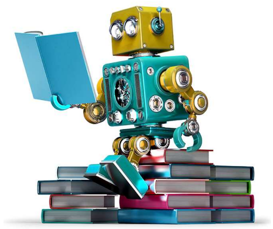
MACHINE LEARNING



MODULE-II
DECISION TREE LEARNING

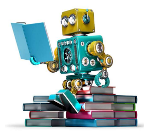
BY
HARIVINOD N
VIVEKANANDA COLLEGE OF ENGINEERING TECHNOLOGY, PUTTUR

Module 2- Outline

MACHINE LEARNING

Decision Tree Learning

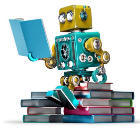
1. Introduction
2. Decision Tree Representation
3. Appropriate Problems for Decision Tree Learning
4. Basic Decision Tree Learning Algorithm (ID3)
5. Hypothesis Space Search in decision Tree Learning
6. Inductive Bias in Decision Tree Learning
7. Issues in Decision Tree Learning



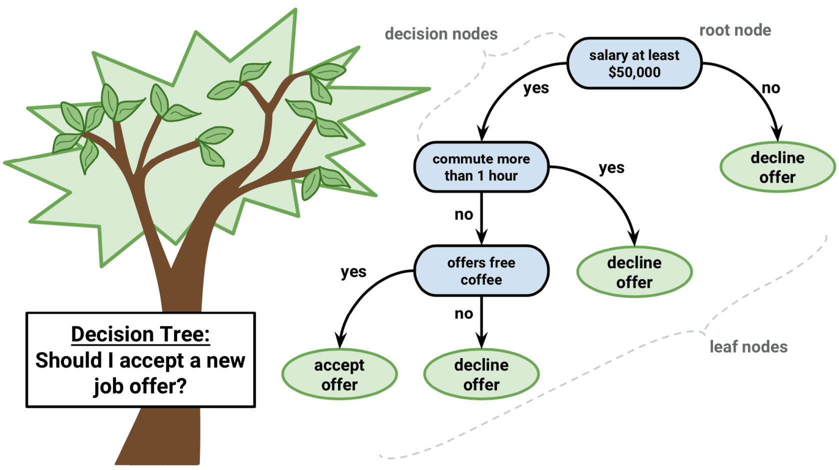
Module 2- Outline

Decision Tree Learning

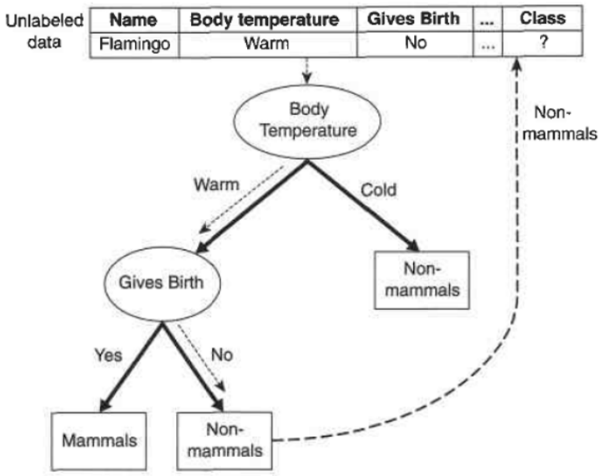
1. **Introduction**
2. Decision Tree Representation
3. Appropriate Problems for Decision Tree Learning
4. Basic Decision Tree Learning Algorithm (ID3)
5. Hypothesis Space Search in decision Tree Learning
6. Inductive Bias in Decision Tree Learning
7. Issues in Decision Tree Learning



1. Introduction



Introduction



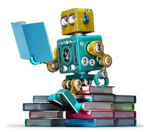
Introduction

- Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree.
- Learned trees can also be re-represented as sets of if-then rules to improve human readability.
- Most popular of inductive inference algorithms

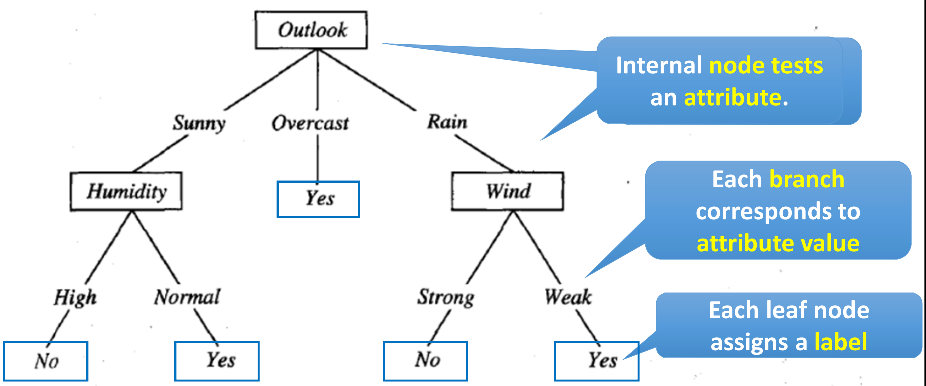
Module 2- Outline

Decision Tree Learning

1. Introduction
2. **Decision Tree Representation**
3. Appropriate Problems for Decision Tree Learning
4. Basic Decision Tree Learning Algorithm (ID3)
5. Hypothesis Space Search in decision Tree Learning
6. Inductive Bias in Decision Tree Learning
7. Issues in Decision Tree Learning



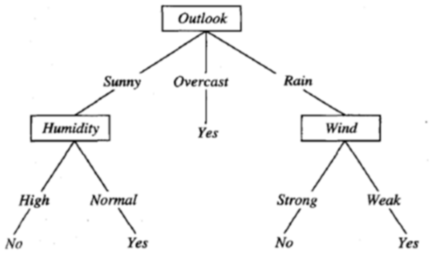
2. Decision tree representation



$$\langle Outlook = Sunny,\ Temperature = Hot,\ Humidity = High,\ Wind = Strong \rangle$$

Decision tree representation

- It represent a disjunction of conjunctions of constraints on the attribute values of instances.

