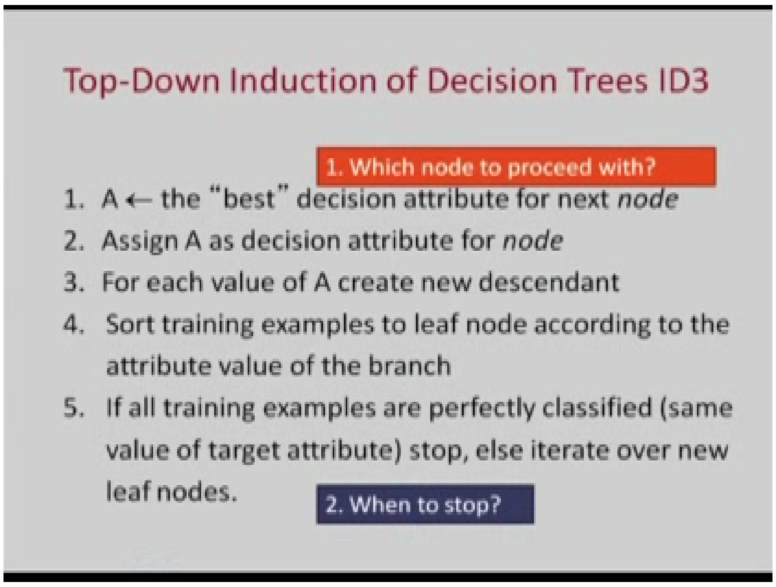
Introduction to Machine Learning

Prof. Sudeshna Sarkar
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Module - 2
Lecture - 07
Learning Decision Tree

Good Morning. Welcome to the next lecture of this class, in the module of Decision Trees. In the last class we introduced decision tree, today we will look at some Learning Algorithm to Learn Decision Tress. So let us recapitulate the basic outline of the greedy decision tree algorithm that we discussed in the last class.

(Refer Slide Time: 00:42)



Top-Down Induction of Decision Trees ID3

1. Which node to proceed with?

1. A ← the "best" decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A create new descendant
4. Sort training examples to leaf node according to the attribute value of the branch
5. If all training examples are perfectly classified (same value of target attribute) stop, else iterate over new leaf nodes.

2. When to stop?

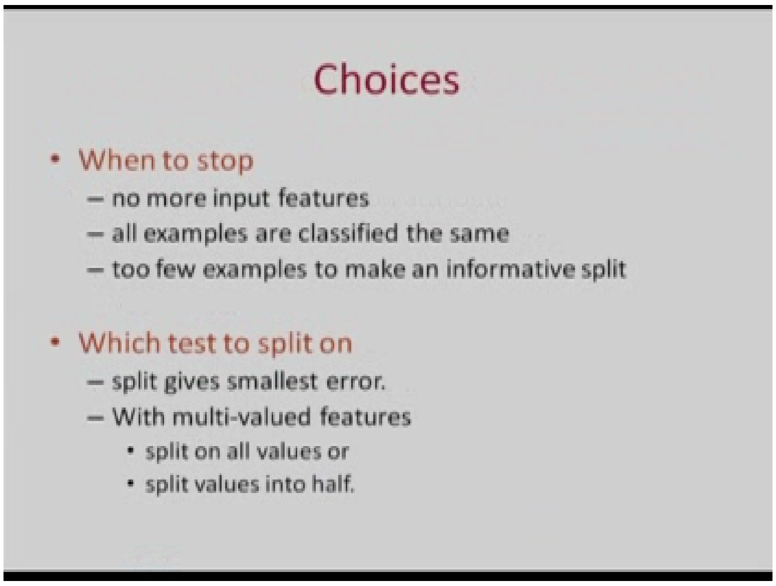
As we said that we start with number of training examples. Let us sat d is the set of training examples. Now we want to decide a test for the route node of the decision tree.

