بسم الله الرحمن الرحيم

Summary & Some Definitions

Inductive Learning/Prediction

- Supervised, Unsupervised
- In supervised learning given examples are in (x,y) or (x, f(x)) pairs. (labelled data)
- In classification problems, y or f(x) is discrete
- In regression problems, y or f(x) is continuous
- In probability estimation problems, y or f(x) will be p(x)

Representation

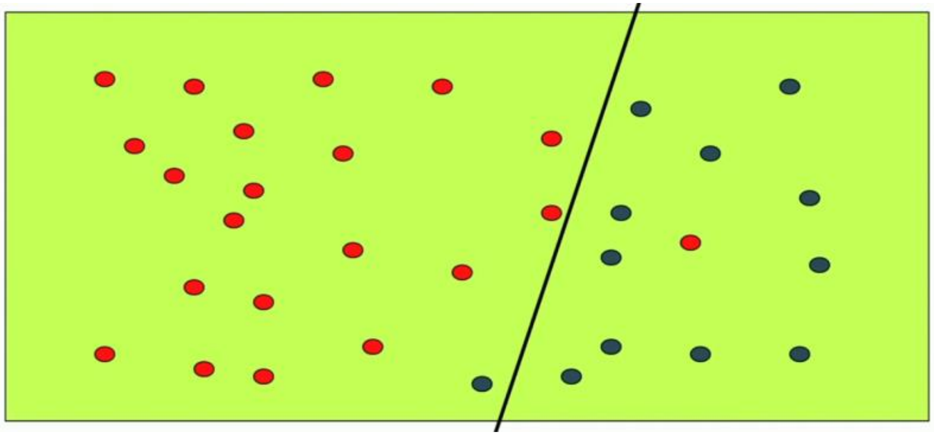
- There are two things that we need to describe a function.

  - Features
  - Language

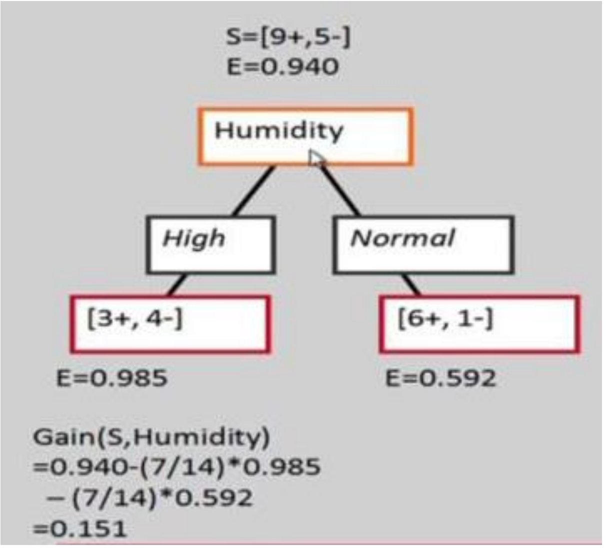
- Based on features and language, we can define our hypothesis space.

Representations

- ➢ Linear Function

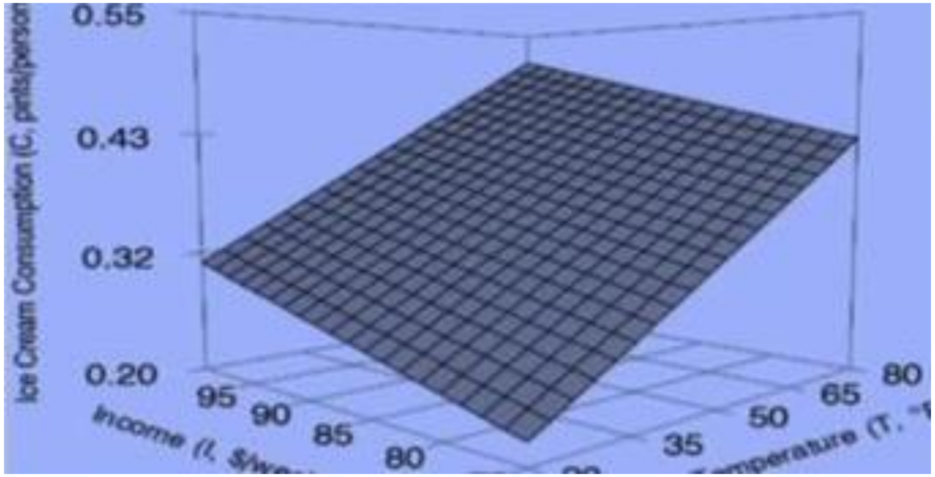


- ➢ Decision Tree

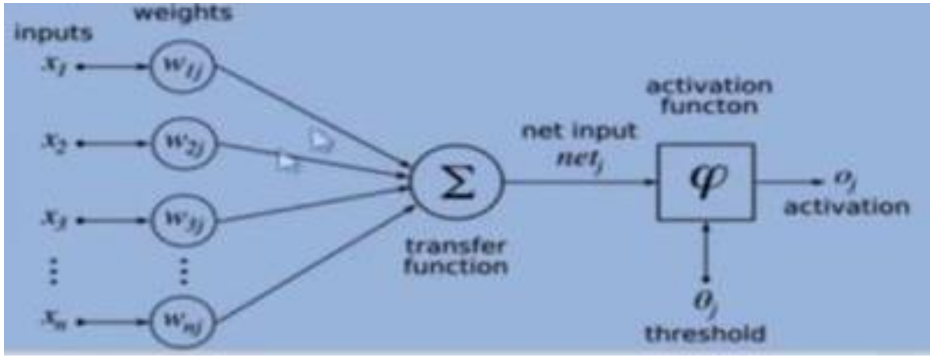


Representations

- Multivariate Linear Function

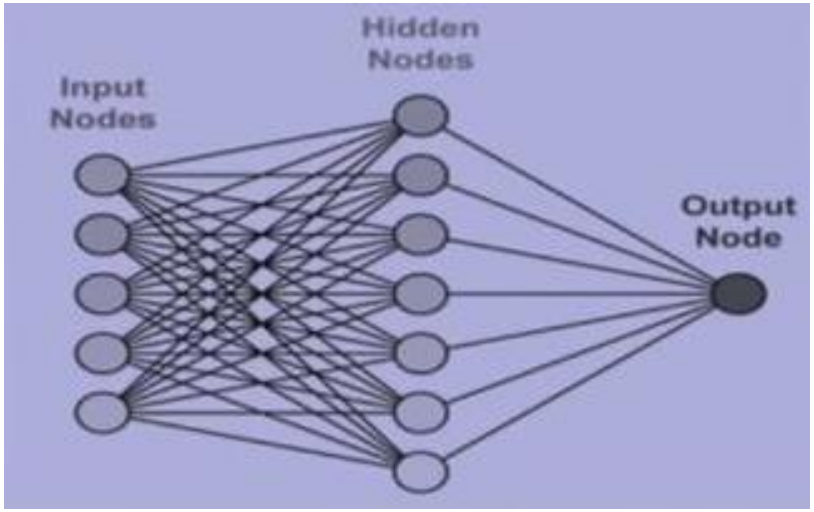


- Single layer Perceptron



Representations

- Multi-layer Neural Network



Hypothesis Space

- The space of all possible hypotheses that can in principle, be output by learning algorithm.

