Induction of Decision Trees

Blaž Zupan, Ivan Bratko

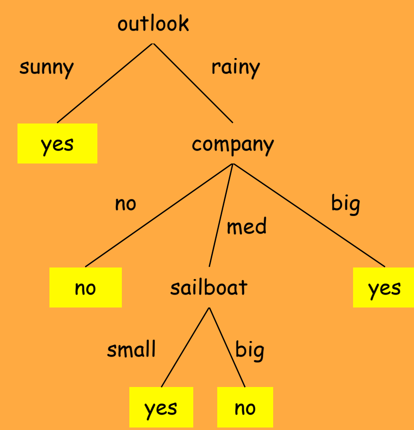
magix.fri.uni-lj.si/predavanja/uisp

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

Classification

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



Induction of Decision Trees

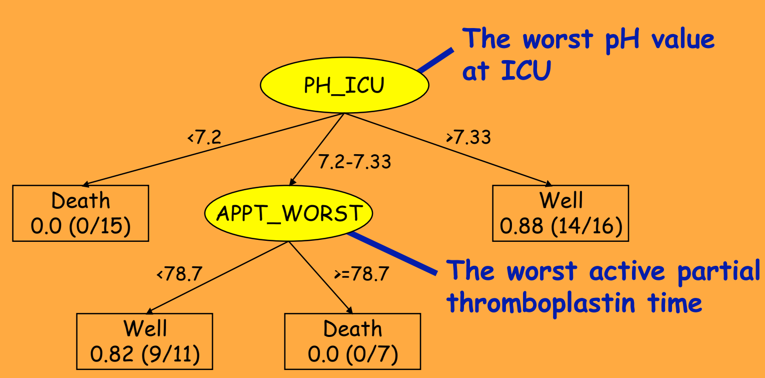
- Data Set (Learning Set)
  - Each example = Attributes + Class
- Induced description = Decision tree

- TDIDT
  - Top Down Induction of Decision Trees
- Recursive Partitioning

Some TDIDT Systems

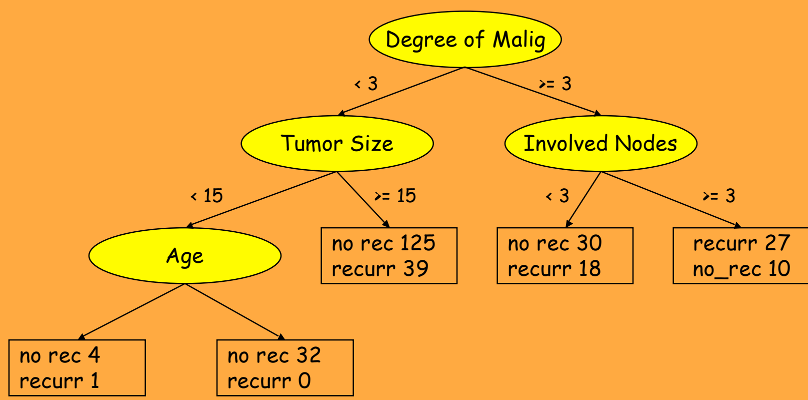
- ID3 (Quinlan 79)
- CART (Brieman et al. 84)
- Assistant (Cestnik et al. 87)
- C4.5 (Quinlan 93)
- See5 (Quinlan 97)
- ...
- Orange (Demšar, Zupan 98-03)

Analysis of Severe Trauma Patients Data



PH_ICU and APPT_WORST are exactly the two factors (theoretically) advocated to be the most important ones in the study by Rotondo et al., 1997.

