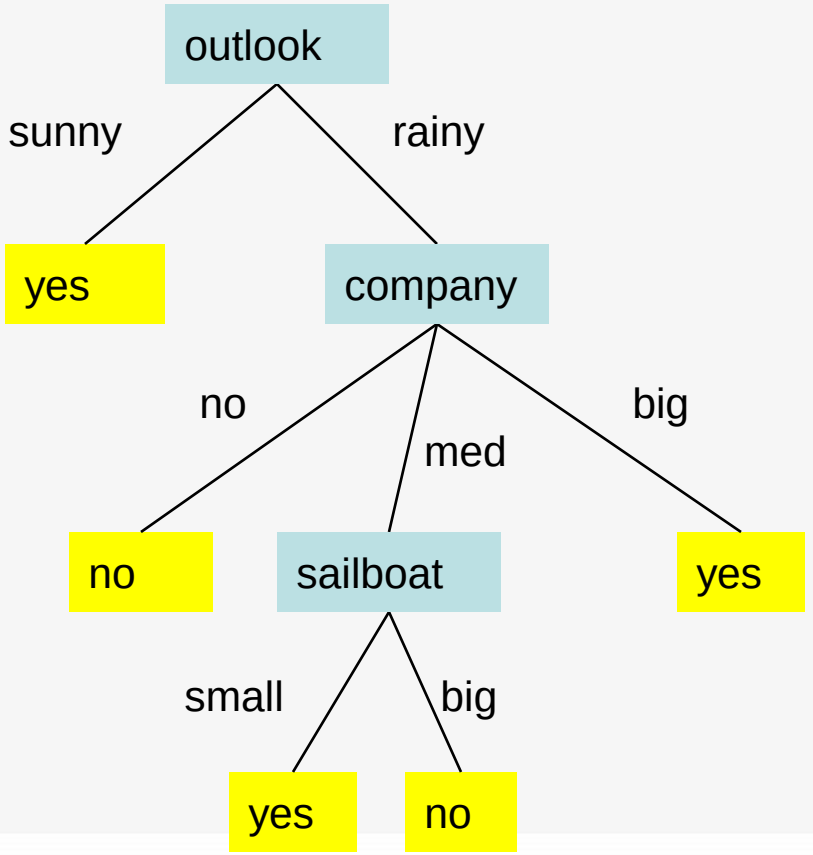
人工智能

决策树

代建华 教授、博士生导师
湖南师范大学信息科学与工程学院

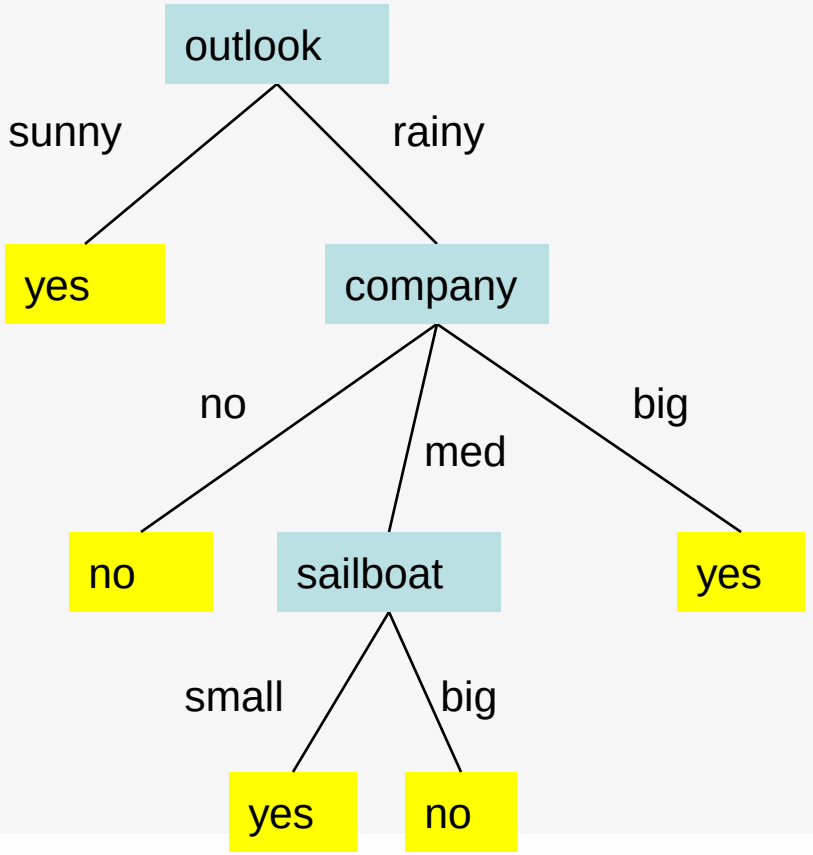
An Example Data Set and Decision Tree

#	*Attribute*			*Class*
	Outlook	Company	Sailboat	Sail?
1	sunny	big	small	yes
2	sunny	med	small	yes
3	sunny	med	big	yes
4	sunny	no	small	yes
5	sunny	big	big	yes
6	rainy	no	small	no
7	rainy	med	small	yes
8	rainy	big	big	yes
9	rainy	no	big	no
10	rainy	med	big	no



