Introduction

Decision tree learning is one of the most widely used and practical methods for inductive inference. It is a method for approximating discrete-valued functions that is robust to noisy data and capable of learning disjunctive expressions. This chapter describes a family of decision tree learning algorithms that includes widely used algorithms such **as** ID3, ASSISTANT, and **C4.5.** These decision tree learning methods search a completely expressive hypothesis space and thus avoid the difficulties of restricted hypothesis spaces. Their inductive bias is a preference for small trees over large trees.

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. These learning methods are among the most popular of inductive inference algorithms and have been successfully applied to a broad range of tasks from learning to diagnose medical cases to learning to assess credit risk of loan applicants.

DECISION TREE REPRESENTATION

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.

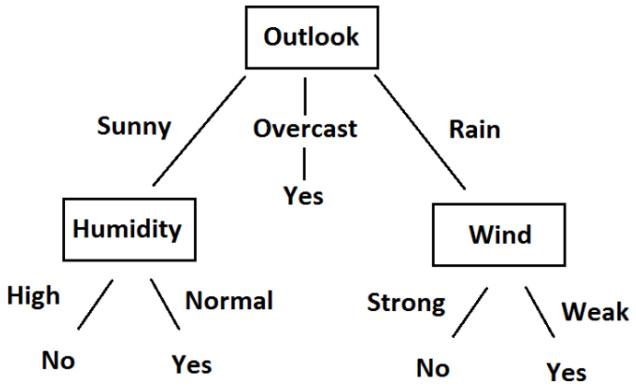


Figure 1 Decision Tree for PlayTennis problem

Figure ***1*** illustrates a typical learned decision tree. This decision tree classifies Saturday mornings according to whether they are suitable for playing tennis.
For example, the instance

(Outlook = Sunny, Temperature = Hot, Humidity = High, Wind = Strong)

would be sorted down the leftmostbranch of this decision tree and would therefore be classified as a negative instance (i.e., the tree predicts that ***PlayTennis*** = ***no).*** This tree and the example used in Table 1 to illustrate the ID3 learning algorithm.

In general, 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 Figure **1** corresponds to the expression

(Outlook = ***Sunny*** **A** ***Humidity*** = ***Normal)*** ∨ ***(Outlook*** = ***Overcast)*** v ***(Outlook*** = ***Rain A Wind*** = ***Weak)***

3.3 APPROPRIATE PROBLEMS FOR DECISION TREE LEARNING

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

- ***Instances are represented by attribute-value pairs.*** Instances are described by a fixed set of attributes (e.g., ***Temperature)*** and their values (e.g., ***Hot).*** The easiest situation for decision tree learning is when each attribute takes on a small number of disjoint possible values (e.g., ***Hot, Mild, Cold).*** However, extensions to the basic algorithm allow handling real-valued attributes as well (e.g., representing ***Temperature*** numerically).
- ***The target function has discrete output values.*** The decision tree in Figure.1 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. A more substantial extension allows learning target functions with real-valued outputs, though the application of decision trees in this
  setting is less common.
- ***Disjunctive descriptions may be required.*** As noted above, decision trees naturally represent disjunctive expressions.
- ***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.
- ***The training data may contain missing attribute values.*** Decision tree methods can be used even when some training examples have unknown values (e.g., if the ***Humidity*** of the day is known for only some of the training examples).

3.4 THE BASIC DECISION TREE LEARNING ALGORITHM

A Step by Step ID3 Decision Tree Example Post navigation

Decision tree algorithms transform raw data to rule based decision-making trees. Herein, ID3 is one of the most common decision tree algorithms. Firstly, It was introduced in 1986 and it is acronym of **Iterative Dichotomiser**.

