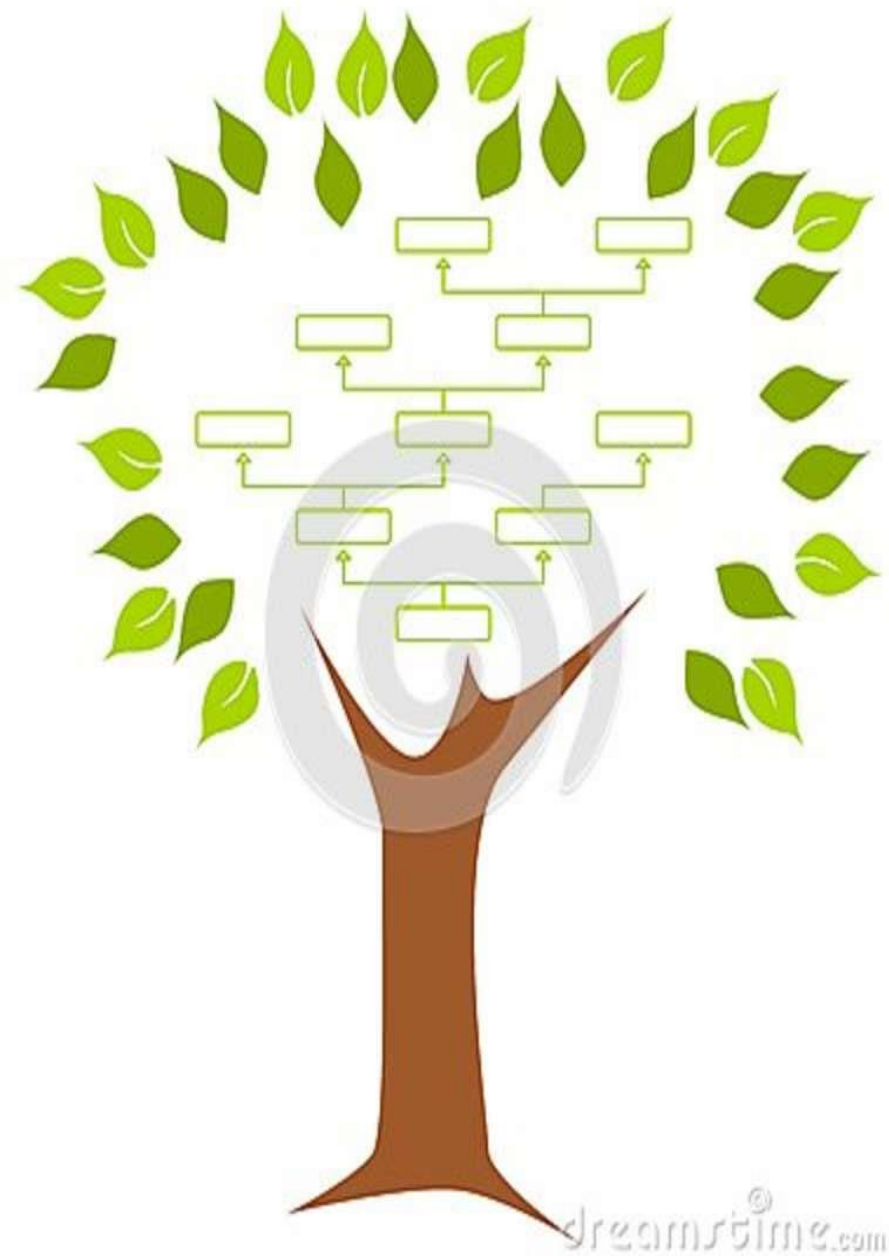
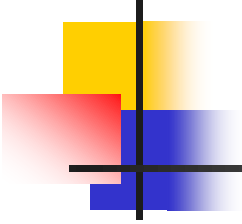


Chapter#4

Decision Trees



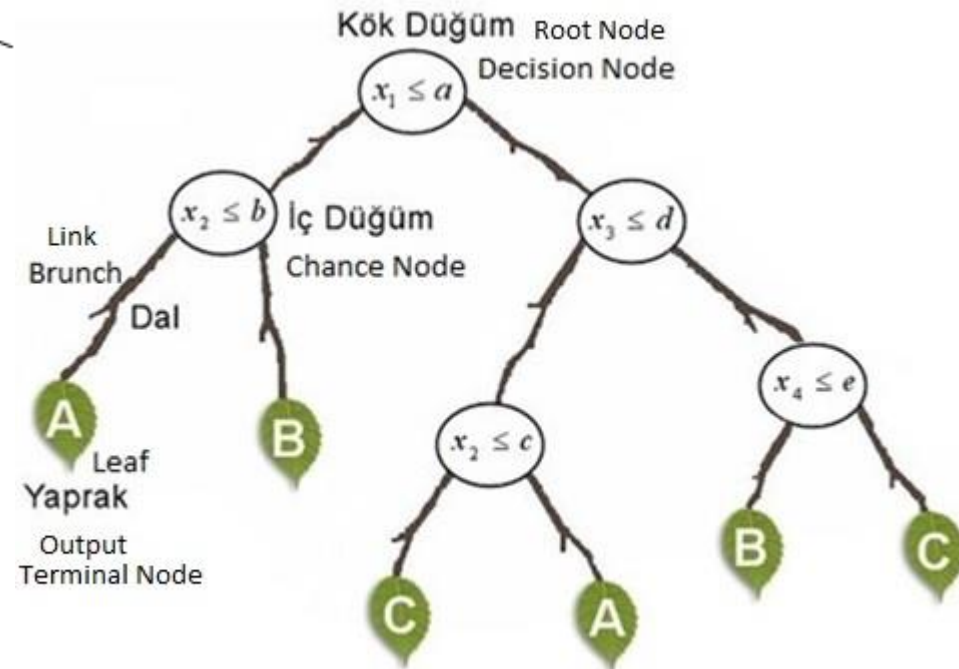
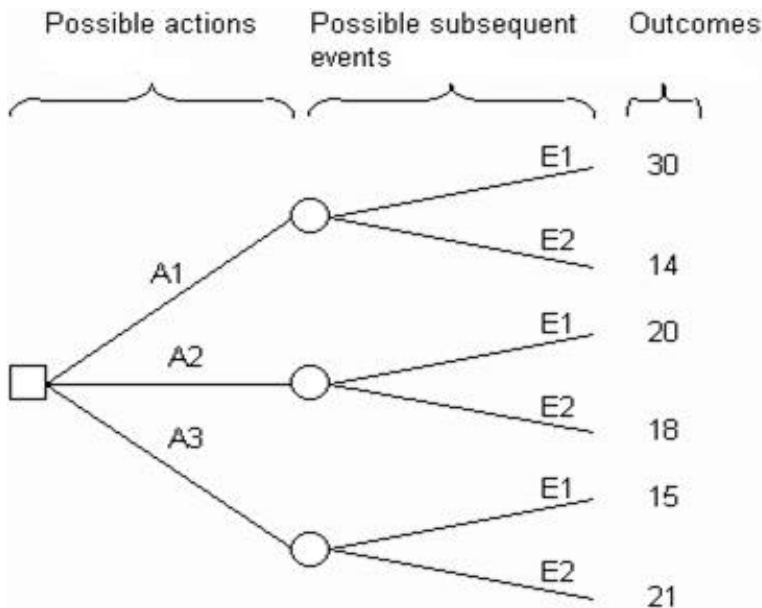
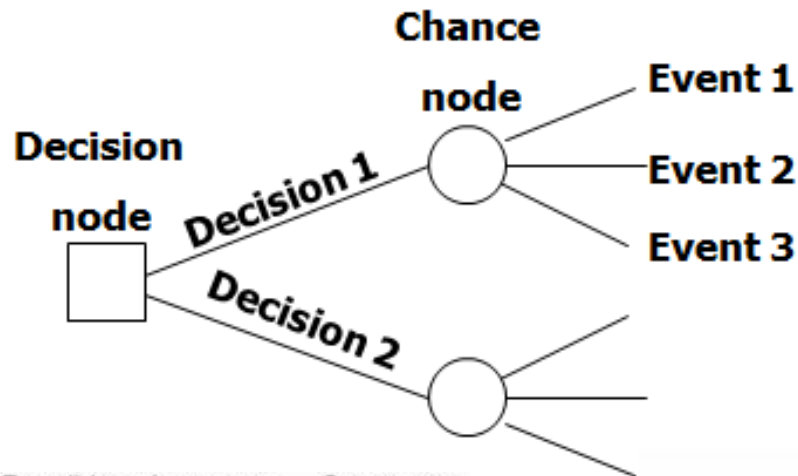
- 
-
- Also known as
 - – Hierarchical classifiers
 - – Tree classifiers
 - – Multistage classification
 - – Divide & conquer strategy



Decision Trees

- Classify a pattern through a sequence of questions (20-question game); next question asked depends on the answer to the current question
- This approach is particularly useful for non-metric data; questions can be asked in a “yes-no” or “true-false” style that do not require any notion of metric
- Sequence of questions is displayed in a directed decision tree
- Root node, links or branches, leaf or terminal nodes
- Classification of a pattern begins at the root node until we reach the leaf node; pattern is assigned the category of the leaf node
- Benefit of decision tree:
 - Interpretability: a tree can be expressed as a logical expression
 - Rapid classification: a sequence of simple queries
 - Higher accuracy & speed

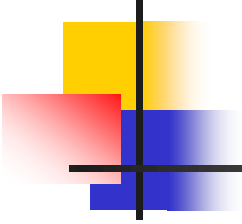
Decision Tree Structure





When to consider Decision Trees

- Instances describable by attribute-value pairs
 - e.g Humidity: *High, Normal*
- Target function is discrete valued
 - e.g Play tennis; *Yes, No*
- Disjunctive hypothesis may be required
 - e.g *Outlook=Sunny* \vee *Wind=Weak*
- Possibly noisy training data
- Missing attribute values
- Application Examples:
 - Medical diagnosis
 - Credit risk analysis
 - Object classification for robot manipulator (Tan 1993)

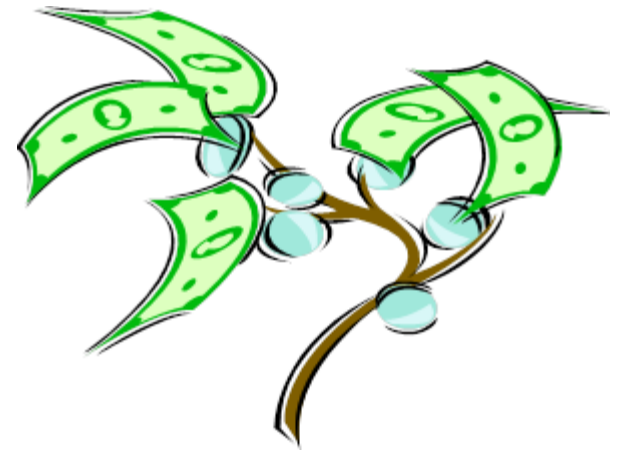
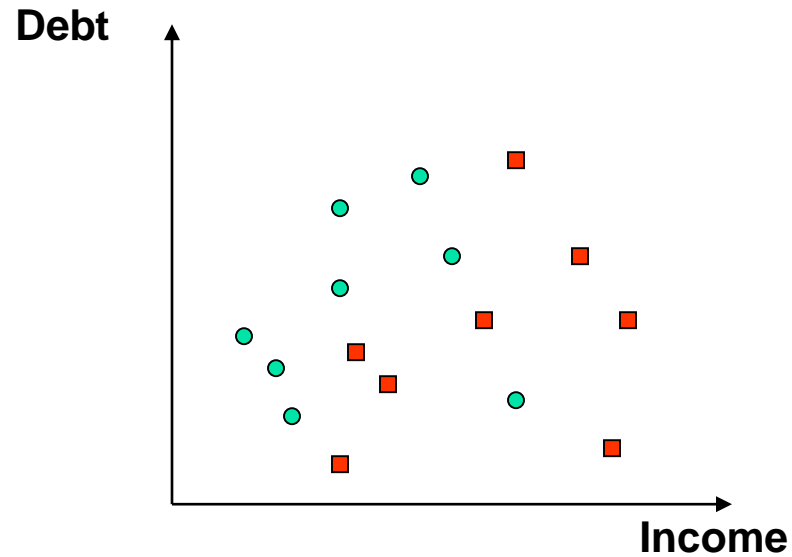
- 
-
- Decision tree representation
 - ID3 learning algorithm
 - Entropy & Information gain
 - Examples
 - Gini Index
 - Overfitting



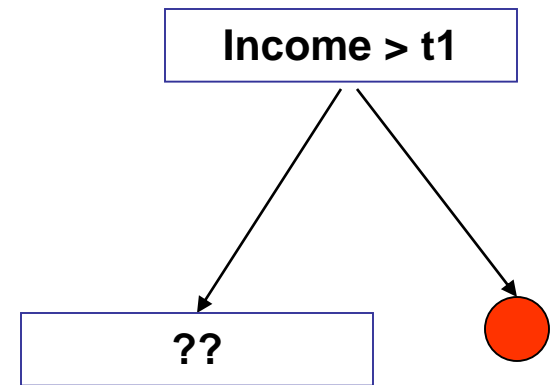
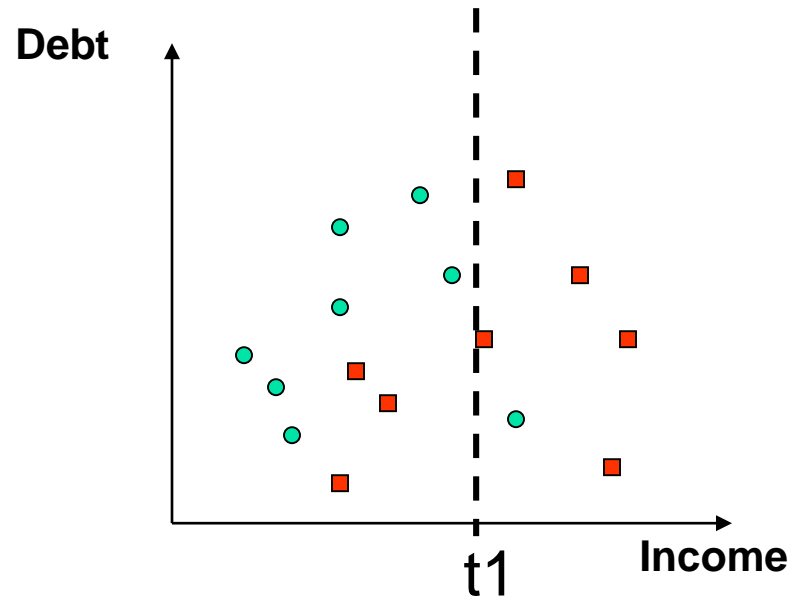
■ **Decision Trees**

- Decision tree is a simple but powerful learning paradigm. In this method a set of training examples is broken down into smaller and smaller subsets while at the same time an associated decision tree gets incrementally developed. At the end of the learning process, a decision tree covering the training set is returned.
- The decision tree can be thought of as a set of sentences (in Disjunctive Normal Form) written in propositional logic.
- Some characteristics of problems that are well suited to Decision Tree Learning are:
 - Attribute-value paired elements
 - Discrete target function
 - Disjunctive descriptions (of target function)
 - Works well with missing or erroneous training data

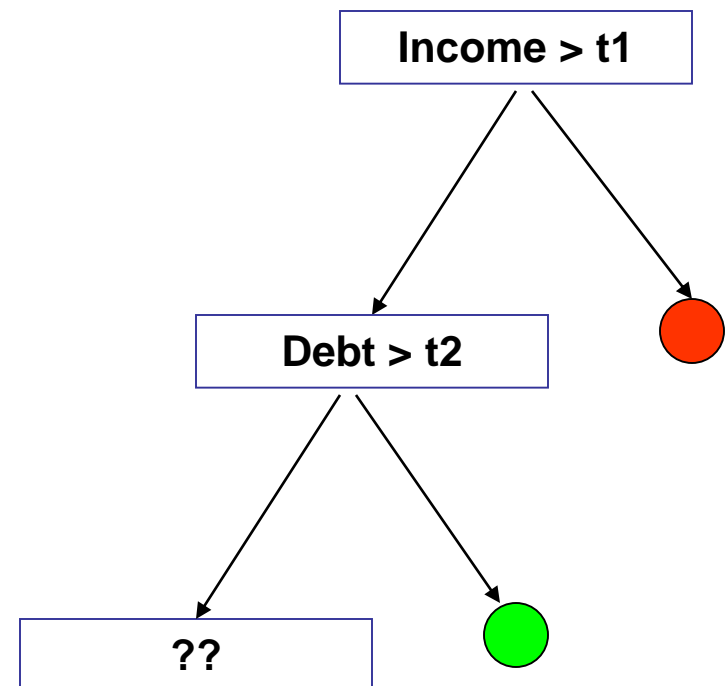
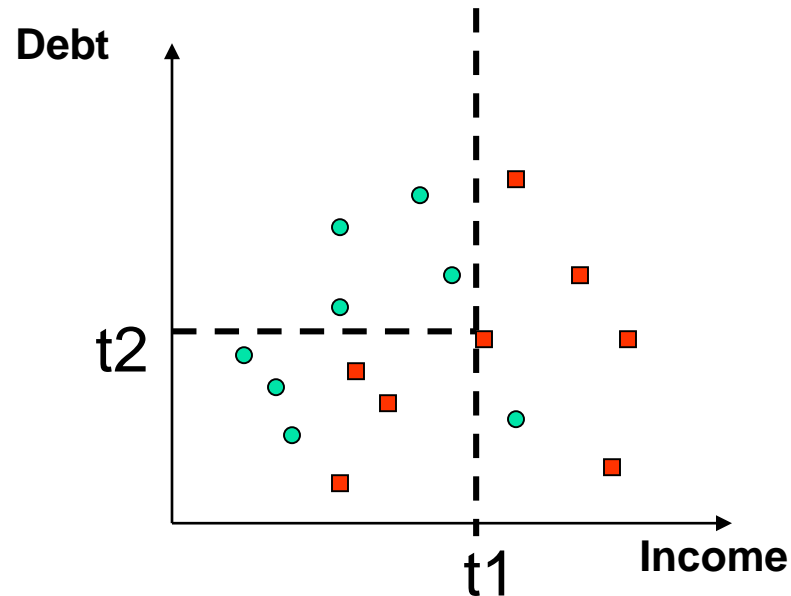
Decision Tree Example



