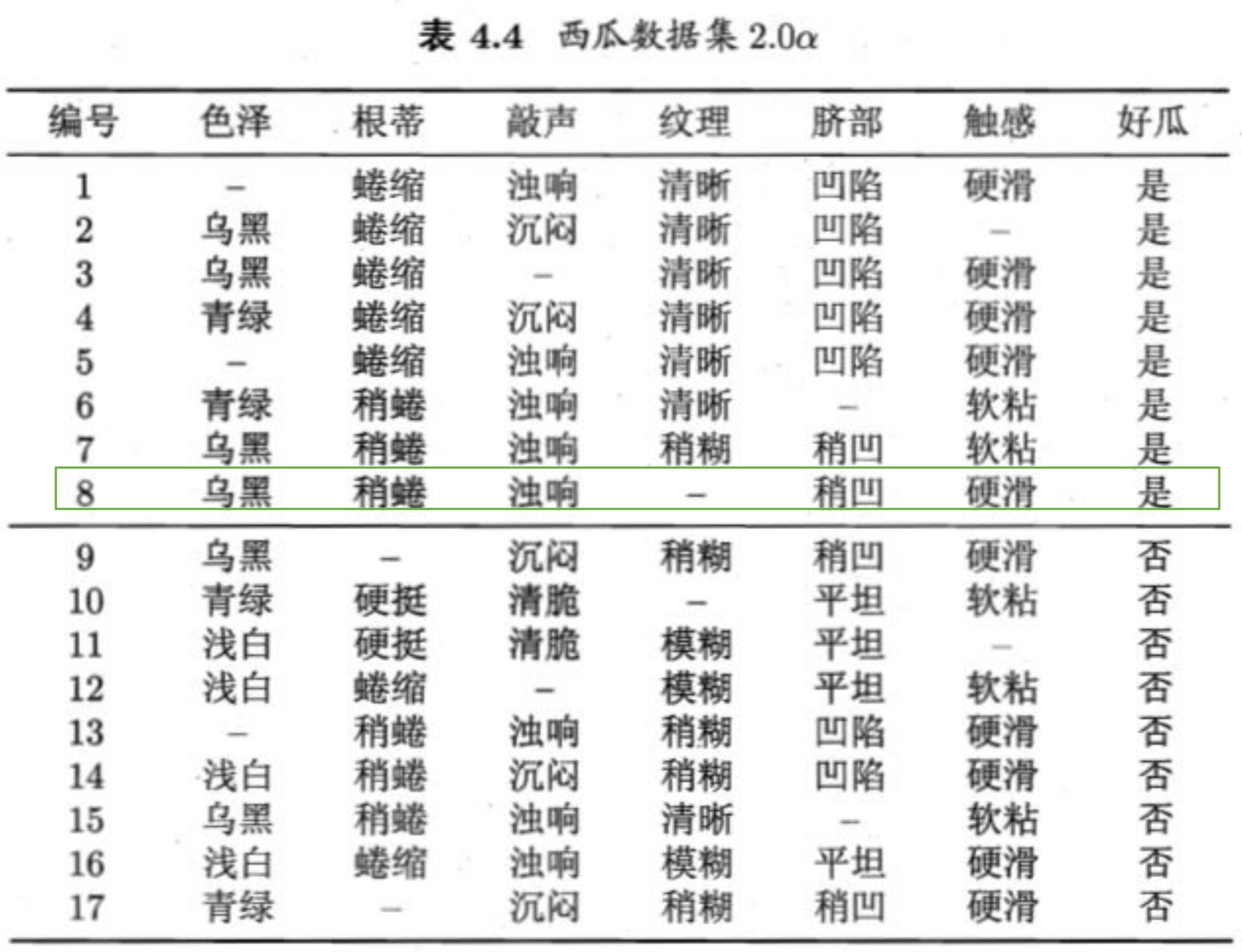


表 4.4 西瓜数据集 2.0α

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

Random Forest

- Decision Tree algorithm only generate one tree.
- Random Forest can generate multiple trees to improve the generalization ability.

- Random Forest
  - Repeat m times
    - Randomly draw n samples
    - Randomly select p attributes/features when growing the tree
  - Ensemble the m trees to predict each sample.

Tools for Decision Tree

- ID3, C4.5, C5.0
  http://www.rulequest.com/Personal/
- J4.8
  http://www.cs.waikato.ac.nz/ml/weka/

Tools for Decision Tree

- *Machine Learning: A Probabilistic Perspective*: Chapter 16.2
- *Pattern Recognition and Machine Learning*: Chapter 14.4