First of all, dichotomisation means dividing into two completely opposite things. That's why, the algorithm iteratively divides attributes into two groups which are the most dominant attribute and others to construct a tree. Then, it calculates the entropy and information gains of each attribute. In this way, the most dominant attribute can be founded. After then, the most dominant one is put on the tree as decision node. Thereafter, entropy and gain scores would be

calculated again among the other attributes. Thus, the next most dominant attribute is found. Finally, this procedure continues until reaching a decision for that branch. That's why, it is called Iterative Dichotomiser.

For instance, the following table informs about decision making factors to play tennis at outside for previous 14 days.

Table 1:

Day	Outlook	Temp.	Humidity	Wind	Decision
1.	Sunny	Hot	High	Weak	No
2.	Sunny	Hot	High	Strong	No
3.	Overcast	Hot	High	Weak	Yes
4.	Rain	Mild	High	Weak	Yes
5.	Rain	Cool	Normal	Weak	Yes
6.	Rain	Cool	Normal	Strong	No
7.	Overcast	Cool	Normal	Strong	Yes
8.	Sunny	Mild	High	Weak	No
9.	Sunny	Cool	Normal	Weak	Yes
10.	Rain	Mild	Normal	Weak	Yes
11.	Sunny	Mild	Normal	Strong	Yes
12.	Overcast	Mild	High	Strong	Yes
13.	Overcast	Hot	Normal	Weak	Yes
14.	Rain	Mild	High	Strong	No

The central choice in the ID3 algorithm is selecting which attribute to test at each node in the tree. We would like to select the attribute that is most useful for classifying examples. We will define a statistical property, called ***information gain,*** which measures how well a given attribute separates the training examples according to their target classification. ID3 uses this information gain measure to select among the candidate attributes at each step while growing the tree.

ENTROPY MEASURES HOMOGENEITY OF EXAMPLES

In order to define information gain precisely, we begin by defining a measure commonly used in information theory, called ***entropy,*** that characterizes the (im)purity of an arbitrary collection of examples. Given a collection S, containing positive and negative examples of some target concept, the entropy of S relative to this boolean classification is

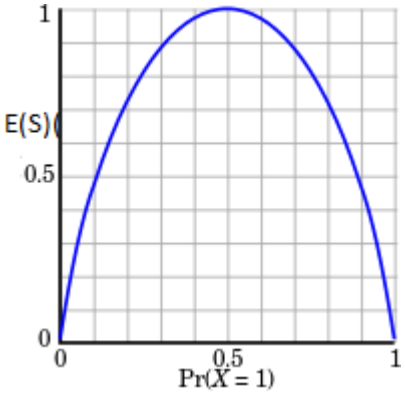
$$Entrop(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

where p, is the proportion of positive examples in S and p, is the proportion of negative examples in S. In all calculations involving entropy we define 0log0 to be 0.

To illustrate, suppose S is a collection of 14 examples of some Boolean concept, including 9 positive and 5 negative examples (we adopt the notation [9+, **5-1** to summarize such a sample of data). Then the entropy of S relative to this boolean classification is

$Entrop(9+,5-) = -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.940$

Notice that the entropy is 0 if all members of S belong to the same class. For example, if all members are positive (pe = I), then p, is 0, and Entropy(S) = -1 .log2(1)- 0 .log20 = -1 . 0 - 0 .log20 = 0. Note 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 3.2 shows the form of the entropy function relative to a boolean classification, as **p,** varies between **0** and 1



Thus far we have discussed entropy in the special case where the target classification is boolean. More generally, if the target attribute can take on c different values, then the entropy of ***S*** relative to this c-wise classification is defined as

$$Entrop(S) \equiv -p_i \log_2 p_i$$

where p_i ***i*** is the proportion of ***S*** belonging to class ***i*** **.** Note the logarithm is still base 2 because entropy is a measure of the expected encoding length measured in ***bits.*** Note also that if the target attribute can take on c possible values, the entropy can be as large as log, p .

