MACHINE LEARNING

Subject Code	17CS73	IA Marks 40
Number of Lecture Hours/Week	03	Exam Marks 60
Total Number of Lecture Hours	50	Exam Hours 03

Prof. Yogesh N
Assistant Professor
Department of ISE
Acharya Institute Of Technology

MODULE -2

DECISION TREE LEARNING

Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree.

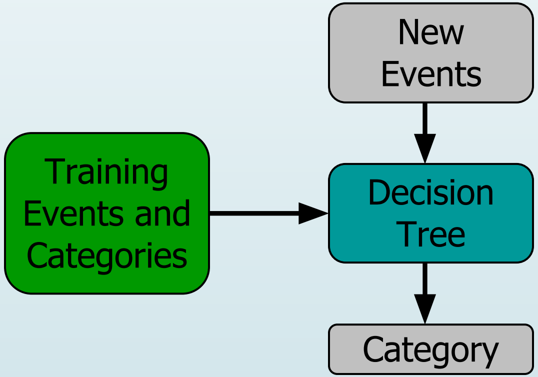
Introduction

Goal: Categorization

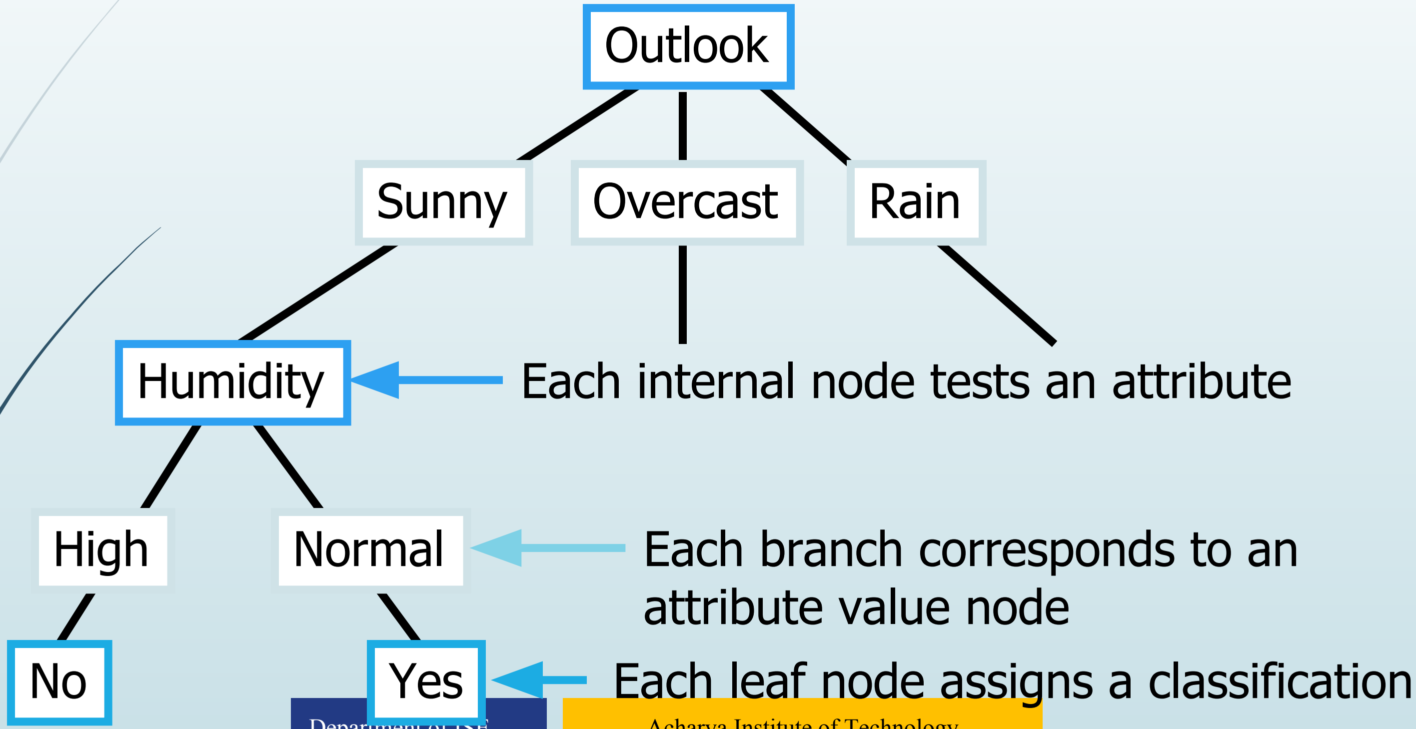
- Given an event, predict is category. Examples:
  - Who won a given ball game?
  - How should we file a given email?
  - What word sense was intended for a given occurrence of a word?