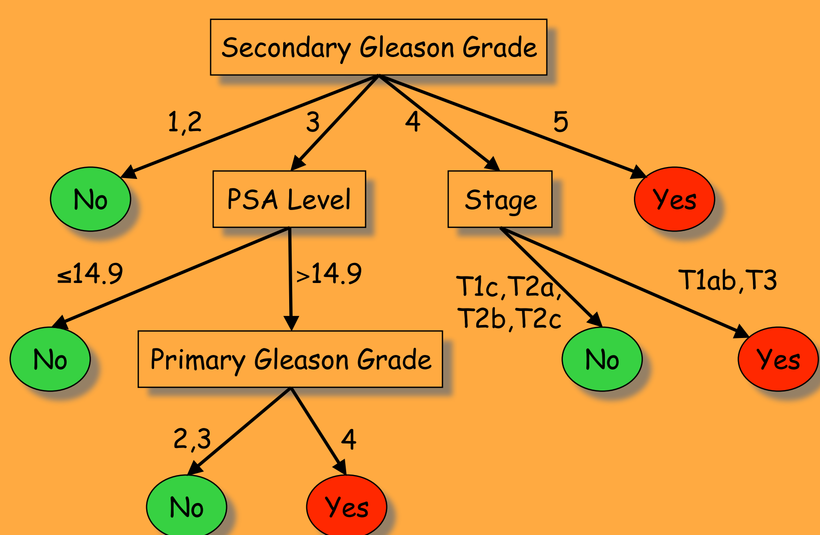
Breast Cancer Recurrence



Tree induced by Assistant Professional

Interesting: Accuracy of this tree compared to medical specialists

Prostate cancer recurrence

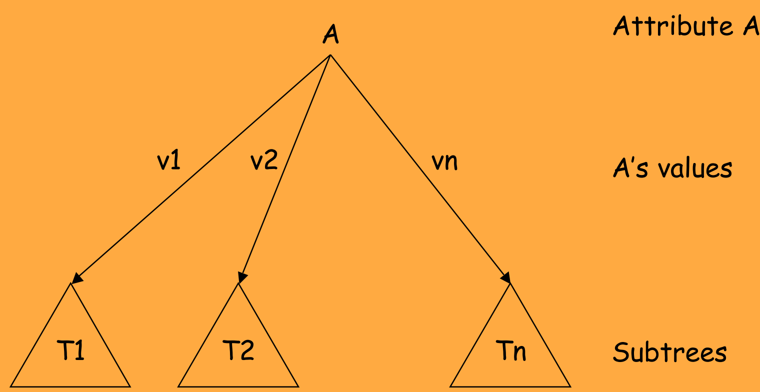


TDIDT Algorithm

- Also known as ID3 (Quinlan)
- To construct decision tree T from learning set S:
  - **If** all examples in S belong to some class C **Then** make leaf labeled C
  - **Otherwise**
    - select the "most informative" attribute A
    - partition S according to A's values
    - recursively construct subtrees T1, T2, ..., for the subsets of S

TDIDT Algorithm

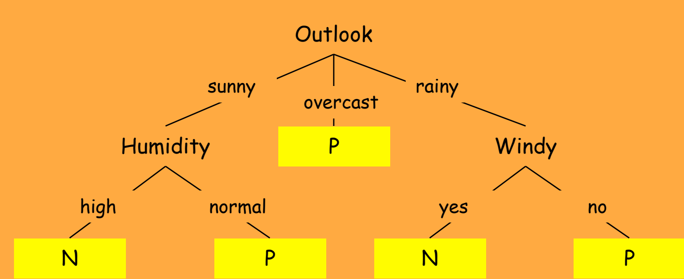
- Resulting tree T is:



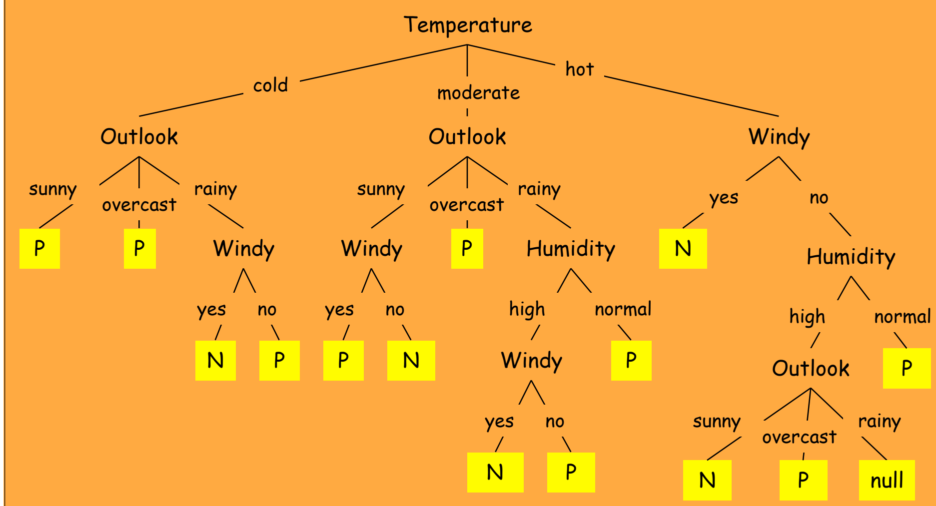
Another Example

#	Attribute				Class
	Outlook	Temperature	Humidity	Windy	Play
1	sunny	hot	high	no	N
2	sunny	hot	high	yes	N
3	overcast	hot	high	no	P
4	rainy	moderate	high	no	P
5	rainy	cold	normal	no	P
6	rainy	cold	normal	yes	N
7	overcast	cold	normal	yes	P
8	sunny	moderate	high	no	N
9	sunny	cold	normal	no	P
10	rainy	moderate	normal	no	P
11	sunny	moderate	normal	yes	P
12	overcast	moderate	high	yes	P
13	overcast	hot	normal	no	P
14	rainy	moderate	high	yes	N