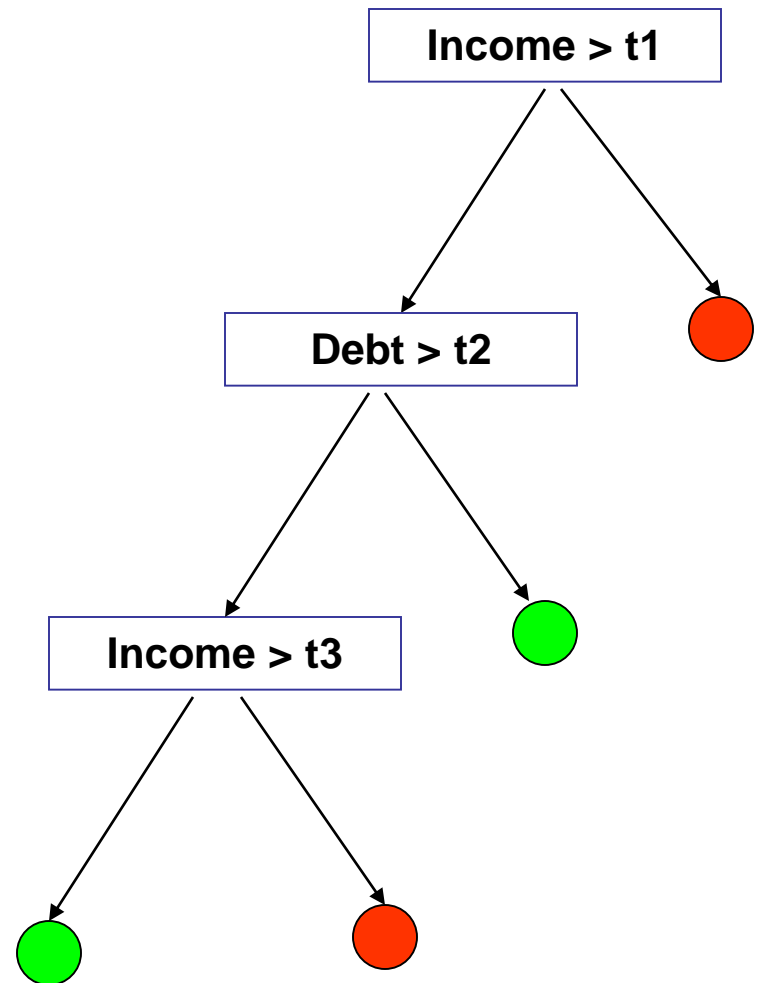
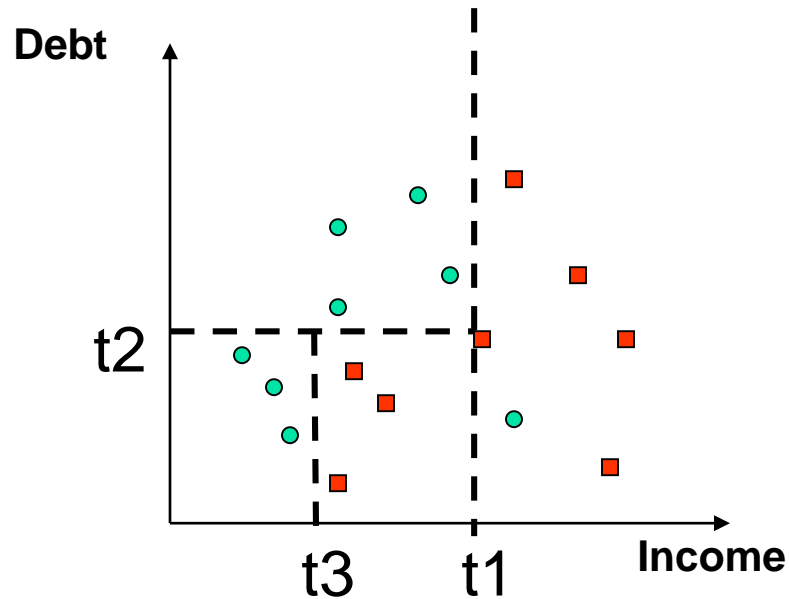
Decision Tree Example



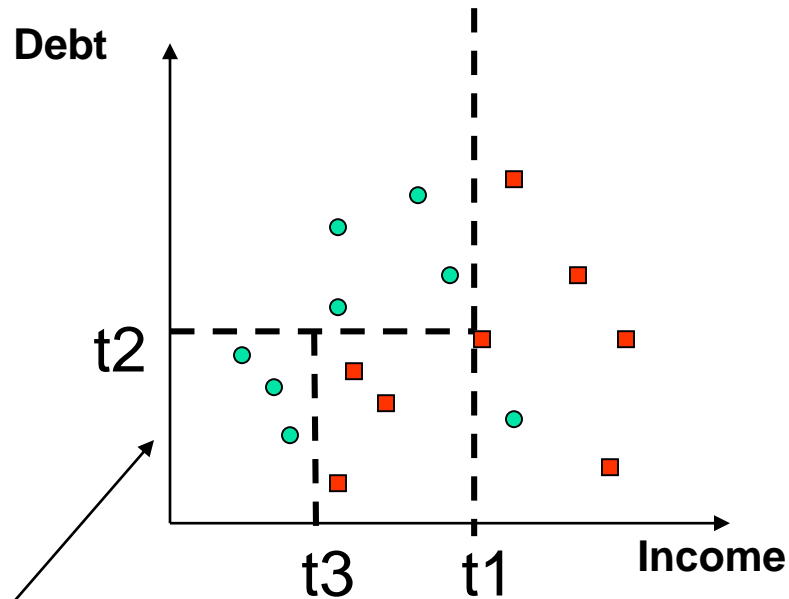
Decision Tree Example



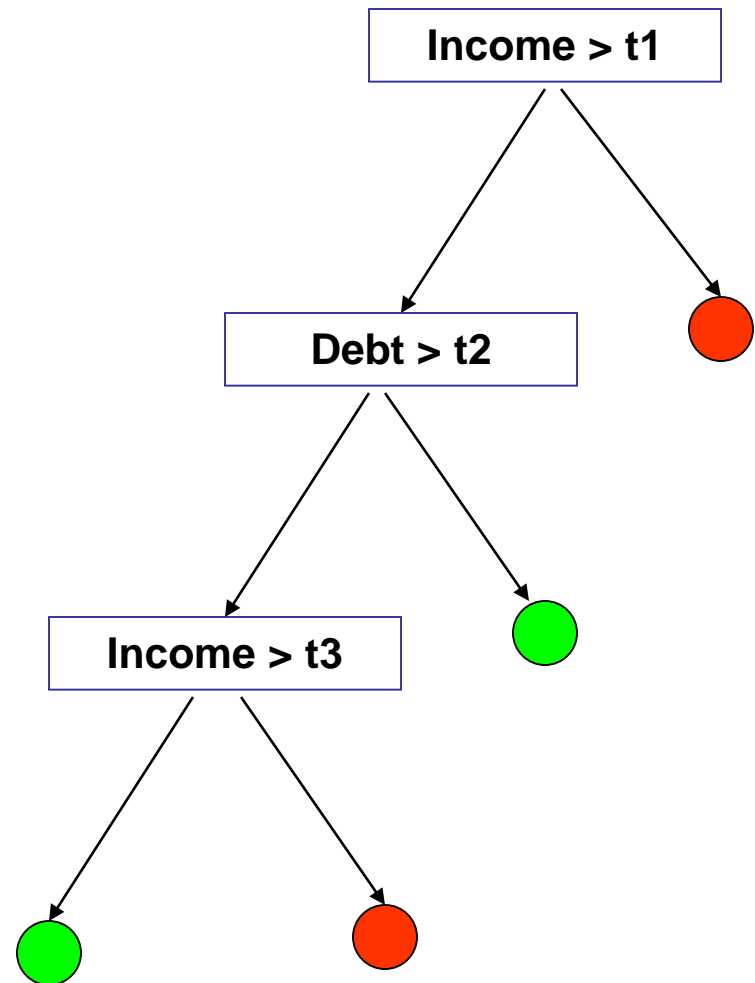
Decision Tree Example



Decision Tree Construction



Note: tree boundaries are piecewise linear and axis-parallel





ID3 Algorithm

- Is the algorithm to construct a decision tree
- Using Entropy to generate the information gain
- The best value then be selected



Top-Down Induction of Decision Trees ID3

1. $A \leftarrow$ the “best” decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A create new descendant
4. Sort training examples to leaf node according to the attribute value of the branch
5. If all training examples are perfectly classified (same value of target attribute) stop, else iterate over new leaf nodes.



Building a Decision Tree

1. First test all attributes and select the one that would function as the best root;
2. Break-up the training set into subsets based on the branches of the root node;
3. Test the remaining attributes to see which ones fit best underneath the branches of the root node;
4. Continue this process for all other branches until
 - a. all examples of a subset are of one type
 - b. there are no examples left (return majority classification of the parent)
 - c. there are no more attributes left (default value should be majority classification)



Decision Tree for Playing Tennis

Consider the following table

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



Decision Tree for Playing Tennis

- We want to build a decision tree for the tennis matches
- The schedule of matches depend on the weather (Outlook, Temperature, Humidity, and Wind)
- So to apply what we know to build a decision tree based on this table



Decision Tree for Playing Tennis

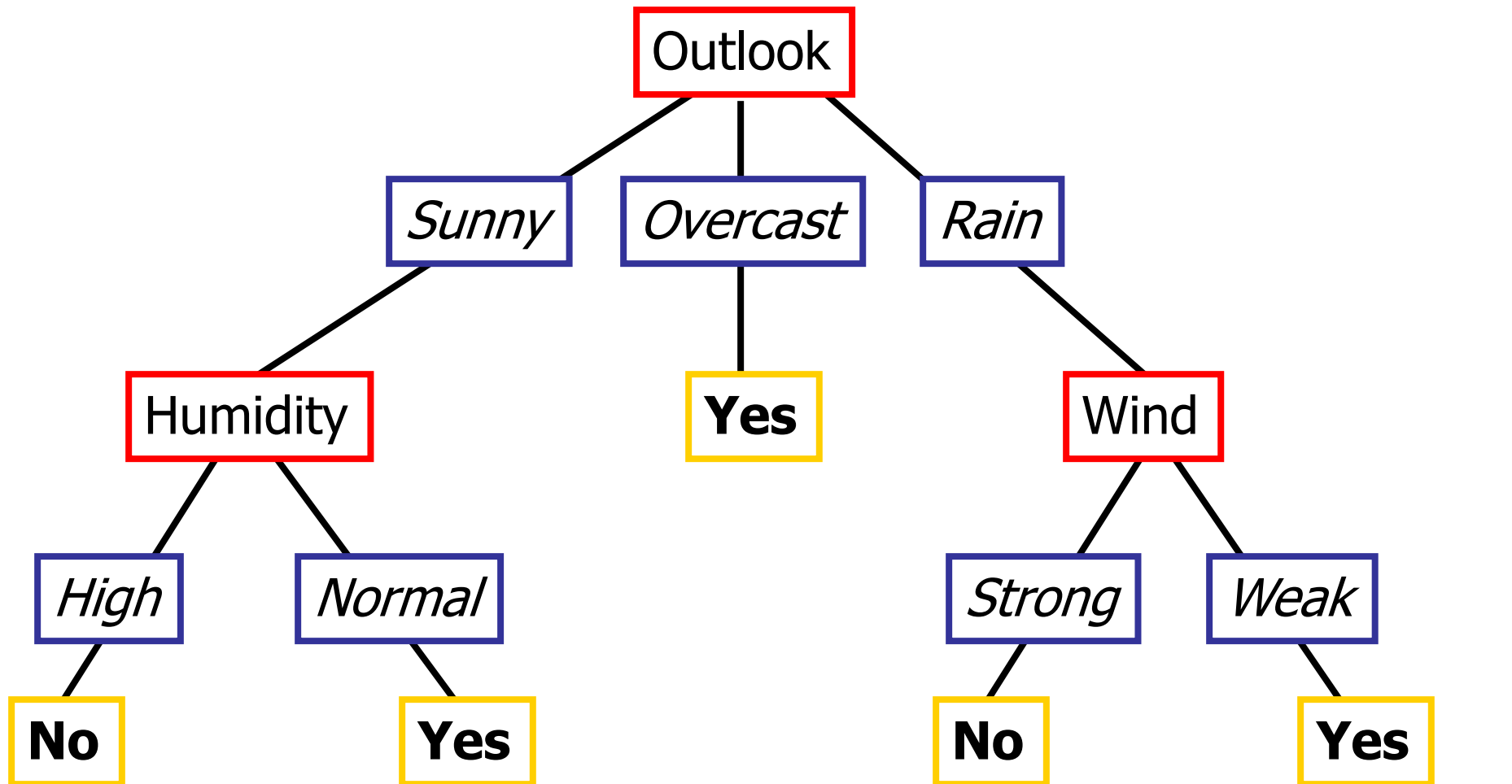
- Calculating the information gains for each of the weather attributes:
 - For the Wind
 - For the Humidity
 - For the Outlook
 - For the Temperature



Decision Tree for Playing Tennis

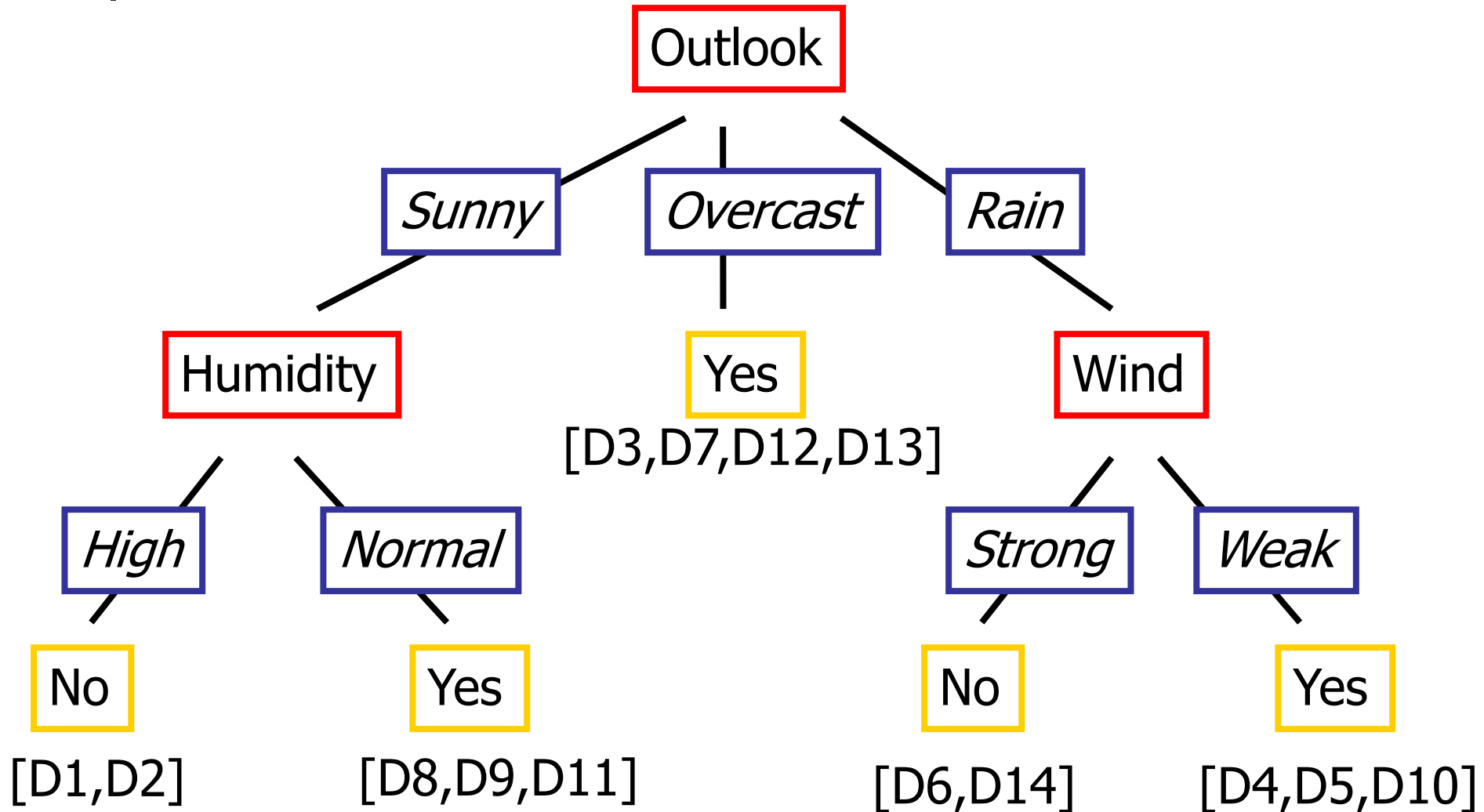
- Attributes and their values:
 - Outlook: *Sunny, Overcast, Rain*
 - Humidity: *High, Normal*
 - Wind: *Strong, Weak*
 - Temperature: *Hot, Mild, Cold*
- Target concept - Play Tennis: *Yes, No*

Decision Tree for Playing Tennis

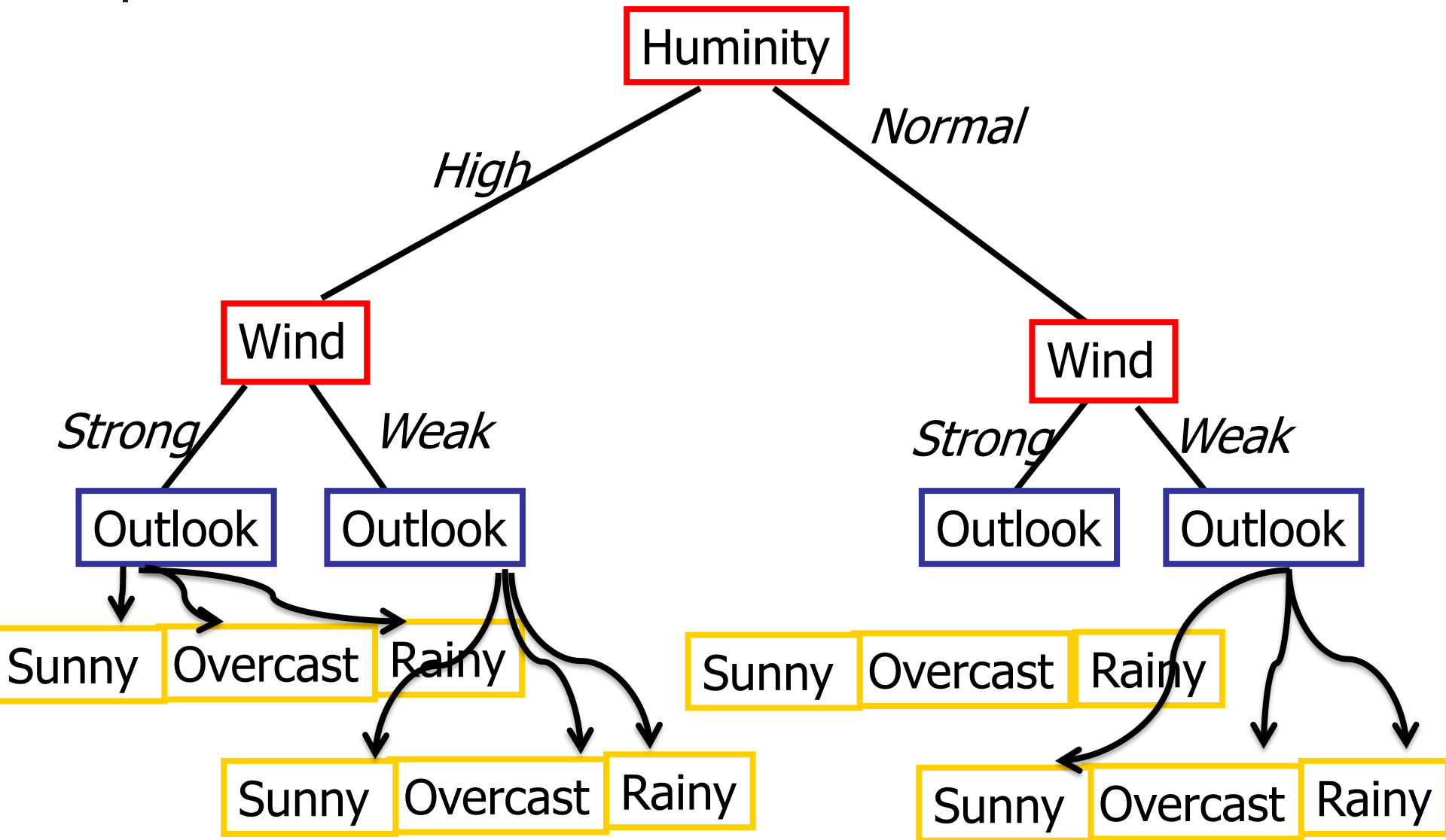


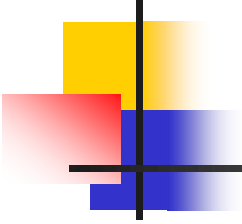
$(\text{Outlook} = \text{Sunny} \wedge \text{Humidity} = \text{Normal}) \vee (\text{Outlook} = \text{Overcast}) \vee (\text{Outlook} = \text{Rain} \wedge \text{Wind} = \text{Weak})$

Complete Tree



OR Complete Tree



- 
-
- Decision Trees are a good solution but not optimum.
 - To obtain an optimum decision tree a rule is needed.
 - We need to find the root node according to some calculations (Entropy & Gain).
 - After find the root node it will continue up to leaves.