- Event = list of features. Examples:
  - Ball game: Which players were on offense?
  - Email: Who sent the email?
  - Disambiguation: What was the preceding word?

- Use a decision tree to predict categories for new events.
- Use training data to build the decision tree.



Decision Tree for PlayTennis



DECISION TREE REPRESENTATION

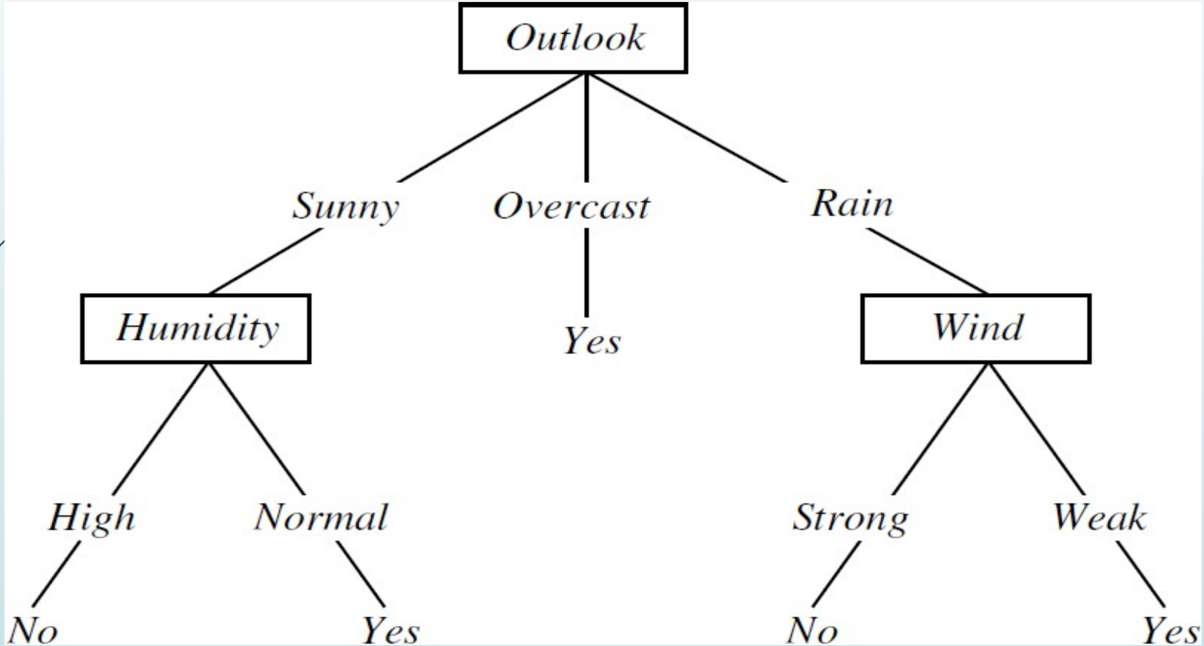


FIGURE: A decision tree for the concept ***PlayTennis***. An example is classified by sorting it through the tree to the appropriate leaf node, then returning the classification associated with this leaf

- Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance.

- Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute.

- An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the subtree rooted at the new node.