- We can think of supervised learning machine as a device that explores the “Hypothesis Space”.

  - Each setting of the parameters is different hypothesis about the function that maps input vectors to output vectors.

Terminology

- **Example (x,y):** instance x with label y = f(x)
- **Training Data S:** collection of examples observed by learning algorithm
- **Instance Space X**: set of all possible objects that can be described by the features.
- **Target Function f**: Maps each instance x to y

Classifier

- **Hypothesis h:** function that approximates f
- **Hypothesis Space H:** set of functions we allow for approximating f
- The set of hypothesis that can be produced, may be restricted further by specifying a **language bias**.
- **Input:** Training set $S \subseteq X$
- **Output:** A hypothesis $h \in H$

Underfitting & Overfitting

- **Underfitting:** Model is too "**simple**" to represent all the relevant class characteristics.

    - High bias & low variance
    - High Training error and high test error.

- **Overfitting:** Model is too "**complex**" and it fits irrelevant characteristics (noise) in the data.

    - Low bias & high variance
    - Low Training error and high test error.

Decision Trees

Decision Trees

- A tree structured Classifier which contains
  - Decision nodes : Each internal or decision node tests one discrete valued attribute or feature x_j.
  - Branch from a node selects one value for x_j.
  - Leaf nodes predicts y

Example:

Make a decision tree to approve or disapprove a loan based on Employment (Y/N), Credit Score (H/L) and/or Income (H/L) of the client.

Function Approximation

Problem Setting

- Set of possible instances $\mathcal{X}$
- Set of possible labels $\mathcal{Y}$
- Unknown target function $f : \mathcal{X} \to \mathcal{Y}$
- Set of function hypotheses $H = \{h \mid h : \mathcal{X} \to \mathcal{Y}\}$

Input: Training examples of unknown target function f

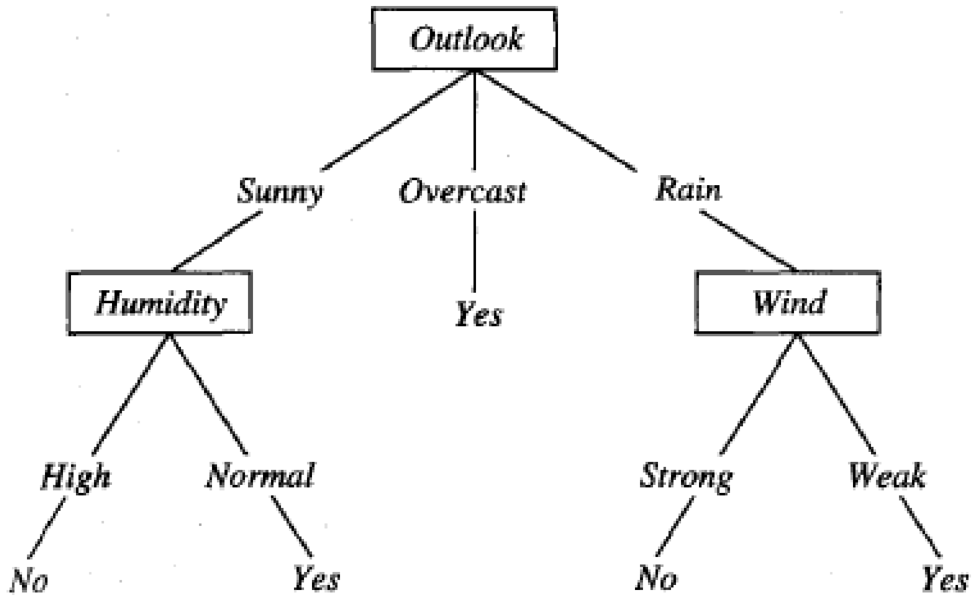
$$\{\langle \boldsymbol{x}_i, y_i \rangle\}_{i=1}^{n} = \{\langle \boldsymbol{x}_1, y_1 \rangle, \ldots, \langle \boldsymbol{x}_n, y_n \rangle\}$$

Output: Hypothesis $h \in H$ that best approximates f

Sample Data Set

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
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	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	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

Possible Decision Tree for the Data



- Decision nodes : Each internal or decision node tests one discrete valued attribute or feature x_j.
- Branch from a node selects one value for x_j.
- Leaf nodes predicts y

<outlook=sunny, temperature=hot, humidity=high, wind=weak> ?

Decision Tree Learning

Problem Setting

- Set of possible instances X
  - Each instance x in X is a feature vector
  - e.g <Humidity = low, wind = weak, temp = hot, outlook = rain>
- Unknown target function f : X→Y
  - Y = 1, if the person plays tennis otherwise 0.
- Set of function hypotheses H = {h|h:X →Y}
  - Each hypothesis is a decision tree

Inputs: Training Examples

Outputs: $h \in H$ that best approximates f

Decision Trees

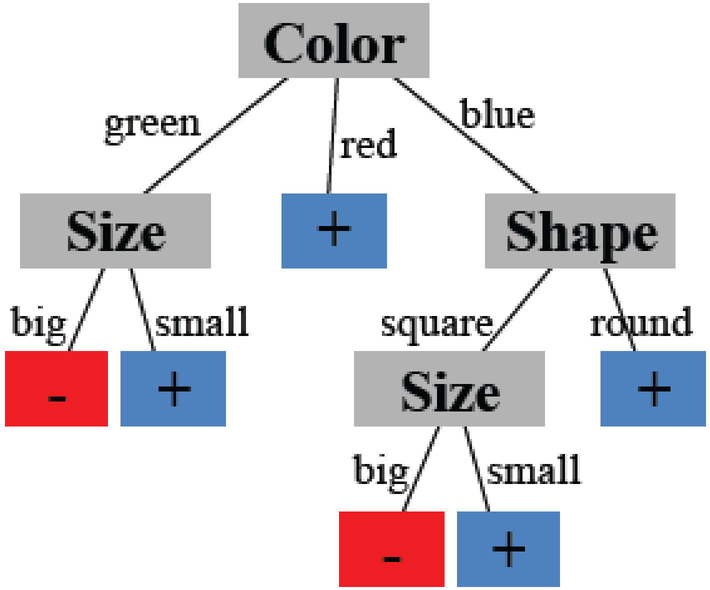
Suppose X= <x_1, x_2, …….x_d>, where x_j are Boolean valued variables.