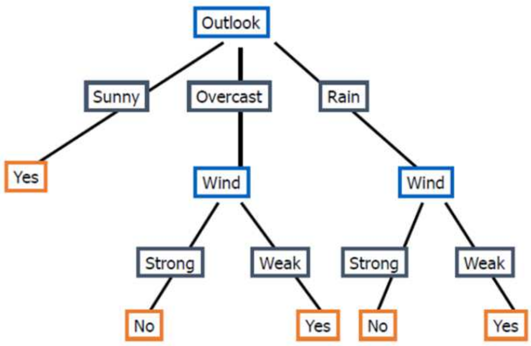


$$(Outlook = Sunny \wedge Humidity = Normal)$$
$$\vee \quad (Outlook = Overcast)$$
$$\vee \quad (Outlook = Rain \wedge Wind = Weak)$$

Decision tree representation

- Decision tree for Disjunction

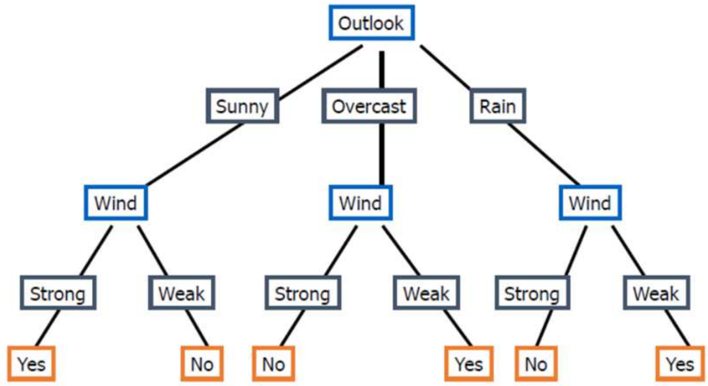
Outlook=Sunny ∨ Wind=Weak



Decision tree representation

- Decision tree for XOR

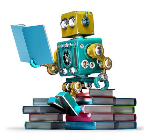
Outlook = Sunny XOR Wind = Weak



Module 2- Outline

Decision Tree Learning

1. Introduction
2. Decision Tree Representation
3. **Appropriate Problems for Decision Tree Learning**
4. Basic Decision Tree Learning Algorithm (ID3)
5. Hypothesis Space Search in decision Tree Learning
6. Inductive Bias in Decision Tree Learning
7. Issues in Decision Tree Learning



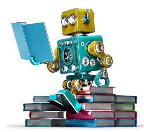
3. Appropriate problems

- Instances are represented by attribute-value pairs.
  - Attributes like Temperature (e.g., Hot, Mild, Cold).
- 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.
- Disjunctive descriptions may be required.
- The training data may contain errors.
  - Decision tree learning methods are robust to errors (error in attributes / error in targets)
- The training data may contain missing attribute values.

Module 2- Outline

Decision Tree Learning

1. Introduction
2. Decision Tree Representation
3. Appropriate Problems for Decision Tree Learning
4. **Basic Decision Tree Learning Algorithm (ID3)**
5. Hypothesis Space Search in decision Tree Learning
6. Inductive Bias in Decision Tree Learning
7. Issues in Decision Tree Learning



Basic Decision Tree Learning Algorithm (ID3)

MACHINE LEARNING

- The core algorithm that employs a top-down, greedy search through the space of possible decision trees.
- This approach is demonstrated by the ID3 algorithm (Iterative Dichotomiser 3)
- In real time we use variations of ID3.
- ID3 basic algorithm, learns decision trees by constructing them top-down, beginning with the question "which attribute should be tested at the root of the tree?"
- The best attribute is selected based on the statistical test at the root node of the tree.

Inventor

MACHINE LEARNING

- **John Ross Quinlan**
- He is a computer science researcher in data mining and decision theory.
- He has contributed extensively to the development of decision tree algorithms, including inventing the ID3 & canonical C4.5 algorithms.

Algorithm ID3 (Examples, TargetAttribute, Attributes)

Recursive algorithm

1. Create a *Root* node for the tree
2. If all *Examples* are positive, Return the single-node tree *Root*, with label = +
3. If all *Examples* are negative, Return the single-node tree *Root*, with label = -
4. If *Attributes* is empty,
   - Return the single-node tree Root, with label = most common value of *TargetAttribute* in *Examples*