ACHARYA

- Decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances.
- Each path from the tree root to a leaf corresponds to a conjunction of attribute tests, and the tree itself to a disjunction of these conjunctions

For example,

The decision tree shown in above figure corresponds to the expression (Outlook = Sunny $\wedge$ Humidity = Normal)

$\vee$ (Outlook = Overcast)

$\vee$ (Outlook = Rain $\wedge$ Wind = Weak)

APPROPRIATE PROBLEMS FOR DECISION TREE LEARNING

Decision tree learning is generally best suited to problems with the following characteristics:

1. ***Instances are represented by attribute-value pairs*** – Instances are described by a fixed set of attributes and their values
2. ***The target function has discrete output values*** – The decision tree assigns a Boolean classification (e.g., yes or no) to each example. Decision tree methods easily extend to learning functions with more than two possible output values.
3. ***Disjunctive descriptions may be required***

ACHARYA

4. ***The training data may contain errors*** – Decision tree learning methods are robust to errors, both errors in classifications of the training examples and errors in the attribute values that describe these examples.

5. ***The training data may contain missing attribute values*** – Decision tree methods can be used even when some training examples have unknown values

- Decision tree learning has been applied to problems such as learning to classify *medical patients by their disease, equipment malfunctions by their cause*, and *loan applicants by their likelihood of defaulting on payments*.

- Such problems, in which the task is to classify examples into one of a discrete set of possible categories, are often referred to as ***classification problems.***

THE BASIC DECISION TREE LEARNING ALGORITHM

- Most algorithms that have been developed for learning decision trees are variations on a core algorithm that employs a top-down, greedy search through the space of possible decision trees. This approach is exemplified by the ID3 algorithm and its successor C4.5

What is the ID3 algorithm?

- ID3 stands for Iterative Dichotomiser 3
- ID3 is a precursor to the C4.5 Algorithm.
- The ID3 algorithm was invented by Ross Quinlan in 1975
- Used to generate a decision tree from a given data set by employing a top-down, greedy search, to test each attribute at every node of the tree.
- The resulting tree is used to classify future samples.

ID3 algorithm

ID3(Examples, Target_attribute, Attributes)

Examples are the training examples. Target_attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given Examples.

- Create a Root node for the tree
- If all Examples are positive, Return the single-node tree Root, with label = +
- If all Examples are negative, Return the single-node tree Root, with label = -
- If Attributes is empty, Return the single-node tree Root, with label = most common value of Target_attribute in Examples

- Otherwise Begin
  - A ← the attribute from Attributes that best* classifies Examples
  - The decision attribute for Root ← A
  - For each possible value, v_i, of A,
    - Add a new tree branch below *Root*, corresponding to the test A = v_i
    - Let *Examples* $_{vi}$, be the subset of Examples that have value v_i for *A*
    - If *Examples* $_{vi}$, is empty
      - Then below this new branch add a leaf node with label = most common value of Target_attribute in Examples
      - Else below this new branch add the subtree ID3(*Examples* $_{vi}$, Targe_tattribute, Attributes – {A}))
- End
- Return Root

* The best attribute is the one with highest information gain

Which Attribute Is the Best Classifier?

- The central choice in the ID3 algorithm is selecting which attribute to test at each node in the tree.
- A statistical property called ***information gain*** that measures how well a given attribute separates the training examples according to their target classification.
- ID3 uses ***information gain*** measure to select among the candidate attributes at each step while growing the tree.

ENTROPY MEASURES HOMOGENEITY OF EXAMPLES

- To define information gain, we begin by defining a measure called entropy.

Entropy measures the impurity of a collection of examples.

- Given a collection S, containing positive and negative examples of some target

concept, the entro

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

Where,

p_+ is the proportion of positive examples in S

p_- is the proportion of negative examples in S.