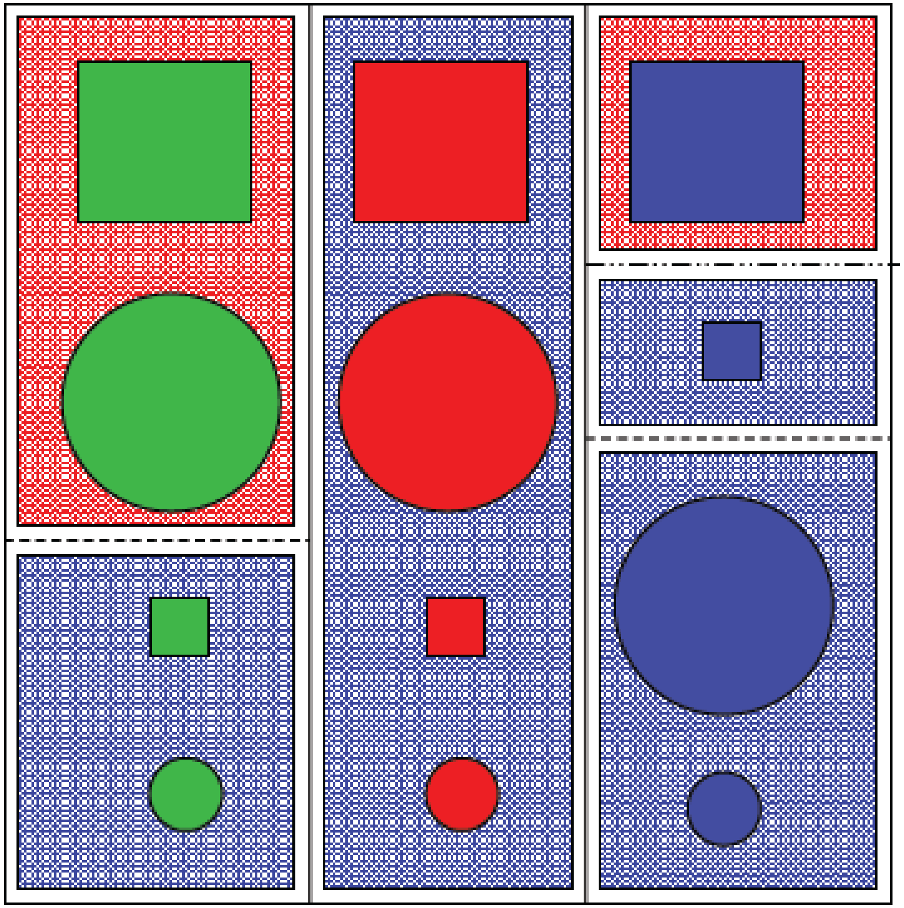
How would you represent following functions using decision trees?

- $y = x_2 x_3$
- $y = x_2 \vee x_3$
- $y = x_2 x_3 \vee x_4 x_5 (\neg x_1)$

Can we represent any arbitrary Boolean Function with decision tree or only few?

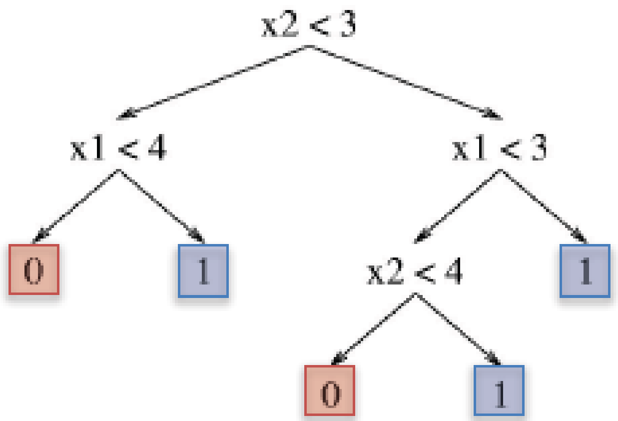
Decision Tree Induced Partition





Decision Tree Induced Partition

- Decision trees divide the feature space into axis-parallel (hyper-)rectangles



Top-Down Induction of Decision Trees

[ID3, C4.5 by Quinlan]

node = root of decision tree

Main loop:

1. $A \leftarrow$ the "best" decision attribute for the next node.
2. Assign A as decision attribute for *node*.
3. For each value of A, create a new descendant of *node*.
4. Sort training examples to leaf nodes.
5. If training examples are perfectly classified, stop. Else, recurse over new leaf nodes.

How to choose the best attribute?