INFORMATION GAIN MEASURES THE EXPECTED REDUCTION IN ENTROPY

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

where ***Values(A)*** is the set of all possible values for attribute A, and ***S,*** is the subset of S for which attribute A has value v (i.e., S_v = ***{ s ∈ S/A (s) = v))***

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
  - $A \leftarrow$ the attribute from *Attributes* that best* classifies *Examples*
  - The decision attribute for *Root* $\leftarrow A$
  - For each possible value, v_i, of A,
    - Add a new tree branch below *Root*, corresponding to the test $A = v_i$
    - Let $Examples_{v_i}$ be the subset of *Examples* that have value v_i for A
    - If $Examples_{v_i}$ 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($Examples_{v_i}$, $Target\_attribute$, $Attributes - \{A\}$))
- End
- Return *Root*

We can summarize the ID3 algorithm as illustrated below

$$Entrop(S) \equiv -p_i \log_2 p_i$$

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

We need to calculate the entropy first. Decision column consists of 14 instances and includes two labels: yes and no. There are 9 decisions labeled yes, and 5 decisions labeled no.

$$Entrop(9+, 5-) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

For each attribute, the gain is calculated and the highest gain is used in the decision node. The weather attributes are outlook, temperature, humidity, and wind speed. They can have the following values:

outlook = { sunny, overcast, rain }
temperature = {hot, mild, cool }
humidity = { high, normal }
wind = {weak, strong }
Now, we need to find the most dominant factor for decision.

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

E(PlayGolf, Outlook) = **P**(Sunny)*__E__(3,2) + **P**(Overcast)*__E__(4,0) + **P**(Rainy)*__E__(2,3)

= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971

= 0.693

G(PlayGolf, Outlook) = **E**(PlayGolf) – **E**(PlayGolf, Outlook)

= 0.940 – 0.693 = 0.247

Wind factor on decision

Gain(S,Wind)=Entropy(S)-(8/14)*Entropy(Sweak)-(6/14)*Entropy(Sstrong)

Entropy(S_{weak}) = - (6/8)*$\log_2$(6/8) - (2/8)*$\log_2$(2/8) = 0.811

Entropy(S_{strong}) = - (3/6)*$\log_2$(3/6) - (3/6)*$\log_2$(3/6) = 1.00

Gain(S,Wind)=Entropy(S)-(8/14)*Entropy(Sweak)-(6/14)*Entropy(Sstrong)

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

= 0.048

Humidity factor on decision

Gain(S,Humidity)=Entropy(S)-(7/14)*Entropy(S_{High})-(7/14)*Entropy(S_{Normal})

Entropy(S_{High}) = - (3/7)*$\log_2$(3/7) - (4/7)*$\log_2$(4/7) = 0.524+0.461=0.985

Entropy(S_{Normal}) = - (6/7)*$\log_2$(6/7) - (1/7)*$\log_2$(1/7) = 0.19+0.4009=0.5909

Gain(S, Humidity)=0.940-(7/14)*0.985-(7/14)*0.5909=**0.152**

Temperature factor on decision

Gain(S, Temperature)=

Entropy(S)-(4/14)*Entropy(S_{Hot})-(6/14)*Entropy(S_{Mild})- (4/14)Entropy(S_{Cool})

Entropy(S_{Hot}) = 1

Entropy(S_{Mild}) = - (4/6)*$\log_2$(4/6) - (2/6)*$\log_2$(2/6)

= 0.667*log(0.667)-0.333*log(0.333)

=0.3905+0.528=0.918

Entropy(S_{Cool})=-(3/4)log(3/4)-(1/4)log(1/4)= 0.3112+0.5=0.8112

Gain(S, Temperature)=0.940-(4/14)*1-(6/14)*0.918-(4/14)*0.8112
=0.940-0.286-0.3934-0.2312**=0.029**

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

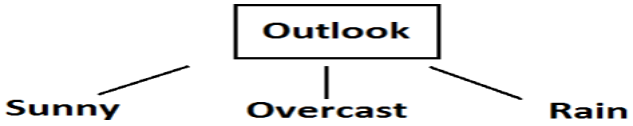
		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		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			

1- Gain(Decision, Outlook) = 0.246
2- Gain(Decision, Temperature) = 0.029
3- Gain(Decision, Humidity) = 0.151
4. Gain(S, Temperature)= =0.029

As seen, outlook factor on decision produces the highest score. That's why, outlook decision will appear in the root node of the tree.