Classification

#	*Attribute*			*Class*
	Outlook	Company	Sailboat	Sail?
1	sunny	no	big	?
2	rainy	big	small	?

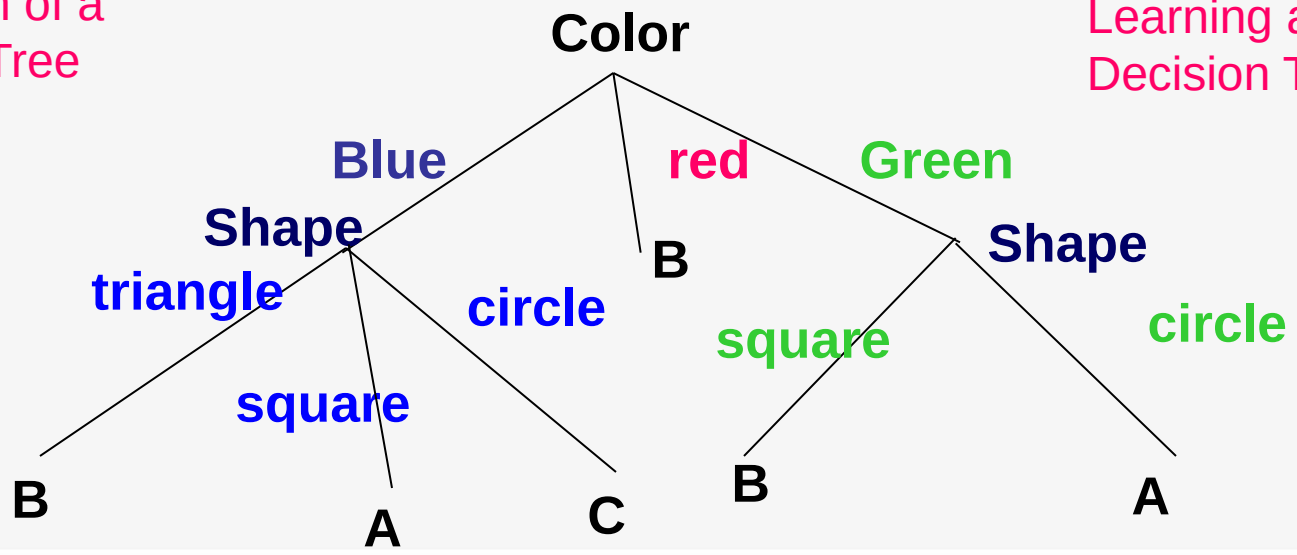


Decision Trees: The Representation

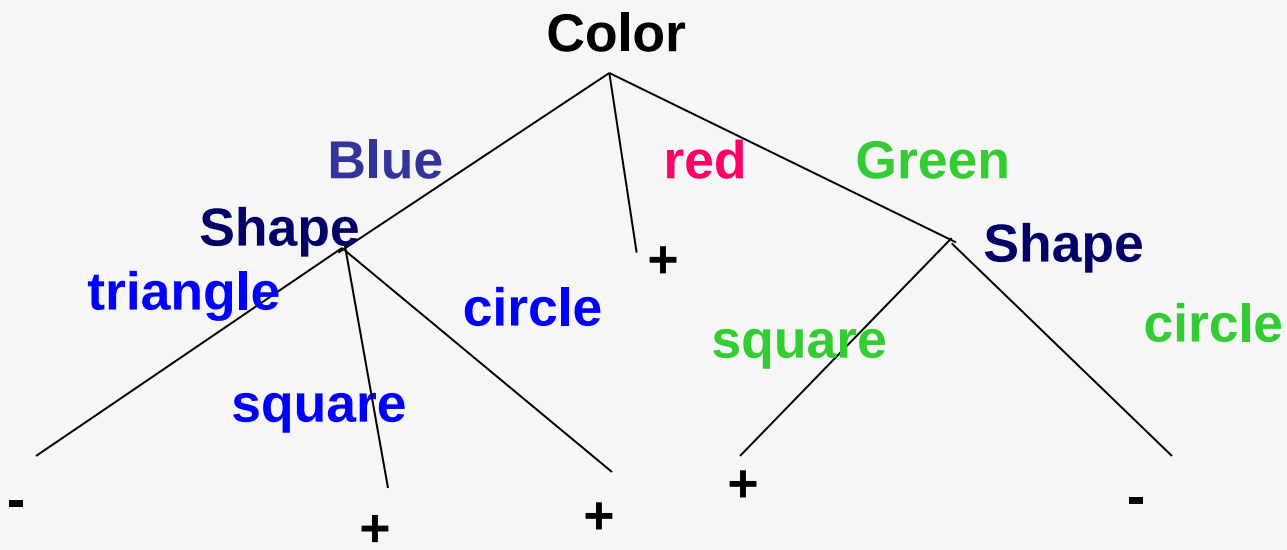
- Decision Trees are classifiers for instances represented as features vectors. (color= ;shape= ;label=)
- Nodes are tests for feature values;
- There is one branch for each value of the feature
- Leaves specify the categories (labels)
- Can categorize instances into multiple disjoint categories

Evaluation of a Decision Tree

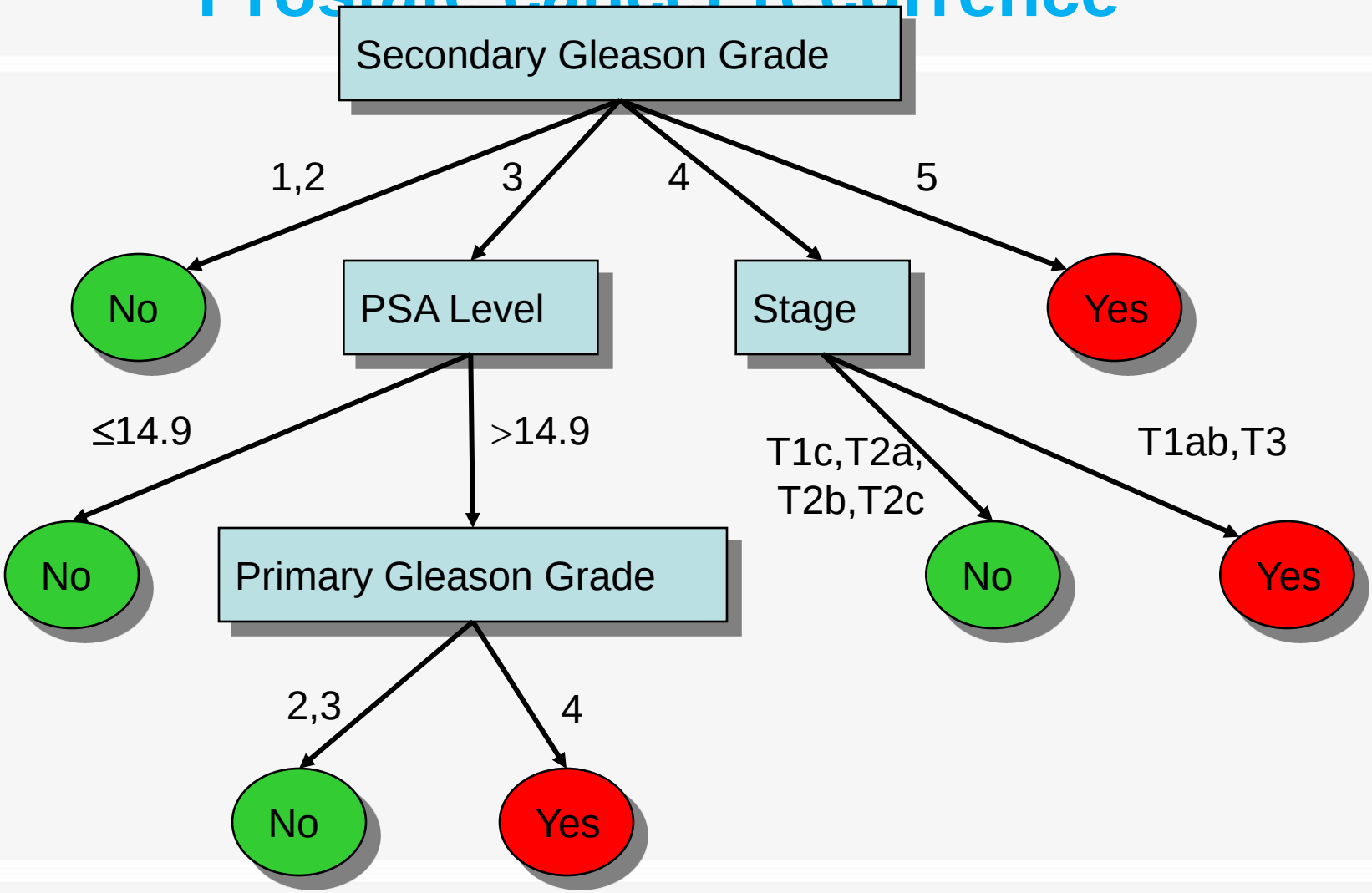
Learning a Decision Tree



Decision Trees



Prostate cancer recurrence



Learning Algorithm

- DT(Examples, Attributes)