(Refer Slide Time: 00:57)

So, we come up with we want to find out, suppose A is the set of attributes and then there are P attributes. We want to check, we want to find one of the attributes based on which we will make decision of the route node. So let A be the attribute that we have selected. Now based on that attribute I will look at the different values of the attribute, suppose A test two values and then corresponding to that we will have two children of A and the set up training examples D which is associated initially with route A that will be split into D 1 and D 2. Suppose, this particular branch corresponds to A equal to true, this corresponds to A equal to false, so D 1 will contain those examples from which A 1 equal to true, D 2 will contain those examples for which A equal to false.

Now at this test when we look at D 1 we have to either decide that we will stop growing the decision tree at this point or we have to select an attribute and we recursively go on. And the basic outline of the algorithm is given in the slide. Now as we discussed in the last class there are two decisions we have to take. Now out of these two open nodes we have to decide which one we should start with, and then we have to decide. If we decide to start with a particular node we have to decide at that node whether we should stop or whether we should grow the tree. If we grow the tree we have to decide which attribute to split out. These are the choices that one has to make when on goes for a decision tree.

(Refer Slide Time: 03:06)

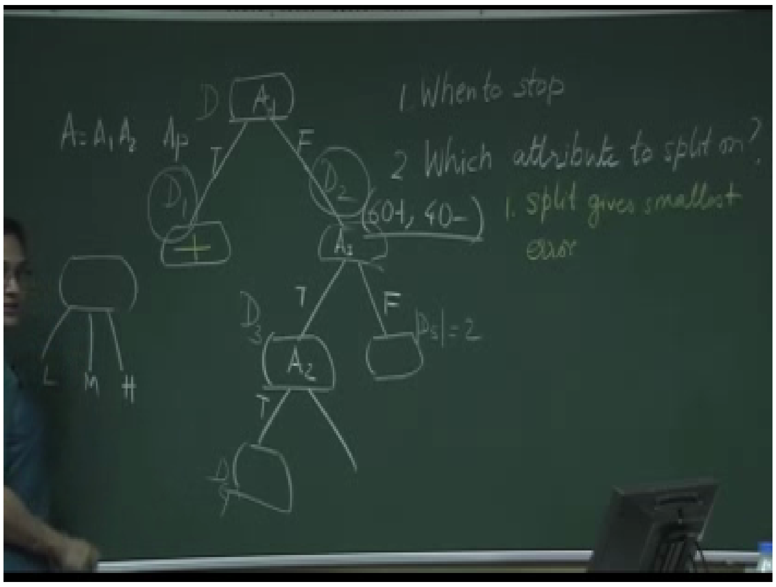


Choices

- When to stop
  - no more input features
  - all examples are classified the same
  - too few examples to make an informative split
- Which test to split on
  - split gives smallest error.
  - With multi-valued features
    - split on all values or
    - split values into half.

Now, let us see how we take those decisions. First of all let us look at the decision about when to stop.

(Refer Slide Time: 03:10)



There are several cases where you may have to stop, first of all if you considered that there are three features or three attributes and at a particular point suppose you have chosen A 1 the feature A 1 and based on that you have A 1 equal to first you have got this split, and then suppose you have chosen A 3 and based on A 3 equal to true you have got this split, and here suppose you have chosen A 2 and based on A 2 equal to true or false you have got this split.

Suppose, here we have D 2 then here we have D 3 and here we have D 4. Now, if it is the case that all the 3 attributes have been used up; this particular example contain A 1 equal to falls D 2 equal to true A 2 equal to true. All this things are given here, so we have no more attributes to split on because we have exhausted the attributes. If you have exhausted the attributes we have to stop here, we have though option.

The second reason where we may stop if we find at this place that all the examples in D 1 have the value plus or have the value minus if all examples of D 1 have the same value for the target attribute we can immediately stop their and label this node as a leaf node. Suppose all of them plus we will label with the class of the attribute that we have. Or we must stop if the examples that we have got here too few, suppose D 5 is the set of examples we have and D 5 is a let us say 2, we have two few examples then also we should stop. These are the three of the possible stopping criteria.