Example: Entropy

- Suppose S is a collection of 14 examples of some boolean concept, including 9 positive and 5 negative examples. Then the entropy of S relative to this boolean classification is

$$Entropy([9+, 5-]) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.940$$

ACHARYA

- The entropy is 0 if all members of S belong to the same class
- The entropy is 1 when the collection contains an equal number of positive and negative examples
- If the collection contains unequal numbers of positive and negative examples, the entropy is between 0 and 1

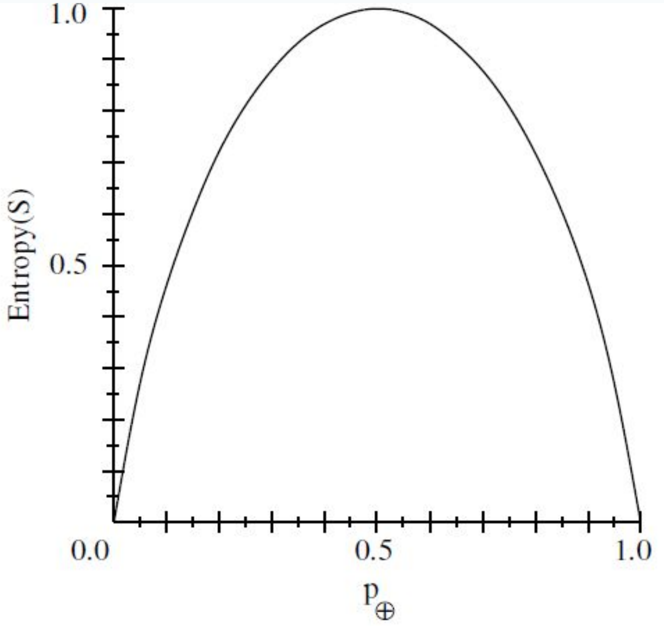


FIGURE The entropy function relative to a boolean classification, as the proportion, $p_\oplus$, of positive examples varies between 0 and 1.

INFORMATION GAIN MEASURES THE EXPECTED REDUCTION IN ENTROPY

- ***Information gain,*** is the expected reduction in entropy caused by partitioning the examples according to this attribute.
- The Information Gain(S, A) of an attribute A, relative to a collection of examples S, is defined as

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

Example: Information gain

Let, $Values(Wind) = \{Weak, Strong\}$

$S = [9+, 5-]$

$S_{Weak} = [6+, 2-]$

$S_{Strong} = [3+, 3-]$

Information gain of attribute *Wind*:

$Gain(S, Wind) = Entropy(S) - 8/14\ Entropy\ (S_{Weak}) - 6/14\ Entropy\ (S_{Strong})$