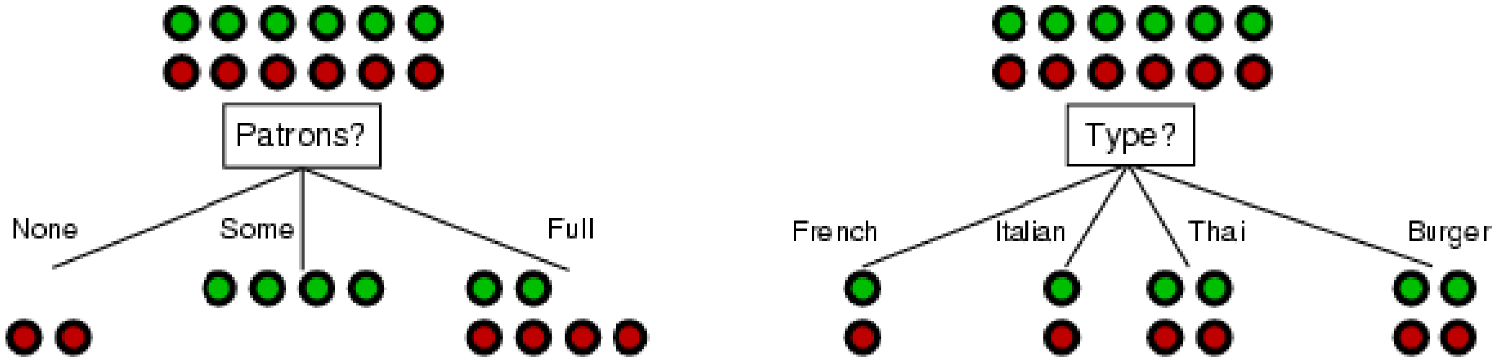
Choosing the Best Attribute

Key problem: choosing which attribute to split a given set of examples

- Some possibilities are:
  - **Random:** Select any attribute at random
  - **Least-Values:** Choose the attribute with the smallest number of possible values
  - **Most-Values:** Choose the attribute with the largest number of possible values
  - **Max-Gain:** Choose the attribute that has the largest expected *information gain*
    - i.e., attribute that results in smallest expected size of subtrees rooted at its children
- The ID3 algorithm uses the Max-Gain method of selecting the best attribute

Choosing an Attribute

Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"

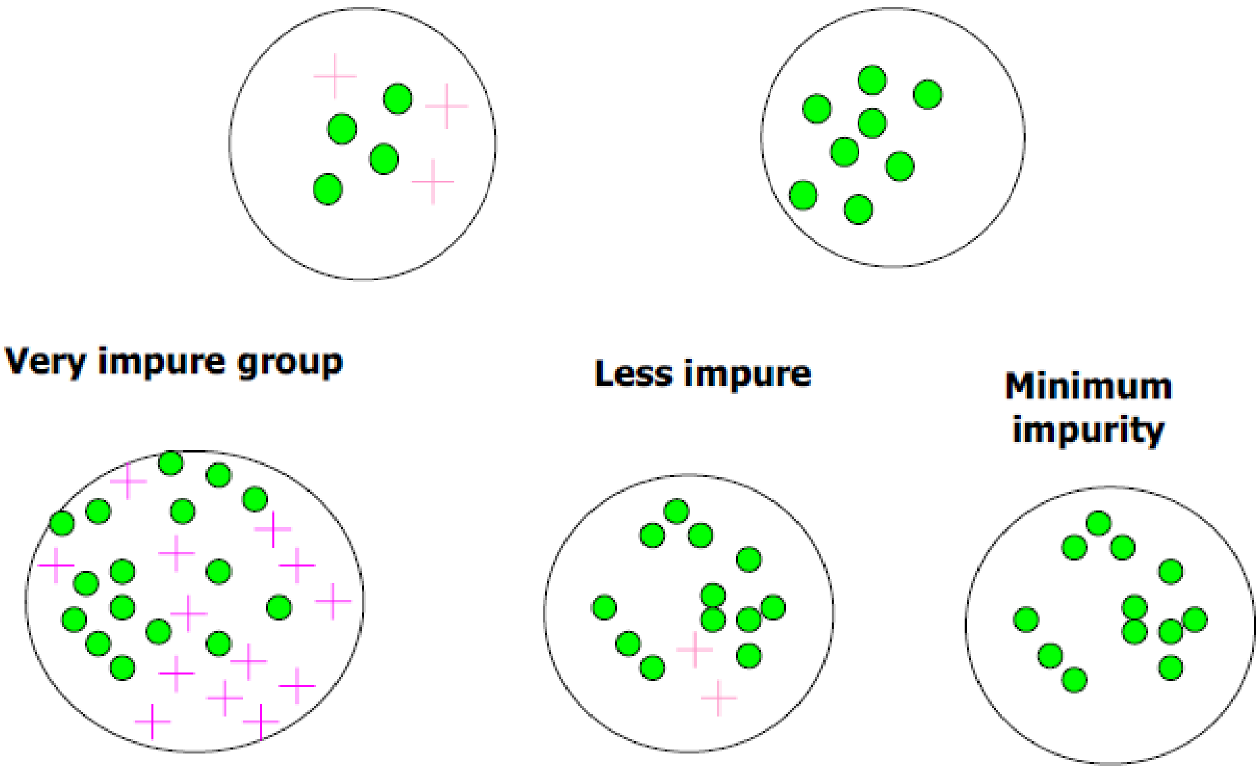


Which split is more informative: *Patrons?* or *Type?*

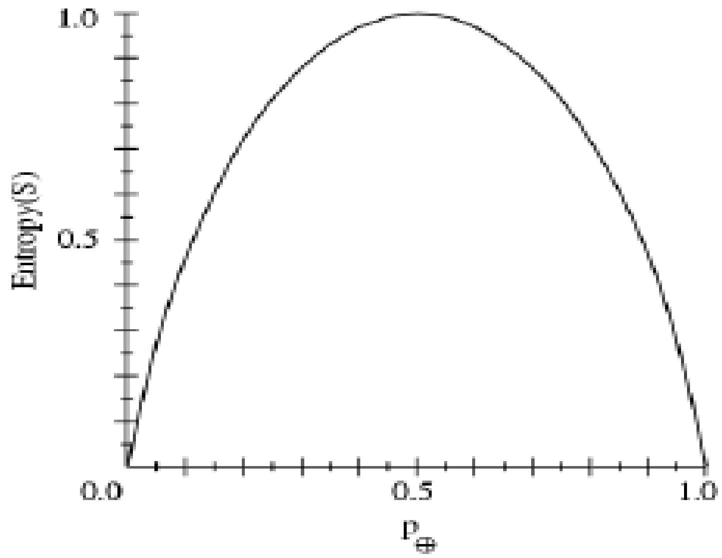
Entropy

Impurity/Entropy (informal)

– Measures the level of **impurity** in a group of examples



Sample Entropy



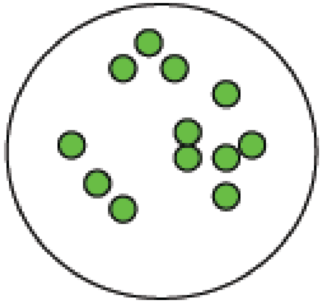
- S is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in S
- $p_{\ominus}$ is the proportion of negative examples in S
- Entropy measures the impurity of S

$$H(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

Sample Entropy

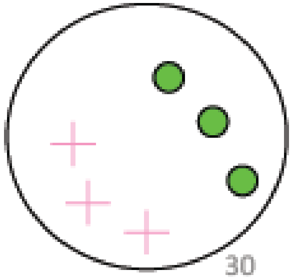
- What is the entropy of a group in which all examples belong to the same class?
  - entropy = $- 1 \log_2 1 = 0$

Minimum impurity



- What is the entropy of a group with 50% in either class?
  - entropy = $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

Maximum impurity



Information Gain

Information gain is the expected reduction in entropy caused by partitioning the examples according to some attribute. The information gain, Gain (S,A) of an attribute A, relative to collection of examples S is given as:

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Information Gain

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- We want to determine which attribute in a given set of training feature vectors is most useful for discriminating between the classes to be learned.