Simple Tree



Complicated Tree



Attribute Selection Criteria

- Main principle
  - Select attribute which partitions the learning set into subsets as "pure" as possible
- Various measures of purity
  - Information-theoretic
  - Gini index
  - X^2
  - ReliefF
  - ...
- Various improvements
  - probability estimates
  - normalization
  - binarization, subsetting

Information-Theoretic Approach

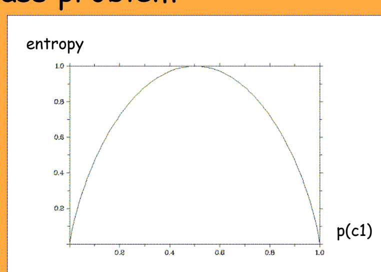
- To classify an object, a certain information is needed
  - I, information
- After we have learned the value of A, we only need some remaining amount of information to classify the object
  - Ires, residual information
- Gain
  - Gain(A) = I – Ires(A)
- The most 'informative' attribute is the one that minimizes Ires, *i.e.*, maximizes Gain

Entropy

- The average amount of information *I* needed to classify an object is given by the entropy measure

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

- For a two-class problem:



Residual Information

- After applying attribute *A*, *S* is partitioned into subsets according to values *v* of *A*
- *Ires* is equal to weighted sum of the amounts of information for the subsets

$$I_{res} = -\sum_v p(v) \sum_c p(c|v) \log_2 p(c|v)$$

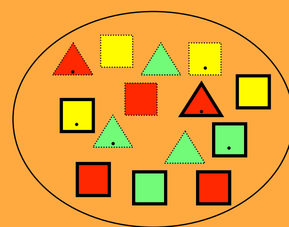
Triangles and Squares

#	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

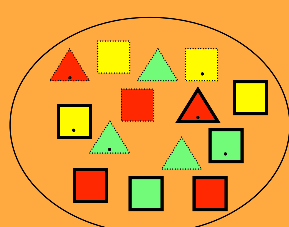
Triangles and Squares

#	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



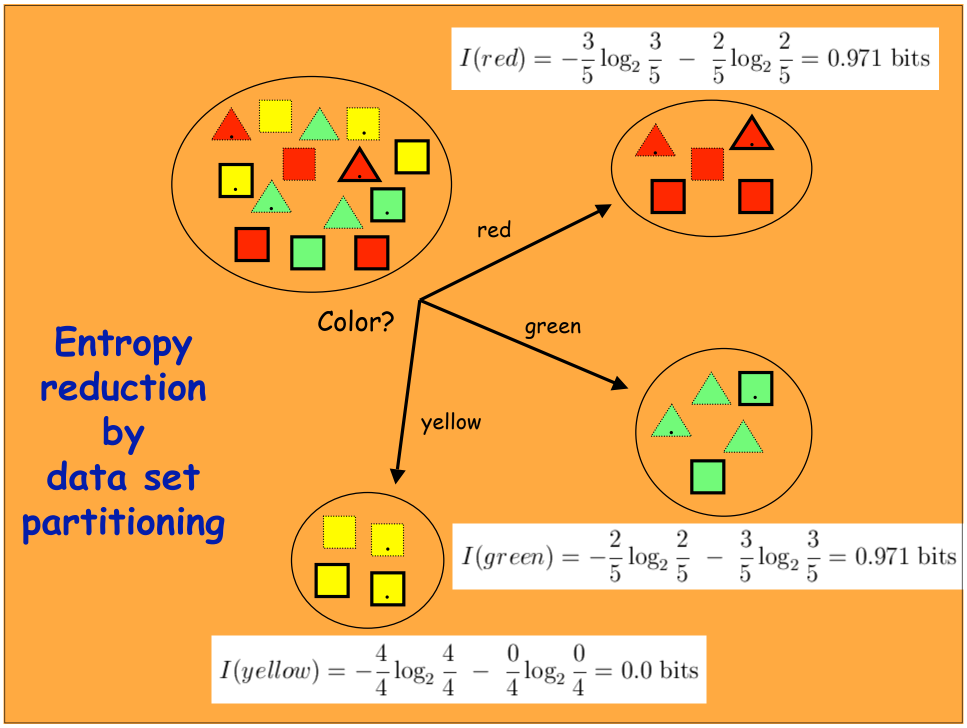
- 5 triangles
- 9 squares
- class probabilities

