Computational Learning and Discovery

GEORGE MASON UNIVERSITY

CSI 873 / MATH 689

Instructor: I. Griva

Wednesday 7:20 – 10:00 pm

Decision tree learning –

method for approximating discrete-valued functions, in which the learned function is approximated by a decision tree

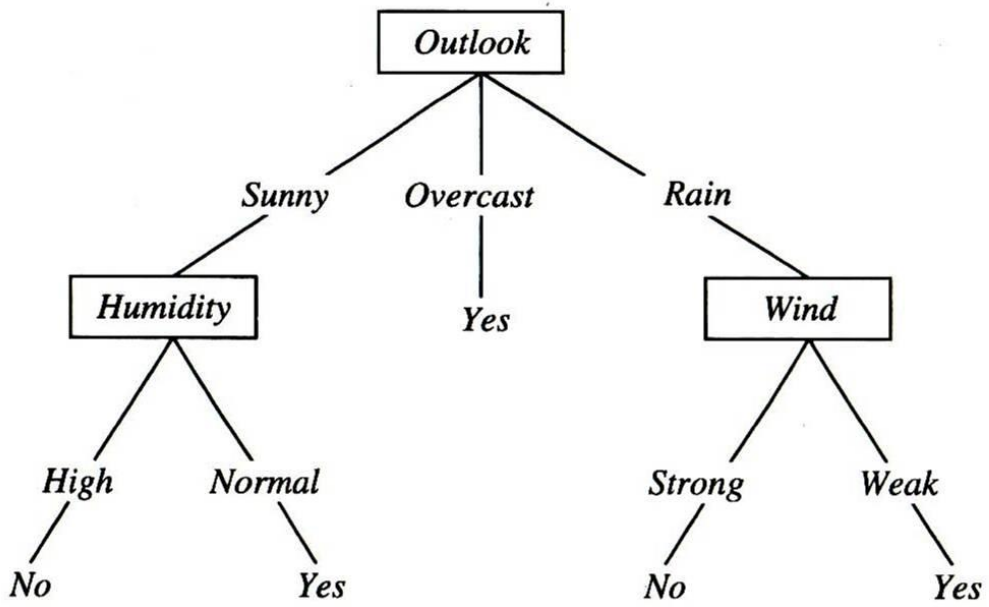


FIGURE 3.1
A decision tree for the concept *PlayTennis*. An example is classified by sorting it through the tree to the appropriate leaf node, then returning the classification associated with this leaf (in this case, *Yes* or *No*). This tree classifies Saturday mornings according to whether or not they are suitable for playing tennis.

Appropriate problems for decision tree learning

- instances are represented by attribute-value pairs
- the target function has discrete output value
- disjunctive descriptions maybe required
- the training data may contain errors
- the training data may contain missing attribute values

Successfully applied for medical diagnosis, equipment malfunction detection, loan application decisions

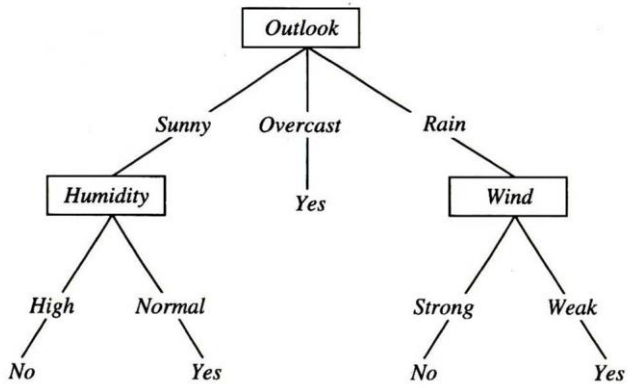


FIGURE 3.1
A decision tree for the concept *PlayTennis*. An example is classified by sorting it through the tree to the appropriate leaf node, then returning the classification associated with this leaf (in this case, *Yes* or *No*). This tree classifies Saturday mornings according to whether or not they are suitable for playing tennis.

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

TABLE 3.2
Training examples for the target concept *PlayTennis*.

ID3 is the basic decision tree learning algorithm

ID3 is the basic decision tree learning algorithm

Uses entropy to measure homogeneity of data

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

ID3 is the basic decision tree learning algorithm

Uses entropy to measure homogeneity of data

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

Information gain

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

Which attribute is the best classifier?