$= 0.94 - (8/14)*\ 0.811 - (6/14)\ *1.00$

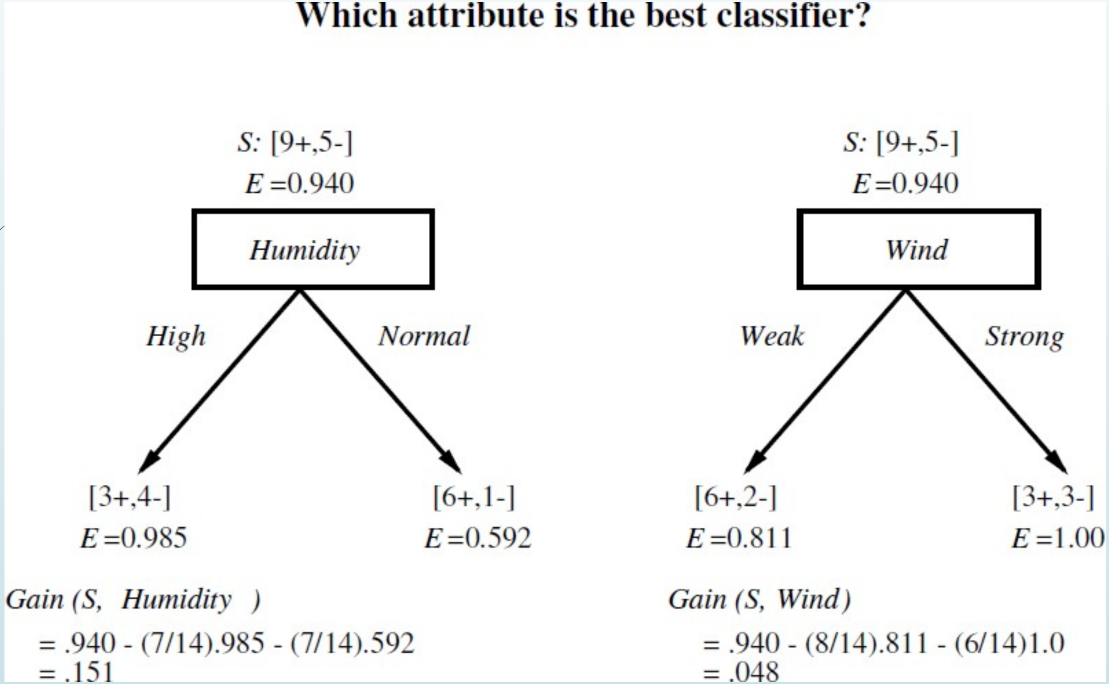
$= 0.048$

An Illustrative Example

- To illustrate the operation of ID3, consider the learning task represented by the training examples of below table.
- Here the target attribute ***PlayTennis***, which can have values ***yes*** or ***no*** for different days.