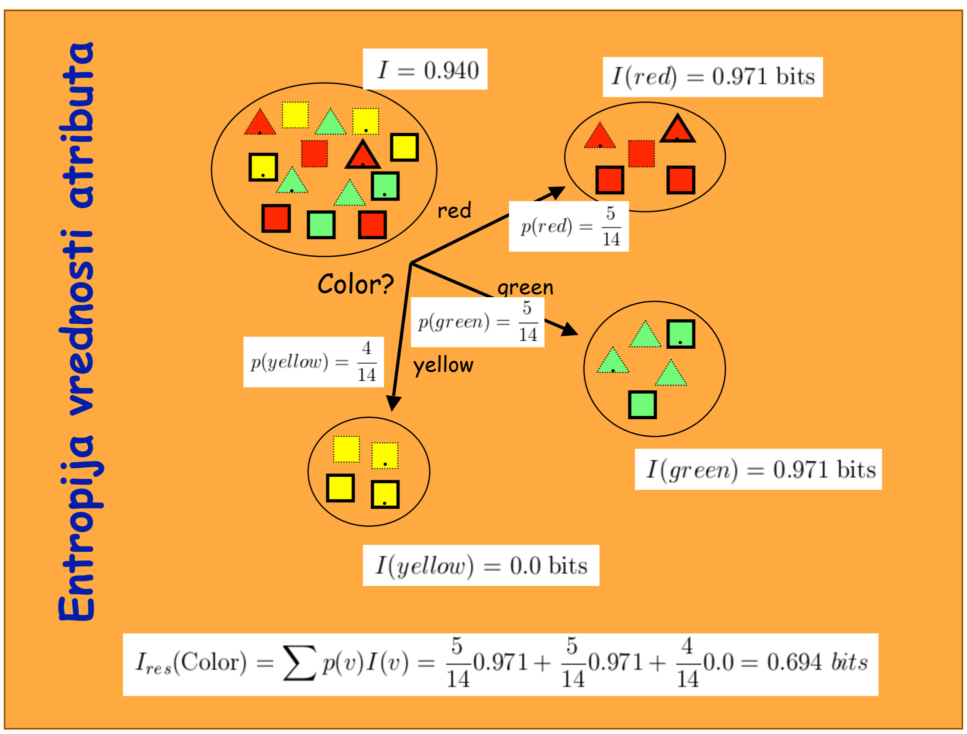
$$p(\square) = \frac{9}{14}$$

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

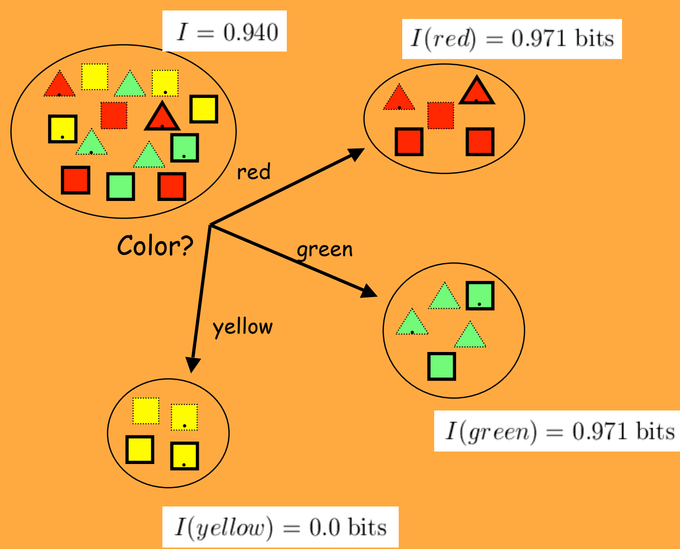
- entropy

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


$I(red) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.971 \text{ bits}$
red
Color?
green
Entropy reduction by data set partitioning
yellow
$I(green) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971 \text{ bits}$
$I(yellow) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0.0 \text{ bits}$


Entropija vrednosti atributa
$I = 0.940$
$I(red) = 0.971 \text{ bits}$
red
$p(red) = \frac{5}{14}$
Color?
green
$p(green) = \frac{5}{14}$
$p(yellow) = \frac{4}{14}$
yellow
$I(green) = 0.971 \text{ bits}$
$I(yellow) = 0.0 \text{ bits}$
$I_{res}(\text{Color}) = \sum p(v)I(v) = \frac{5}{14}0.971 + \frac{5}{14}0.971 + \frac{4}{14}0.0 = 0.694 \; bits$