  If all Examples have same label: return a leaf node with Label

  Else

  If Attributes is empty: return a leaf with majority Label

  Else

  Pick an attribute A as root

  For each value v of A

  Let Examples(v) be all the examples for which A=v

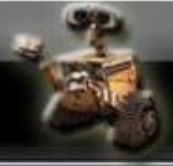
  Add a branch out of the root for the test A=v

  If Examples(v) is empty

  create a leaf node labeled with the majority label in Examples

  Else recursively create subtree by calling

  DT(Examples(v), Attribute-{A})



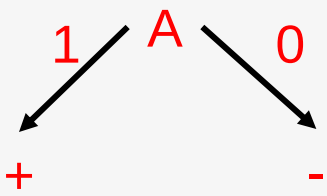
Picking the Root Attribute

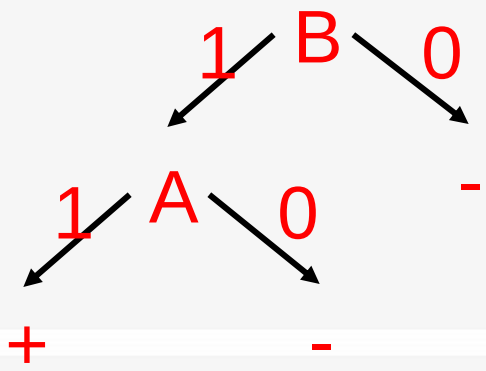
- The goal is to have the resulting decision tree as small as possible (Occam's Razor)
- Finding the minimal decision tree consistent with the data is NP-hard
- The recursive algorithm is a greedy heuristic search for a simple tree, but cannot guarantee optimality.
- The main decision in the algorithm is the selection of the next attribute to condition on.

Picking the Root Attribute

- Consider data with two Boolean attributes (A,B).
  - < (A=0,B=0), - >: 50 examples
  - < (A=0,B=1), - >: 50 examples
  - < (A=1,B=0), - >: 0 examples
  - < (A=1,B=1), + >: 100 examples

- What should be the first attribute we select?
- Splitting on A: we get purely labeled nodes.





- Splitting on B: we don't get purely labeled nodes.
- What if we have: <(A=1,B=0), - >: 2 examples

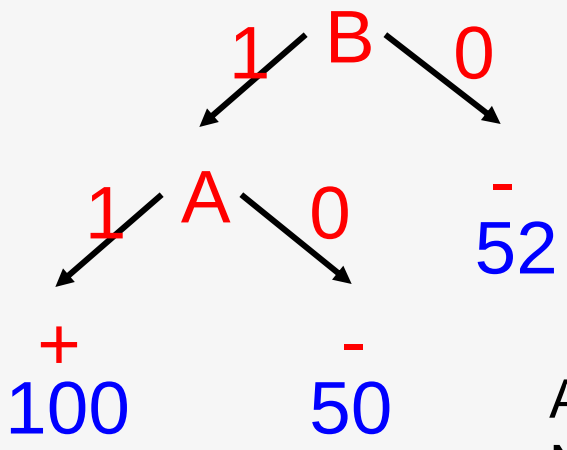
Picking the Root Attribute

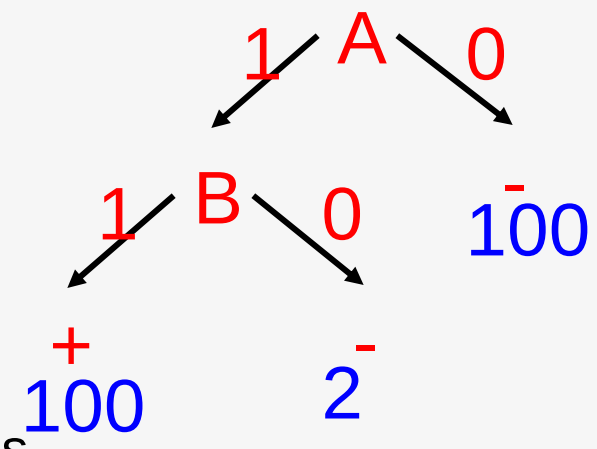
- Consider data with two Boolean attributes (A,B).

< (A=0,B=0), - >: 50 examples
< (A=0,B=1), - >: 50 examples
< (A=1,B=0), - >: 0 examples
< (A=1,B=1), + >: 100 examples

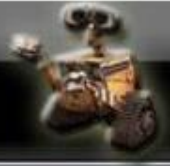
2 examples

- Trees looks structurally similar; which attribute should we choose?





Advantage A. But…
Need a way to quantify things



Picking the Root Attribute

- The goal is to have the resulting decision tree as small as possible (Occam's Razor)
- The main decision in the algorithm is the selection of the next attribute to condition on.
- We want attributes that split the examples to sets that are relatively pure in one label; this way we are closer to a leaf node.
- The most popular heuristics is based on information gain, originated with the ID3 system of Quinlan.

Entropy

- Entropy (impurity, disorder) of a set of examples, S, relative to a binary classification is:

$$Entropy(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

where p_+ is the proportion of positive examples in S and

p_- is the proportion of negatives.

- If all the examples belong to the same category: *Entropy* = 0
- If the examples are equally mixed (0.5,0.5) *Entropy* = 1

In general, when p_i is the fraction of examples labeled i:

$$Entropy(\{p_1, p_2, \ldots p_k\}) = -\sum_{i=1}^{k} p_i \log(p_i)$$

Entropy can be viewed as the number of bits required, on average, to encode the class of labels. If the probability for + is 0.5, a single bit is required for each example; if it is 0.8 -- can use less then 1 bit.