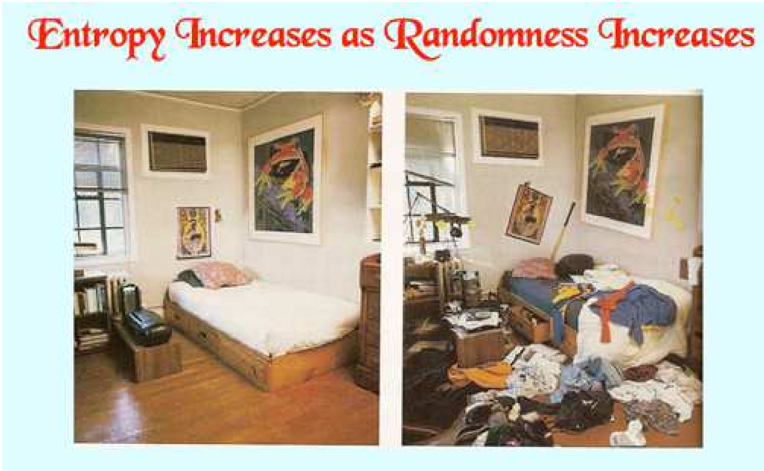
   else
   - A ← the attribute from *Attributes* that best classifies *Examples*
   - The decision attribute for *Root* ← A
   - For each possible value, vi, of A,
     - Add a new tree branch below *Root*, corresponding to the test A = vi
     - Let $Examples_{vi}$ be the subset of *Examples* that have value vi for A
     - If $Examples_{vi}$ is empty Then below this new branch add a leaf node with label = most common value of *TargetAttribute* in *Examples*

       Else below this new branch add subtree ID3($Examples_{vi}$, *TargetAttribute*, *Attributes*–{A})

   endif
5. Return Root

How to select the attribute from *Attributes* that best classifies *Examples?*

Definition: Entropy



Definition: Entropy

- It is a Measurement of Homogeneity of Examples
- Given a collection S, containing +ve and -ve examples of some target concept, the entropy of S is given by

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

- where $\mathbf{p_+}$ is the proportion of positive examples in S and $\mathbf{p_-}$ is the proportion of negative examples in S
- In all calculations involving entropy we define 0.log 0 = 0.
- In general for c class classification

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

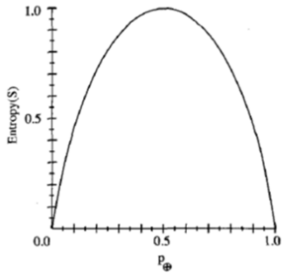
Entropy - Illustration

MACHINE LEARNING

- Let S is a collection of 14 examples of some boolean concept
- Let 9 positive and 5 negative examples [9+, 5-]
- Then the entropy of S relative to this boolean classification is

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

- Entropy function relative to a boolean classification, as p_+, varies between 0 and 1



Definition: Information Gain

MACHINE LEARNING

- It is the expected reduction in entropy caused by partitioning the examples according to some attribute A
- Split the node with attribute having highest Gain

$$Gain(S,A) = \underbrace{Entropy(S)}_{\text{original entropy of S}} - \underbrace{\sum_{v \in values(A)} \frac{|S_v|}{|S|} \cdot Entropy(S_v)}_{\text{relative entropy of S}}$$

- *S* – a collection of examples
- *A* – an attribute
- *Values(A)* – possible values of attribute A;
- S_v – the subset of S for which attribute A has value v.

$$(\text{i.e., } S_v = \{s \in S | A(s) = v\}).$$

Definition: Information Gain

MACHINE LEARNING

- Values (Wind) = Weak, Strong
- S = [9+,5-]
- $S_{weak} = [\ 6+,\ \ 2-]$
- $S_{strong} = [3+, 3\ -]$

$$Gain(S, wind) = Entropy(S) - \sum_{v \in \{weak, strong\}} (|S_v| / |S|)\ Entropy(S_v)$$

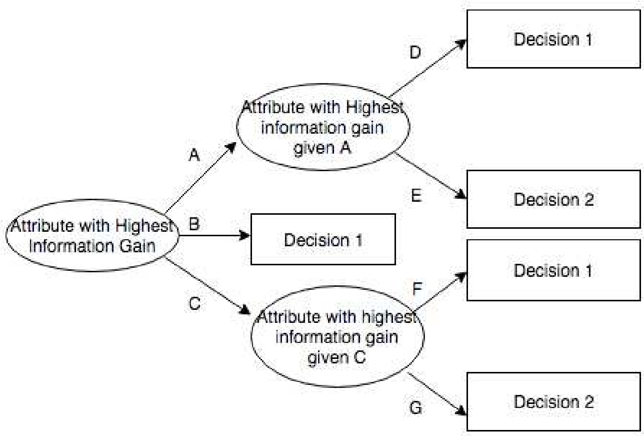
$= Entropy(S) – (8/14)Entropy(S_{weak}) – (6/4)Entropy(S_{strong})$

$= 0.940 – (8/14)0.811 – (6/14)1.00$

$= 0.048$

ID3 using Information Gain

MACHINE LEARNING



ID3: Illustration

Day	Outlook	Temp.	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	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

1. Level 0: To identify Root Node

Entropy(S) =

Day	Outlook	Temp.	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	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

a) To compute Gain(S, Outlook)

Outlook	PlayTennis	
	Yes	No
Sunny		
Overcast		
Rainy		

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

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

b) To compute Gain(S, Outlook)

Temp	PlayTennis	
	Yes	No
Hot		
Mild		
Cool		

Day	Outlook	Temp.	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	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

$$Entropy(S) \equiv \sum^{c} -p_i \log_2 p_i$$

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

c) To compute Gain (S, Humidity)

Humidity	PlayTennis	
	Yes	No
High		
Normal		

Day	Outlook	Temp.	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	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

$$Entropy(S) \equiv \sum^{c} -p_i \log_2 p_i$$

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

d) To compute Gain (S, Wind)