Root decision on the tree

Now, we need to test dataset for custom subsets of outlook attribute.

Overcast outlook on decision

Basically, decision will always be yes if outlook were overcast.

Day	Outlook	Temp.	Humidity	Wind	Decision
3	Overcast	Hot	High	Weak	Yes
7	Overcast	Cool	Normal	Strong	Yes

12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes

Sunny outlook on decision

Day	**Outlook**	**Temp.**	**Humidity**	**Wind**	**Decision**
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Here, there are 5 instances for sunny outlook. Decision would be probably 3/5 percent no, 2/5 percent yes.

Gain(Outlook=Sunny, Temperature)
=Entropy(Sunny)-(2/5)*Entropy($Temp_{hot}$)-(1/5)Entropy($Temp_{Cool}$)-(2/5)Entropy($Temp_{Mild}$)

Entropy(Sunny)=[2+,3-]=-2/5*log(2/5)-3/5*log(3/5)=0.528+0.442=0.970

Entropy($Temp_{hot}$=0

Entropy($Temp_{Cool}$)=0

Entropy($Temp_{Mild}$)=1

Gain(Outlook=Sunny, Temperature)=0.970-(2/5)*0-(1/5)*0-(2/5)*1=0.97-0.4*1
=0.97-0.4=**0.570**

Gain(Outlook=Sunny, Humidity)
=Entropy(Sunny)-(3/5)*Entropy($Humidity_{High}$)-(2/5)Entropy($Humidity_{Normal}$)

Entropy($Temp_{high}$)=E[0+,3-]=0

Entropy($Temp_{Normal}$)=E[2+,0-]=0

Gain(Outlook=Sunny, Humidity)=**0.970**

Gain(Outlook=Sunny, Wind)
= Entropy(Sunny)-(2/5)*Entropy($Wind_{False}$)-3/5*Entropy($Wind_{True}$)
= 0.970-(2/5)*1-(3/5)*(0.918)=0.970-0.4-0.550= 0.970-0.4-0.5508=0.019

Gain(Outlook=Sunny, Wind) = **0.019**

1- Gain(Outlook=Sunny|Temperature) = 0.570
2- Gain(Outlook=Sunny|Humidity) = 0.970
3- Gain(Outlook=Sunny|Wind) = 0.019
Now, humidity is the decision because it produces the highest score if outlook were sunny.
At this point, decision will always be no if humidity were high.

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No

On the other hand, decision will always be yes if humidity were normal

Day	Outlook	Temp.	Humidity	Wind	Decision
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Finally, it means that we need to check the humidity and decide if outlook were sunny.

Rain outlook on decision

Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
10	Rain	Mild	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

1- Gain(Outlook=Rain | Temperature)= E(rain)-3/5*E(T_{Mild0})-2/5E(T_{Cool})
=0.970-3/5*0.918-(2/5)*1 =0.970-0.5508-0.4=**0.0192**
2- Gain(Outlook=Rain | Humidity)=E(Rain)-3/5*E(H Normal)-2/5*E(H High)=**0.192**

3- Gain(Outlook=Rain | Wind)=E(Rain)-3/5*E(W Week)-2/5*E(W Strong)=0.970-0-0=**0.970**

Here, wind produces the highest score if outlook were rain. That's why, we need to check wind attribute in 2nd level if outlook were rain.