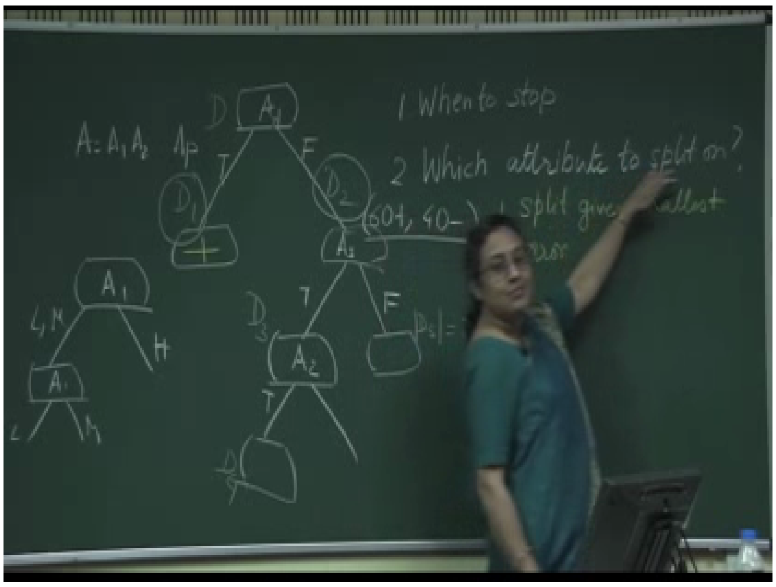
Next we will look at the decision about which node to split on, rather which attribute to split on that is the test that we will apply at a particular node. For example, here we have the choice of using attributes A 2 or A 3, here we have the choice of using attributes A 1 A 2 or A 3, which attribute should we choose. So, earlier we have talked about a bias criteria which says that we want to choose a very simple function we prefer simpler functions or we can say in the context of decision trees we prefer smaller decision trees. We could think of a heuristic method of choosing the attribute so that based on this choice of the attribute the decision tree is expected to be smaller.

Another thing that we could do is that we could think of, if you haves an examples here let us say D 2 and suppose it is A 2 class problem D 2 may have mixture of positive and negative examples. If D 2 has only one type of example we can stop at that node, if D 2 has a mixture of positive and negative examples we can looked at you know if you say that if you did stop a D 2 we will out put the majority class. Suppose D 2 has 60 positive examples and 40 negative examples so if we had stooped D 2 we could have said D 2 is positive and then we would have made an error and that error is 40 percent.

So, we can stop so that the split gives the smallest error. And there are slightly more sophisticated methods based on which we can do this which we will discuss presently. And if you have multi valid features, so this is one of the things that we can do and we will see more on this.

Now in this example our attributes has two values, if the attributes have multiple values then there are two choices; one is that we could take that attribute and suppose that attribute has 4 values we could have 4 children. But you know sometimes the attribute can be real valued and the number of children can be extremely large. If you have few children then we could have multiple valued attributes or we could split those values into half so that we have two children.

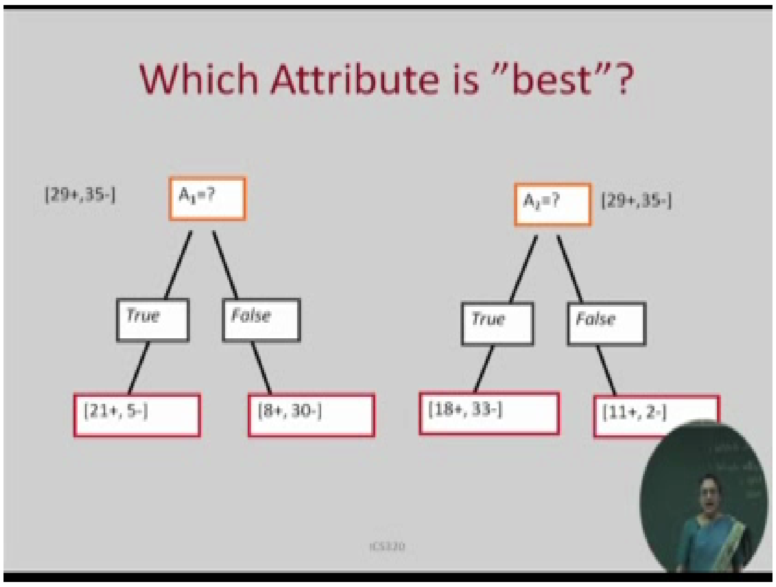
(Refer Slide Time: 08:54)



Suppose, a particular attribute has the value low, medium, high, so we could have 3 children corresponding to these 3 values low, medium, high or we could have 2 children and then we could say this is low and medium and this is high. If you use attribute A 1 as low and medium comes here, in a feature below this node we could also again split and A 1 let us say A 1 is low A 1 is medium.