Steps for a desion tree construction:

1. Decision tree needs a **decision** (Go to Play Tennis or not)
2. **Entropy** calculation of the system
3. Root Node selection (according to **Information Gain**)



Entropy & Gain

- Determining which attribute is best (Entropy & Gain)
- Entropy (E) is the minimum number of bits needed in order to classify an arbitrary example as yes or no
- $E(S) = \sum_{i=1}^c -p_i \log_2 p_i$
 - **E(S)=0** , if all variables are the **same**
 - **E(S)=1** , if variables are distributed **equally**.
 - **0<E(S)<1** , if variables are distributed **randomly**.
- The information gain $G(S,A)$ where A is an attribute
- $G(S,A) \equiv E(S) - \sum_{v \text{ in Values}(A)} (|S_v| / |S|) * E(S_v)$


Entropy

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

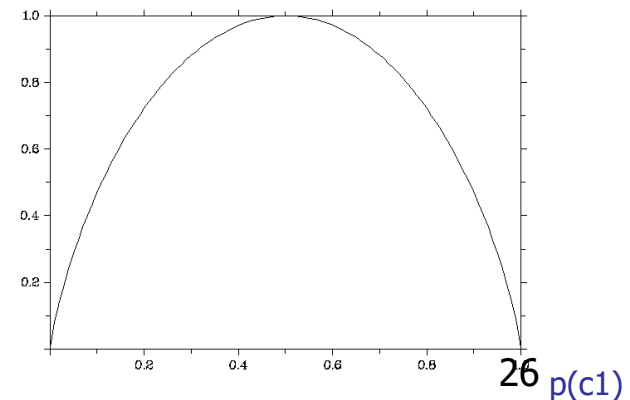
$$Entropy(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

- Where S is a set of training examples,
 - c is the number of classes, and
 - p_i is the proportion of the training set that is of class i
- Zero Entropy occurs when the entire set is from one class

For our entropy equation $0 \log_2 0 = 0$

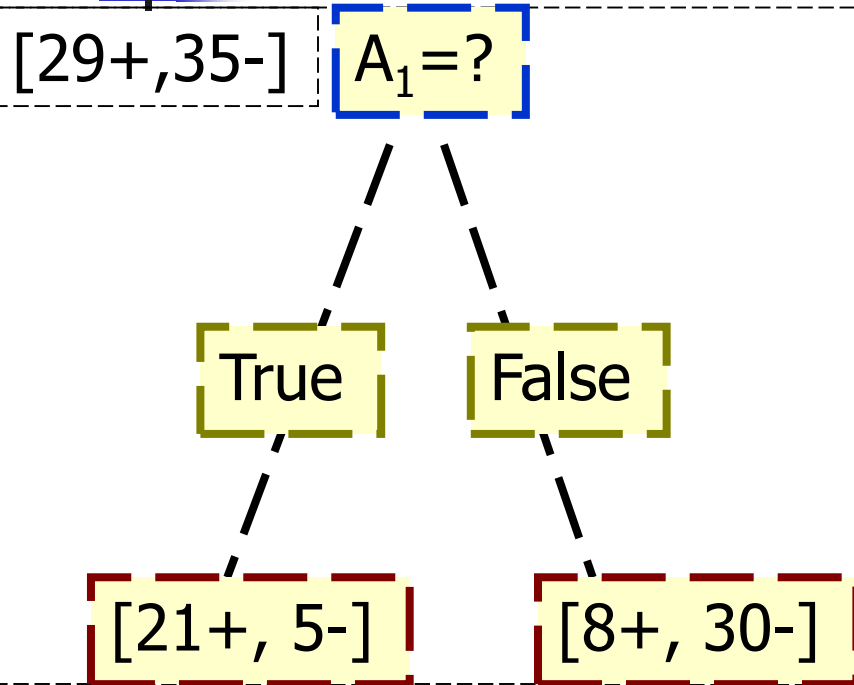
- Also it can be written as: **$E(S) = -(p_+)*\log_2(p_+) - (p_-)*\log_2(p_-)$**
- Where p_+ is the positive samples
- Where p_- is the negative samples
- Where S is the sample of attributions
- For a two-class problem: 

entropy



Example

The Entropy of A1 is computed as the following:



$$\begin{aligned} E(\mathbf{A}) &= -29/(29+35) \cdot \log_2(29/(29+35)) - \\ &\quad 35/(35+29) \log_2(35/(35+29)) \\ &= 0.9937 \end{aligned}$$

- The Entropy of True:

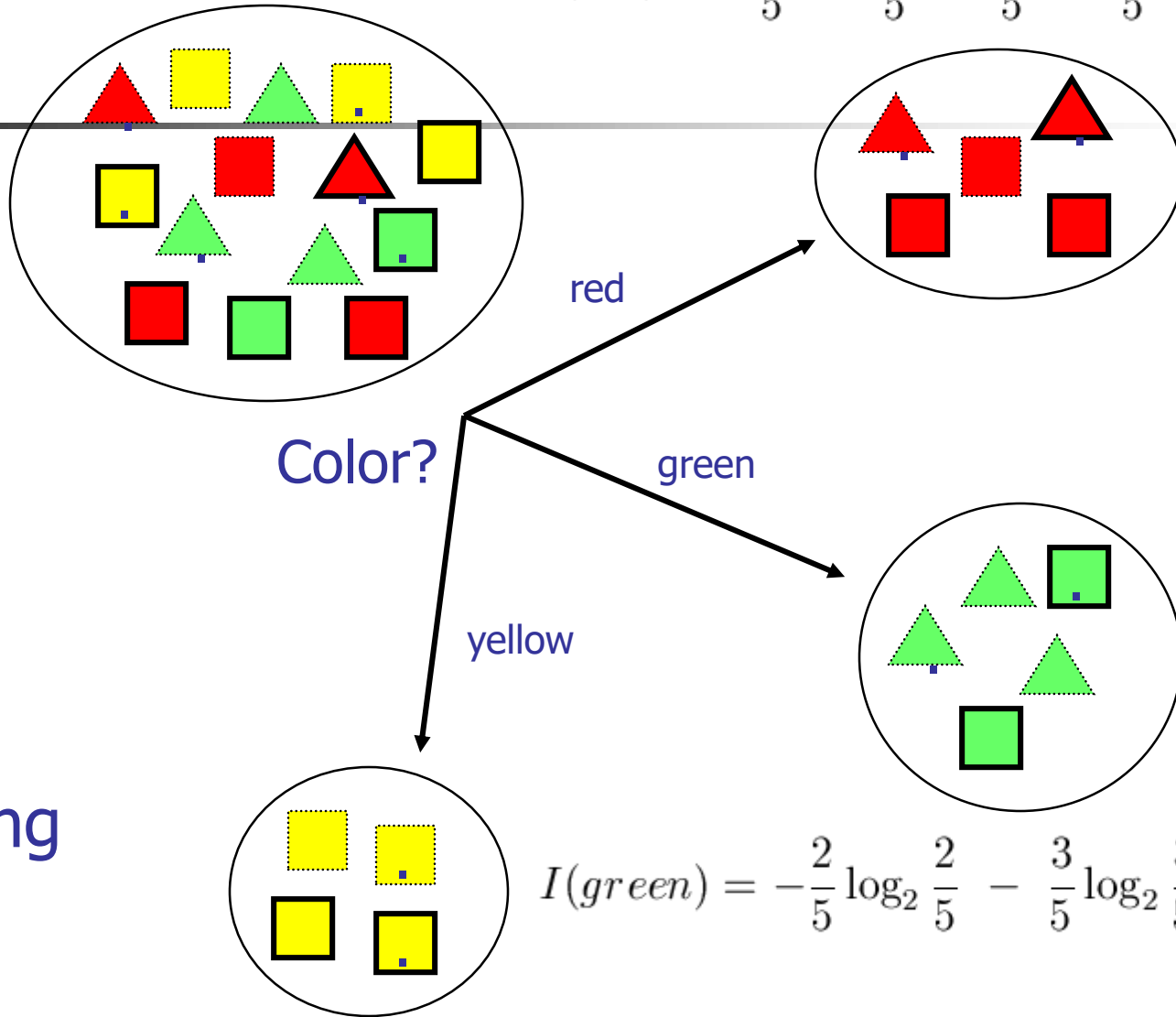
$$\begin{aligned} E(\mathbf{TRUE}) &= -21/(21+5) \cdot \log_2(21/(21+5)) \\ &\quad - 5/(5+21) \cdot \log_2(5/(5+21)) = 0.7960 \end{aligned}$$

$$\begin{aligned} E(\mathbf{FALSE}) &= -8/(8+30) \cdot \log_2(8/(8+30)) - \\ &\quad 30/(30+8) \cdot \log_2(30/(30+8)) \end{aligned}$$

- The Entropy of False:
 $= 0.7426$

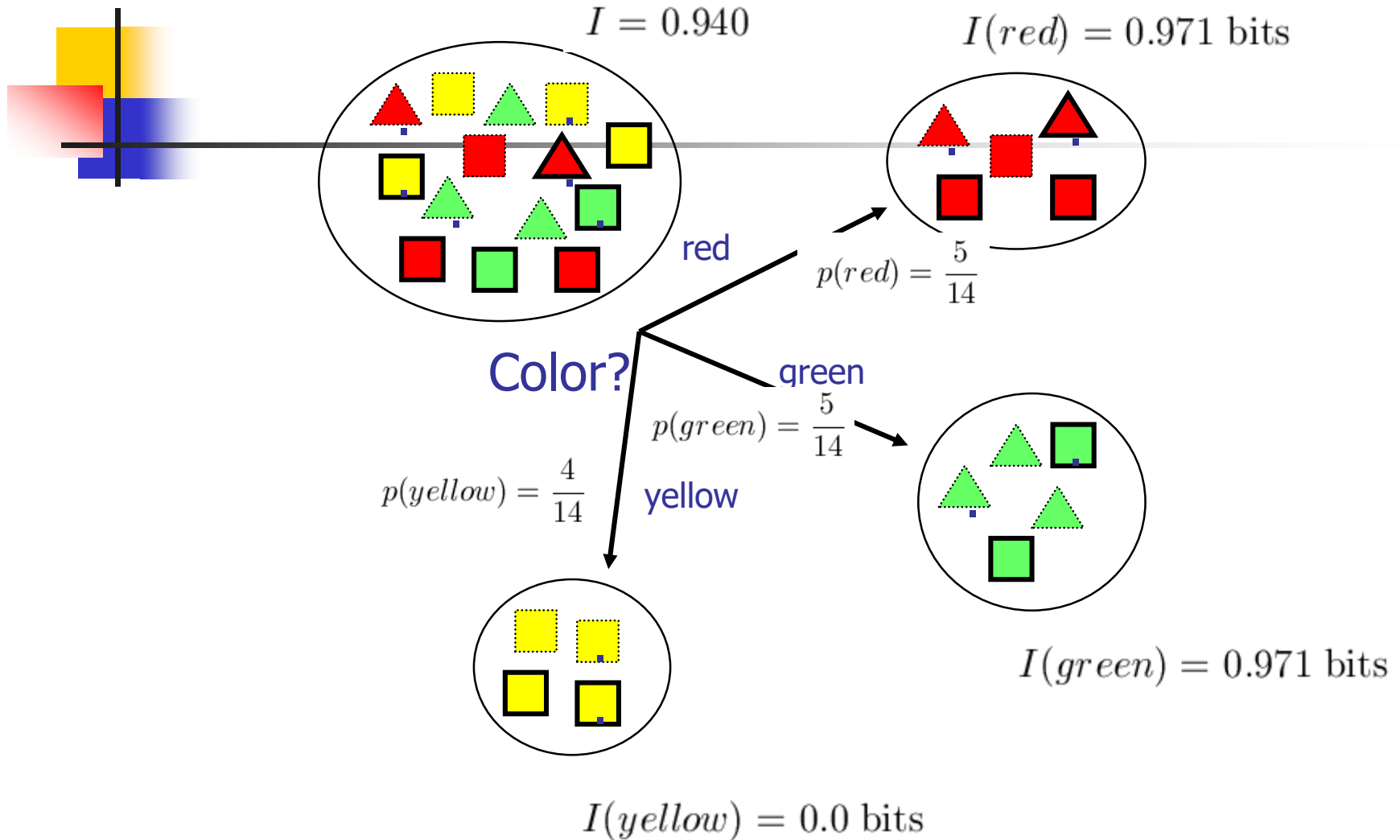
$$I(\text{red}) = -\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5} = 0.971 \text{ bits}$$

Entropy
reduction
by
data set
partitioning



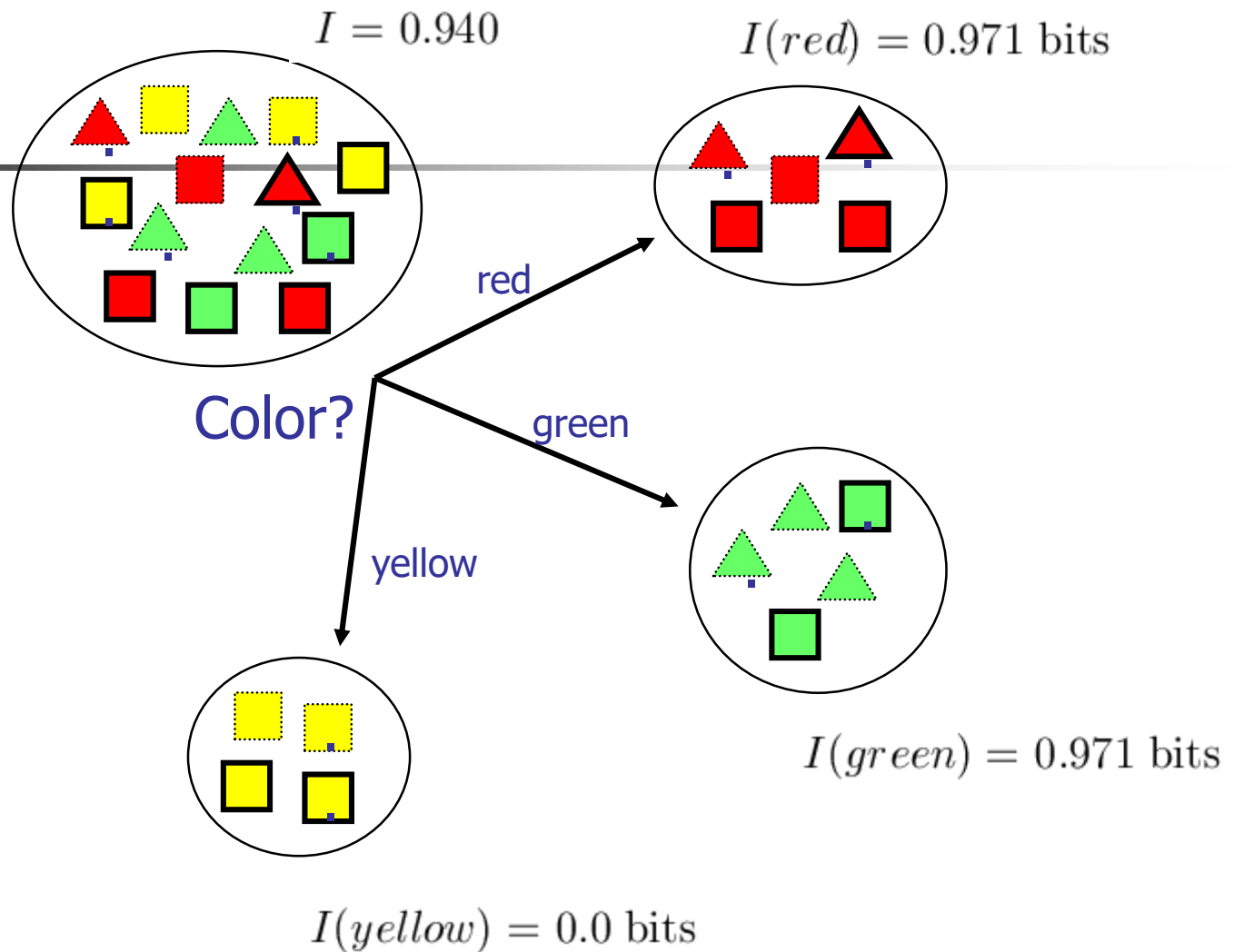
$$I(\text{green}) = -\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} = 0.971 \text{ bits}$$

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



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

Information Gain

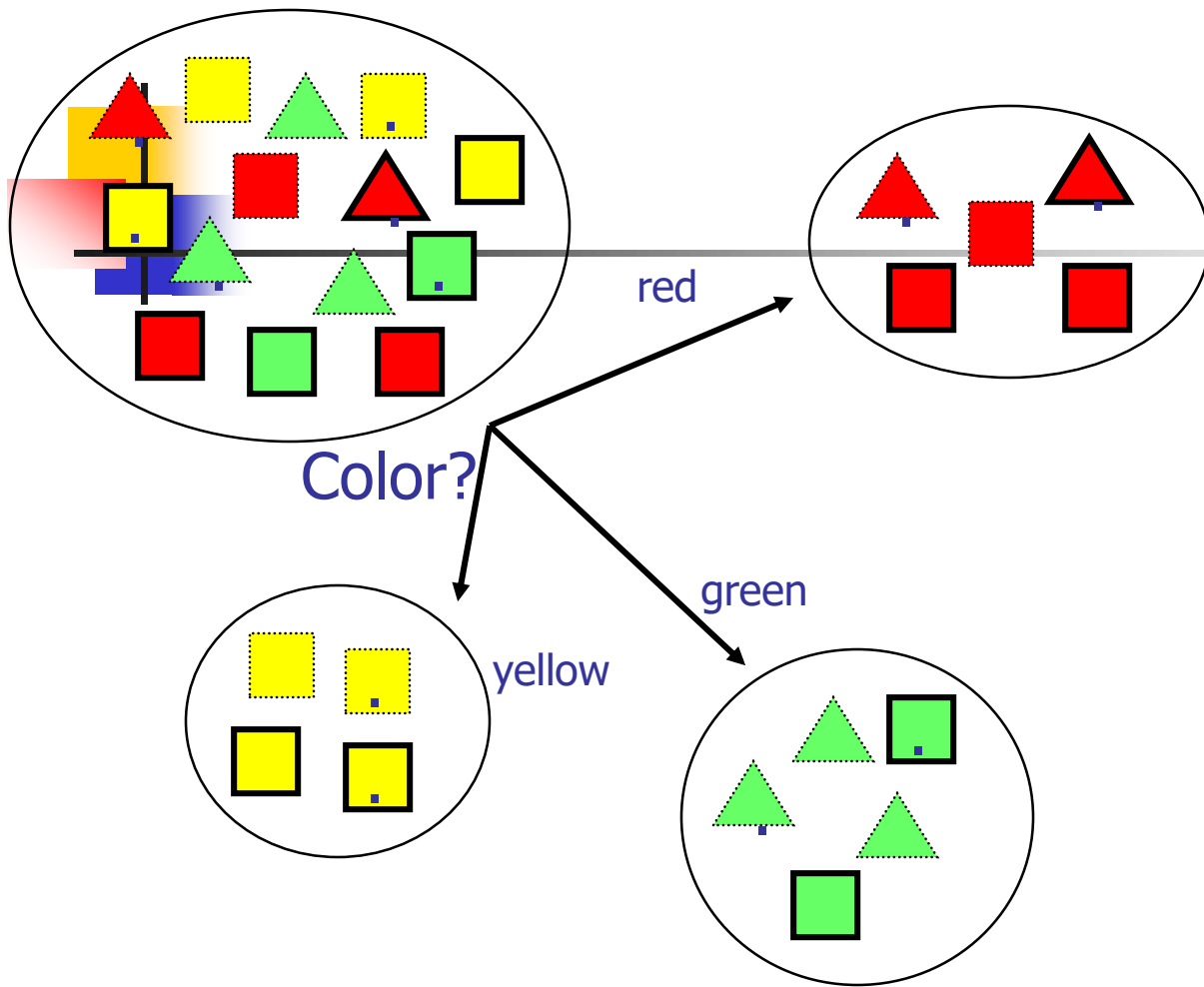


$$\text{Gain}(\text{Color}) = I - I_{\text{res}}(\text{Color}) = 0.940 - 0.694 = 0.246 \text{ bits}$$