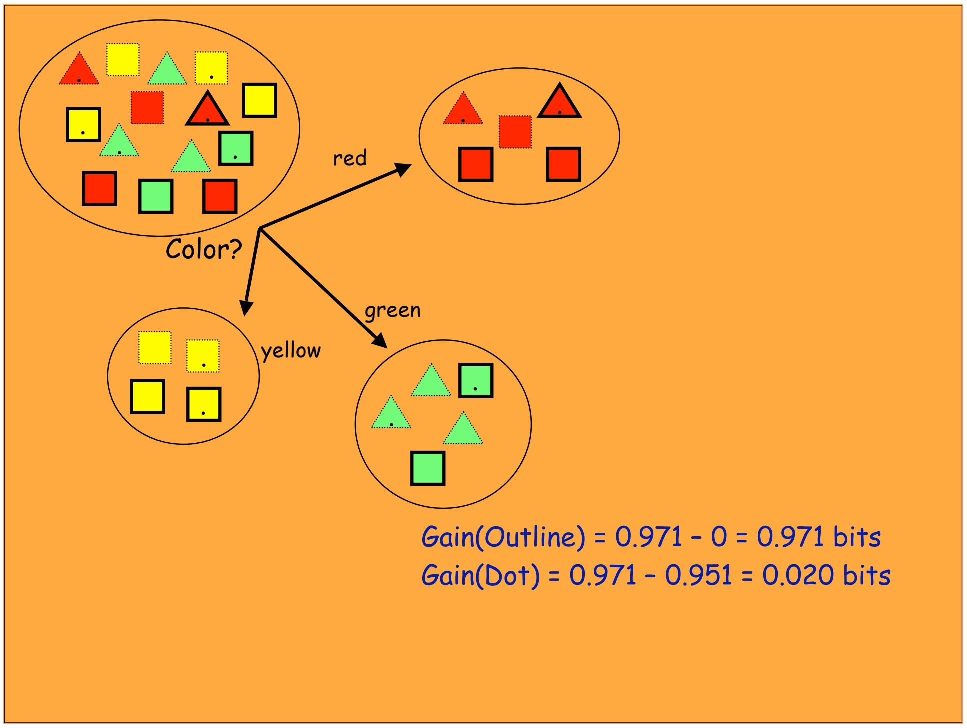
Information Gain

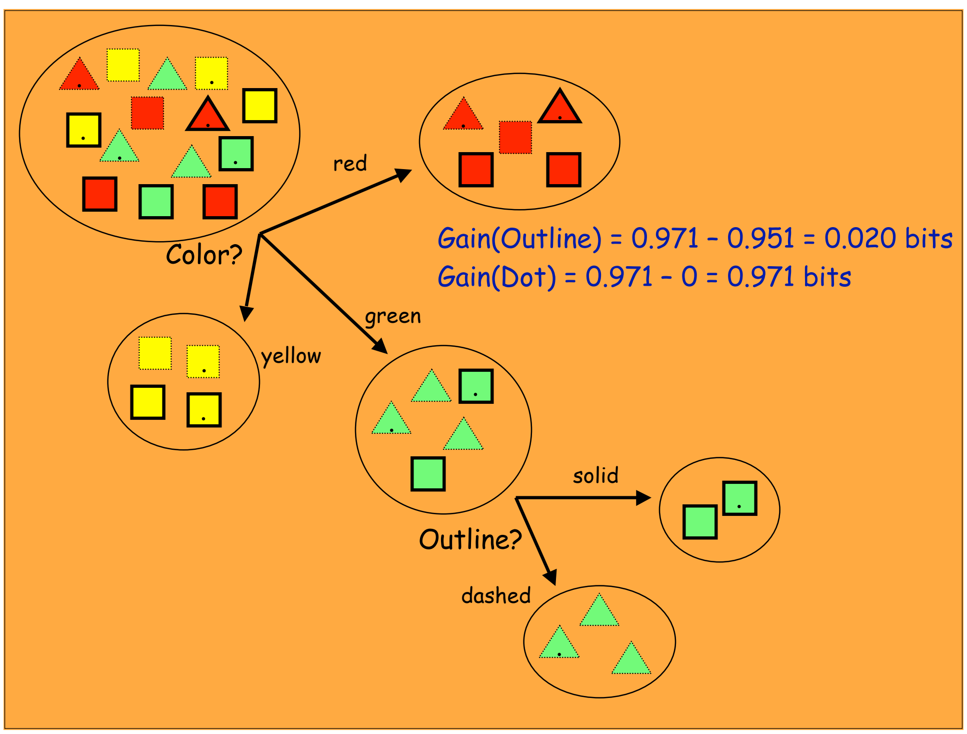


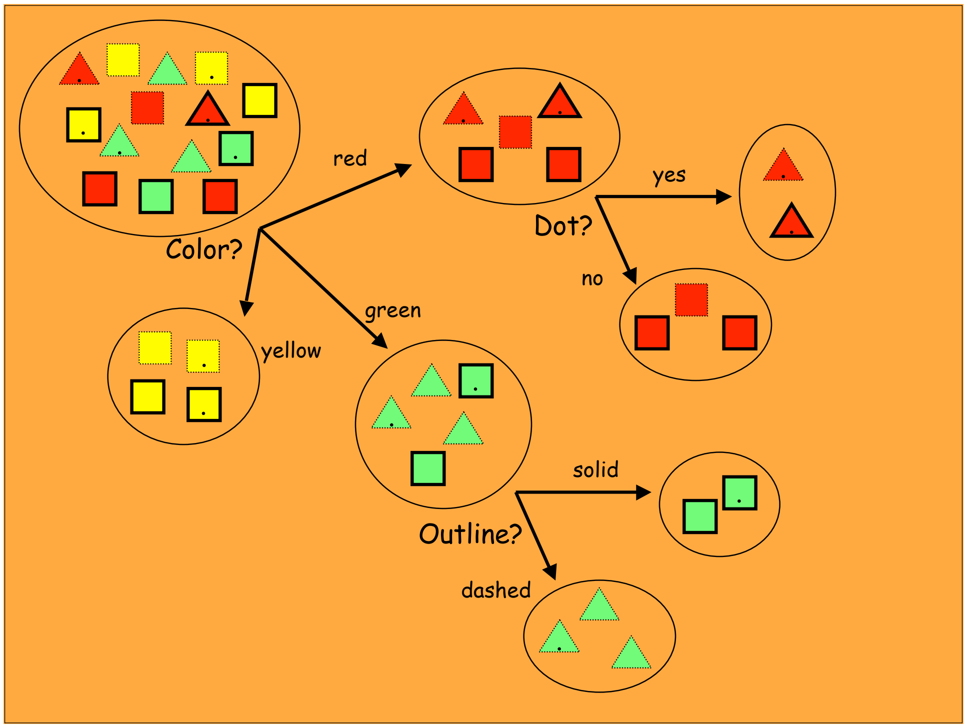
$$Gain(\text{Color}) = I - I_{res}(\text{Color}) = 0.940 - 0.694 = 0.246\ bits$$

Information Gain of The Attribute

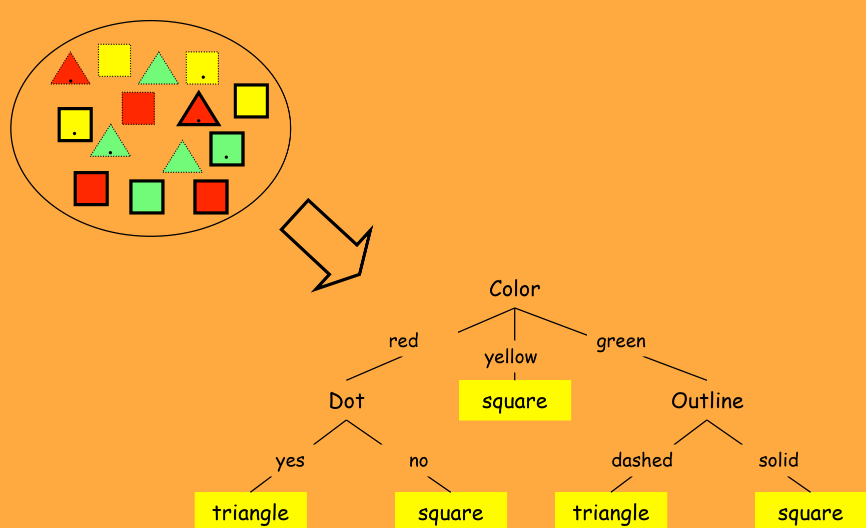
- Attributes
  - Gain(Color) = 0.246
  - Gain(Outline) = 0.151
  - Gain(Dot) = 0.048
- Heuristics: attribute with the highest gain is chosen
- This heuristics is local (local minimization of impurity)


red
Color?
green
yellow
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



Decision Tree



A Defect of *Ires*

- *Ires* favors attributes with many values
- Such attribute splits S to many subsets, and if these are small, they will tend to be pure anyway
- One way to rectify this is through a corrected measure of **information gain ratio**.