So, for valued attribute when we use less number of splits we can use the attribute again to split that node. Now let us look at how we choose an attribute some more principle criteria about which attribute to split on.

(Refer Slide Time: 09:59)



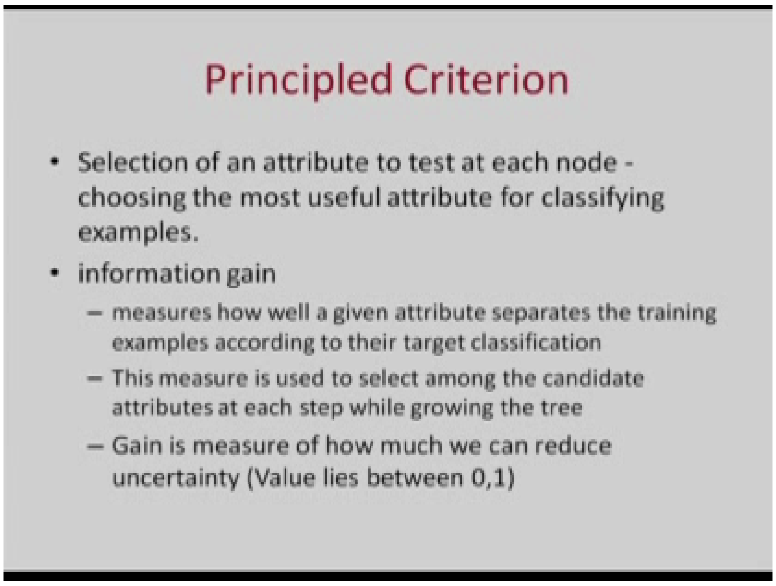
Which Attribute is "best"?

Now, let us look at this slide to look at two examples. We have some training examples, let us say 64 training examples here and if you choose attribute A 1 to split on, A 1 is a binary attribute having true and false then if for A 1 equal to true we get 26 examples on this side and 38 example for A 1 equal to false. Out of this 26 examples 21 is positive 5 is negative. For A 1 equal to false 8 or positive 30 are negative. However, if you split on A 2 for A 2 equal to true we have 18 positive, 33 negative. For A 2 equal to false we have 11 positive and 2 negative.

Now based on this information we want to decide whether A 1 or A 2 is a better candidate first splity. To given answer to this there are multiple methods that one could use and we will introduce one very popular method which is based on entropy and information gain. So, entropy as you know is a measure of disorder in a system. If it a particular node all the examples have positive or all the examples are negative that is all examples belong to the same class then it is homogeneous set of examples and entropy is 0, entropy is low. However, if we have 2 class and all the examples half belong to one class, half belong to another class then entropy is highest.

Now leaf node is one ideally where all the examples belong to the same class that is entropy is low. So we would prefer that we quickly get nodes which have low entry. So based on this we have a heuristic to decide which attribute to split on.