Wind	PlayTennis	
	Yes	No
Weak		
Strong		

Day	Outlook	Temp.	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	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

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

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

Illustration

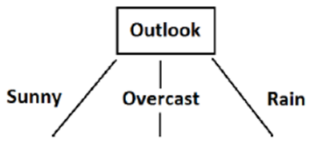
MACHINE LEARNING

		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			



2. Level 1: (1st branch) Outlook=Sunny

MACHINE LEARNING

Entropy(S_{sunny}) =

Day	Outlook	Temp.	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes

a) To compute **Gain (S_{sunny}, Temp)**

	PlayTennis	
Temp	Yes	No
Hot		
Mild		
Cool		

b) To compute **Gain (S_{sunny}, Humidity)**

MACHINE LEARNING

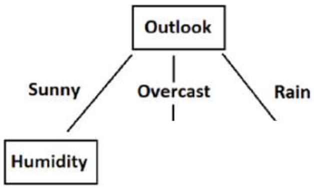
Humidity	PlayTennis	
	Yes	No
High		
Normal		

Day	Outlook	Temp.	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes

c) To compute **Gain (S_{sunny}, Wind)**

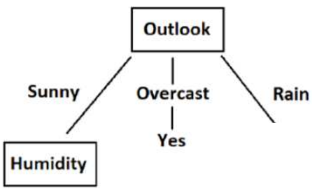
Wind	PlayTennis	
	Yes	No
Weak		
Strong		

Day	Outlook	Temp.	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes



- 3. Level 1: (2nd branch) Outlook = Overcast,
- All are yes,
- No Splitting required.

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D3	Overcast	Hot	High	Weak	Yes
D7	Overcast	Cool	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes



Illustration

MACHINE LEARNING

4. Level 1: (3rd branch)
Outlook = Rain

Entropy(S_{rain})

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D10	Rain	Mild	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

a) To compute **Gain (S_{Rain} , Temp)**

Temp	PlayTennis	
	Yes	No
Hot		
Mild		
Cool		

Illustration

MACHINE LEARNING

b) To compute **Gain (S_{Rain} , Humidity)**

Humidity	PlayTennis	
	Yes	No
High		
Normal		

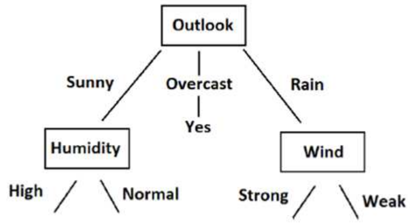
Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D10	Rain	Mild	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Illustration

b) To compute **Gain (S_{Rain} , Wind)**

Wind	PlayTennis	
	Yes	No
Weak		
Strong		

Day	*Outlook*	*Temp.*	*Humidity*	*Wind*	*Play Tennis*
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D10	Rain	Mild	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



Illustration

5. Level 2: (1st branch) Outlook=Sunny, Humidity = High ....No Splitting

Day	*Outlook*	*Temp.*	*Humidity*	*Wind*	*Play Tennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D8	Sunny	Mild	High	Weak	No

6. Level 2: (2st branch) Outlook=Sunny, Humidity = Normal ....No Splitting

Day	*Outlook*	*Temp.*	*Humidity*	*Wind*	*Play Tennis*
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes

Illustration

MACHINE LEARNING

7. Level 2: (3rd branch) Outlook = Rain, Wind=Weak ….No Splitting

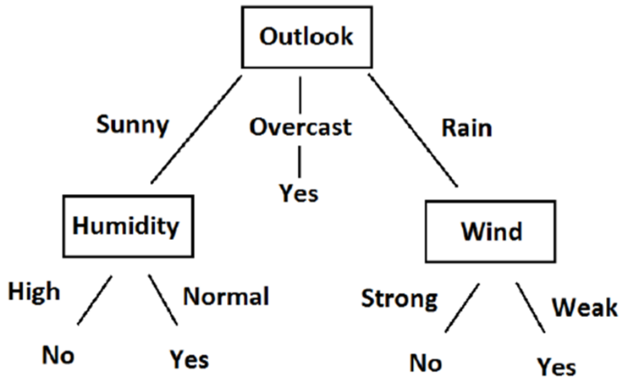
Day	*Outlook*	*Temp.*	*Humidity*	*Wind*	*Play Tennis*
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes

8. Level 2: (3rd branch) Outlook = Rain, Wind=Strong….No Splitting

Day	*Outlook*	*Temp.*	*Humidity*	*Wind*	*Play Tennis*
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No

Illustration: Final Decision Tree

MACHINE LEARNING



Algorithm ID3 (Examples, TargetAttribute, Attributes)

MACHINE LEARNING

Recursive algorithm

1. Create a *Root* node for the tree
2. If all *Examples* are positive, Return the single-node tree *Root*, with label = +
3. If all *Examples* are negative, Return the single-node tree *Root*, with label = -
4. If *Attributes* is empty,
   - Return the single-node tree Root, with label = most common value of *TargetAttribute* in *Examples*

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

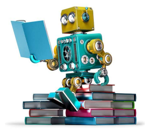
   endif
5. Return Root

Module 2- Outline

MACHINE LEARNING

Decision Tree Learning

1. Introduction
2. Decision Tree Representation
3. Appropriate Problems for Decision Tree Learning
4. Basic Decision Tree Learning Algorithm (ID3)
5. **Hypothesis Space Search in decision Tree Learning**
6. Inductive Bias in Decision Tree Learning
7. Issues in Decision Tree Learning