Information Gain Ratio

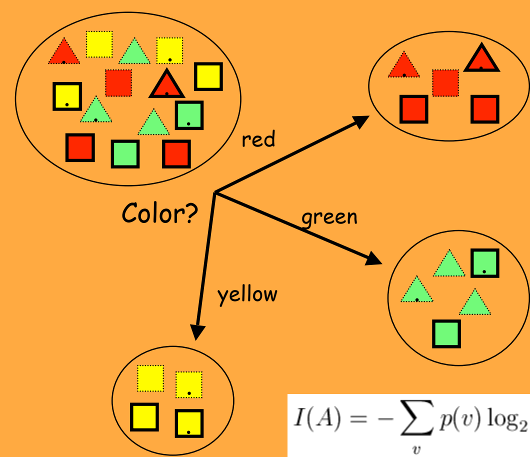
- I(A) is amount of information needed to determine the value of an attribute A

$$I(A) = -\sum_v p(v)\log_2(p(v))$$

- Information gain ratio

$$GainRatio(A) = \frac{Gain(A)}{I(A)} = \frac{I - I_{res}(A)}{I(A)}$$

Information Gain Ratio



$$I(A) = -\sum_v p(v)\log_2(p(v))$$

$$I(\text{Color}) = -\frac{5}{14}\log_2\frac{5}{14} - \frac{5}{14}\log_2\frac{5}{14} - \frac{4}{14}\log_2\frac{4}{14} = 1.58 \text{ bits}$$

$$\mathit{GainRatio}(\text{Color}) = \frac{\mathit{Gain}(\text{Color})}{I(\text{Color})} = \frac{0.940 - 0.694}{1.58} = 0.156$$

Information Gain and Information Gain Ratio

A	|v(A)|	Gain(A)	GainRatio(A)
Color	3	0.247	0.156
Outline	2	0.152	0.152
Dot	2	0.048	0.049

Gini Index

- Another sensible measure of impurity
  (i and j are classes)

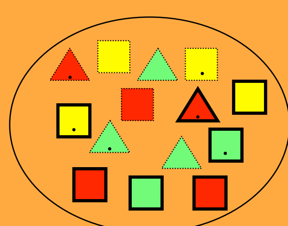
$$Gini = \sum_{i \neq j} p(i)p(j)$$

- After applying attribute A, the resulting Gini index is

$$Gini(A) = \sum_{v} p(v) \sum_{i \neq j} p(i|v)p(j|v)$$

- Gini can be interpreted as expected error rate

Gini Index



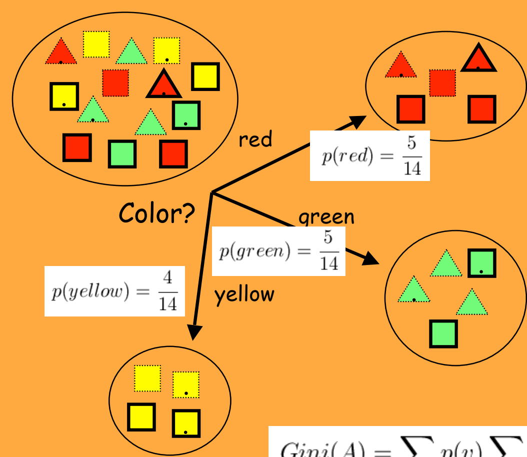
$$p(\square) = \frac{9}{14}$$

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

$$Gini = \sum_{i \neq j} p(i)p(j)$$

$$Gini = \frac{9}{14} \times \frac{5}{14} = 0.230$$

Gini Index for Color



$$Gini(A) = \sum_{v} p(v) \sum_{i \neq j} p(i|v)p(j|v)$$

$$Gini(\text{Color}) = \frac{5}{14} \times (\frac{3}{5} \times \frac{2}{5}) + \frac{5}{14} \times (\frac{2}{5} \times \frac{3}{5}) + \frac{4}{14} \times (\frac{4}{4} \times \frac{0}{4}) = 0.171$$

Gain of Gini Index

$$Gini = \frac{9}{14} \times \frac{5}{14} = 0.230$$

$$Gini(\text{Color}) = \frac{5}{14} \times (\frac{3}{5} \times \frac{2}{5}) + \frac{5}{14} \times (\frac{2}{5} \times \frac{3}{5}) + \frac{4}{14} \times (\frac{4}{4} \times \frac{0}{4}) = 0.171$$

$$\boxed{GiniGain(\text{Color}) = 0.230 - 0.171 = 0.058}$$

Three Impurity Measures

A	Gain(A)	GainRatio(A)	GiniGain(A)
Color	0.247	0.156	0.058
Outline	0.152	0.152	0.046
Dot	0.048	0.049	0.015

- These impurity measures assess the effect of a single attribute
- Criterion "most informative" that they define is local (and "myopic")
- It does not reliably predict the effect of several attributes applied jointly

Orange: Shapes Data Set

shape.tab

Color	Outline	Dot	Shape
d	d	d	d
			class
green	dashed	no	triange
green	dashed	yes	triange
yellow	dashed	no	square
red	dashed	no	square
red	solid	no	square
red	solid	yes	triange
green	solid	no	square
green	dashed	no	triange
yellow	solid	yes	square
red	solid	no	square
green	solid	yes	square
yellow	dashed	yes	square
yellow	solid	no	square