- Consider the first step through the algorithm, in which the topmost node of the decision tree is created.

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

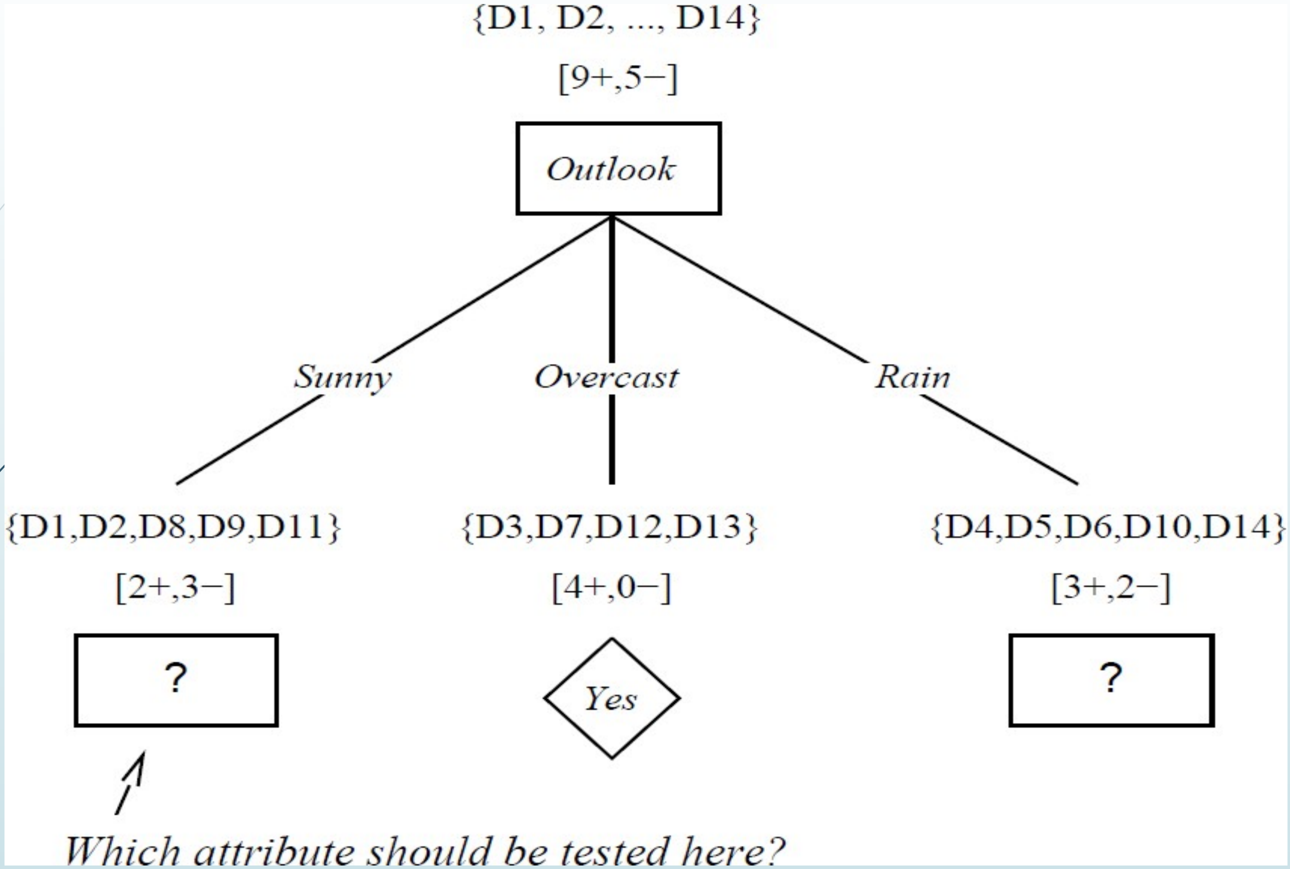
ID3 determines the information gain for each candidate attribute (i.e., Outlook, Temperature, Humidity, and Wind), then selects the one with highest information gain