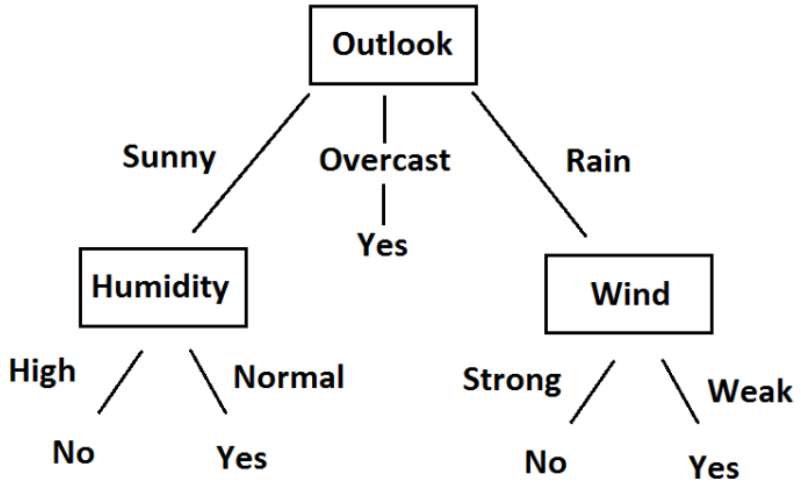
So, it is revealed that decision will always be yes if wind were weak and outlook were rain.

Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes

What's more, decision will be always no if wind were strong and outlook were rain.

Day	Outlook	Temp.	Humidity	Wind	Decision
6	Rain	Cool	Normal	Strong	No
14	Rain	Mild	High	Strong	No

So, decision tree construction is over. We can use the following rules for decisioning.



Final version of decision tree

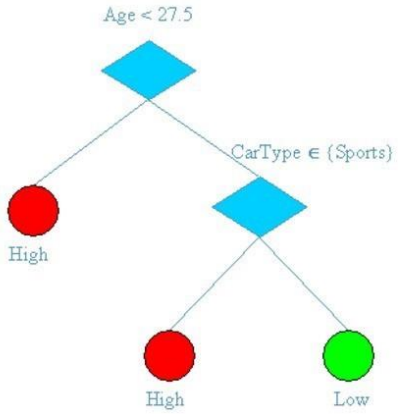
So, decision tree algorithms transform the raw data into rule based mechanism. In this post, we have mentioned one of the most common decision tree algorithm named as ID3. They can use nominal attributes whereas most of common machine learning algorithms cannot. However, it is required to transform numeric attributes to nominal in ID3. Besides, its evolved version C4.5exists which can handle nominal data. Even though decision tree algorithms are powerful, they have long training time. On the other hand, they tend to fall over-fitting.

Besides, they have evolved versions named random forests which tend not to fall over-fitting issue and have shorter training times.

Problem 2:

Tid	Age	Car Type	Class
0	23	Family	High
1	17	Sports	High
2	43	Sports	High
3	68	Family	Low
4	32	Truck	Low
5	20	Family	High

Numeric Categorical



1) Age < 27.5 ⇒ High

2) Age >= 27.5 and CarType = Sports ⇒ High

3) Age >= 27.5 and CarType ≠ Sports ⇒ High

3.5 HYPOTHESIS SPACE SEARCH IN DECISION TREE LEARNING

ID3 can be characterized as **searching a space of hypothesis** 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 hypothesis in search of a decision tree that correctly classifies the training data. The evaluation function that guides this hill-climbing search is the **information gain measure.** This search is depicted in **Figure 3.5.**