Hypothesis Space Search in ID3

MACHINE LEARNING

- Hypothesis space:
  - The hypothesis space searched by ID3 is the set of possible decision trees.
  - It is a complete space of finite discrete-valued functions, relative to the available attributes
- Search Method:
  - ID3 performs a simple-to complex, hill-climbing search.
- Evaluate Function: Information Gain

Hypothesis Space Search in ID3

MACHINE LEARNING

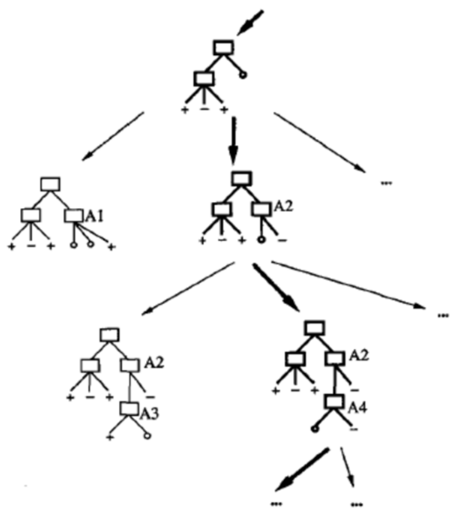


FIGURE 3.5
Hypothesis space search by ID3. ID3 searches through the space of possible decision trees from simplest to increasingly complex, guided by the information gain heuristic.

Capabilities and Limitations of ID3

- ID3 uses all training examples at each step
- ID3 can be easily extended to handle noisy training data
  by modifying its termination criterion
  to accept hypotheses that imperfectly fit the training data.
- No backtracking. No guaranty of optimality

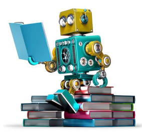
Advantages :
- Computationally Inexpensive
- Handles both numerical and categorical attributes
- Outputs are easy to interpret
- Works well with both linear and nonlinear data
- Sensitive to small variations in the training data
- Robust with redundant and correlated data

Disadvantages :
- Overfitting
- Too many Layers
- Lack of training Data
- Biased Data in training set
- Multicollinearity among variables

Module 2- Outline

Decision Tree Learning

1. Introduction
2. Decision Tree Representation
3. Appropriate Problems for Decision Tree Learning
4. Basic Decision Tree Learning Algorithm (ID3)
5. Hypothesis Space Search in decision Tree Learning
6. **Inductive Bias in Decision Tree Learning**
7. Issues in Decision Tree Learning



Inductive bias

- Typically there are many decision trees consistent with training examples.
- It chooses the first acceptable tree it encounters in its simple-to-complex, hill-climbing search through the space of possible trees
- Approximate inductive bias of ID3: Shorter trees are preferred over larger trees.
- A closer approximation to the inductive bias of ID3
  - Shorter trees are preferred over longer trees.
  - Trees that place high information gain attributes close to the root are preferred over those that do not.

6.1 Restriction Biases and Preference Biases

- The inductive bias of ID3 is thus a preference for certain hypotheses over others (e.g., for shorter hypotheses)
  - This form of bias is typically called a preference bias (or, alternatively, a search bias).
- In contrast, the bias of the CEA is in the form of a categorical restriction on the set of hypotheses considered.
  - This form of bias is typically called a restriction bias (or, alternatively, a language bias).
- A preference bias is more desirable than a restriction bias
- ID3 exhibits a purely preference bias and CEA is a purely restriction bias whereas some learning systems combine both.

6.2 Why prefer short hypothesis?

- Occam's razor: (Problem Solving Principle)
  - **Prefer the simplest hypothesis that fits the data.**
  - (The term razor is frequency and effectiveness with which he used it)

Why prefer short hypotheses?

Argument 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

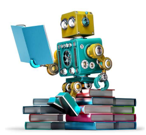
Argument opposed:

- There are many ways to define small sets of hypotheses
- What is so special about small sets based on *size* of hypothesis

Module 2- Outline

Decision Tree Learning

1. Introduction
2. Decision Tree Representation
3. Appropriate Problems for Decision Tree Learning
4. Basic Decision Tree Learning Algorithm (ID3)
5. Hypothesis Space Search in decision Tree Learning
6. Inductive Bias in Decision Tree Learning
7. **Issues in Decision Tree Learning**



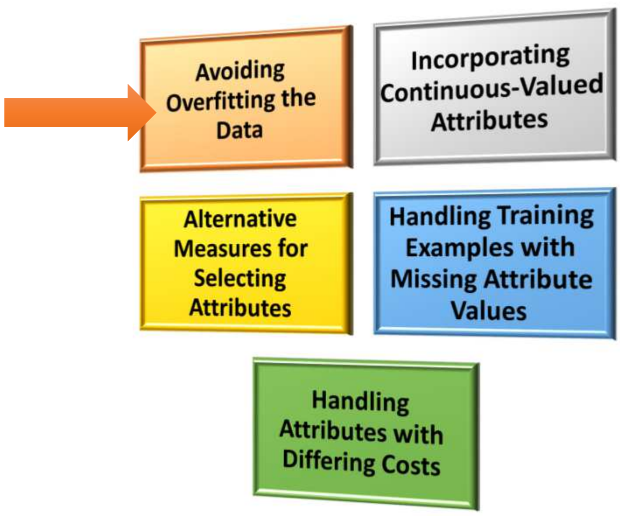
Issues in Decision tree learning

MACHINE LEARNING

- Practical issues in learning decision trees include
  - determining how deeply to grow the decision tree,
  - handling continuous attributes,
  - choosing an appropriate attribute selection measure,
  - handling training data with missing attribute values,
  - handling attributes with differing costs, and
  - improving computational efficiency.
- we discuss each of these issues and extensions to the basic ID3 algorithm that address them.
- ID3 has itself been extended to address most of these issues, with the resulting system renamed C4.5.