Example

$$Values(Wind) = Weak, Strong$$

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

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

$$S_{Strong} \leftarrow [3+, 3-]$$

$$Gain(S, Wind) = Entropy(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

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

$$= 0.940 - (8/14)0.811 - (6/14)1.00$$

$$= 0.048$$

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

TABLE 3.2
Training examples for the target concept *PlayTennis*.

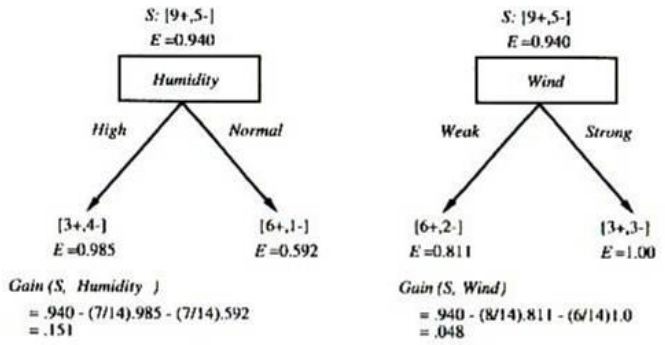


FIGURE 3.3
Humidity provides greater information gain than *Wind*, relative to the target classification. Here, E stands for entropy and S for the original collection of examples. Given an initial collection S of 9 positive and 5 negative examples, $[9+, 5-]$, sorting these by their *Humidity* produces collections of $[3+, 4-]$ ($Humidity = High$) and $[6+, 1-]$ ($Humidity = Normal$). The information gained by this partitioning is .151, compared to a gain of only .048 for the attribute *Wind*.

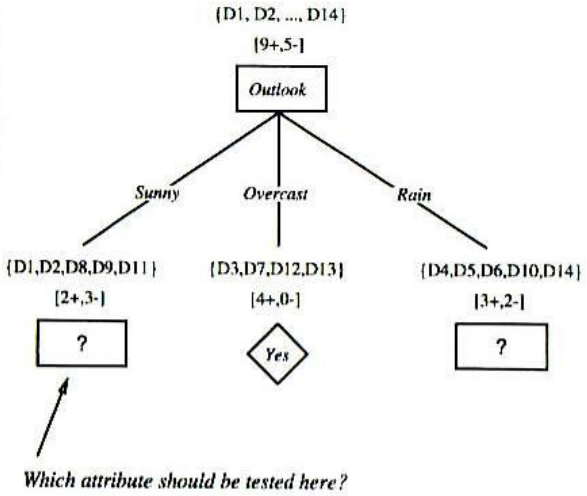
$$Gain(S, Outlook) = 0.246$$

$$Gain(S, Humidity) = 0.151$$

$$Gain(S, Wind) = 0.048$$

$$Gain(S, Temperature) = 0.029$$

Example



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

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

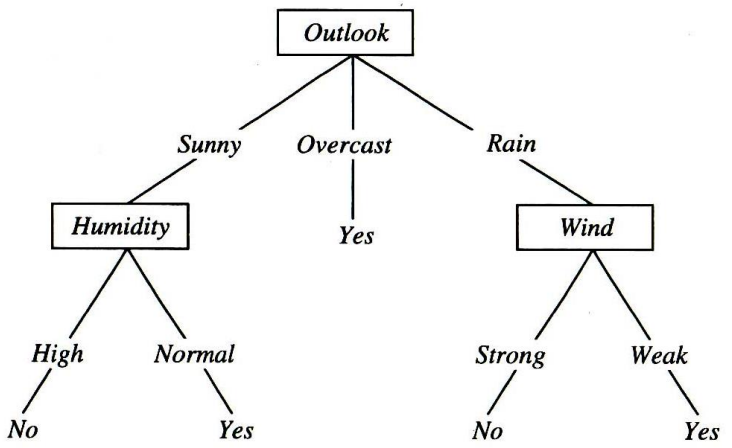
$Gain(S_{sunny}, Temperature)$ = .970 - (2/5) 0.0 - (2/5) 1.0 - (1/5) 0.0 = .570

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

FIGURE 3.4
The partially learned decision tree resulting from the first step of ID3. The training examples are sorted to the corresponding descendant nodes. The *Overcast* descendant has only positive examples and therefore becomes a leaf node with classification *Yes*. The other two nodes will be further expanded, by selecting the attribute with highest information gain relative to the new subsets of examples.