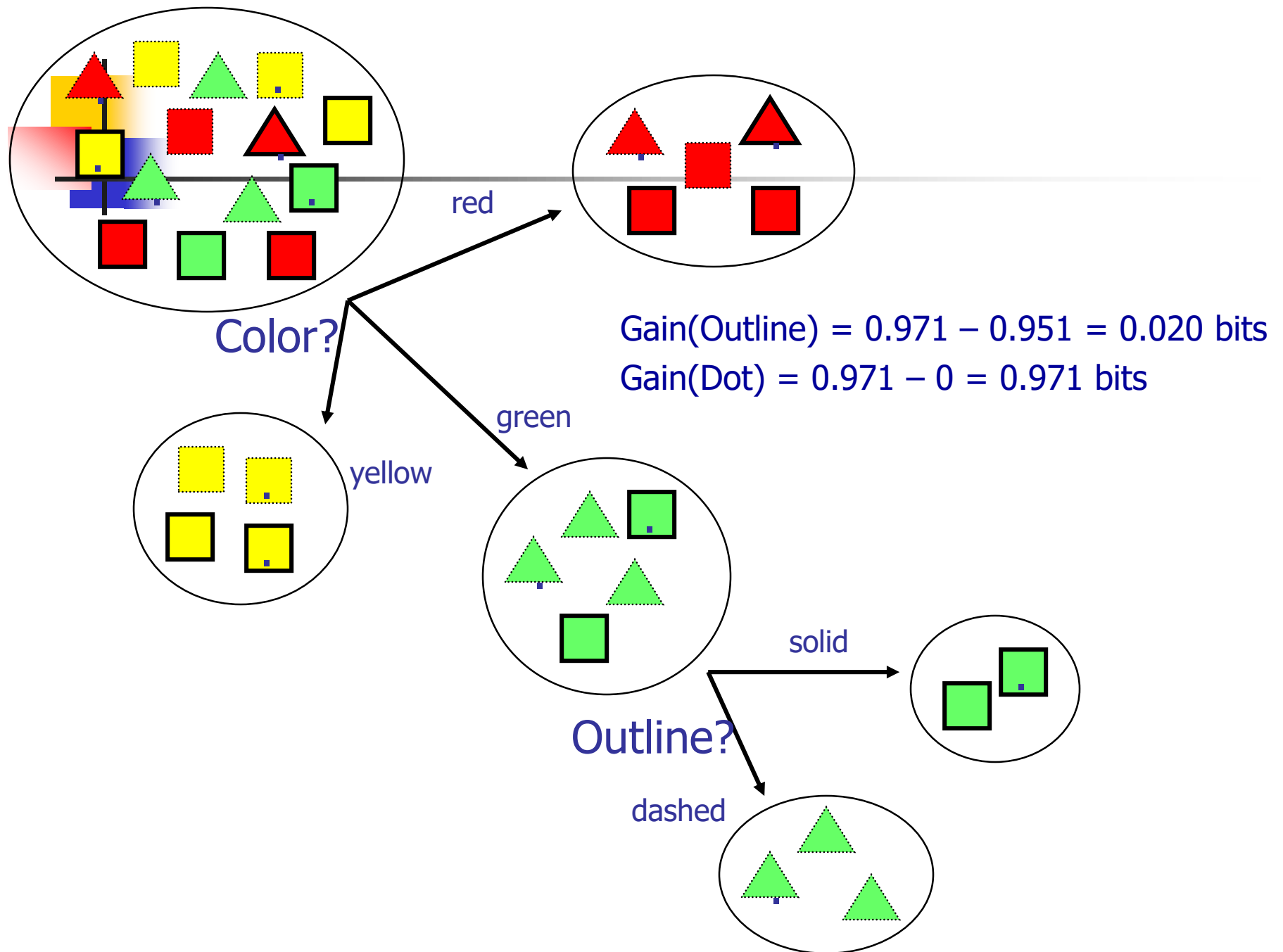
Information Gain of The Attribute

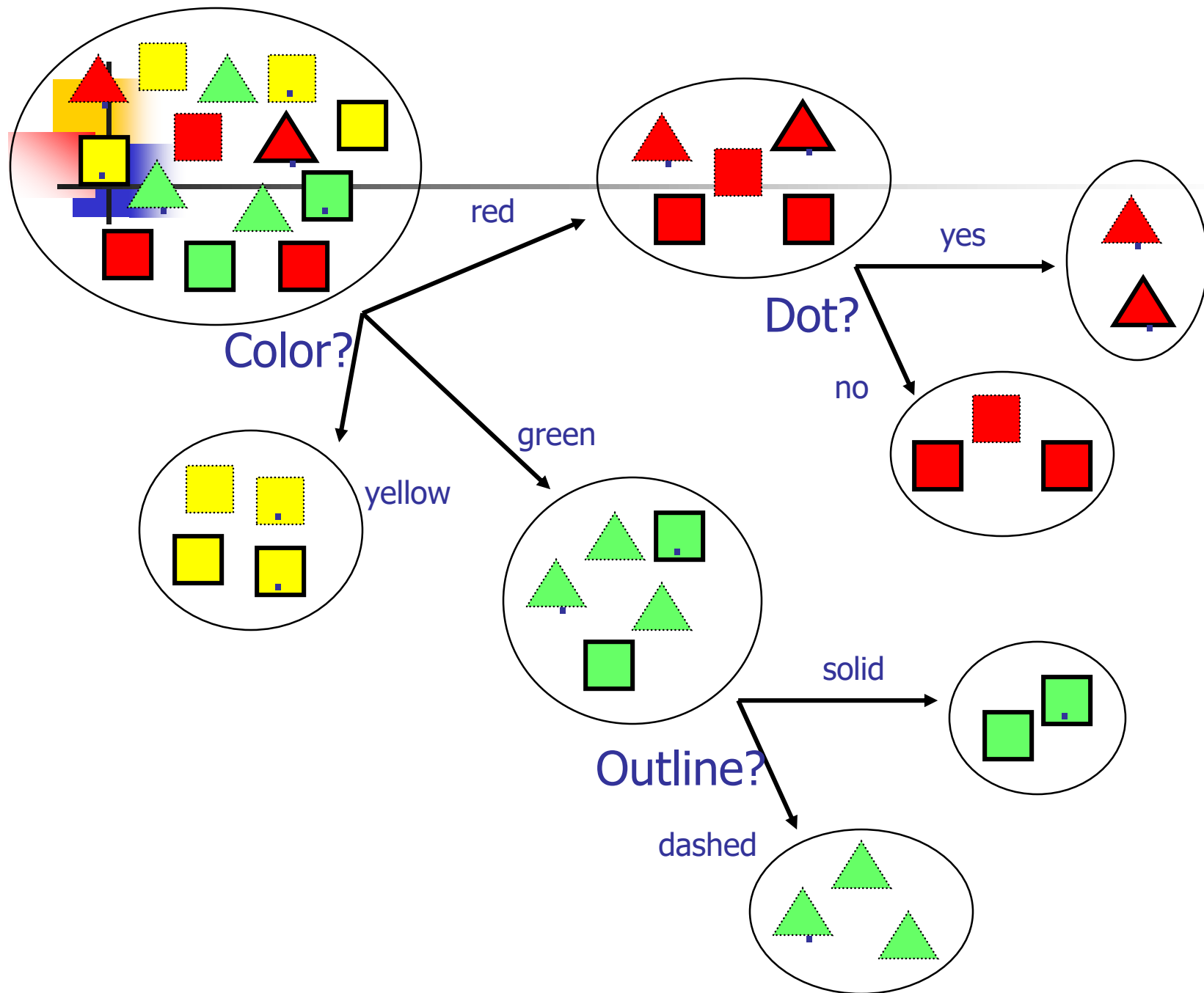
- Attributes
 - $\text{Gain}(\text{Color}) = 0.246$
 - $\text{Gain}(\text{Outline}) = 0.151$
 - $\text{Gain}(\text{Dot}) = 0.048$
- Heuristics: attribute with the highest gain is chosen
- This heuristics is local (local minimization of impurity)



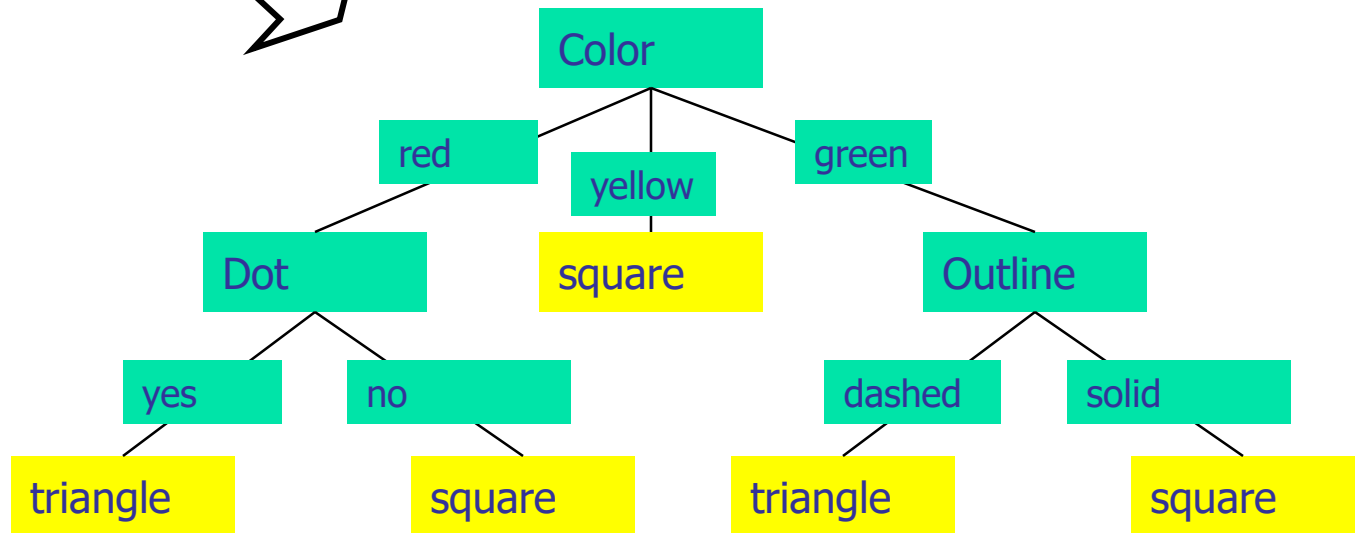
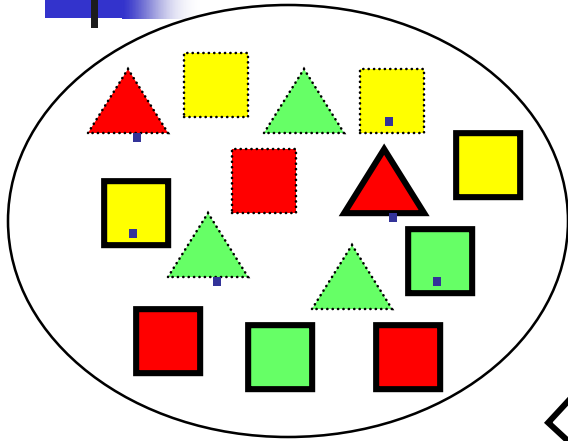
$$\text{Gain(Outline)} = 0.971 - 0 = 0.971 \text{ bits}$$

$$\text{Gain(Dot)} = 0.971 - 0.951 = 0.020 \text{ bits}$$

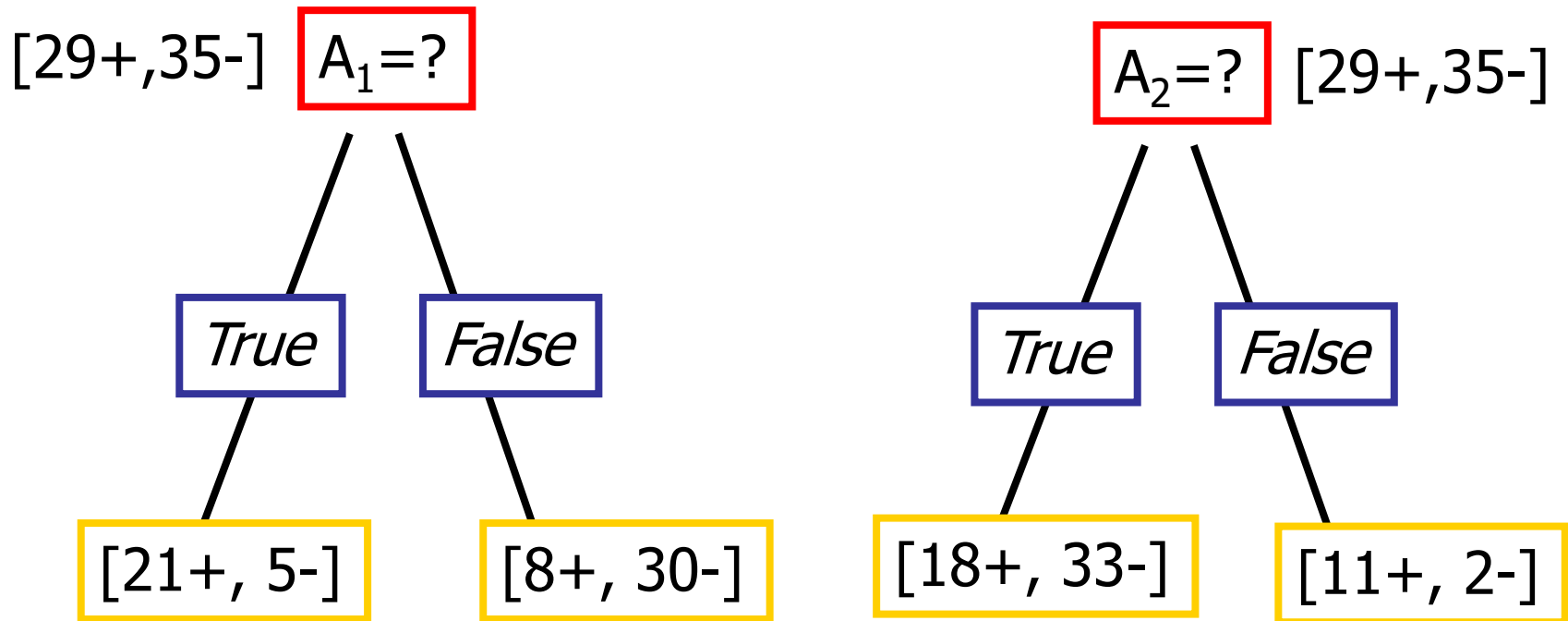




Decision Tree



Which Attribute is "best"?



Information Gain

$$\text{Entropy}([21+, 5-]) = 0.71$$

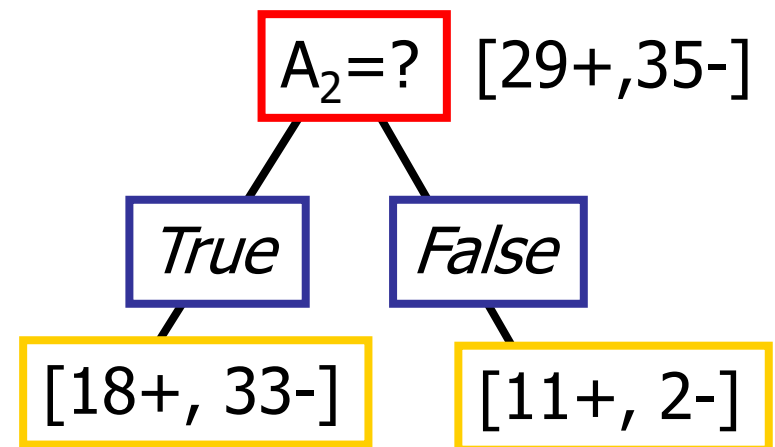
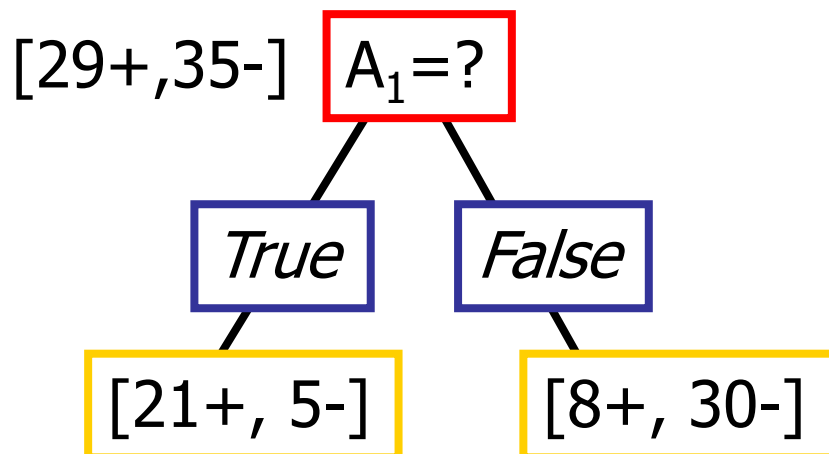
$$\text{Entropy}([8+, 30-]) = 0.74$$

$$\begin{aligned}\text{Gain}(S, A_1) &= \text{Entropy}(S) \\ &\quad - 26/64 * \text{Entropy}([21+, 5-]) \\ &\quad - 38/64 * \text{Entropy}([8+, 30-]) \\ &= 0.27\end{aligned}$$

$$\text{Entropy}([18+, 33-]) = 0.94$$

$$\text{Entropy}([8+, 30-]) = 0.62$$

$$\begin{aligned}\text{Gain}(S, A_2) &= \text{Entropy}(S) \\ &\quad - 51/64 * \text{Entropy}([18+, 33-]) \\ &\quad - 13/64 * \text{Entropy}([11+, 2-]) \\ &= 0.12\end{aligned}$$



Playing Tennis or Not Example Calculations

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

Selecting the Next Attribute

$S=[9+,5-]$
 $E=0.940$

Humidity

High

Normal

$[3+, 4-]$

$E=0.985$

$[6+, 1-]$

$E=0.592$

$$\begin{aligned}\text{Gain}(S, \text{Humidity}) &= 0.940 - (7/14) * 0.985 \\ &\quad - (7/14) * 0.592 \\ &= 0.151\end{aligned}$$

$S=[9+,5-]$
 $E=0.940$

Wind

Weak

Strong

$[6+, 2-]$

$E=0.811$

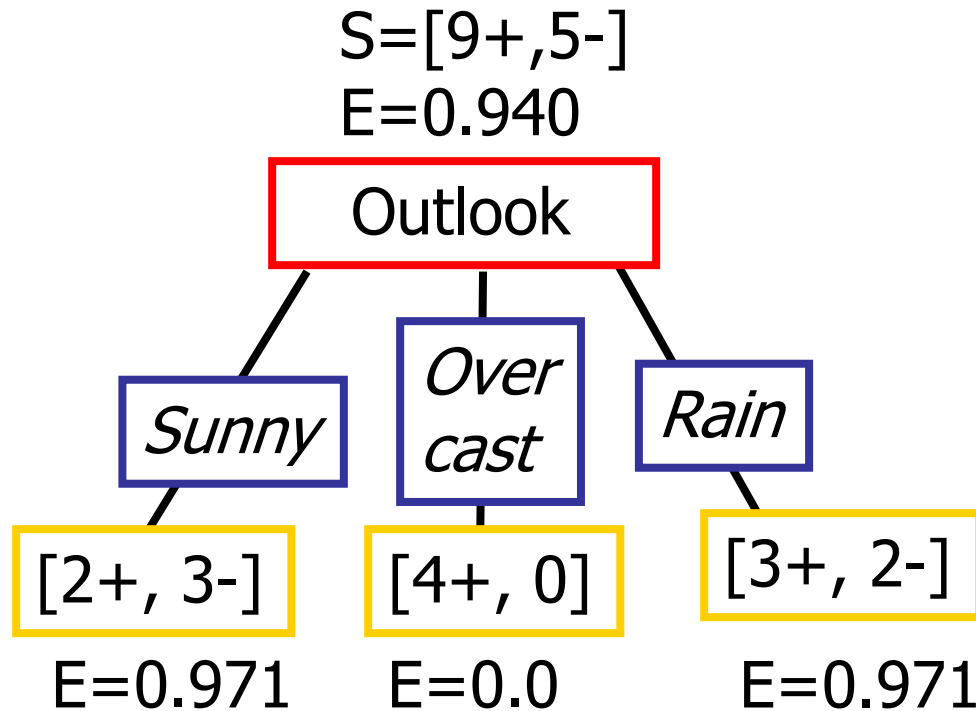
$[3+, 3-]$

$E=1.0$

$$\begin{aligned}\text{Gain}(S, \text{Wind}) &= 0.940 - (8/14) * 0.811 \\ &\quad - (6/14) * 1.0 \\ &= 0.048\end{aligned}$$