Issues in Decision Tree learning

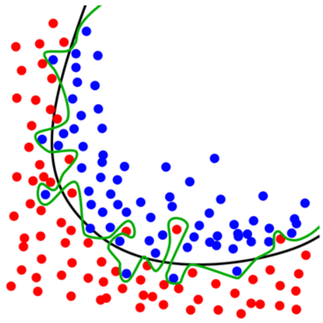
MACHINE LEARNING



Overfitting

MACHINE LEARNING

- Consider 2D data. +ve examples are plotted in Blue, -ve are in Red
- The green line represents an overfitted model and the black line represents a regularized model.
- While the green line best follows the training data, it is too dependent on that data and it is likely to have a higher error rate on new unseen data, compared to the black line.

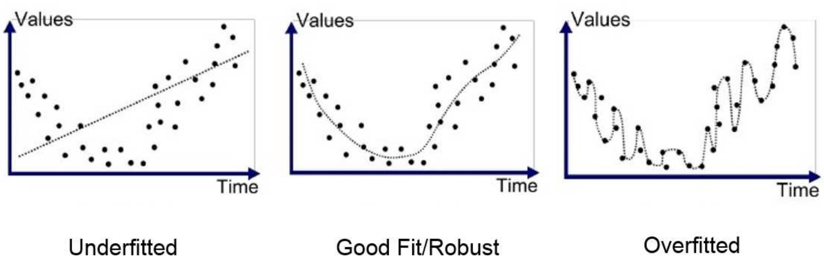


(Source: wikipedia)

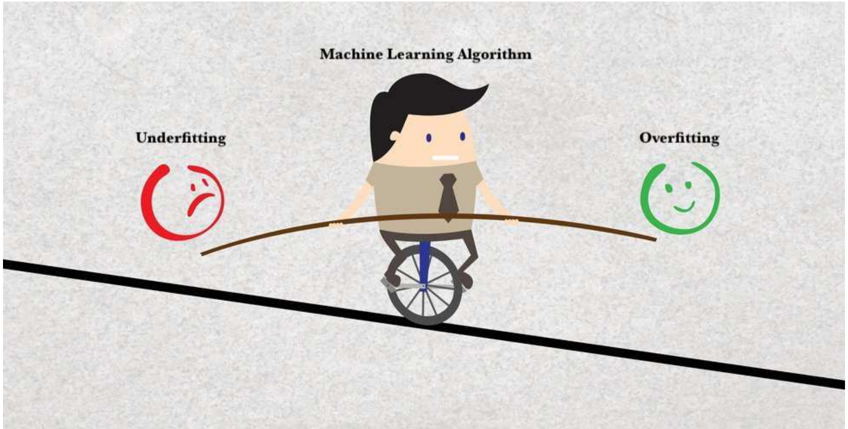
Underfitting

MACHINE LEARNING

- **Underfitting** occurs when a statistical model cannot adequately capture the underlying structure of the data.



Souce: http://blog.algotrading101.com/design-theories/what-is-curve-fitting-overfitting-in-trading/

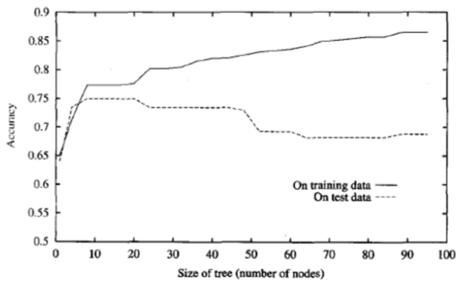


Overfitting

Definition: Overfitting

Given a hypothesis space H,
a hypothesis h ∈ H is said to **overfit** the training data
if there exists some alternative hypothesis h' ∈ H,
such that h has smaller error than h' over the training examples,
but h' has a smaller error than h over the entire distribution of instances.

ID3: Tree Nodes vs Accuracy



Overfitting

- **reasons for overfitting:**
  - noise in the data
  - number of training examples is to small too produce a representative sample of the target function
- **how to avoid overfitting:**
  - **stop the tree grow earlier**, before it reaches the point where it perfectly classifies the training data
  - allow overfitting and then **post-prune** the tree (more successful in practice!)
- **how to determine the perfect tree size:**
  - separate validation set to evaluate utility of post-pruning
  - apply statistical test to estimate whether expanding (or pruning) produces an improvement

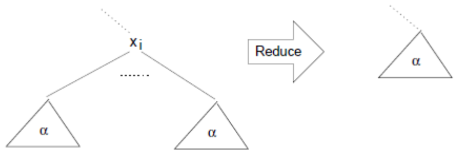
Validation Set

- How exactly might we use a validation set to prevent overfitting?
  - **Reduced Error Pruning**
  - **Rule Post-Pruning**