Entropy

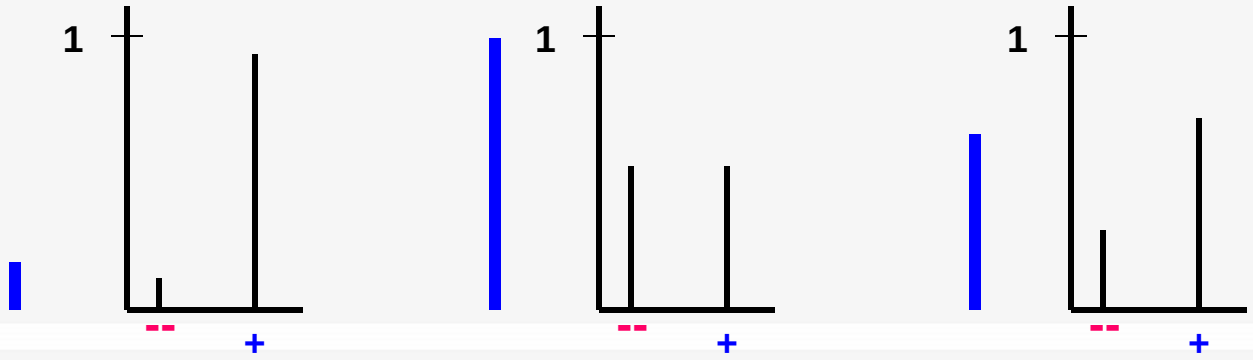
- Entropy (impurity, disorder) of a set of examples, S, relative to a binary classification is:

$$\text{Entropy}(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

where p_+ is the proportion of positive examples in S and

p_- is the proportion of negatives.

- If all the examples belong to the same category: *Entropy* = 0
- If the examples are equally mixed (0.5,0.5) *Entropy* = 1



Entropy

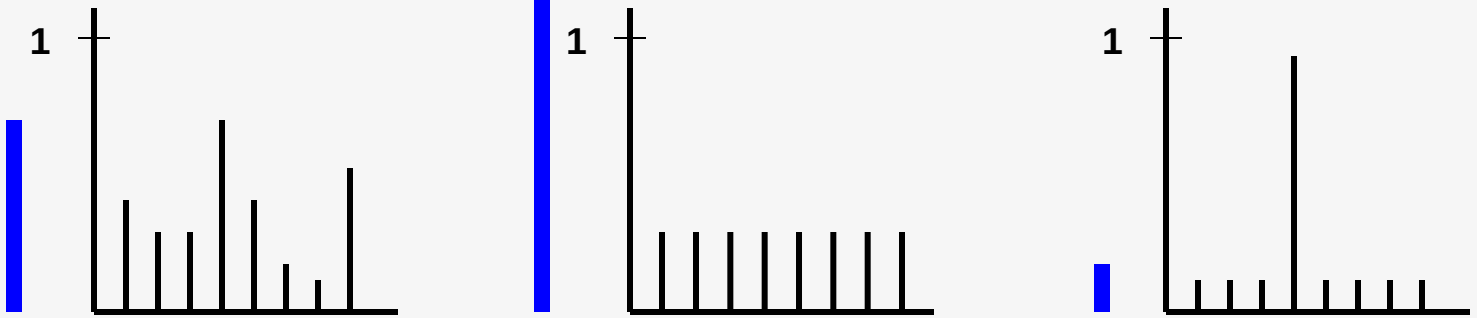
- Entropy (impurity, disorder) of a set of examples, S, relative to a binary classification is:

$$Entropy(S) = -p_+ \log(p_+) - p_- \log(p_-)$$

where p_+ is the proportion of positive examples in S and

p_- is the proportion of negatives.

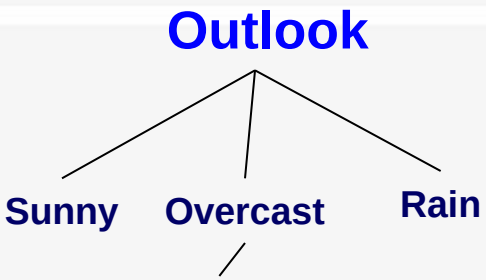
- If all the examples belong to the same category: *Entropy* = 0
- If the examples are equally mixed (0.5,0.5) *Entropy* = 1



Information Gain

- The information gain of an attribute *a* is the expected reduction in entropy caused by partitioning on this attribute.

$$Gain(S, a) = Entropy(S) - \sum_{v \in values(a)} \frac{|S_v|}{|S|} Entropy(S_v)$$

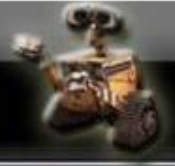


where S_v is the subset of S for which attribute *a* has value *v*

and the entropy of partitioning the data is calculated by weighing the entropy of each partition by its size relative to the original set

Partitions of low entropy lead to high gain

Go back to check which of the A, B splits is better



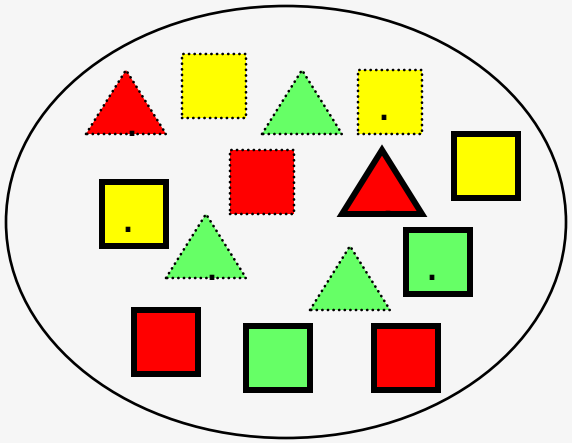
Triangles and Squares

#	Attribute			Shape
	Color	Outline	Dot	
1	green	dashed	no	triange
2	green	dashed	yes	triange
3	yellow	dashed	no	square
4	red	dashed	no	square
5	red	solid	no	square
6	red	solid	yes	triange
7	green	solid	no	square
8	green	dashed	no	triange
9	yellow	solid	yes	square
10	red	solid	no	square
11	green	solid	yes	square
12	yellow	dashed	yes	square
13	yellow	solid	no	square
14	red	dashed	yes	triange