The information gain values for all four attributes are

- Gain(S, Outlook) = 0.246
- Gain(S, Humidity) = 0.151
- Gain(S, Wind) = 0.048
- Gain(S, Temperature) = 0.029

- According to the information gain measure, the ***Outlook*** attribute provides the best prediction of the target attribute, ***PlayTennis***, over the training examples. Therefore, ***Outlook*** is selected as the decision attribute for the root node, and branches are created below the root for each of its possible values i.e., Sunny, Overcast, and Rain.


{D1, D2, ..., D14}
[9+,5−]
Outlook
Sunny
Overcast
Rain
{D1,D2,D8,D9,D11}
[2+,3−]
?
{D3,D7,D12,D13}
[4+,0−]
Yes
{D4,D5,D6,D10,D14}
[3+,2−]
?
Which attribute should be tested here?

S_{sunny} = {D1,D2,D8,D9,D11}

$$Gain\ (S_{sunny}, Humidity) = .970 - (3/5)\ 0.0 - (2/5)\ 0.0 = .970$$

$$Gain\ (S_{sunny}, Temperature) = .970 - (2/5)\ 0.0 - (2/5)\ 1.0 - (1/5)\ 0.0 = .570$$

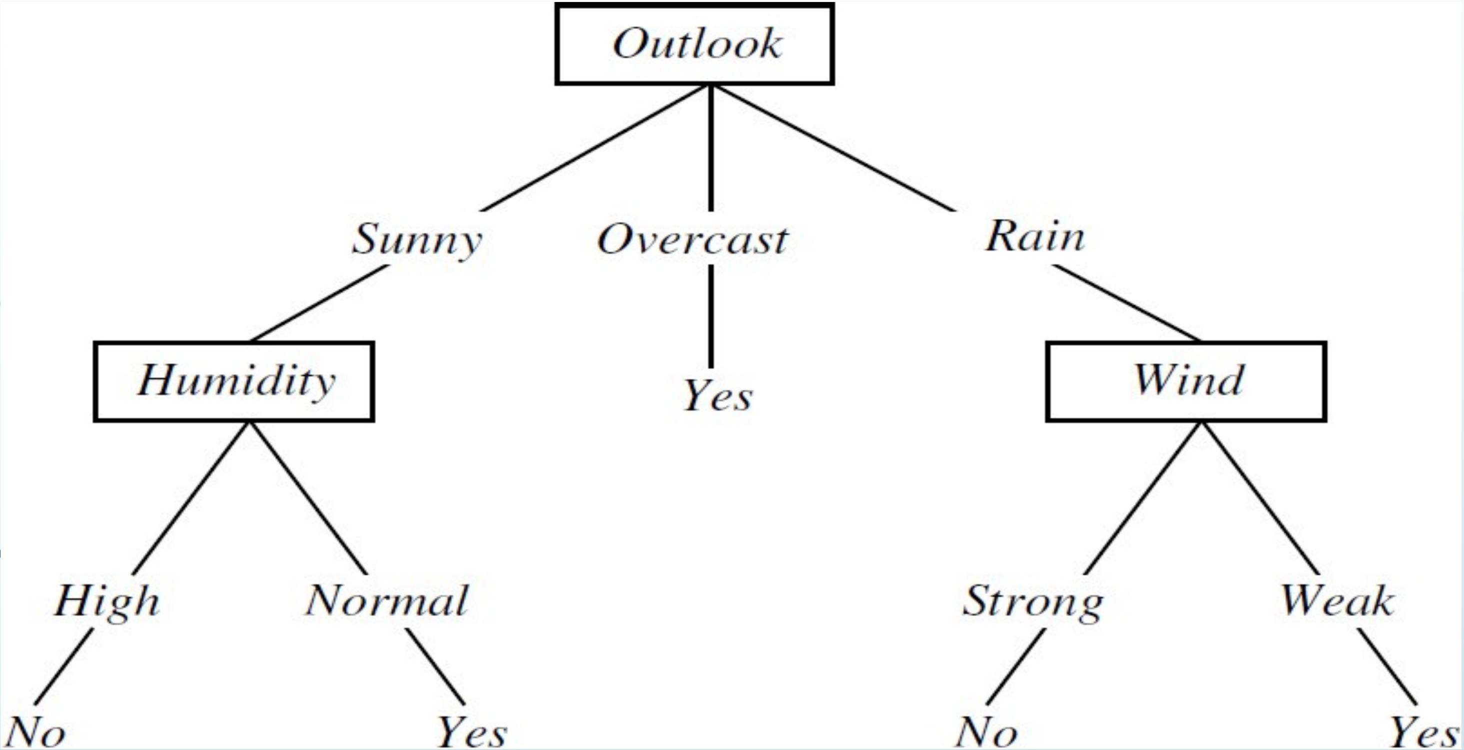
$$Gain\ (S_{sunny}, Wind) = .970 - (2/5)\ 1.0 - (3/5)\ .918 = .019$$

S_{Rain} = { D4, D5, D6, D10, D14}

$$Gain\ (S_{Rain}, Humidity) = 0.970 – (2/5)1.0 – (3/5)0.917 = 0.019$$

$$Gain\ (S_{Rain}, Temperature) = 0.970 – (0/5)0.0 – (3/5)0.918 – (2/5)1.0 = 0.019$$

$$Gain\ (S_{Rain}, Wind) = 0.970 – (3/5)0.0 – (2/5)0.0 = 0.970$$


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

HYPOTHESIS SPACE SEARCH IN DECISION TREE LEARNING

ACHARYA

- ID3 can be characterized as searching a space of hypotheses for one that fits the training examples.
- The hypothesis space searched by ID3 is the set of possible decision trees.
- ID3 performs a ***simple-to complex, hill-climbing search*** through this hypothesis space, beginning with the empty tree, then considering progressively more elaborate hypotheses in search of a decision tree that correctly classifies the training data