Humidity provides greater info. gain than Wind, w.r.t target classification. 39

Selecting the Next Attribute



$$\begin{aligned} \text{Gain}(S, \text{Outlook}) &= 0.940 - (5/14) * 0.971 \\ &\quad - (4/14) * 0.0 - (5/14) * 0.0971 \\ &= 0.247 \end{aligned}$$



Selecting the Next Attribute

The information gain values for the 4 attributes are:

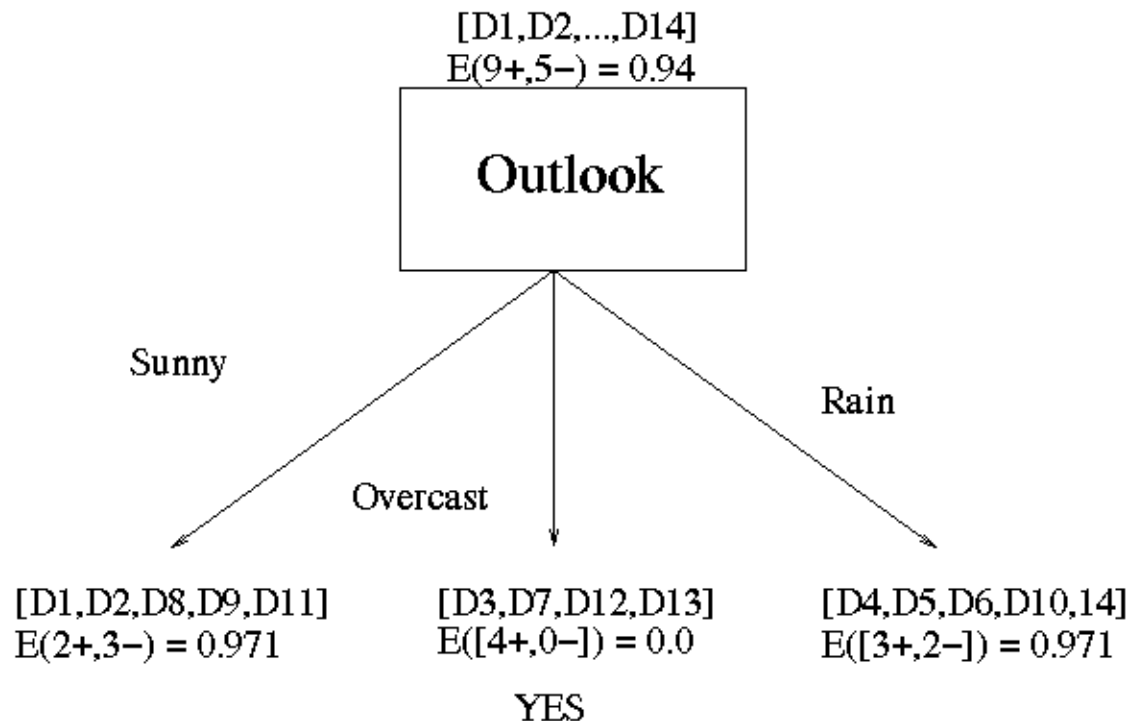
- $\text{Gain}(S, \text{Outlook}) = 0.247$
- $\text{Gain}(S, \text{Humidity}) = 0.151$
- $\text{Gain}(S, \text{Wind}) = 0.048$
- $\text{Gain}(S, \text{Temperature}) = 0.029$

where S denotes the collection of training examples

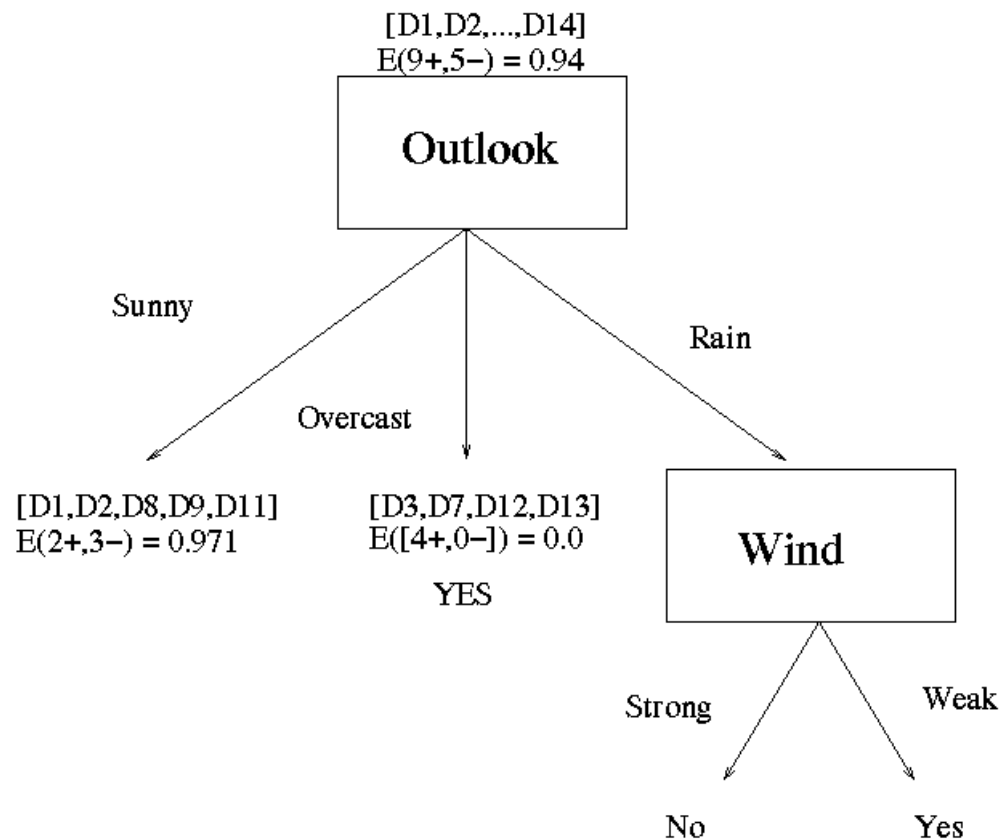
Note: $0\log_2 0 = 0$

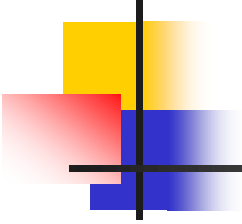
Decision Tree Learning: A Simple Example

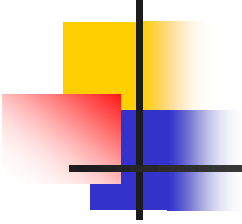
- Outlook is our winner!



- Now our decision tree looks like:



- 
- Let
 - $E([X+, Y-])$ represent that there are X positive training elements and Y negative elements.
 - Therefore the Entropy for the training data, $E(S)$, can be represented as $E([9+, 5-])$ because of the 14 training examples 9 of them are **yes** and 5 of them are **no**.
 - Let's start off by calculating the Entropy of the Training Set.
 - $E(S) = E([9+, 5-]) = (-9/14 \log_2 9/14) + (-5/14 \log_2 5/14)$
 - $= 0.94$
 - Next we will need to calculate the information gain $G(S, A)$ for each attribute A where A is taken from the set {Outlook, Temperature, Humidity, Wind}.

- 
- **$G(S, \text{Humidity}) = 0.94 - [7/14 * E(\text{Humidity}=\text{high}) + 7/14 * E(\text{Humidity}=\text{normal})]$**
 - $G(S, \text{Humidity}) = 0.94 - [7/14 * E([3+, 4-]) + 7/14 * E([6+, 1-])]$
 - $G(S, \text{Humidity}) = 0.94 - [7/14 * 0.985 + 7/14 * 0.592]$
 - **$G(S, \text{Humidity}) = 0.1515$**

- Now that we have discovered the root of our decision tree
- We must now recursively find the nodes that should go below **Sunny, Overcast,** and **Rain.**
- **$G(\text{Outlook}=\text{Rain}, \text{Humidity}) = 0.971 - [2/5 * E(\text{Outlook}=\text{Rain} \wedge \text{Humidity}=\text{high}) + 3/5 * E(\text{Outlook}=\text{Rain} \wedge \text{Humidity}=\text{normal})]$**
- **$G(\text{Outlook}=\text{Rain}, \text{Humidity}) = 0.02$**
- **$G(\text{Outlook}=\text{Rain}, \text{Wind}) = 0.971 - [3/5 * 0 + 2/5 * 0]$**
- **$G(\text{Outlook}=\text{Rain}, \text{Wind}) = 0.971$**



■ The information **gain for Outlook** is:

■ $G(S, \text{Outlook}) = E(S) - [5/14 * E(\text{Outlook}=\text{sunny}) + 4/14 * E(\text{Outlook} = \text{overcast}) + 5/14 * E(\text{Outlook}=\text{rain})]$

■ $G(S, \text{Outlook}) = E([9+, 5-]) - [5/14 * E(2+, 3-) + 4/14 * E([4+, 0-]) + 5/14 * E([3+, 2-])]$

■ $G(S, \text{Outlook}) = 0.94 - [5/14 * 0.971 + 4/14 * 0.0 + 5/14 * 0.971]$

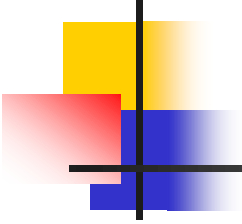
■ **$G(S, \text{Outlook}) = 0.246$**

■ $G(S, \text{Temperature}) = 0.94 - [4/14 * E(\text{Temperature}=\text{hot}) + 6/14 * E(\text{Temperature}=\text{mild}) + 4/14 * E(\text{Temperature}=\text{cool})]$

■ $G(S, \text{Temperature}) = 0.94 - [4/14 * E([2+, 2-]) + 6/14 * E([4+, 2-]) + 4/14 * E([3+, 1-])]$

■ $G(S, \text{Temperature}) = 0.94 - [4/14 + 6/14 * 0.918 + 4/14 * 0.811]$

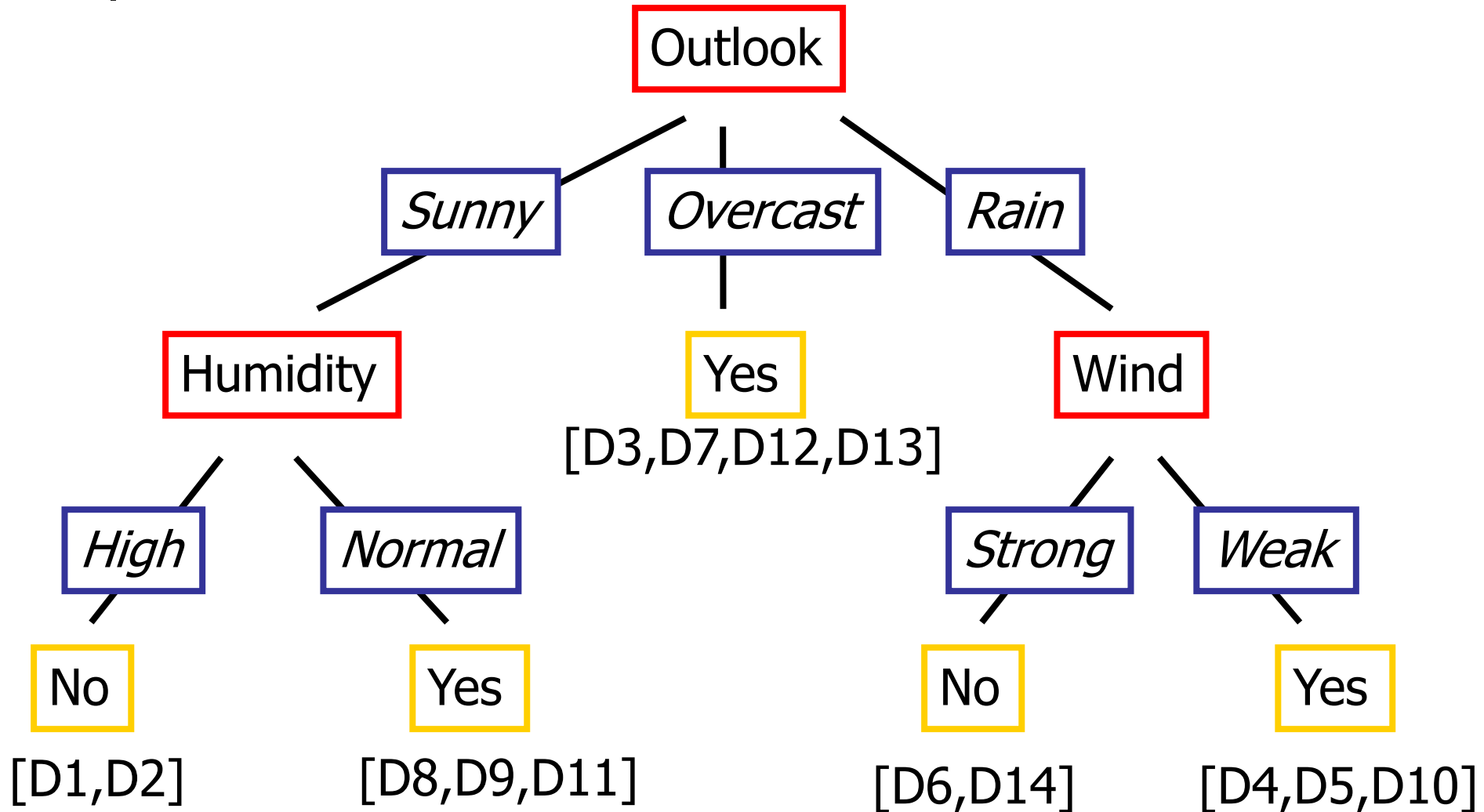
■ **$G(S, \text{Temperature}) = 0.029$**

- 
- **$G(S, \text{Humidity}) = 0.94 - [7/14 * E(\text{Humidity}=\text{high}) + 7/14 * E(\text{Humidity}=\text{normal})]$**
 - $G(S, \text{Humidity}) = 0.94 - [7/14 * E([3+, 4-]) + 7/14 * E([6+, 1-])]$
 - $G(S, \text{Humidity}) = 0.94 - [7/14 * 0.985 + 7/14 * 0.592]$
 - **$G(S, \text{Humidity}) = 0.1515$**

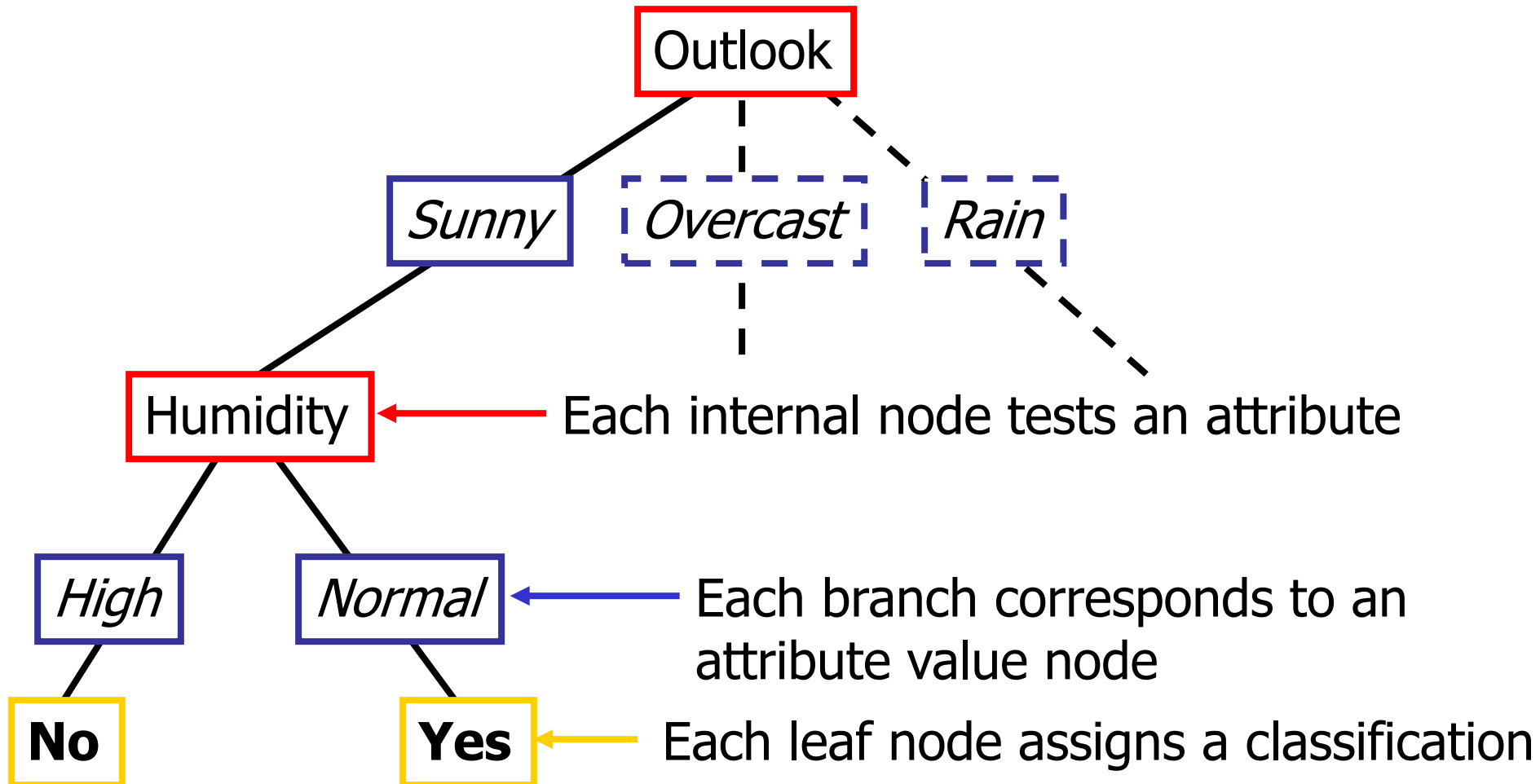
 - **$G(S, \text{Wind}) = 0.94 - [8/14 * 0.811 + 6/14 * 1.00]$**
 - **$G(S, \text{Wind}) = 0.048$**