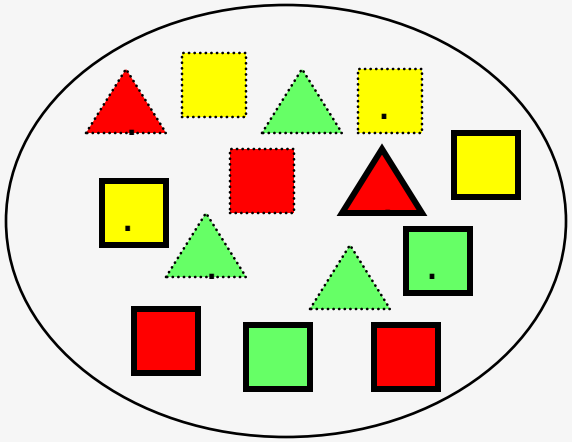
Triangles and Squares

#	Attribute			Shape
	Color	Outline	Dot	
1	green	dashed	no	triange
2	green	dashed	yes	triange
3	yellow	dashed	no	square
4	red	dashed	no	square
5	red	solid	no	square
6	red	solid	yes	triange
7	green	solid	no	square
8	green	dashed	no	triange
9	yellow	solid	yes	square
10	red	solid	no	square
11	green	solid	yes	square
12	yellow	dashed	yes	square
13	yellow	solid	no	square
14	red	dashed	yes	triange

Data Set:
A set of classified objects



Entropy



- 5 triangles
- 9 squares
- class probabilities

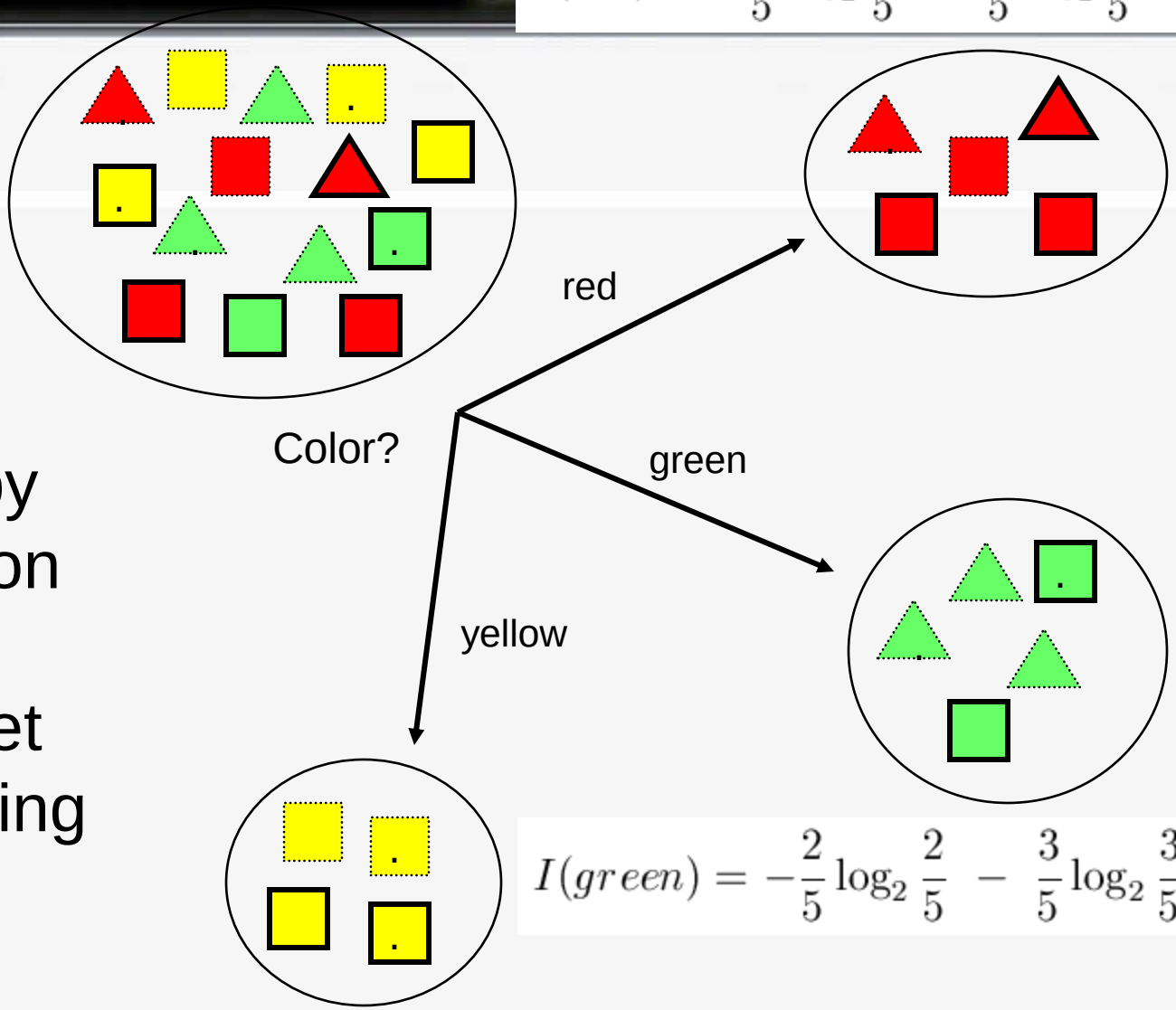
$$p(\square) = \frac{9}{14}$$

$$p(\triangle) = \frac{5}{14}$$

- entropy

$$I = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940 \text{ bits}$$

$$I(red) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.971 \text{ bits}$$



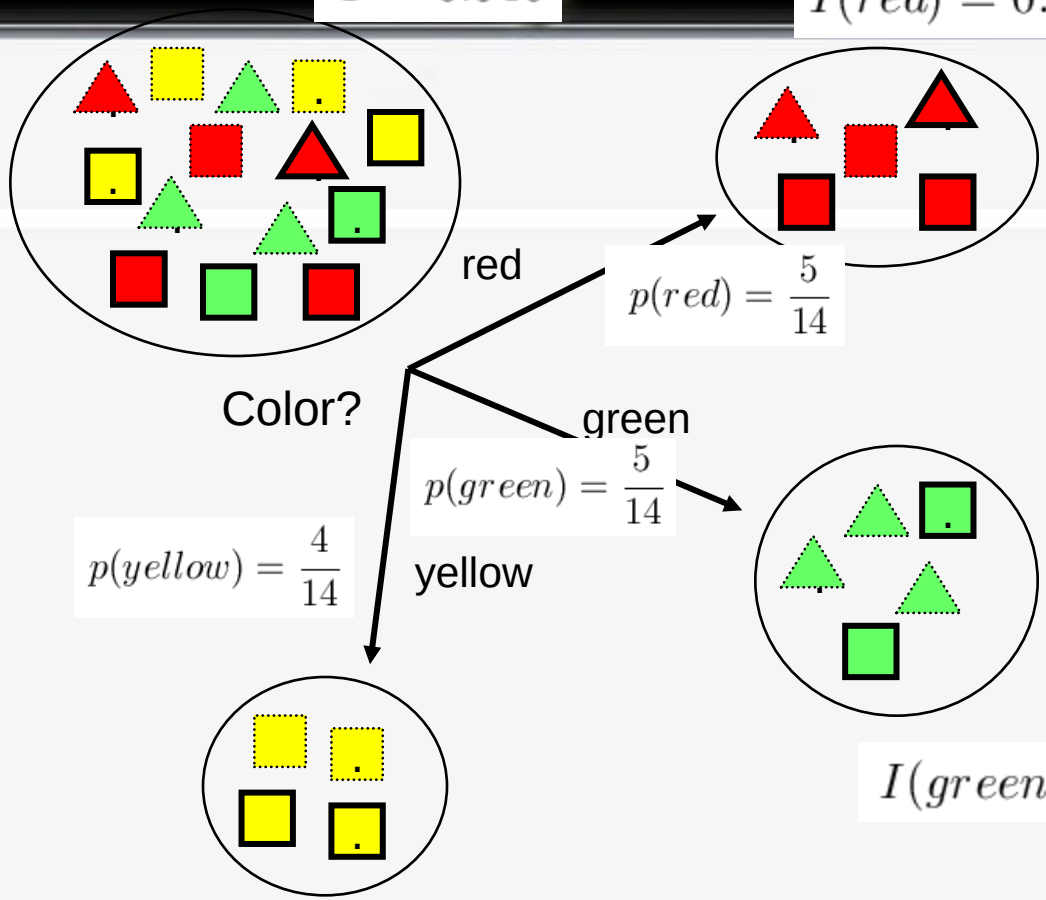
Entropy reduction by data set partitioning