- Information gain tells us how important a given attribute of the feature vectors is.

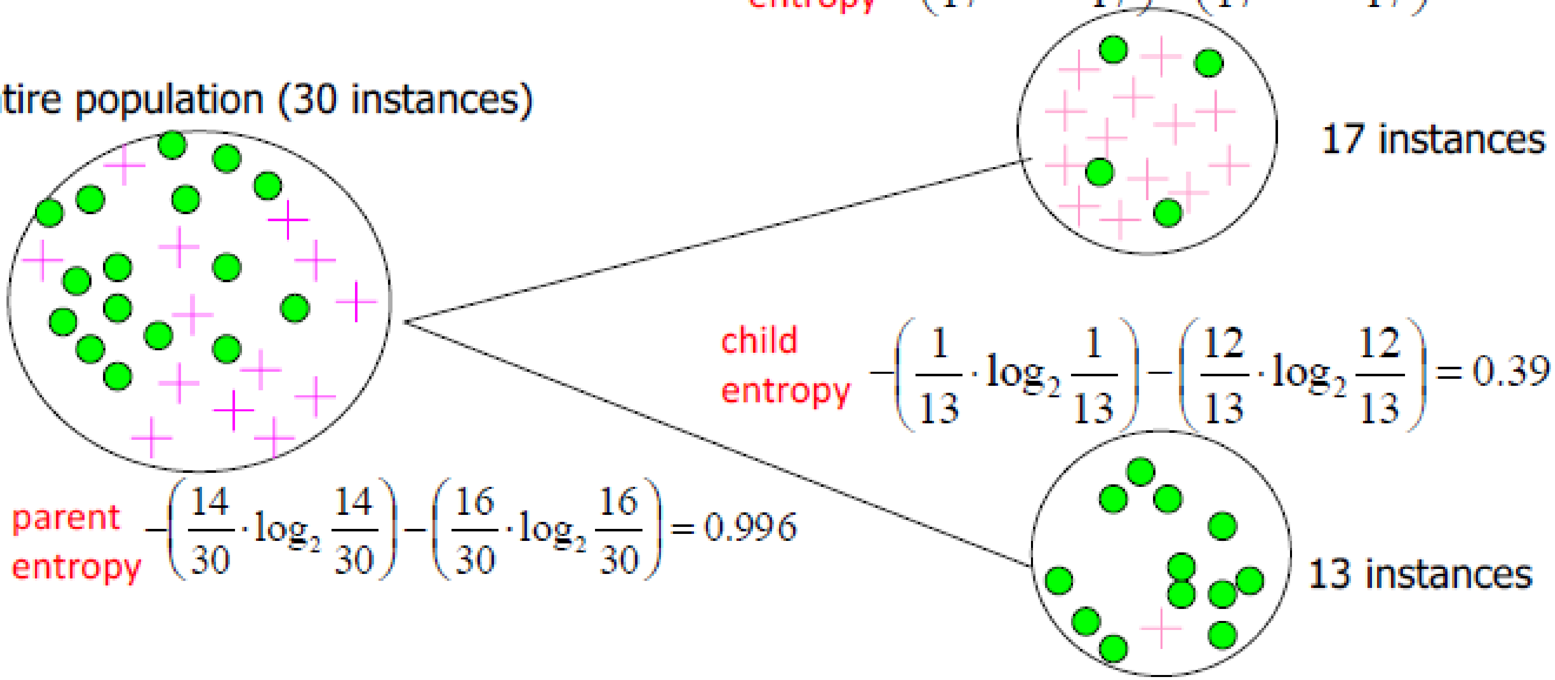
- We will use it to decide the ordering of attributes in the nodes of a decision tree.

Calculating Information Gain

Information Gain = entropy(parent) – [average entropy(children)]

child entropy $-\left(\frac{13}{17}\cdot\log_2\frac{13}{17}\right)-\left(\frac{4}{17}\cdot\log_2\frac{4}{17}\right)=0.787$

Entire population (30 instances)



child entropy $-\left(\frac{1}{13}\cdot\log_2\frac{1}{13}\right)-\left(\frac{12}{13}\cdot\log_2\frac{12}{13}\right)=0.391$

parent entropy $-\left(\frac{14}{30}\cdot\log_2\frac{14}{30}\right)-\left(\frac{16}{30}\cdot\log_2\frac{16}{30}\right)=0.996$

(Weighted) Average Entropy of Children = $\left(\frac{17}{30}\cdot 0.787\right)+\left(\frac{13}{30}\cdot 0.391\right)=0.615$

Training Examples

Day	Outlook	Temp	Humidity	Wind	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	Strong	*Yes*
D8	Sunny	Mild	High	Weak	*No*
D9	Sunny	Cool	Normal	Weak	*Yes*
D10	Rain	Mild	Normal	Weak	*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*