(Refer Slide Time: 12:17)



Principled Criterion

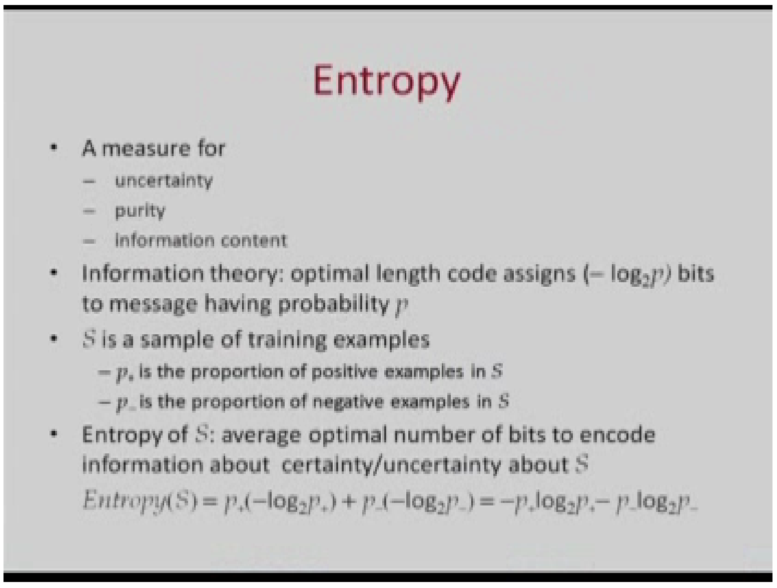
- Selection of an attribute to test at each node - choosing the most useful attribute for classifying examples.
- information gain
  - measures how well a given attribute separates the training examples according to their target classification
  - This measure is used to select among the candidate attributes at each step while growing the tree
  - Gain is measure of how much we can reduce uncertainty (Value lies between 0,1)

Now, when we want to select an attribute we want to choose the most useful attribute. And one of the criteria we use to decide what is the most important attribute is the criteria of information gain. Now what is the information? So if at a particular place we have some training examples where half of positive half is negative and if we are given a random example and ask to predict its class, we have to make a random guest we have no information. If the examples are equally divided among the classes there is no information there, but if all the examples belong to class and we identify the set up examples with that class information is very high.

So, we want to have definition of information and we want to prefer situation where information is high. Now when we decide which attribute to split on we may use the principle of information gain. And this principle, if you can look at this slide this principle for information gain measures how well a given attribute separate the training examples according to their target classification.

So if all the examples have same target classification information is high and information gain is high. Suppose, 90 percent belong to one class then also information is quiet high, but if 50 percent belongs to one class 50 percent to then information is low. The measure of information gain which we will define is used to select among the candidate attributes at each step while growing the decision tree. And the gain is a measure of how much we can reduce uncertainty. If the examples belong to the same class there is no uncertainty, if the examples us spread among the classes almost uniformly there is high uncertainty.

(Refer Slide Time: 14:43)



Entropy

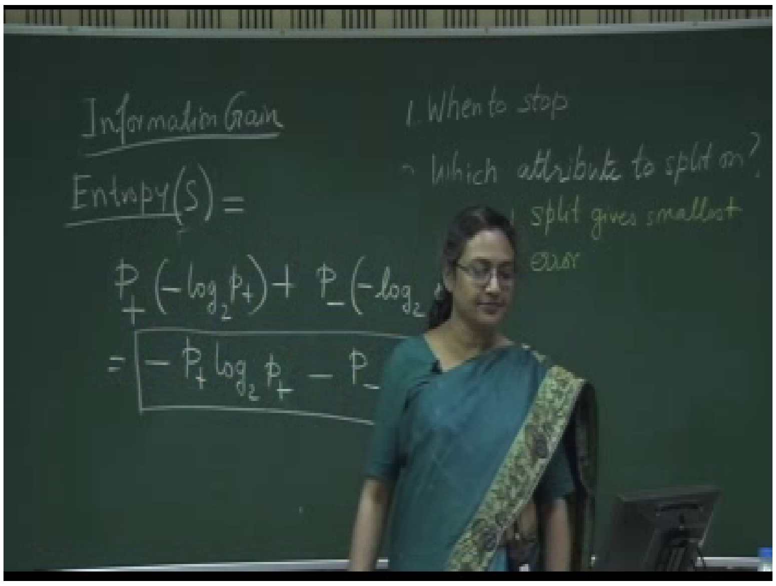
- A measure for
  - uncertainty
  - purity
  - information content
- Information theory: optimal length code assigns ($-\log_2 p$) bits to message having probability p
- S is a sample of training examples
  - p_+ is the proportion of positive examples in S
  - p_- is the proportion of negative examples in S
- Entropy of S: average optimal number of bits to encode information about certainty/uncertainty about S

$$Entropy(S) = p_+(-\log_2 p_+) + p_-(-\log_2 p_-) = -p_+\log_2 p_+ - p_-\log_2 p_-$$