$$I(green) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971 \text{ bits}$$

$$I(yellow) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0.0 \text{ bits}$$

$I = 0.940$

$I(red) = 0.971$ bits



$p(red) = \frac{5}{14}$

$p(green) = \frac{5}{14}$

$p(yellow) = \frac{4}{14}$

$I(green) = 0.971$ bits

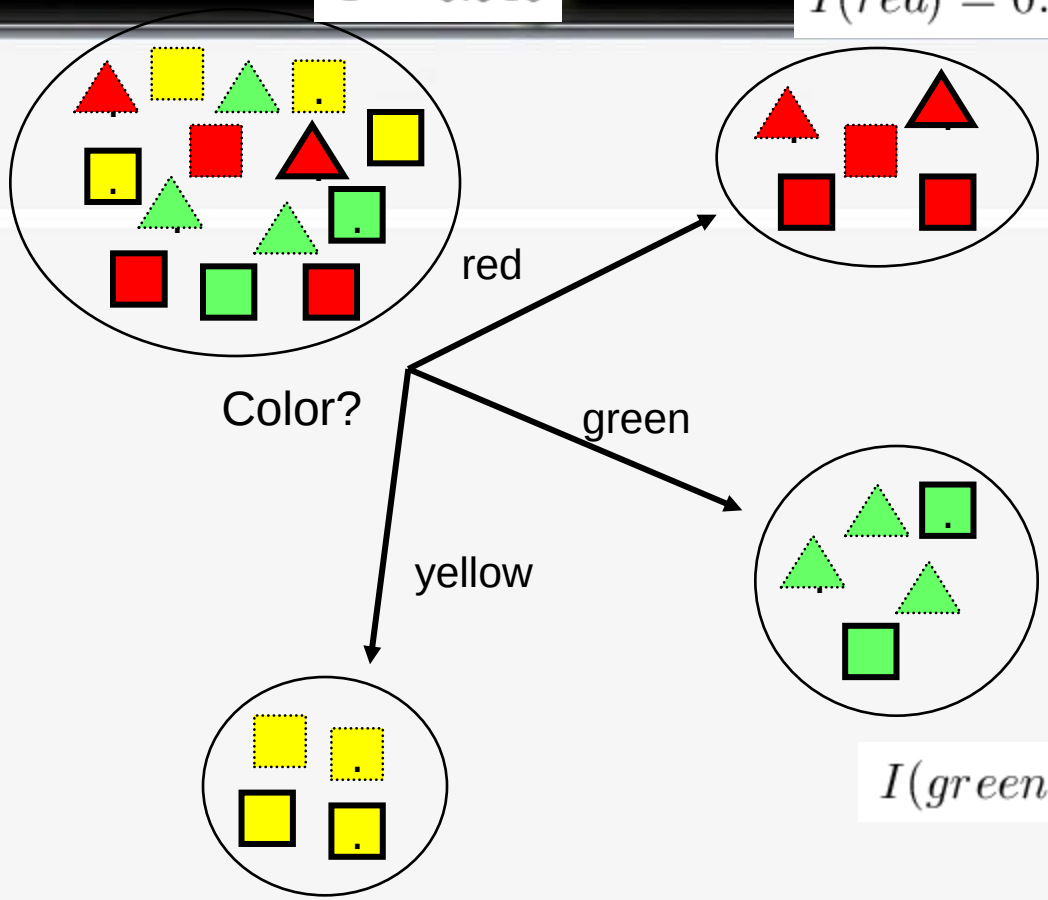
$I(yellow) = 0.0$ bits

$$I_{res}(\text{Color}) = \sum p(v)I(v) = \frac{5}{14}0.971 + \frac{5}{14}0.971 + \frac{4}{14}0.0 = 0.694 \; bits$$