Back

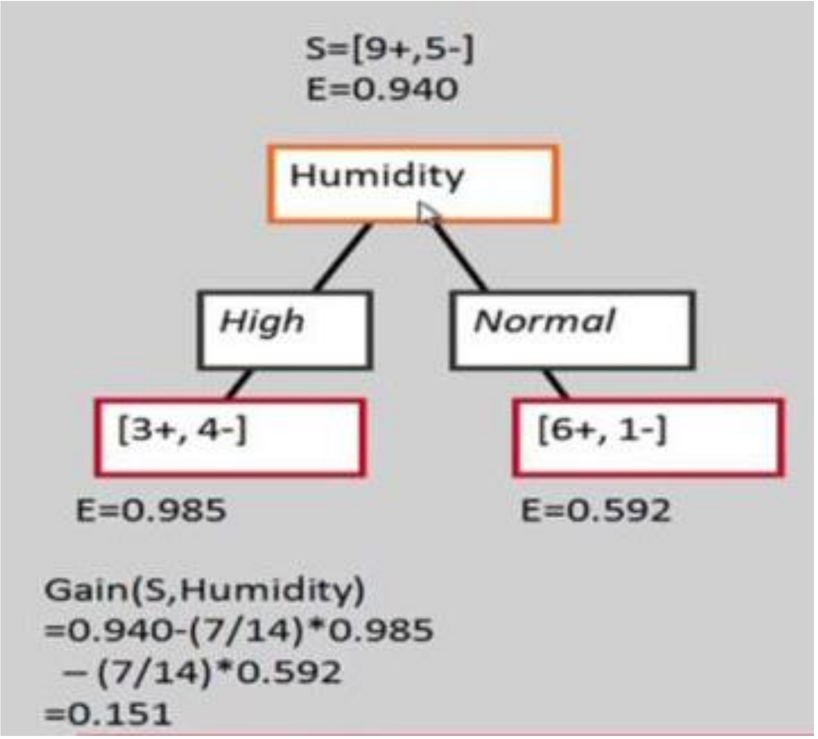
Back

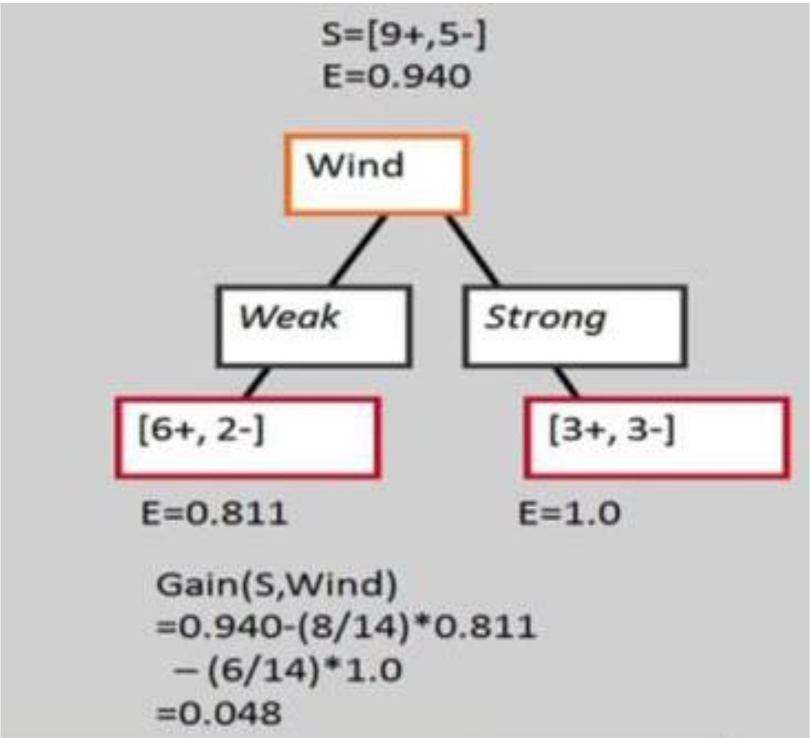
Back

Back

Back

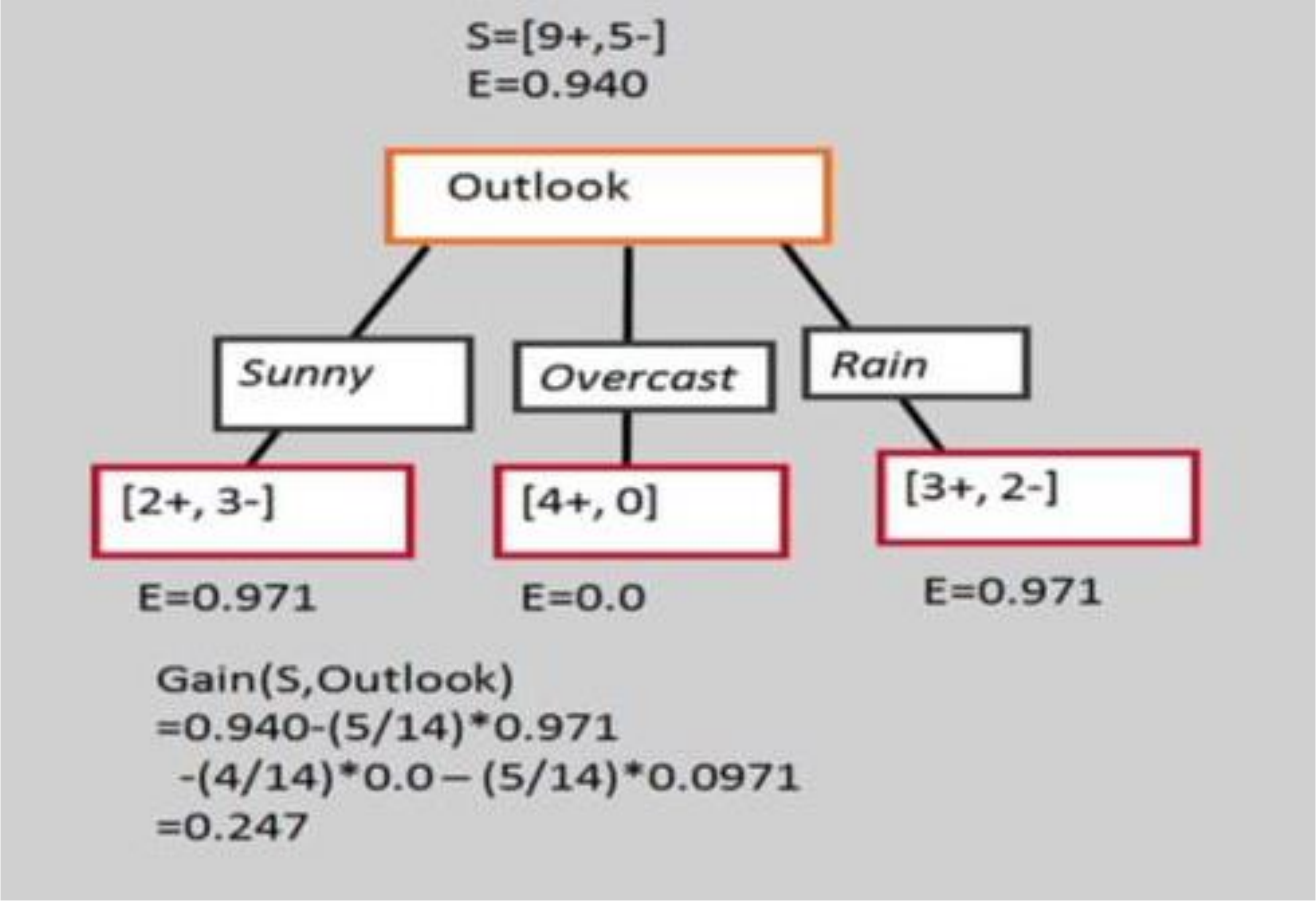
Selecting the Root Attribute





Data Set

Selecting the Root Attribute