Then the Complete Tree should be like this

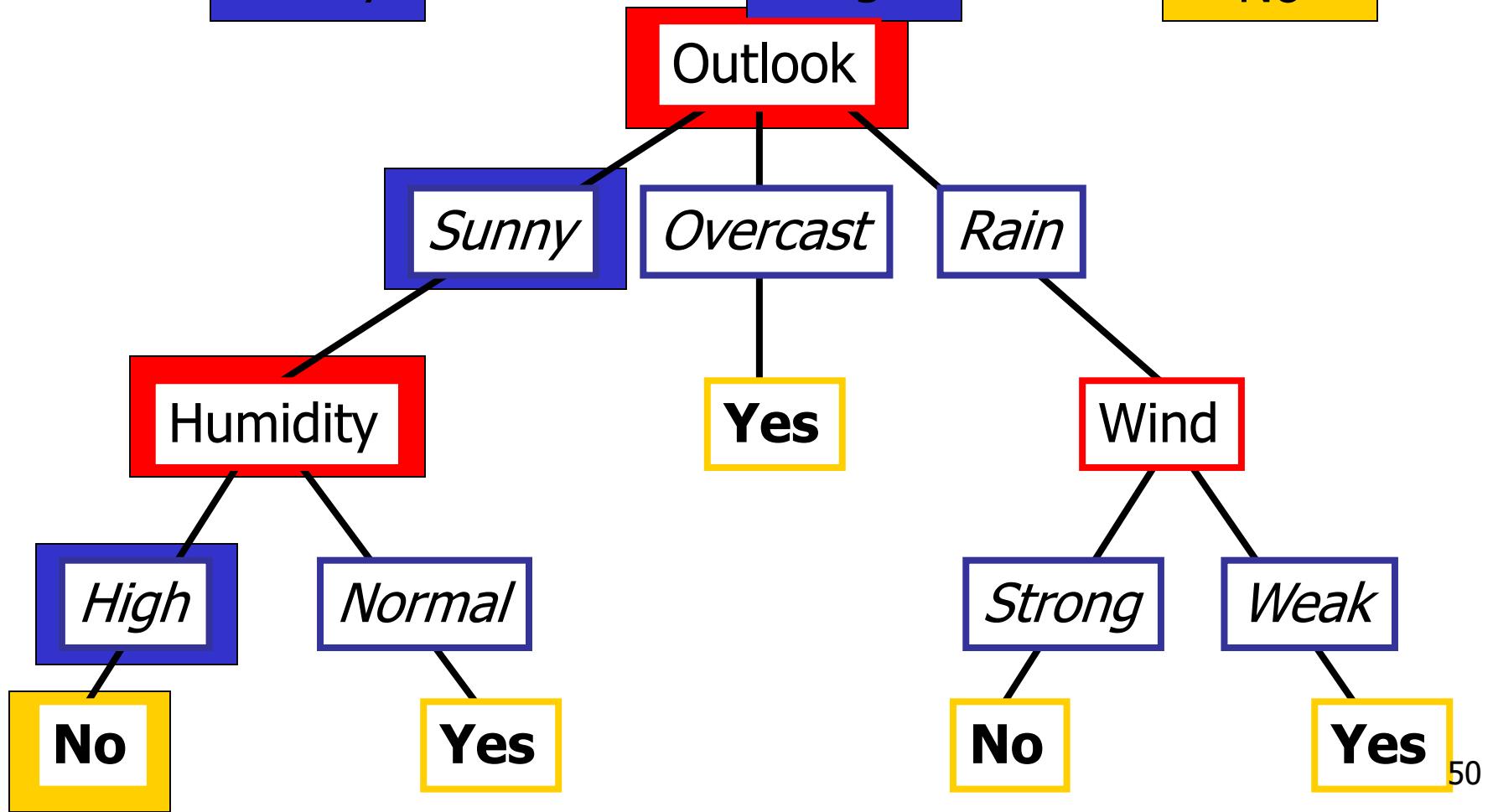


Decision Tree for PlayTennis



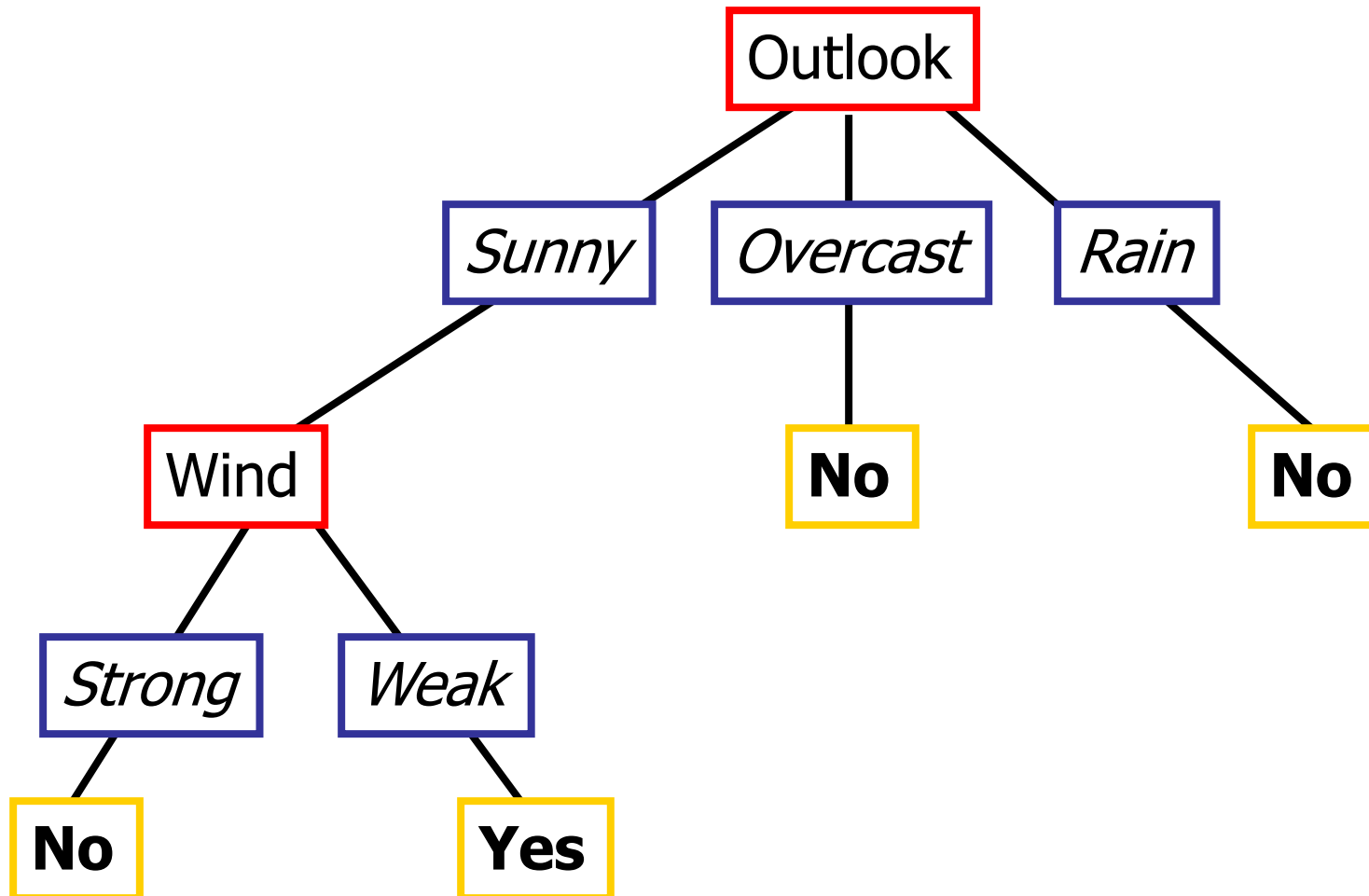
Decision Tree for PlayTennis

Outlook	Temperature	Humidity	Wind	PlayTennis
Sunny	Hot	High	Weak	No



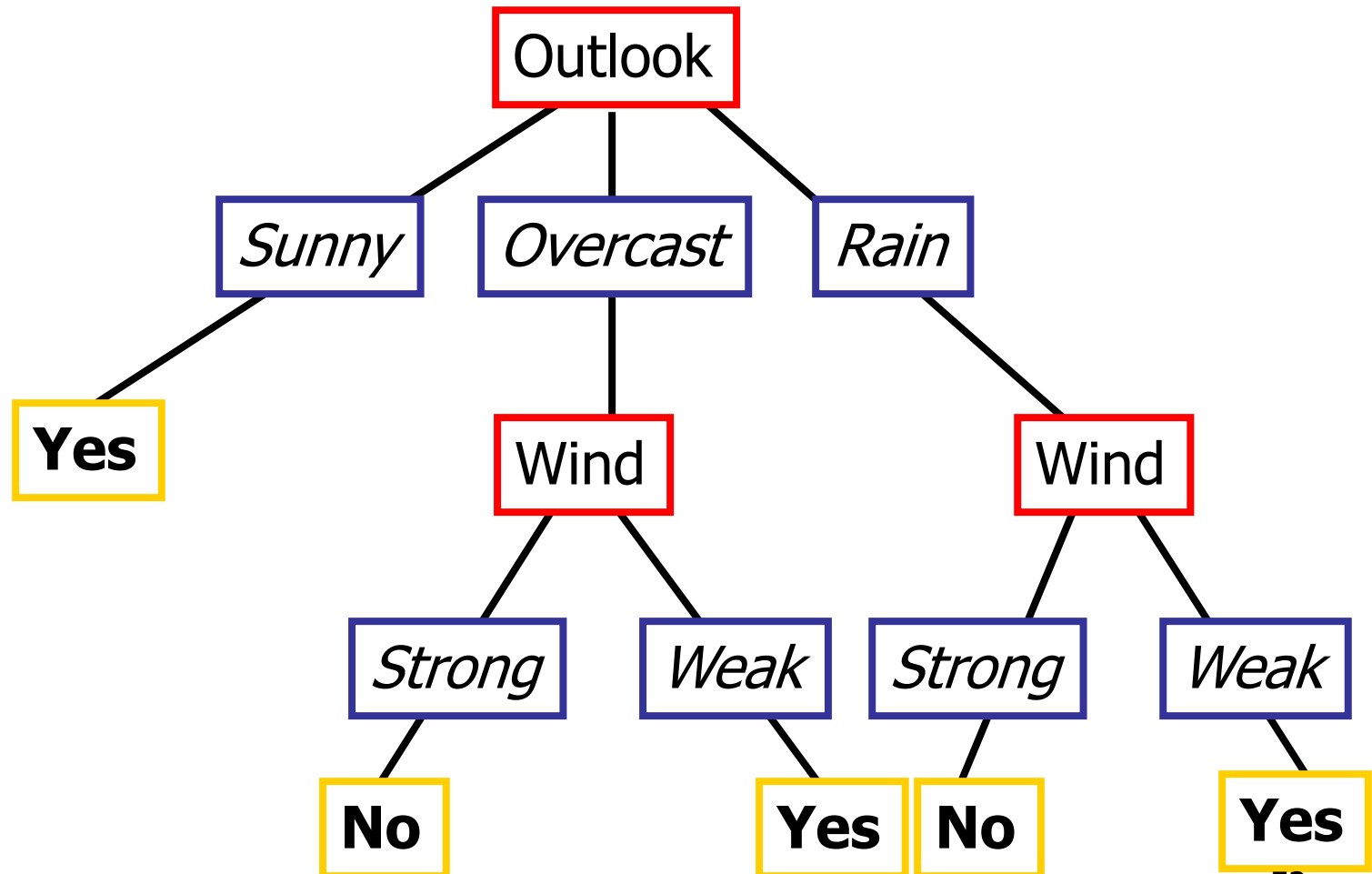
Decision Tree for Conjunction

Outlook=Sunny \wedge Wind=Weak



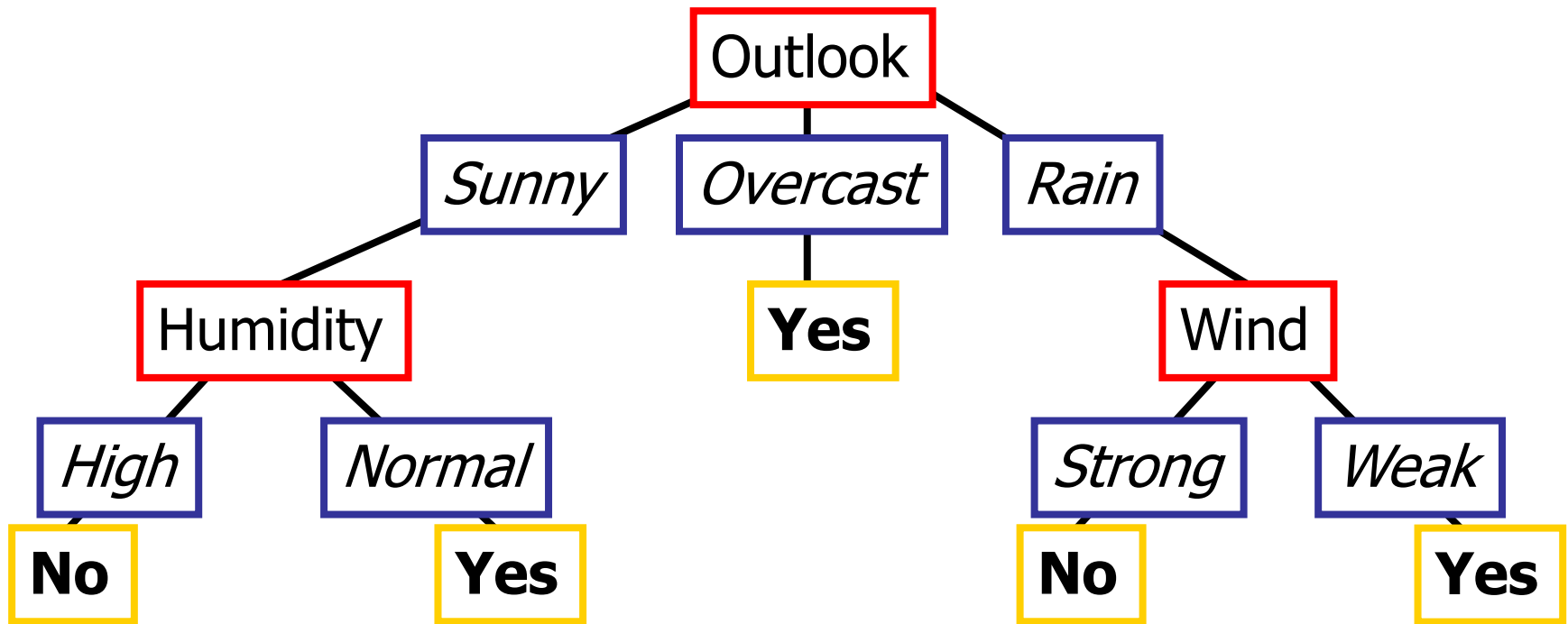
Decision Tree for Disjunction

Outlook=Sunny \vee Wind=Weak



Decision Tree

- Decision trees represent disjunctions of conjunctions



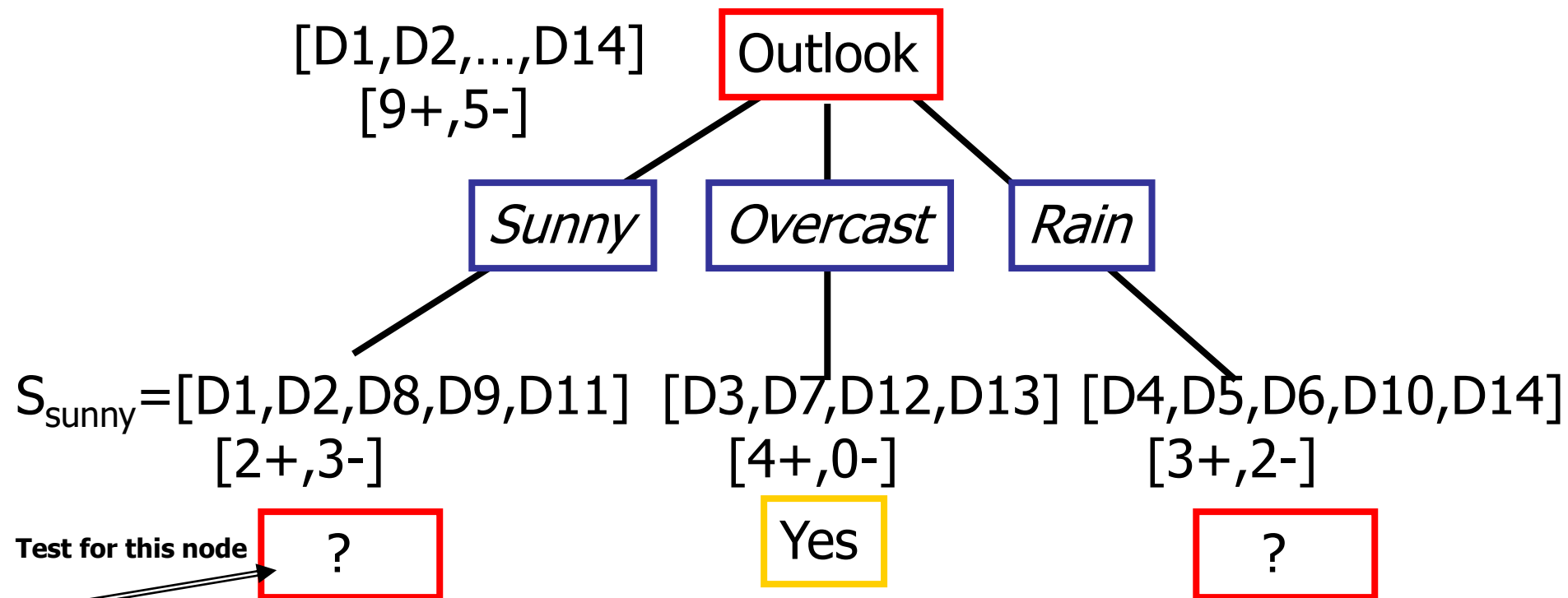
(Outlook=Sunny \wedge Humidity=Normal)

✓ (Outlook=Overcast)

✓ (Outlook=Rain \wedge Wind=Weak)