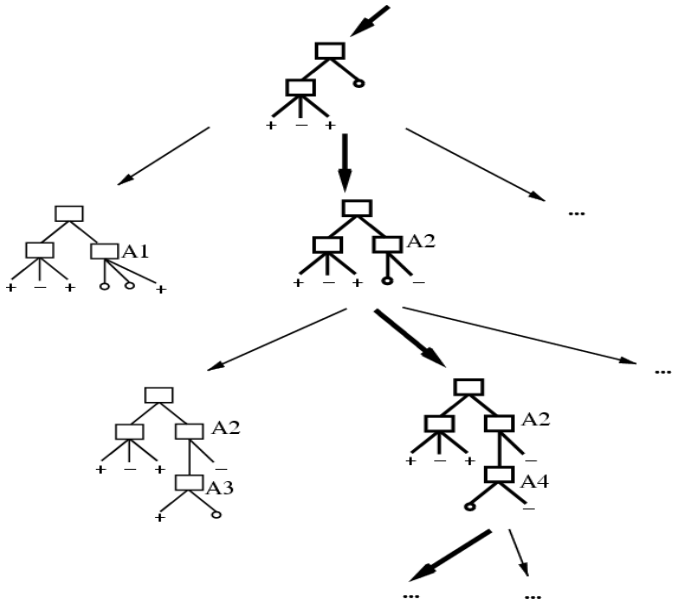


Figure 3.5

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 (such as methods that consider only conjunctive hypotheses): 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. This contrasts, for example, with the earlier version space candidate-elimination method, which maintains the set of ***all*** hypotheses consistent with the available training examples.
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.
4. **ID3** uses all training examples at each step in the search to make statistically based decisions regarding how to refine its current hypothesis. This contrasts with methods that make decisions incrementally, based on individual training examples (e.g., FIND-S or CANDIDATE-ELIMINATOION). One advantage of using statistical properties of all the examples (e.g., information gain) 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.

ISSUES IN DECISION TREE LEARNING

We will say that a hypothesis overfits the training examples if some other hypothesis that fits the training examples less well actually performs better over the entire distribution of instances (i.e., including instances beyond the training set)

Definition: Given a hypothesis space H, a hypothesis $h \varepsilon H$ is said to **overfit** the training data if there exists some alternative hypothesis h'εH, 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.

ISSUES IN DECISION TREE LEARNING

Practical issues in learning decision trees include

1. determining how deeply to grow the decision tree,
2. handling continuous attributes,
3. choosing an appropriate attribute selection measure,
4. handling training data with missing attribute values,
5. handling attributes with differing costs
6. improving computational efficiency.

Module Questions.

1. Give decision trees to represent the following boolean functions:

(a) A ∧˜B

(b) A V [B ∧ C]

(c) A XOR B

(d) [A ∧ B] v [C ∧ D]

2. Consider the following set of training examples:

Instance	Classification	a1	a2
1	+	T	T
2	+	T	T
3	-	T	F
4	+	F	F
5	-	F	T
6	-	F	T

(a) What is the entropy of this collection of training examples with respect to the target function classification?

(b) What is the information gain of a2 relative to these training examples?

3. NASA wants to be able to discriminate between Martians (M) and Humans (H) based on the following characteristics: Green ∈{N, Y} , Legs ∈{2,3} , Height ∈{S, T}, Smelly ∈{N, Y} Our available training data is as follows:

	Species	Green	Legs	Height	Smelly
1	M	N	3	S	Y
2	M	Y	2	T	N
3	M	Y	3	T	N
4	M	N	2	S	Y
5	M	Y	3	T	N
6	H	N	2	T	Y
7	H	N	2	S	N
8	H	N	2	T	N
9	H	Y	2	S	N
10	H	N	2	T	Y

a) Greedily learn a decision tree using the ID3 algorithm and draw the tree.

b) (i) Write the learned concept for Martian as a set of conjunctive rules (e.g., if

(green=Y and legs=2 and height=T and smelly=N), then Martian; else if ... then Martian;...; else Human).

(ii) The solution of part b)i) above uses up to 4 attributes in each conjunction. Find a set of conjunctive rules using only 2 attributes per conjunction that still results in zero error in the training set. Can this simpler hypothesis be represented by a decision tree of depth 2? Justify.

4. Discuss Entropy in ID3 algorithm with an example
5. Compare Entropy and Information Gain in ID3 with an example.
6. Describe hypothesis Space search in ID3 and contrast it with Candidate-Elimination
7. Write the steps of ID3Algorithm
8. Explain the various issues in Decision tree Learning