Now, based on this we will define entropy. First we will define entropy and in terms of entropy we can define information gain. Entropy is a measure of purity, pure (Refer Time: 14:58) of examples will have entropy 0, it can also with thought about as a measure of uncertainty or a measure of information content. Entropy is a very standard term which is used in various domains in thermodynamics, information theory etcetera.

In information theory just as a side, in an information theory the optimum length code assigns minus log p bites. So if you have p positive examples and p negative examples. So you have p positive examples and 1 minus p negative examples. And you want to come up with a code then in information theory the optimum code will use minus log p number of bites to send a message having probability p. We use the idea of entropy for decision trees and entropy is defined as the average optimum number of bites to encode information about certainty, uncertainty about s. This is the background of entropy.

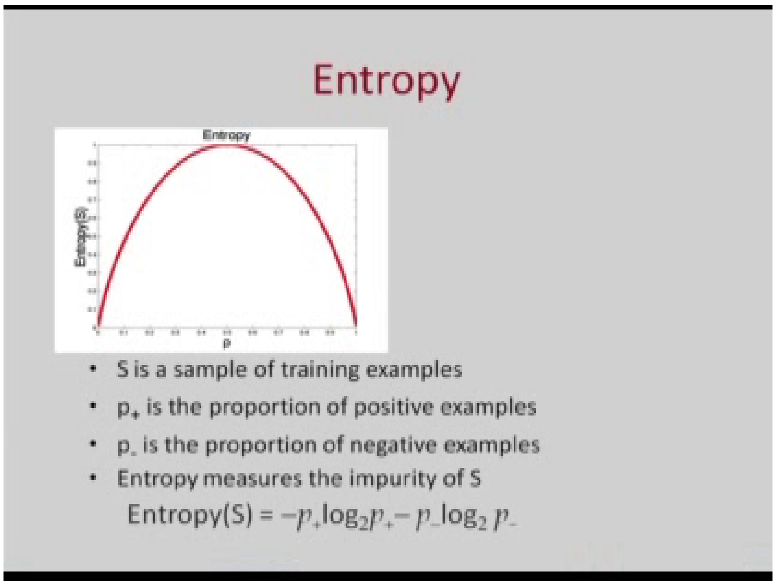
(Refer Slide Time: 16:30)


Information Gain
Entropy(S) =
$p_+(-\log_2 p_+) + p_-(-\log_2$
$= -p_+ \log_2 p_+ - p_-$
1. When to stop
2. Which attribute to split on?
split gives smallest error

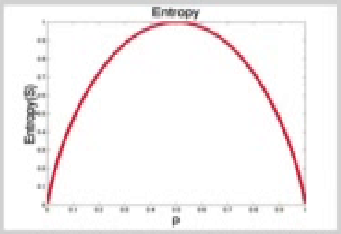
And entropy is defined as; entropy of S. S is a sample that is a set up examples. It is defined as p plus, p plus is the fraction of positive examples in this sample. p plus minus log to the base 2 p plus plus p minus, p minus is the fraction of negative examples in the sample, p minus negative of log 2 p minus. Or we can write this us minus p plus log 2 of p plus minus p minus log 2 of p minus. This is the definition of entropy.

So, if it is homogeneous set, then suppose p plus is 1 and p minus is 0. In that case entropy will be 0. If p plus is 0, p minus is 1 then also entropy will be 0. And entropy that is the values of this function is highest when p plus is equal to p minus equal to half.