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

TABLE 3.2
Training examples for the target concept *PlayTennis*.



The hypothesis space is the space of possible decision trees.

Capabilities and limitations:

- **Complete space of finite discrete valued functions considered**
- **Finds only a single hypothesis**
- **No backtracking**
- **All training examples are used to make decision how to refine the current hypothesis**

Inductive bias in decision tree learning:

Approximately: Shorter tree are preferred over longer trees. Trees that place high information gain attributes close to the root are preferred over those that do not.

Decision Tree algorithms have a preference bias while the Candidate Elimination algorithm has a restriction bias.

Shorter hypothesis?

Occam's razor:
Prefer the simplest hypothesis that fits the data.

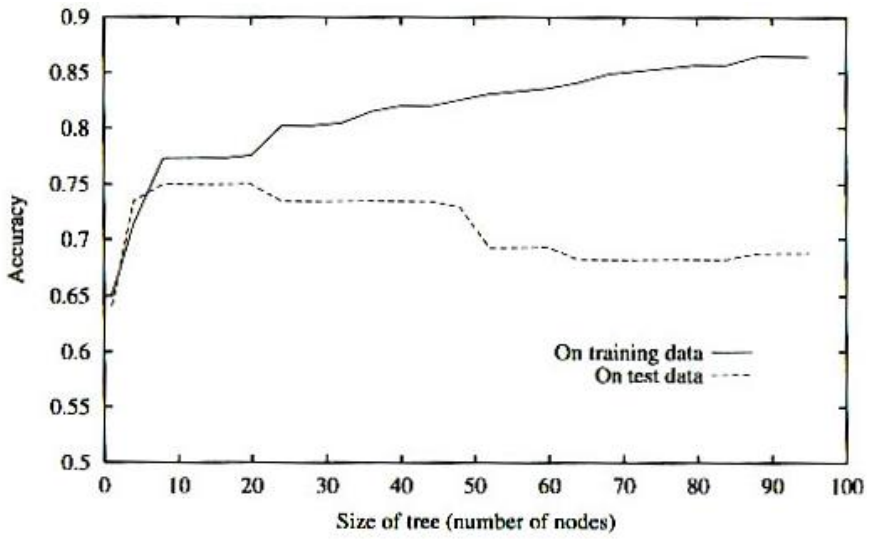
Some issues and details of Decision Tree Learning

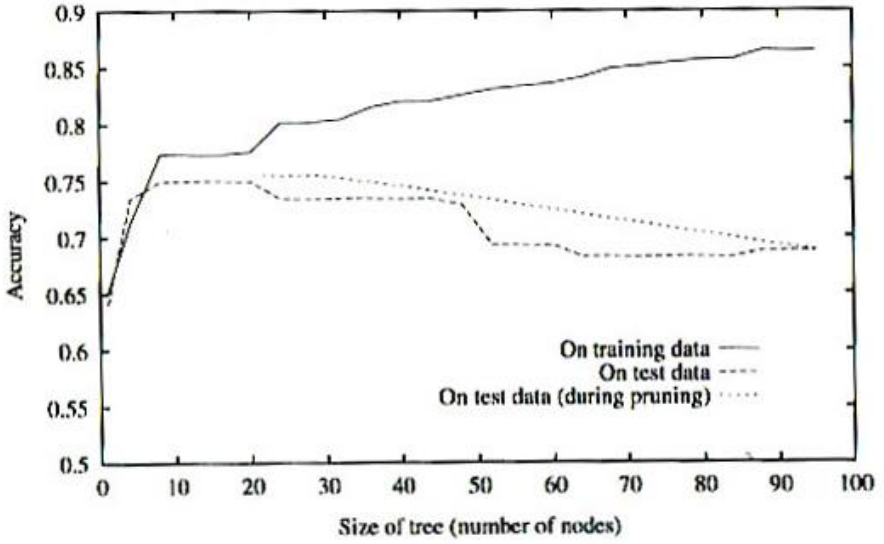
- Overfitting
- Post-pruning
- Continuous-valued attributes
- Missing attributes values
- Attributes with different costs
- Various measures for selecting attributes

Overfitting the Data

Definition.

Given a hypothesisspace H, a hypothesis$h \in H$ is said to overfit the training data if there exists ome alternative hypothesis$h' \in H$, such that h has a smaller error than h' over thraining examples, but h' has a smaller error than h over the entire distribution of instances.