Data Set

Selecting the Root Attribute

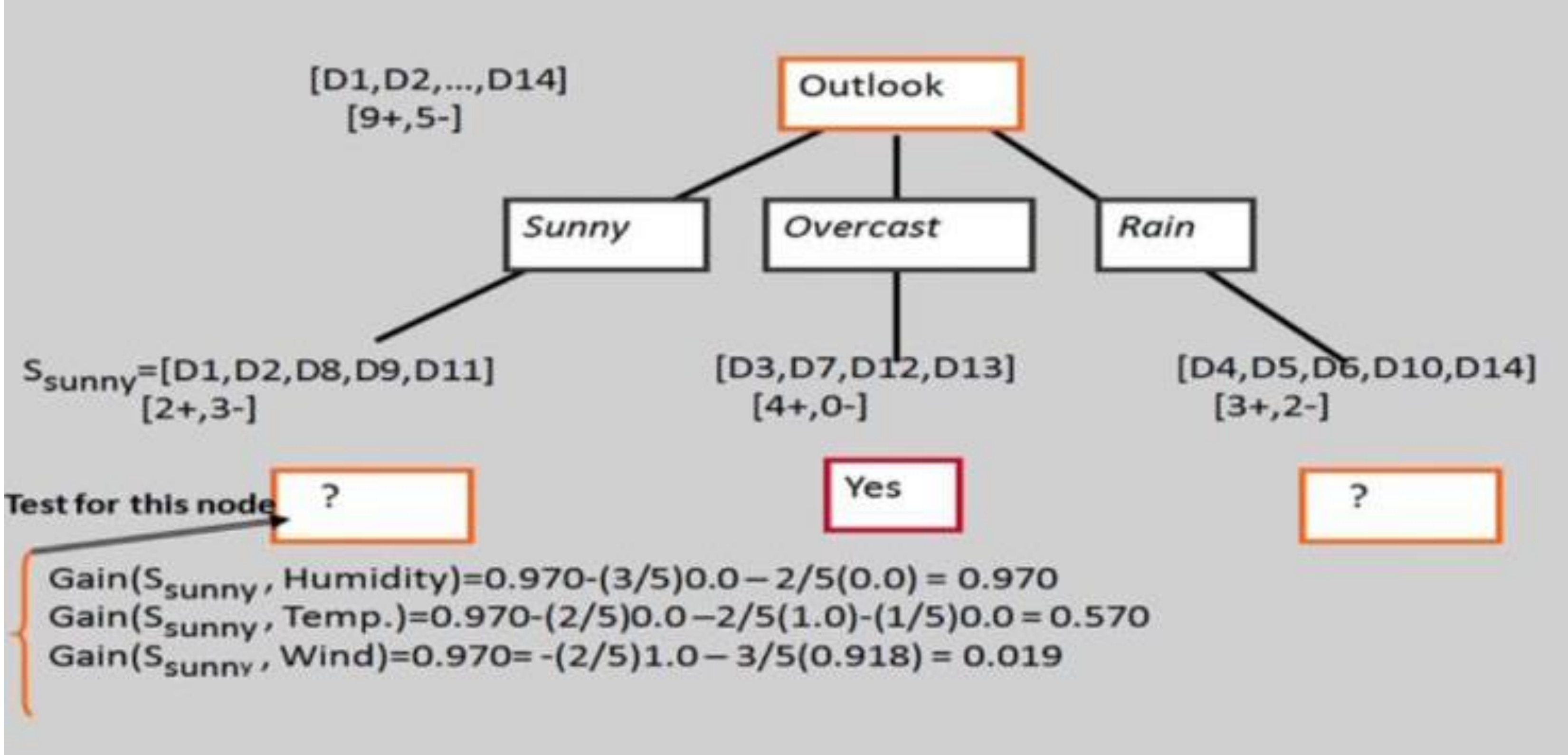
The information gain values for the 4 attributes are:

- Gain(S,Outlook) =0.247
- Gain(S,Humidity) =0.151
- Gain(S,Wind) =0.048
- Gain(S,Temperature) =0.029