1. Reduced Error Pruning..(1)

MACHINE LEARNING

- each of the decision nodes is considered to be candidate for pruning

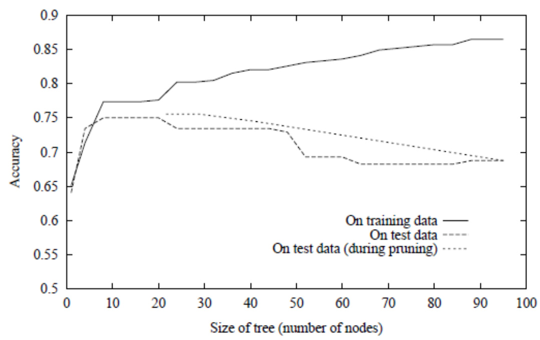


- **pruning** a decision node consists of removing the subtree rootet at the node, making it a leaf node and assigning the most common classification of the training examples affiliated with that node
- nodes are removed only if the resulting tree performs **not worse** than the original tree over the validation set
- pruning starts wit the node whose removal most increases accuracy and continues until further pruning is harmful

1. Reduced Error Pruning ..(2)

MACHINE LEARNING

- **effect of reduced error pruning:**

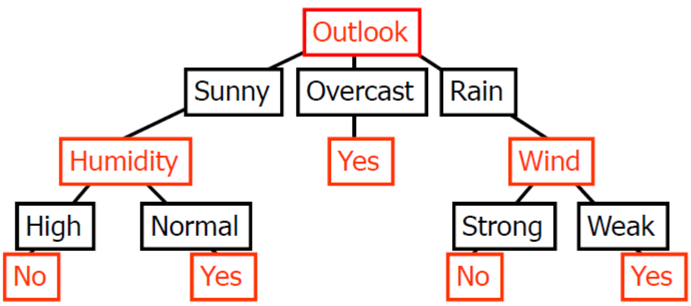


- any node added to coincidental regularities in the training set is likely to be pruned

2. Rule Post Pruning

- rule post-pruning involves the following steps:
  1. Infer the decision tree from the training set (Overfitting allowed!)
  2. Convert the tree into a set of rules
  3. Prune each rule by removing any preconditions that result in improving its estimated accuracy
  4. Sort the pruned rules by their estimated accuracy
- one method to estimate rule accuracy is to use a separate validation set
- pruning rules is more precise than pruning the tree itself

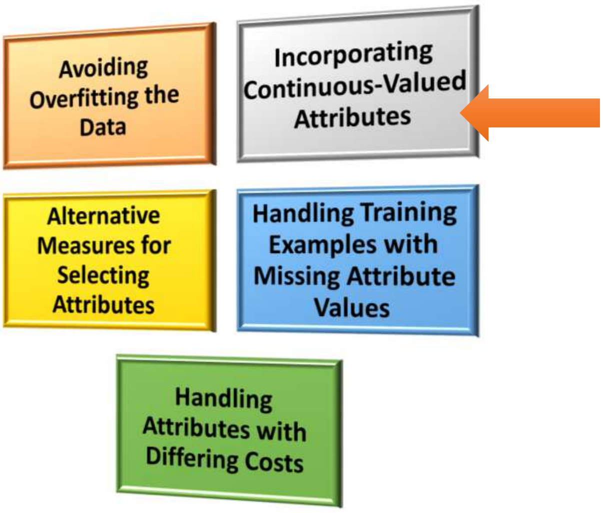
Converting decision trees into rules



R1: If (Outlook=Sunny) ∧ (Humidity=High) Then PlayTennis=No
R2: If (Outlook=Sunny) ∧ (Humidity=Normal) Then PlayTennis=Yes
R3: If (Outlook=Overcast) Then PlayTennis=Yes
R4: If (Outlook=Rain) ∧ (Wind=Strong) Then PlayTennis=No
R5: If (Outlook=Rain) ∧ (Wind=Weak) Then PlayTennis=Yes

Issues in Decision Tree learning

MACHINE LEARNING



Incorporating continuous valued attributes

MACHINE LEARNING

- ID3 is restricted to attributes that take on a discrete set of values.
- Define new discrete valued attributes that partition the continuous attribute value into a discrete set of intervals
- For a continuous-valued attribute A that is, create a new boolean attribute Ac, that is true if $A < c$ and false otherwise.
  - Select c using information gain
  - Sort examples according to the continuous attributeA,
  - Then identify adjacent examples that differ in their target classification
  - Generate candidate thresholds midway between corresponding values of A.
  - The value of c that maximizes information gain must always lie at a boundary.
  - These candidate thresholds can then be evaluated by computing the information gain associated with each.

Example

Temperature: 40 48 60 72 80 90
PlayTennis : No No Yes Yes Yes No

Temperature:	40	48	60	72	80	90
PlayTennis :	No	No	Yes	Yes	Yes	No

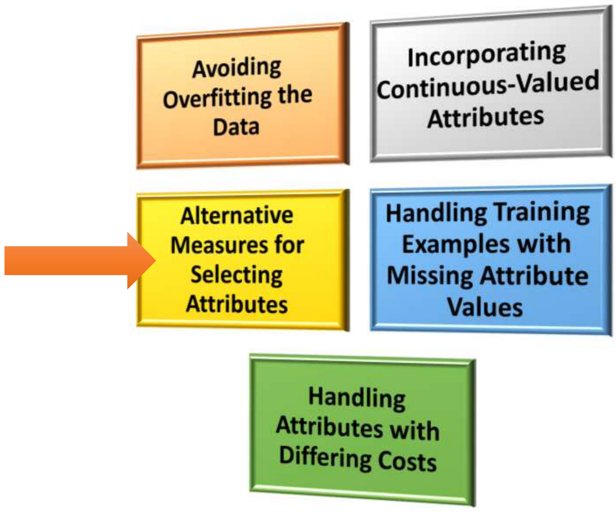
Two candidate thresholds: (48+60)/2=54 (80+90)/2=85

Check the information gain for new boolean attributes:

$\text{Temperature}_{>54}$ $\text{Temperature}_{>85}$

Use these new new boolean attributes same as other discrete valued attributes.