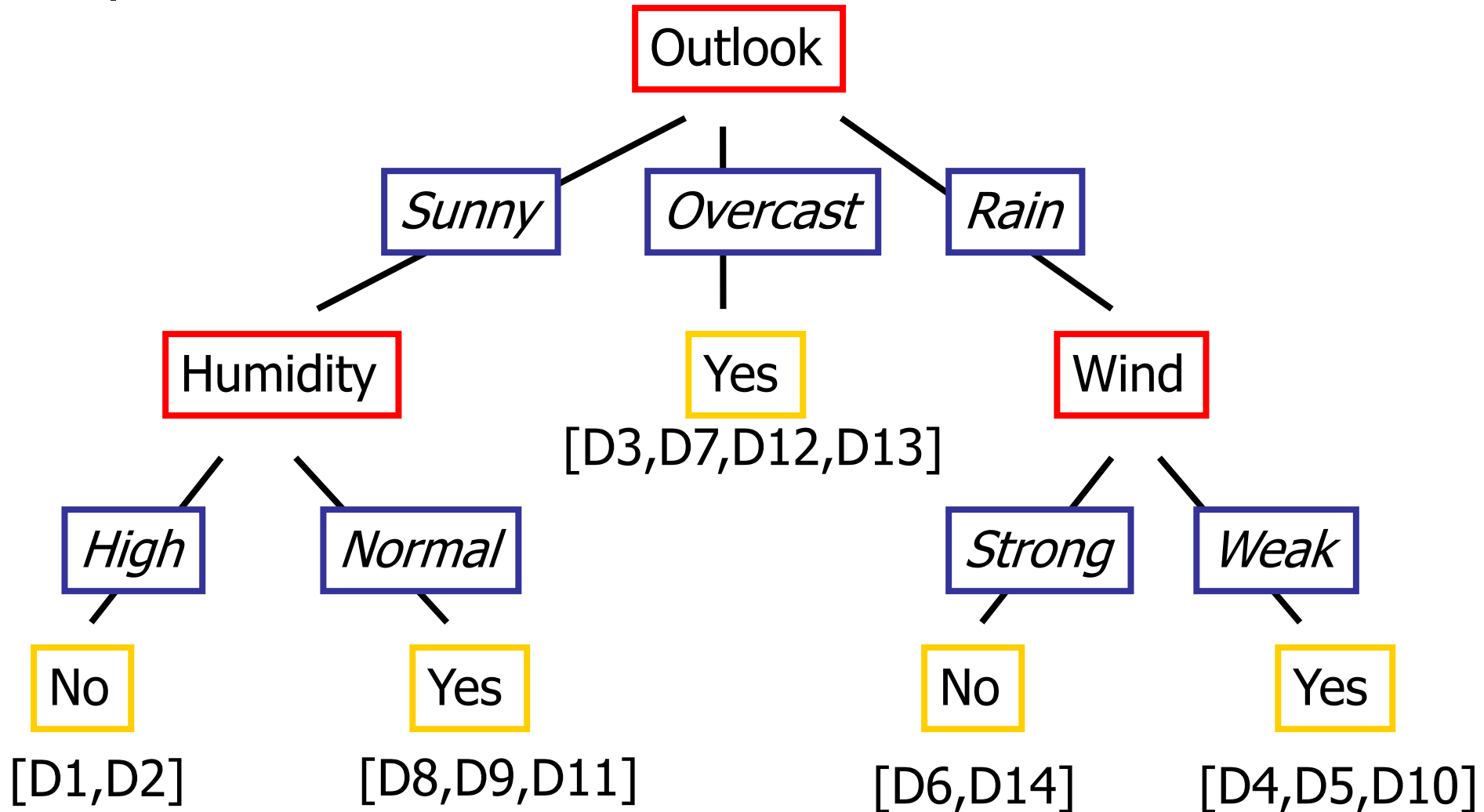
ID3 Algorithm

Note: $0\log_2 0 = 0$

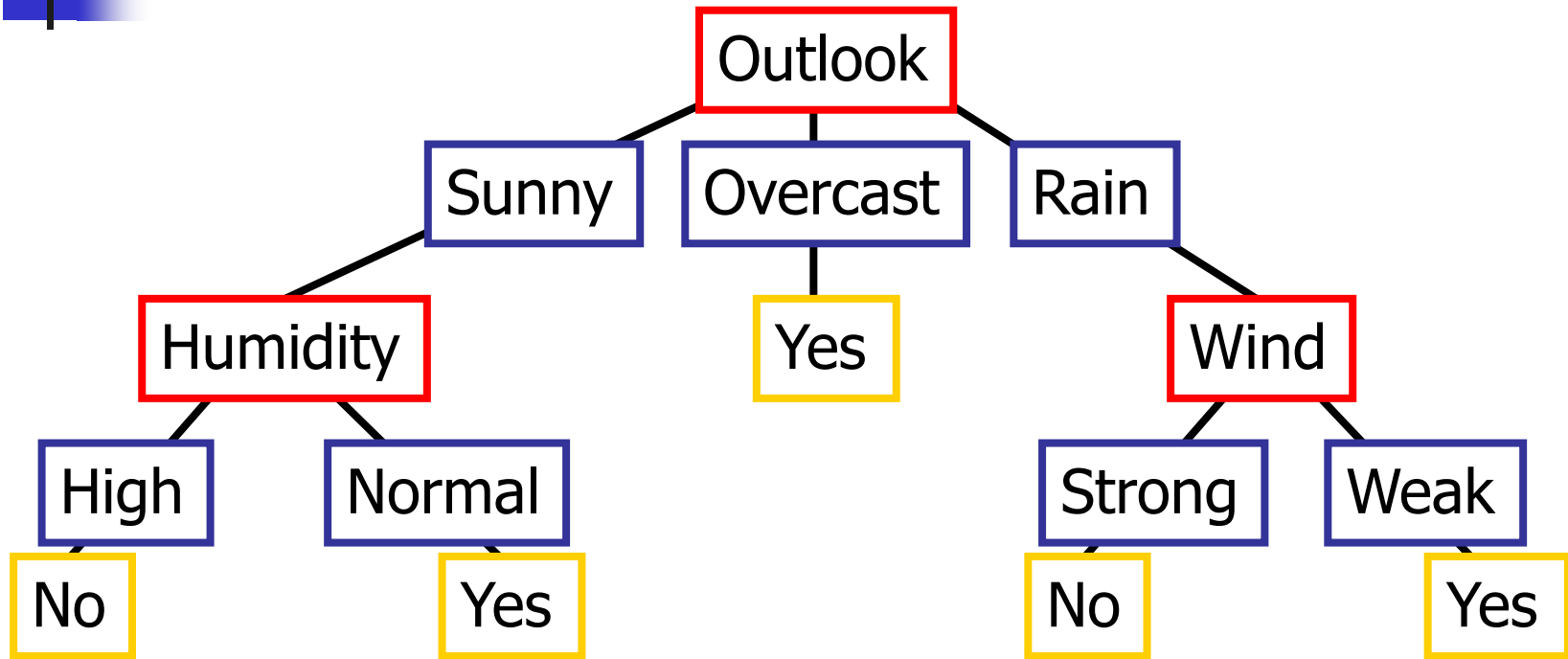


$$\begin{aligned} \text{Gain}(S_{\text{sunny}}, \text{Humidity}) &= 0.970 - (3/5)0.0 - 2/5(0.0) = 0.970 \\ \text{Gain}(S_{\text{sunny}}, \text{Temp.}) &= 0.970 - (2/5)0.0 - 2/5(1.0) - (1/5)0.0 = 0.570 \\ \text{Gain}(S_{\text{sunny}}, \text{Wind}) &= 0.970 - (2/5)1.0 - 3/5(0.918) = 0.019 \end{aligned}$$

ID3 Algorithm



Converting a Tree to Rules



R_1 : If (Outlook=Sunny) \wedge (Humidity=High) Then PlayTennis=No

R_2 : If (Outlook=Sunny) \wedge (Humidity=Normal) Then PlayTennis=Yes

R_3 : If (Outlook=Overcast) Then PlayTennis=Yes

R_4 : If (Outlook=Rain) \wedge (Wind=Strong) Then PlayTennis=No

R_5 : If (Outlook=Rain) \wedge (Wind=Weak) Then PlayTennis=Yes



Advantages/Disadvantages of Decision Trees

Advantages

- Easy construction
- Simple to understand and interpret
- Understandable rules can be obtained.
- Requires little data preparation. (Other techniques often require data normalisation, dummy variables need to be created and blank values to be removed)

Disadvantages

- Lots of class makes it hard to predict or learn.
- Both construction and deconstruction is so high that it causes confusion for learning.



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

- Gain (Sample, Attributes) or Gain (S,A) is expected reduction in entropy due to sorting S on attribute A

$$\text{Gain}(S,A) = \text{Entropy}(S) - \sum_{v \in \text{values}(A)} |S_v|/|S| \text{Entropy}(S_v)$$

So, for the previous example, the Information gain is calculated:

- $$\begin{aligned} G(A1) &= E_{(A1)} - (21+5)/(29+35) * E_{(TRUE)} \\ &\quad - (8+30)/(29+35) * E_{(FALSE)} \\ &= E_{(A1)} - 26/64 * E_{(TRUE)} - 38/64 * E_{(FALSE)} \\ &= 0.9937 - 26/64 * 0.796 - 38/64 * 0.7426 \\ &= 0.5465 \end{aligned}$$



Information Gain Ratio

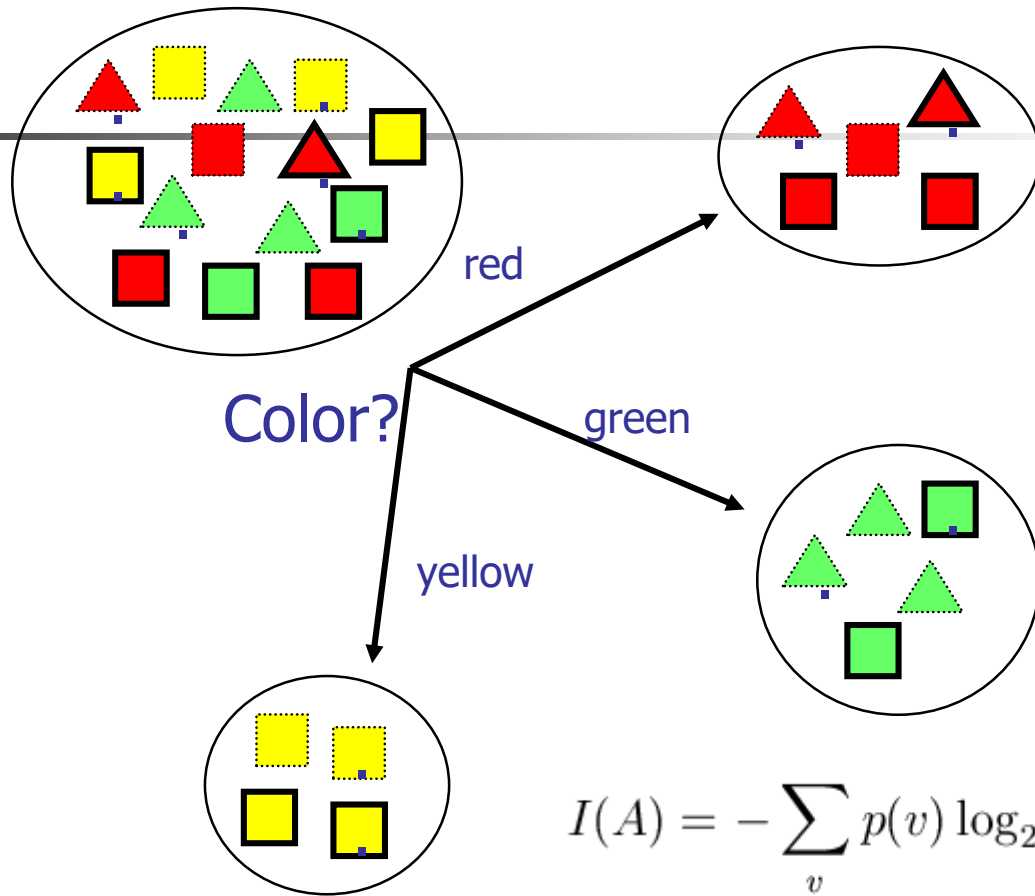
- $I(A)$ is amount of information needed to determine the value of an attribute A

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

- Information gain ratio

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

Information Gain Ratio



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

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

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



Information Gain and Information Gain Ratio

A	$v(A)$	<i>Gain(A)</i>	<i>GainRatio(A)</i>
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)

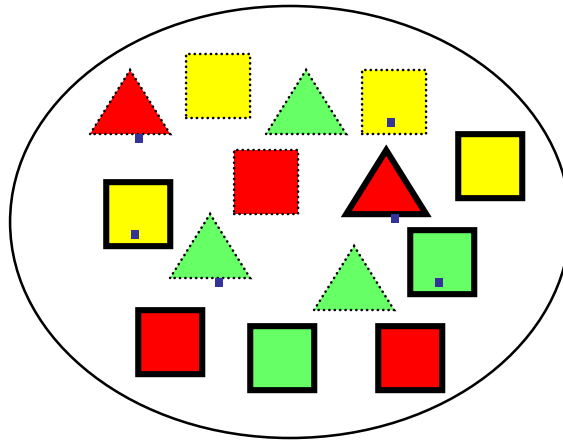
$$Gini = \sum_{i \neq j} p(i)p(j)$$

- 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



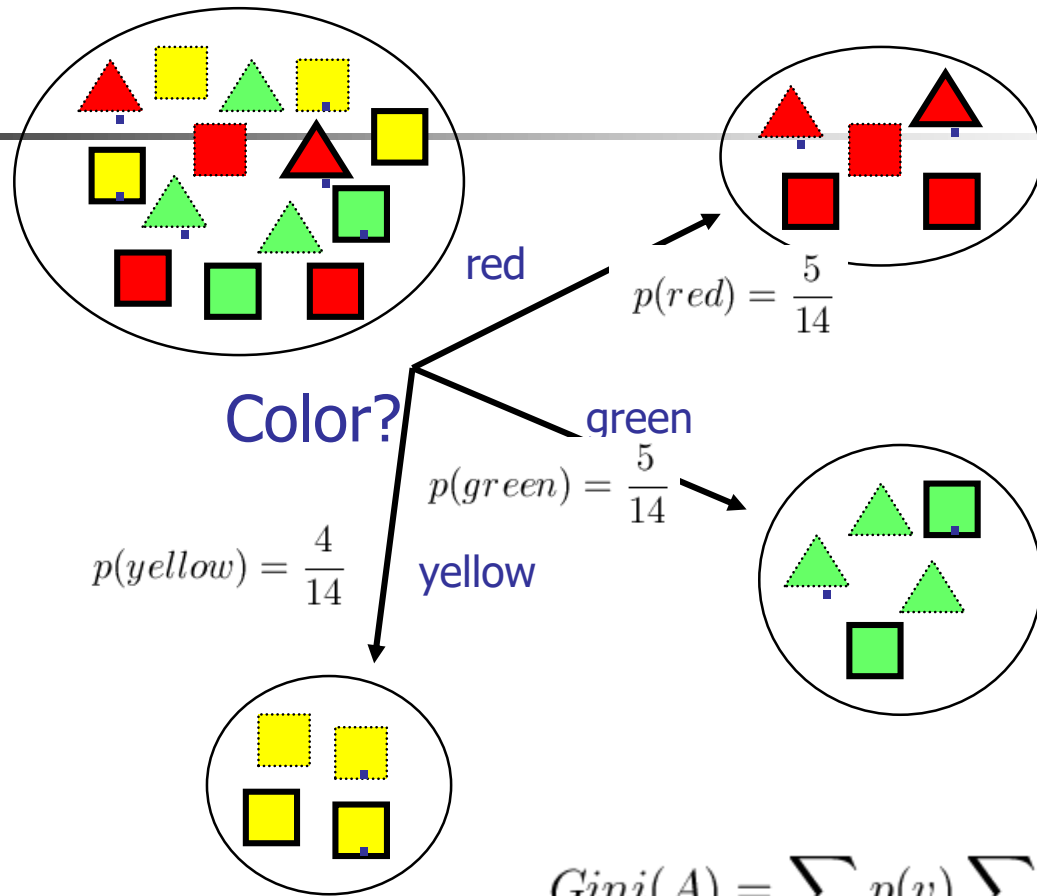
$$p(\square) = \frac{9}{14}$$

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

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

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

Gini Index for Color



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

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

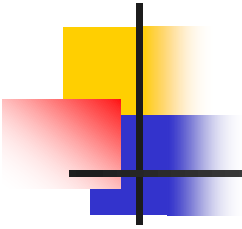


Gain of Gini Index

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

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

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



- Used by the CART (classification and regression tree) algorithm.
- Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it were randomly labeled according to the distribution of labels in the subset.
- Gini impurity can be computed by summing the probability of each item being chosen times the probability of a mistake in categorizing that item.
- It reaches its minimum (zero) when all cases in the node fall into a single target category.



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



Occam's Razor

"If two theories explain the facts equally well, then the simpler theory is to be preferred"

Arguments in favor:

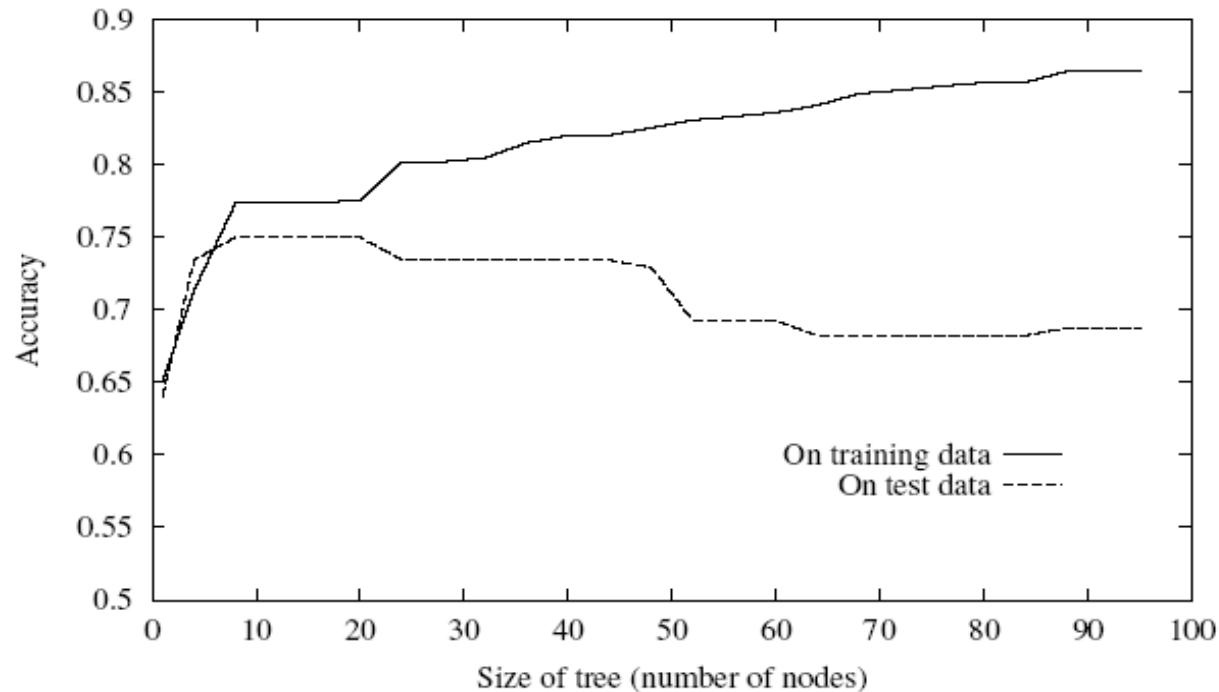
- Fewer short hypotheses than long hypotheses
- A short hypothesis that fits the data is unlikely to be a coincidence
- A long hypothesis that fits the data might be a coincidence

Arguments opposed:

- There are many ways to define small sets of hypotheses

Overfitting in Decision Tree

- One of the biggest problems with decision trees is **Overfitting**





Avoid Overfitting

How can we avoid overfitting?

- Stop growing when data split not statistically significant
- Grow full tree then post-prune

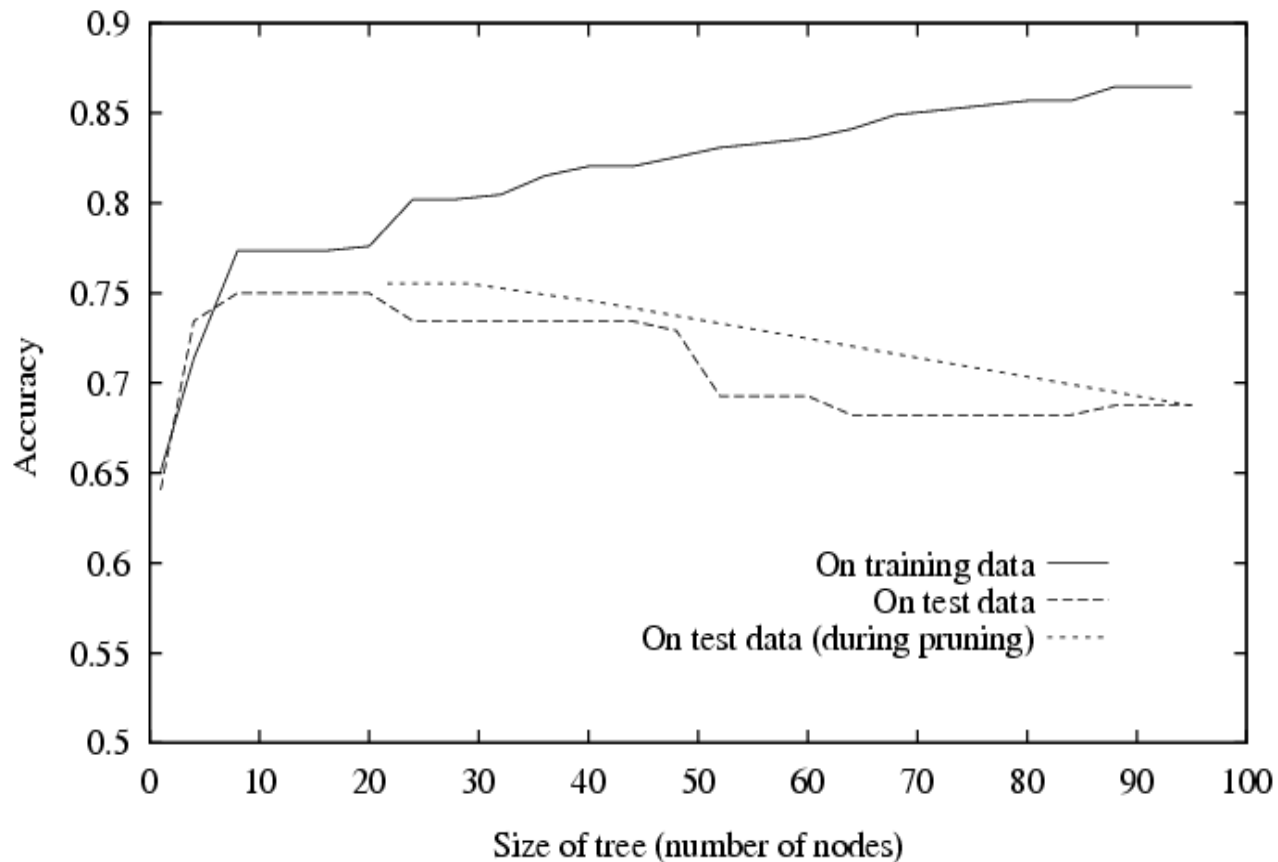
Select “best” tree:

- measure performance over separate validation data set(training data)
- $\min(|\text{tree}| + |\text{misclassifications}(\text{tree})|)$
- Minimum description length (MDL):

Minimize:

$\text{size}(\text{tree}) + \text{size}(\text{misclassifications}(\text{tree}))$

Effect of Reduced Error Pruning





Unknown Attribute Values

What if some examples have missing values of A ?