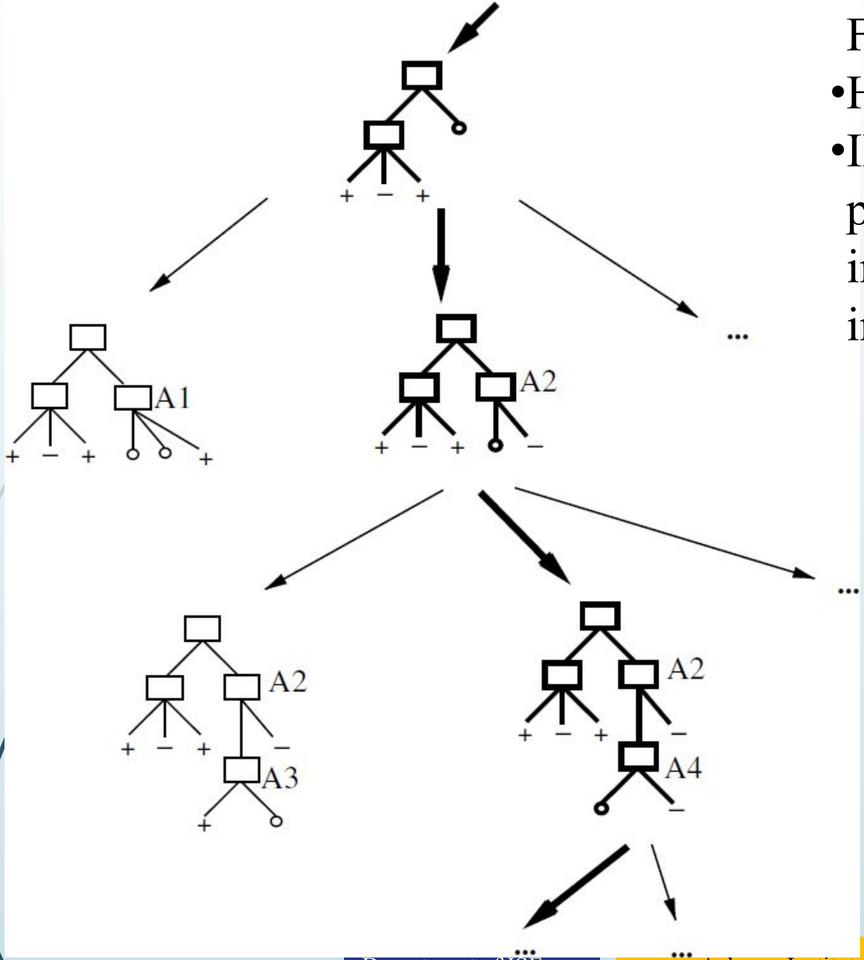


Figure:

- Hypothesis space search by ID3.
- ID3 searches through the space of possible decision trees from simplest to increasingly complex, guided by the information gain heuristic

By viewing ID3 in terms of its search space and search strategy, we can get some insight into its capabilities and limitations

1. ID3's hypothesis space of all decision trees is a ***complete*** space of finite discrete-valued functions, relative to the available attributes. Because every finite discrete-valued function can be represented by some decision tree

- ID3 avoids one of the major risks of methods that ***search incomplete hypothesis spaces*** : that the hypothesis space might not contain the target function.

2. ID3 maintains ***only a single current hypothesis*** as it searches through the space of decision trees.

For example, with the earlier version space candidate elimination method, which maintains the set of ***all*** hypotheses consistent with the available training examples.

By determining only a single hypothesis, ID3 loses the capabilities that follow from explicitly representing all consistent hypotheses.

For example, it does not have the ability to determine how many alternative decision trees are consistent with the available training data, or to pose new instance queries that optimally resolve among these competing hypotheses

3. **ID3** in its pure form performs ***no backtracking in its search***. Once it selects an attribute to test at a particular level in the tree, it never backtracks to reconsider this choice.

- In the case of **ID3,** a locally optimal solution corresponds to the decision tree it selects along the single search path it explores. However, this locally optimal solution may be less desirable than trees that would have been encountered along a different branch of the search.

4. **ID3** ***uses all training examples at each step*** in the search to make statistically based decisions regarding how to refine its current hypothesis.

- One advantage of using statistical properties of all the examples is that the resulting search is much ***less sensitive to errors*** in individual training examples.
- **ID3** can be easily extended to handle noisy training data by modifying its termination criterion to accept hypotheses that imperfectly fit the training data.


Thank You!