Issues in Decision Tree learning



Alternative Selection Measures

MACHINE LEARNING

- Information gain measure favors attributes with many values
  - separates data into small subsets
  - high gain, poor prediction
- Ex. Date attribute has many values, and may separate training examples into very small subsets (even singleton sets – perfect partitions)
  - Information gain will be very high for Date attribute.
  - Perfect partition →maximum gain : Gain(S,Date) = Entropy(S) – 0 = Entropy(S) because $\log_2 1$ is 0.
  - It has high information gain, but very poor predictor for unseen data.
- There are alternative selection measures such as *GainRatio* measure based on *SplitInformation*

Split information

MACHINE LEARNING

- The *gain ratio* measure penalizes attributes with many values (such as Date) by incorporating a term, called *split informtion*

$$\text{SplitInformation}(S,A) = -\sum_{i=1}^{c} (|Si| / |S|) \log_2 (|Si| / |S|)$$

- Split information for boolean attributes is 1 (= $\log_2 2$),
- Split information fot attributes for n values is $\log_2 n$

$$\text{GainRatio}(S,A) = \text{Gain}(S,A) / \text{SplitInformation}(S,A)$$

- *SplitInformation* term discourages the selection of attributes with many uniformly distributed values.

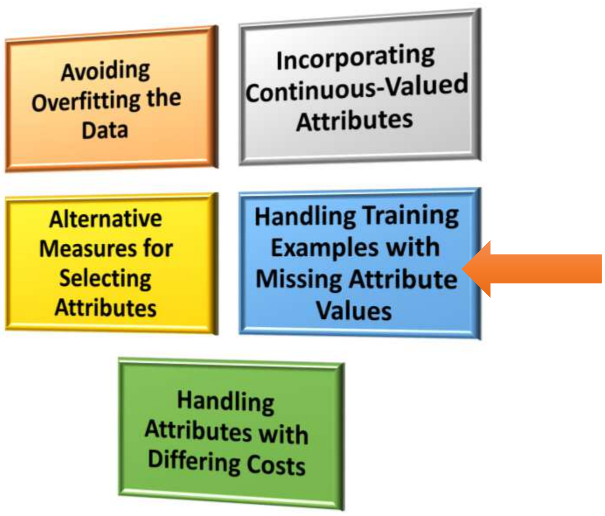
Practical issues on Split information

MACHINE LEARNING

- Some value 'rules'
  - |Si| close to |S|
  - SplitInformation 0 or very small
  - GainRatio undefined or very large
- Apply heuristics to select attributes
  - compute Gain first
  - compute GainRatio only when Gain large enough (above average Gain)

Issues in Decision Tree learning

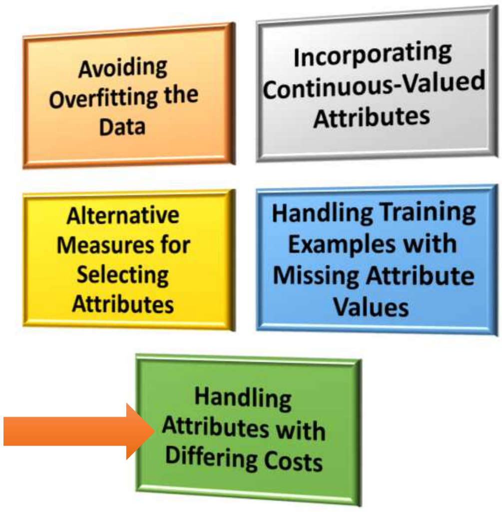
MACHINE LEARNING



Missing Attribute values

- The available data may be missing values for some attributes.
- It is common to estimate the missing attribute value based on other examples for which this attribute has a known value.
- Assume that an example (with classification c) in S has a missing value for attribute A.
  - Assign the most common value of A in S.
  - Assign the most common value of in the examples having c classification in S.
  - Or, use probabilty value for each possible attribute value.

Issues in Decision Tree learning



Attributes with different cost

- Measuring attribute costs something
  - prefer cheap ones if possible
  - use costly ones only if good gain
  - introduce cost term in selection measure
  - no guarantee in finding optimum, but give bias towards cheapest
- Example applications
  - robot & sonar: time required to position
  - medical diagnosis: cost of a laboratory test

Summary

The main points in this module include:

- Decision tree learning provides a practical method for concept learning and for learning other discrete-valued functions.
- ID3 searches a complete hypothesis space
- The inductive bias implicit in ID3 includes a *preference* for smaller trees
- Overfitting the training data is an important issue in decision tree learning.
- A large variety of extensions to the basic ID3 algorithm has been developed by different researchers. These include methods for
  - post-pruning trees,
  - handling real-valued attributes
  - accommodating training examples with missing attribute values
  - incrementally refining decision trees as new training examples available
  - using attribute selection measures other than information gain
  - considering costs associated with instance attributes.


Thank
You