Use training example anyway sort through tree

- If node n tests A , assign most common value of A among other examples sorted to node n .
- Assign most common value of A among other examples with same target value
- Assign probability p_i to each possible value v_i of A
 - Assign fraction p_i of example to each descendant in tree

Classify new examples in the same fashion



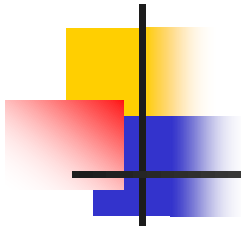
Cross-Validation

- Estimate the accuracy of an hypothesis induced by a supervised learning algorithm
- Predict the accuracy of an hypothesis over future unseen instances
- Select the optimal hypothesis from a given set of alternative hypotheses
 - Pruning decision trees
 - Model selection
 - Feature selection
- Combining multiple classifiers (boosting)

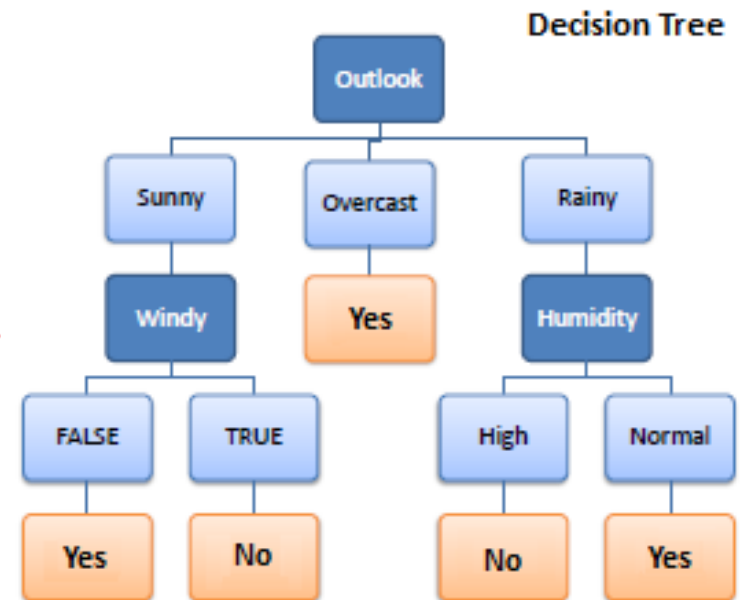


Decision Tree - Exercise

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.



Predictors				Target
Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No





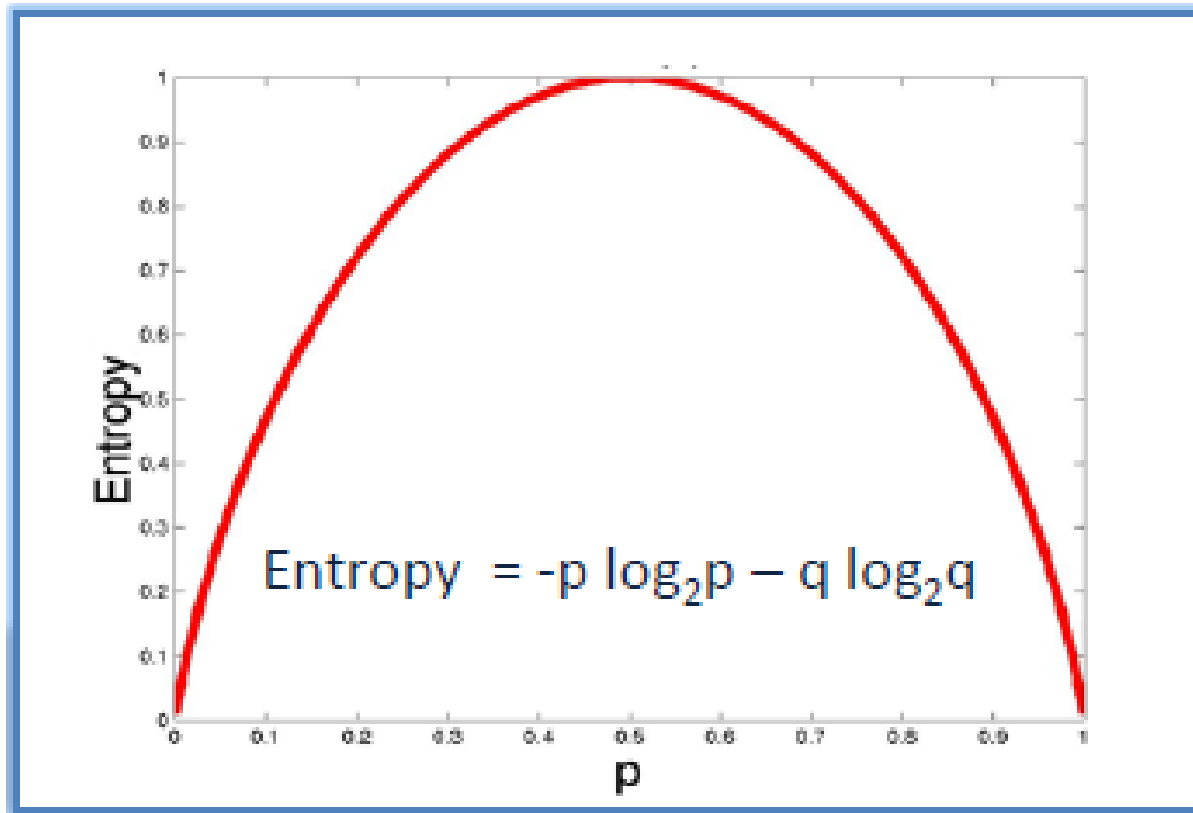
Algorithm

The core algorithm for building decision trees called **ID3** by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. ID3 uses *Entropy* and *Information Gain* to construct a decision tree. In ZeroR model there is no predictor, in OneR model we try to find the single best predictor, naive Bayesian includes all predictors using Bayes' rule and the independence assumptions between predictors but decision tree includes all predictors with the dependence assumptions between predictors.



Entropy

A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). ID3 algorithm uses entropy to calculate the homogeneity of a sample. If the sample is completely homogeneous the entropy is zero and if the sample is an equally divided it has entropy of one.



$$\text{Entropy} = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$$



To build a decision tree, we need to calculate two types of entropy using frequency tables as follows:

- a) Entropy using the frequency table of one attribute:

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

Play Golf	
Yes	No
9	5



$$\begin{aligned}\text{Entropy}(\text{PlayGolf}) &= \text{Entropy}(5,9) \\ &= \text{Entropy}(0.36, 0.64) \\ &= - (0.36 \log_2 0.36) - (0.64 \log_2 0.64) \\ &= 0.94\end{aligned}$$

- 
- b) Entropy using the frequency table of two attributes:

$$E(T, X) = \sum_{c \in X} P(c)E(c)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14



$$\begin{aligned} E(\text{PlayGolf}, \text{Outlook}) &= P(\text{Sunny}) * E(3,2) + P(\text{Overcast}) * E(4,0) + P(\text{Rainy}) * E(2,3) \\ &= (5/14) * 0.971 + (4/14) * 0.0 + (5/14) * 0.971 \\ &= 0.693 \end{aligned}$$

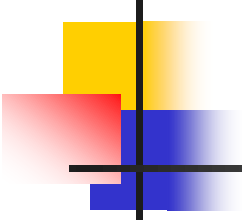



Information Gain

The information gain is based on the decrease in entropy after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches).

- *Step 1*: Calculate entropy of the target.

$$\begin{aligned}\text{Entropy}(\text{PlayGolf}) &= \text{Entropy}(5,9) \\ &= \text{Entropy}(0.36, 0.64) \\ &= - (0.36 \log_2 0.36) - (0.64 \log_2 0.64) \\ &= 0.94\end{aligned}$$

- 
-
- *Step 2:* The dataset is then split on the different attributes. The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split. The resulting entropy is subtracted from the entropy before the split. The result is the Information Gain, or decrease in entropy.



		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

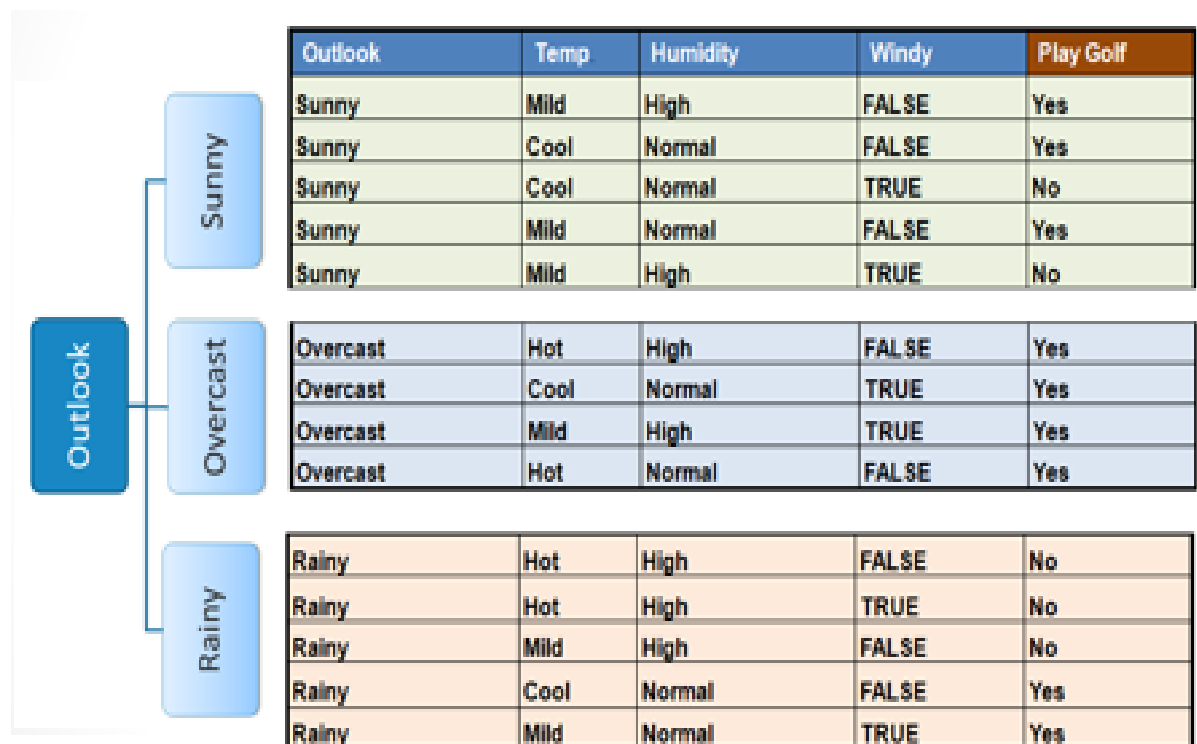
		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

$$\begin{aligned}
 G(\text{PlayGolf}, \text{Outlook}) &= E(\text{PlayGolf}) - E(\text{PlayGolf}, \text{Outlook}) \\
 &= 0.940 - 0.693 = 0.247
 \end{aligned}$$

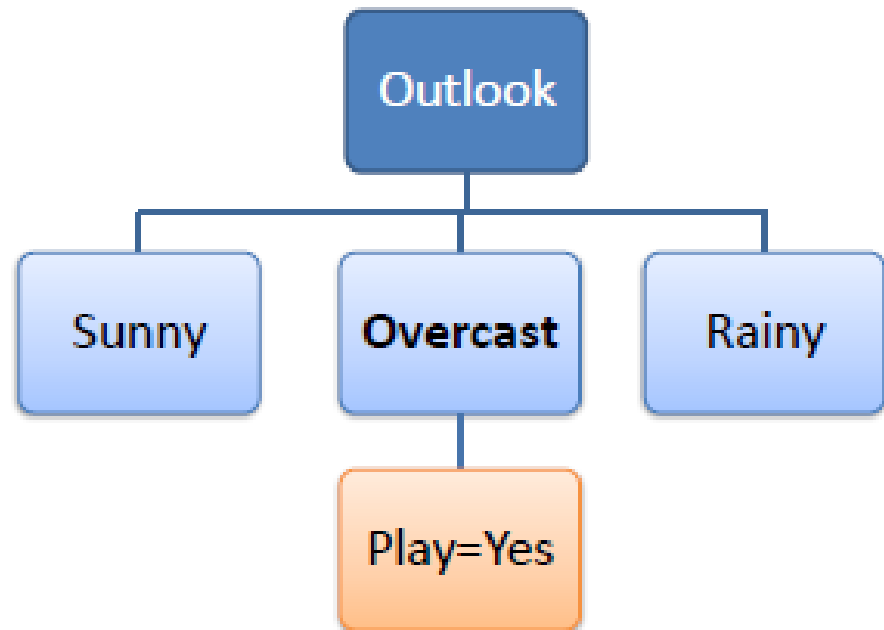
- *Step 3:* Choose attribute with the largest information gain as the decision node, divide the dataset by its branches and repeat the same process on every branch.

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			



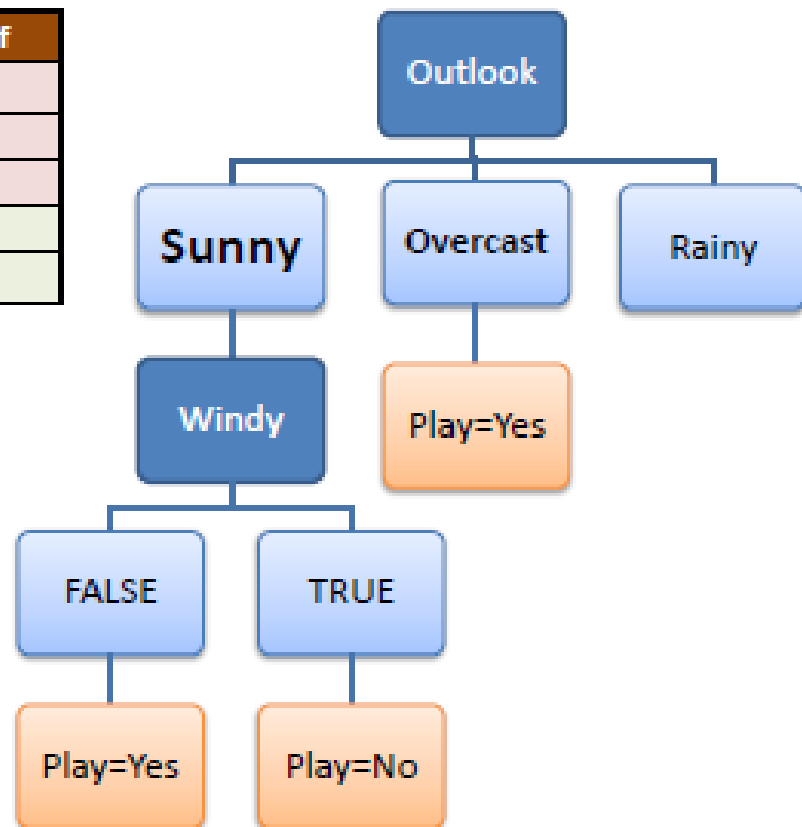
- 
- *Step 4a:* A branch with entropy of 0 is a leaf node.

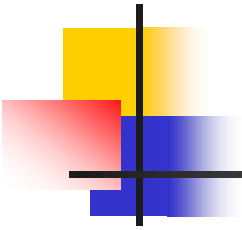
Temp.	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes



- *Step 4b:* A branch with entropy more than 0 needs further splitting.

Temp.	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No





- *Step 5:* The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified.

Decision Tree to Decision Rules

A decision tree can easily be transformed to a set of rules by mapping from the root node to the leaf nodes one by one.

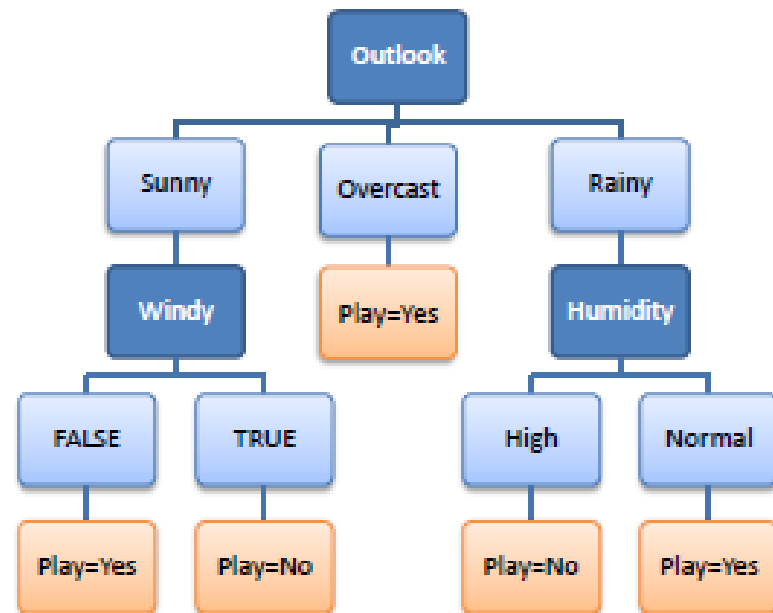
R_1 : IF (Outlook=Sunny) AND
(Windy=FALSE) THEN Play=Yes

R_2 : IF (Outlook=Sunny) AND
(Windy=TRUE) THEN Play=No

R_3 : IF (Outlook=Overcast) THEN
Play=Yes

R_4 : IF (Outlook=Rainy) AND
(Humidity=High) THEN Play=No

R_5 : IF (Outlook=Rain) AND
(Humidity=Normal) THEN
Play=Yes





Reference:

- Alpaydin, Machine Learning, 2010, 2nd Press
- Dr. Lee's Slides, San Jose State University, Spring 2007
- "Building Decision Trees with the ID3 Algorithm", by: Andrew Colin, Dr. Dobbs Journal, June 1996
- "Incremental Induction of Decision Trees", by Paul E. Utgoff, Kluwer Academic Publishers, 1989
- <http://www.cise.ufl.edu/~ddd/cap6635/Fall-97/Short-papers/2.htm>
- <http://decisiontrees.net/node/27>
- Tom M. Mitchell, Machine Learning, McGraw-Hill, 1997
- Barros R. C., Cerri R., Jaskowiak P. A., Carvalho, A. C. P. L. F., A bottom-up oblique decision tree induction algorithm (<http://dx.doi.org/10.1109/ISDA.2011.6121697>). Proceedings of the 11th International
- Breiman, Leo; Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and regression tree Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software. ISBN 978-0-412-04841-8.
- Barros, Rodrigo C., Basgalupp, M. P., Carvalho, A. C. P. L. F., Freitas, Alex A. (2011). A Survey of Evolutionary Algorithms for Decision-Tree Induction (http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5928432). IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, vol. 42, n. 3, p. 291-312, May 2012.
- https://www.saedsayad.com/decision_tree.htm