Reduced-error pruning!

Rule post pruning!

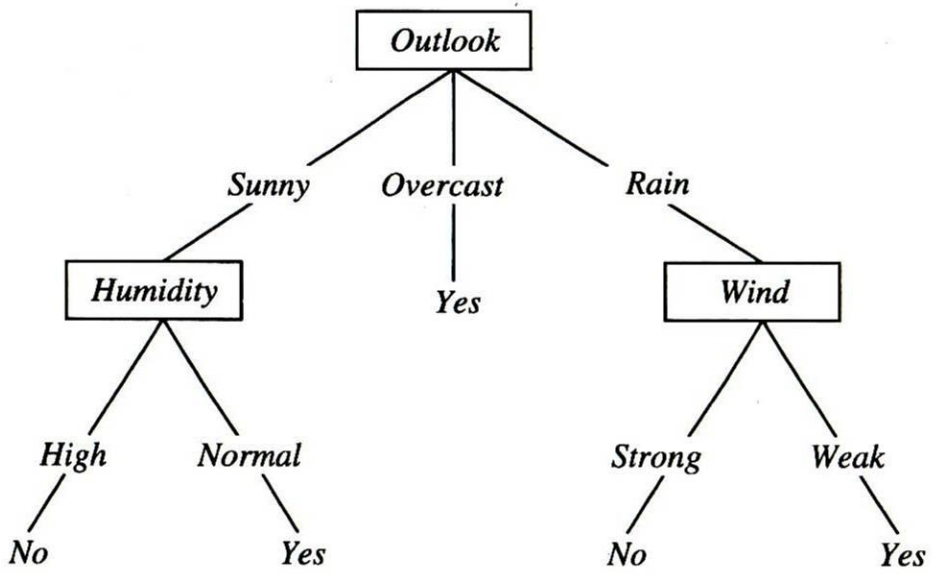


FIGURE 3.1
A decision tree for the concept *PlayTennis*. An example is classified by sorting it through the tree to the appropriate leaf node, then returning the classification associated with this leaf (in this case, *Yes* or *No*). This tree classifies Saturday mornings according to whether or not they are suitable for playing tennis.

IF (*Outlook* = *Sunny*) and (*Humidity* = *High*)
THEN *PlayTennis* = *No*

Incorporation of Continuous-Valued Attributes

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

Attributes:

(Temperature > 54)
(Temperature > 85)

Alternative Measures for Selecting Attributes

$$SplitInformation(S,A) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

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

Alternative Measures for Selecting Attributes

$$SplitInformation(S,A) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

$$GainRatio(S,A) = \frac{Gain(S,A)}{SplitInformation(S,A)+1}$$

Missing Attribute Values

Assign the most common value among training examples at node *n*

Assign probabilities to each of the possible values of attributes

Attributes with different costs

$$\frac{Gain(S,A)}{Cost(A)}$$

$$\frac{Gain^2(S,A)}{Cost(A)}$$

$$\frac{Gain^2(S,A)}{Cost(A)+1}$$

$$\frac{2^{Gain(S,A)}-1}{(Cost(A)+1)^w}$$

Random forest – constructs the family of trees

Each tree is constructed using the following algorithm:

1. Let the number of training cases be M, and the number of variables in the classifier be N.
2. We are told the number n of input variables to be used to determine the decision at a node of the tree; n should be much less than N.
3. Choose a training set for this tree by choosing m times with replacement from all M available training cases . Use the rest of the cases to estimate the error of the tree, by predicting their classes.
4. For each node of the tree, randomly choose n variables on which to base the decision at that node. Calculate the best split based on these n variables in the training set.
5. Each tree is fully grown and not pruned (as may be done in constructing a normal tree classifier).

For prediction a new sample is pushed down the tree. It is assigned the label of the training sample in the terminal node it ends up in. This procedure is iterated over all trees in the ensemble, and the average vote of all trees is reported as random forest prediction.

SUMMARY

- Decision tree learning (DTL) is an error robust method for learning discrete-valued functions (concept learning)

- DTL searches a complete hypothesis space, therefore it avoids the problem of restricted hypothesis space

- The inductive bias of the algorithm is a preference for smaller trees

- Overfitting of data is likely, so the post-pruning is important

- Large variety of the basic DTL includes different ways of post-pruning, accommodating missing attributes, and various measures of attribute selections.