Orange: Impurity Measures

```
import orange
data = orange.ExampleTable('shape')

gain = orange.MeasureAttribute_info
gainRatio = orange.MeasureAttribute_gainRatio
gini = orange.MeasureAttribute_gini

print
print "%15s   %-8s %-8s %-8s" % ("name", "gain", "g ratio", "gini")
for attr in data.domain.attributes:
  print "%15s   %4.3f    %4.3f    %4.3f" % \
    (attr.name, gain(attr, data), gainRatio(attr, data), gini(attr, data))
```

```
           name   gain     g ratio  gini
          Color   0.247    0.156    0.058
        Outline   0.152    0.152    0.046
            Dot   0.048    0.049    0.015
```

Orange: orngTree

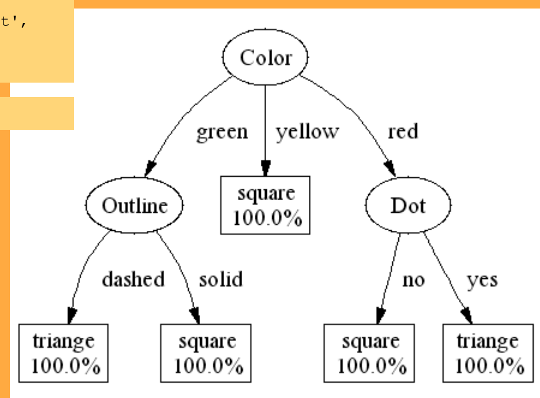
```
import orange, orngTree
data = orange.ExampleTable('shape')

tree = orngTree.TreeLearner(data)
orngTree.printTxt(tree)

print '\nWith contingency vector:'
orngTree.printTxt(tree,
  internalNodeFields=['contingency'],
  leafFields=['contingency'])
```

```
Color green:
|    Outline dashed: triange (100.0%)
|    Outline solid: square (100.0%)
Color yellow: square (100.0%)
Color red:
|    Dot no: square (100.0%)
|    Dot yes: triange (100.0%)

With contingency vector:

Color (<5, 9>) green:
|    Outline (<3, 2>) dashed: triange (<3, 0>)
|    Outline (<3, 2>) solid: square (<0, 2>)
Color (<5, 9>) yellow: square (<0, 4>)
Color (<5, 9>) red:
|    Dot (<2, 3>) no: square (<0, 3>)
|    Dot (<2, 3>) yes: triange (<2, 0>)
```

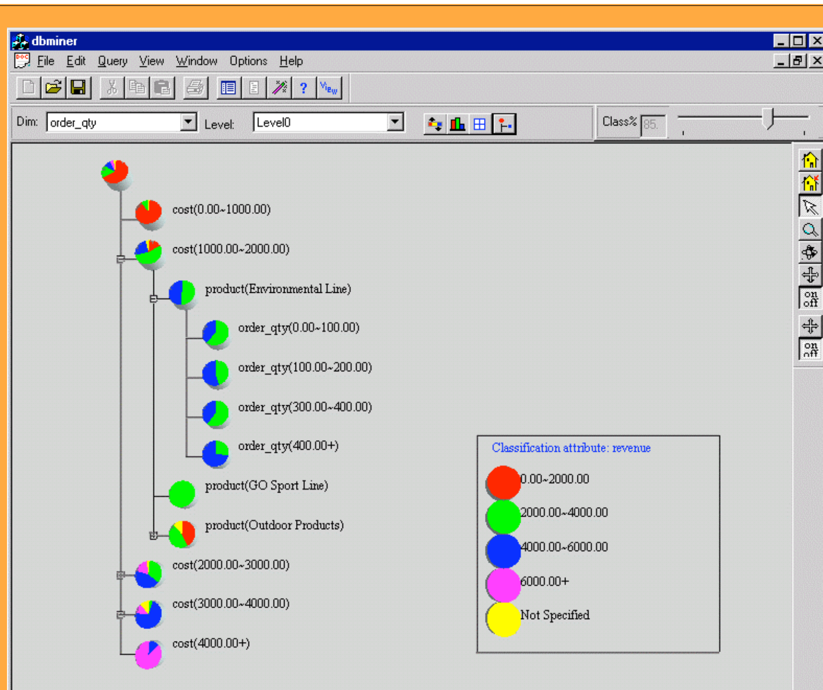
Orange: Saving to DOT

```
import orange, orngTree
data = orange.ExampleTable('shape')

tree = orngTree.TreeLearner(data)

orngTree.printDot(tree, 'shape.dot',
  leafShape='box',
  internalNodeShape='ellipse')
```

```
> dot -Tgif shape.dot > shape.gif
```



DB Miner: visualization



SGI MineSet: visualization