Information Gain

$I = 0.940$

$I(red) = 0.971$ bits



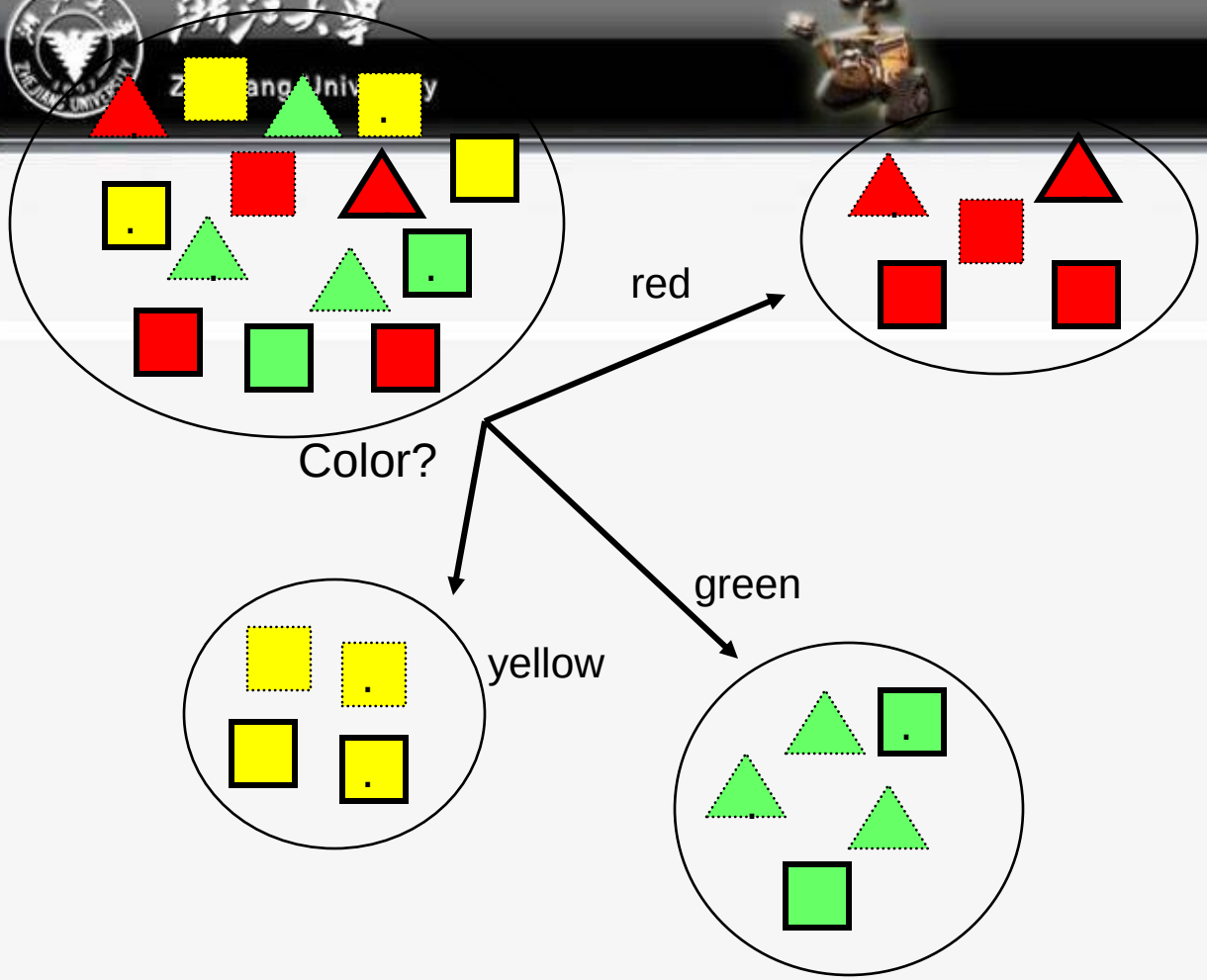
$I(green) = 0.971$ bits

$I(yellow) = 0.0$ bits

$$Gain(\text{Color}) = I - I_{res}(\text{Color}) = 0.940 - 0.694 = 0.246\ bits$$

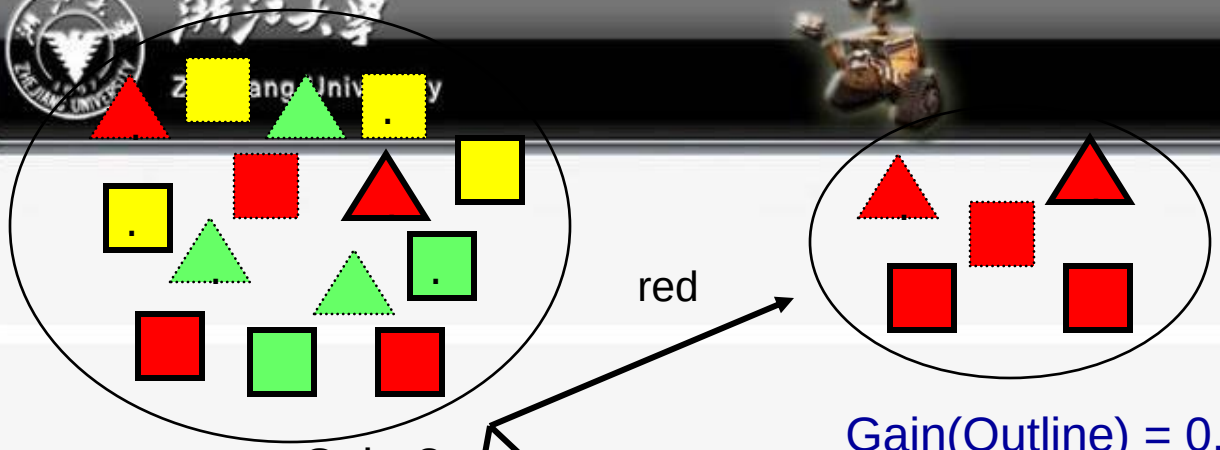
Information Gain of The Attribute

- Attributes
  - Gain(Color) = 0.246
  - Gain(Outline) = 0.151
  - Gain(Dot) = 0.048
- Heuristics: attribute with the highest gain is chosen
- This heuristics is local (local minimization of impurity)



Gain(Outline) = 0.971 – 0 = 0.971 bits

Gain(Dot) = 0.971 – 0.951 = 0.020 bits



red

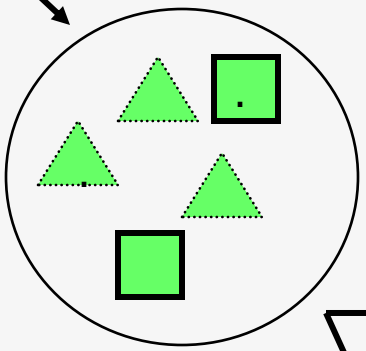
Color?