where S denotes the collection of training examples

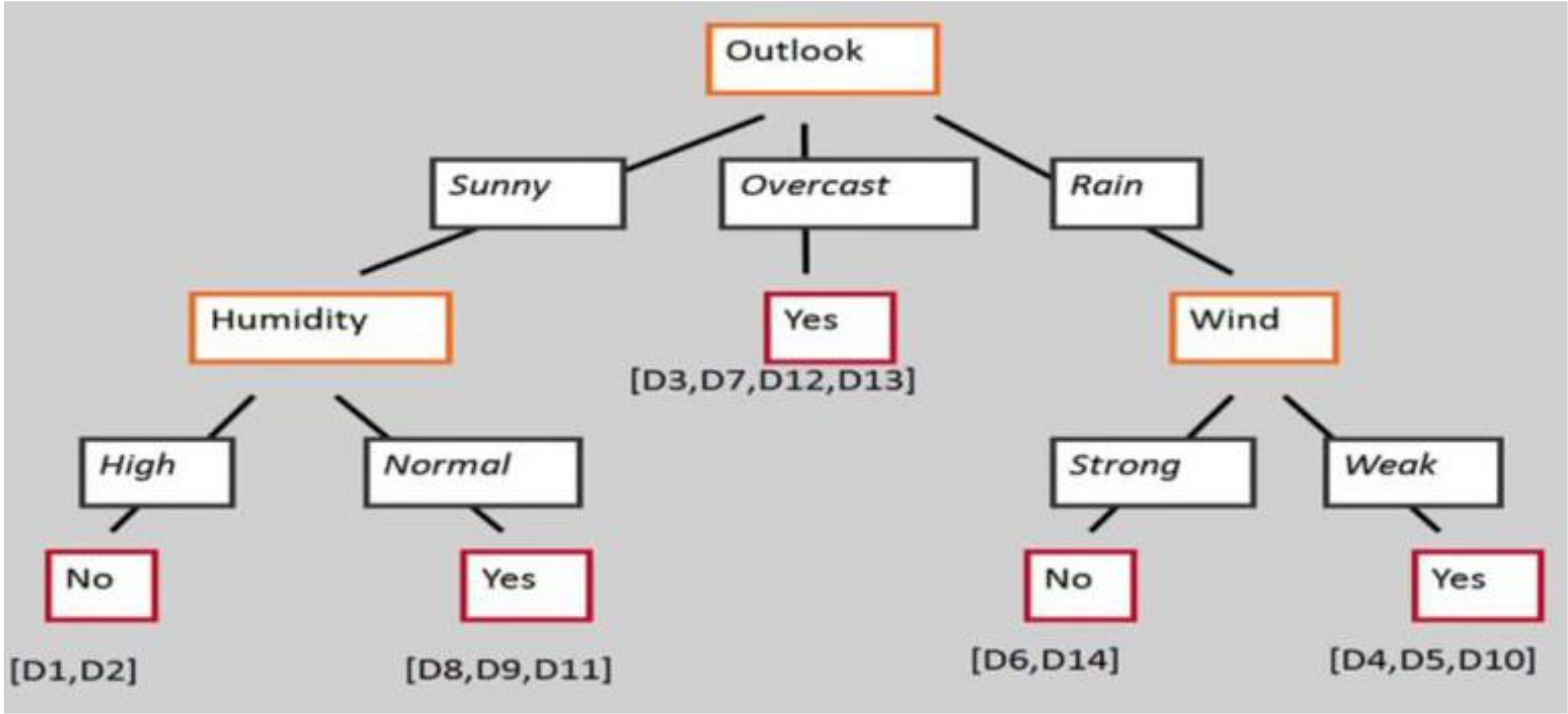
Data Set

Selecting the Next Attribute



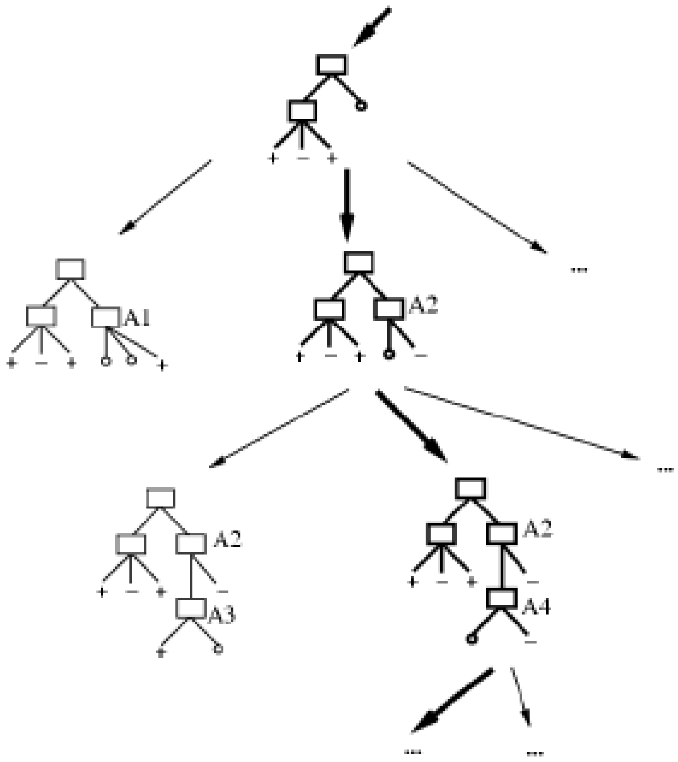
Data Set

Selecting the Next Attribute



Data Set

Which Tree Should We Output?



- ID3 performs heuristic search through space of decision trees
- It stops at smallest acceptable tree. Why?

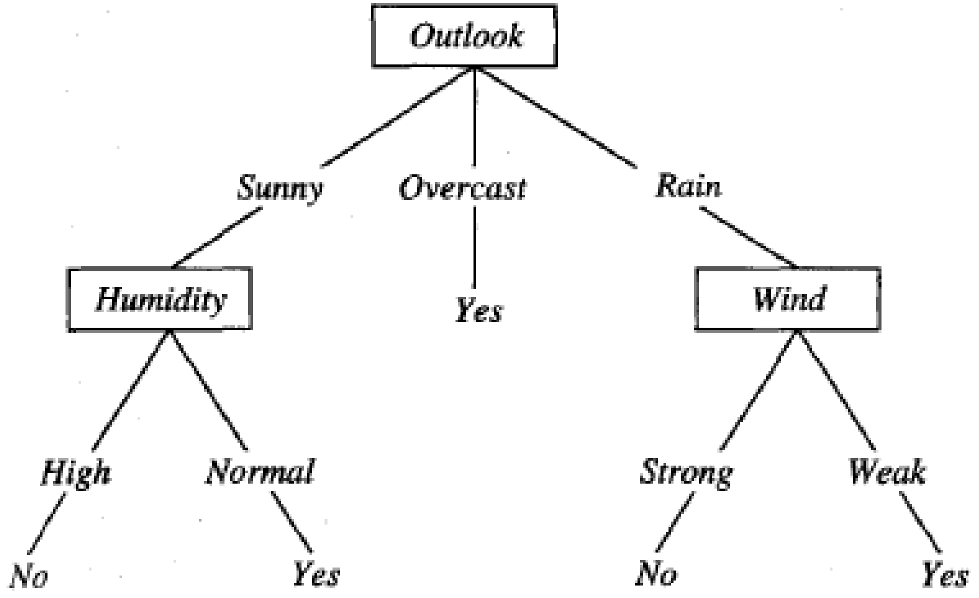
Occam's razor: prefer the simplest hypothesis that fits the data

Overfitting in Decision Trees

Consider adding noisy training example # 15

Sunny, Hot, Normal, Strong, Play Tennis = No

What effect on earlier tree?



Overfitting

Consider a hypothesis h and its

- ***Error rate over training data:*** $error_{train}(h)$
- ***True error rate over all data:*** $error_{true}(h)$

We say that h over fits the training data if

$$Error_{true}(h) > Error_{train}(h)$$

Amount of overfitting = $Error_{true}(h) - Error_{train}(h)$

Overfitting in Decision Trees