Gain(Outline) = 0.971 – 0.951 = 0.020 bits

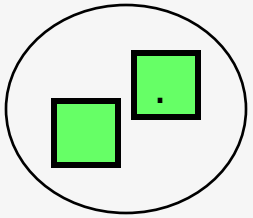
Gain(Dot) = 0.971 – 0 = 0.971 bits

green

yellow

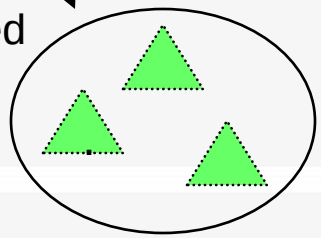


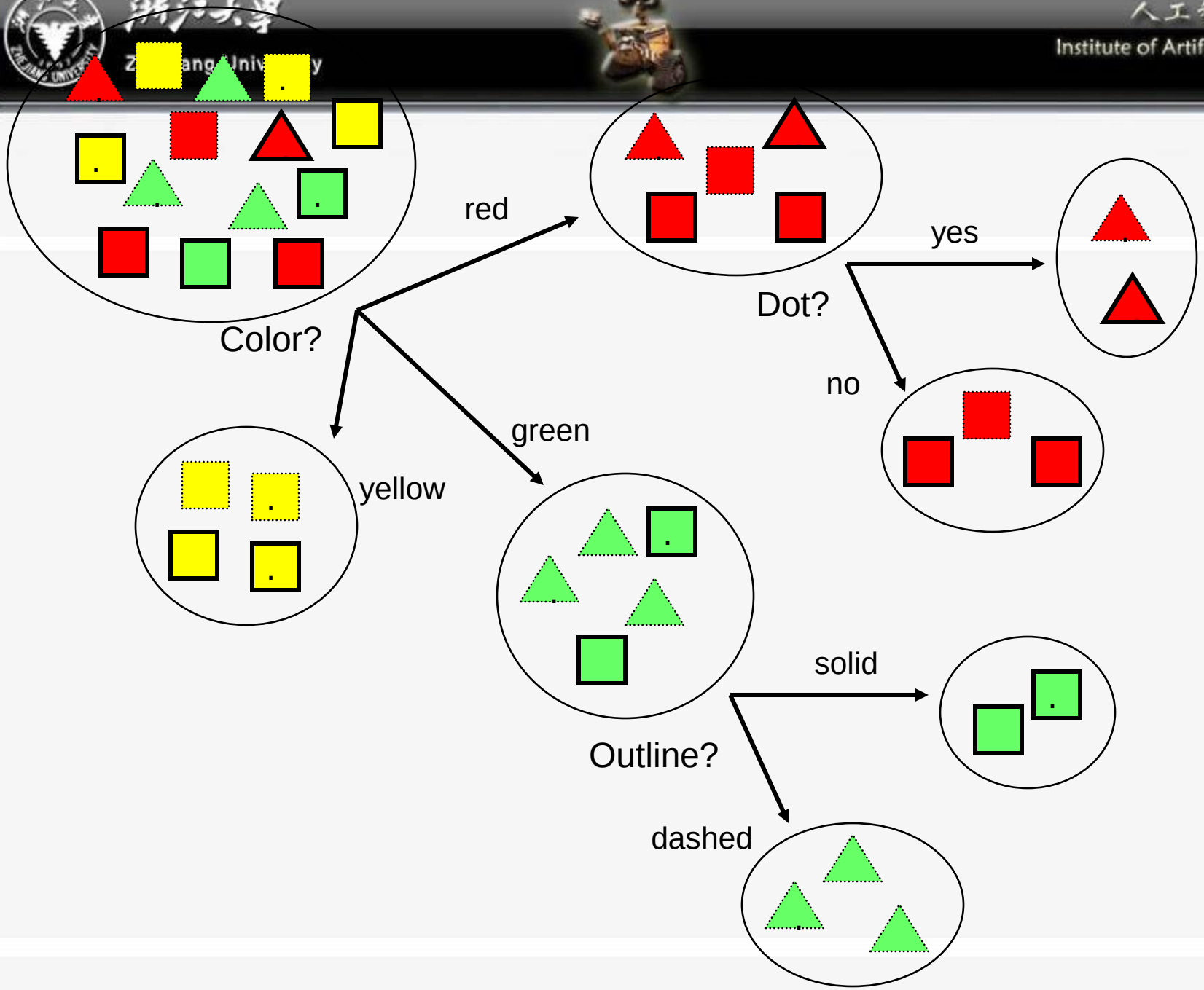
solid



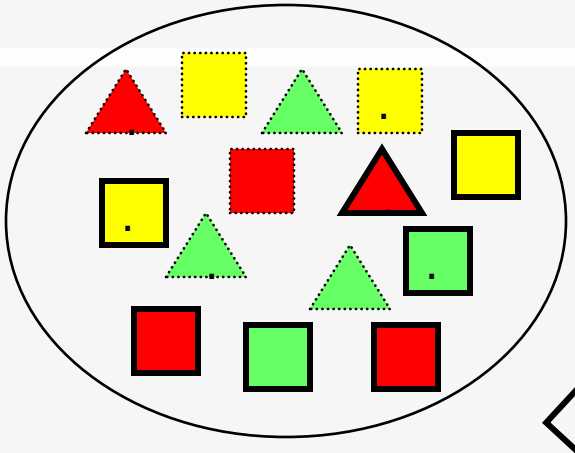
Outline?

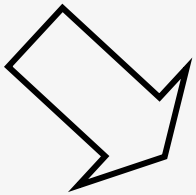
dashed

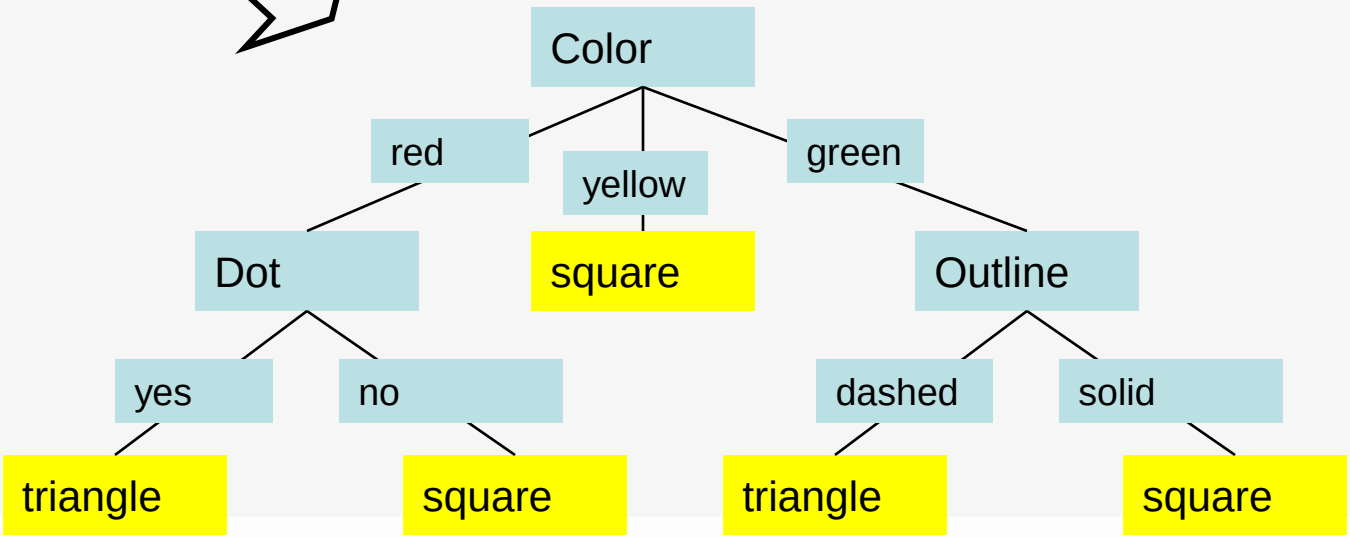



Color?
red
yellow
green
Dot?
yes
no
Outline?
solid
dashed

Decision Tree







Thanks!