(Refer Slide Time: 18:00)



Entropy

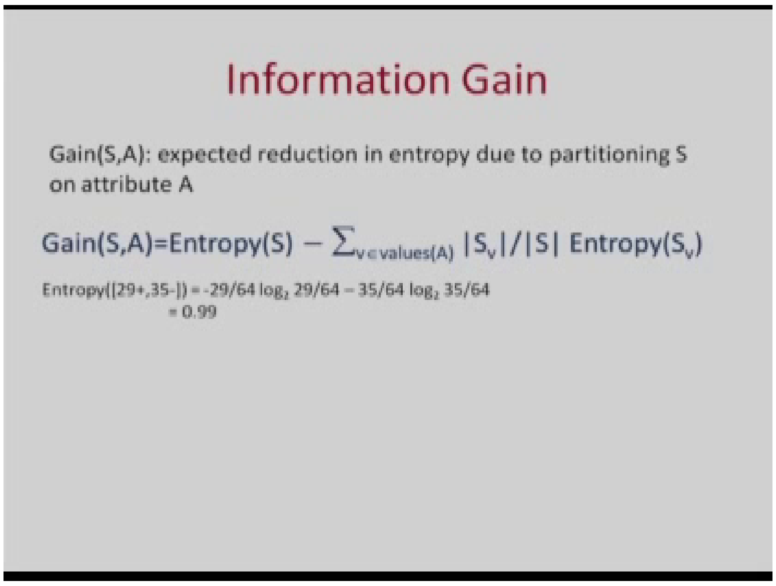


- S is a sample of training examples
- p_+ is the proportion of positive examples
- p_- is the proportion of negative examples
- Entropy measures the impurity of S

$$\text{Entropy}(S) = -p_+\log_2 p_+ - p_-\log_2 p_-$$

Now, if you look at this slide. This slide shows with the value of p, what is the value of entropy. We can take e equal to p plus in that case p minus equal to 1 minus p. So when p equal to - 0 entropy is 0, when p equal to 1 entropy is 0, and when p equal to half entropy is 1. So, you put half here what you get minus half log 2 half in to 2 and this is equal to 1. The highest value of entropy is 1 when p plus equal to p minus equal to half and this is the shape of the entropy curve. The entropy is 0 is the out coming certain, the entropy is maximum if you have no knowledge of the system.

(Refer Slide Time: 19:01)



Information Gain

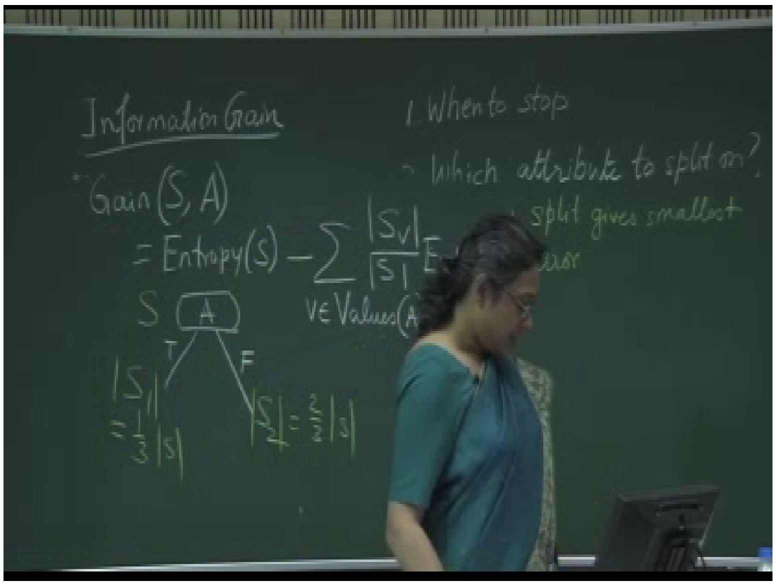
Gain(S,A): expected reduction in entropy due to partitioning S on attribute A

$$\text{Gain}(S,A)=\text{Entropy}(S) - \sum_{v\in \text{values}(A)} |S_v|/|S|\ \text{Entropy}(S_v)$$

$\text{Entropy}([29+,35-]) = -29/64 \log_2 29/64 - 35/64 \log_2 35/64$
$= 0.99$

Now based on entropy we can now define Information Gain.

(Refer Slide Time: 19:19)

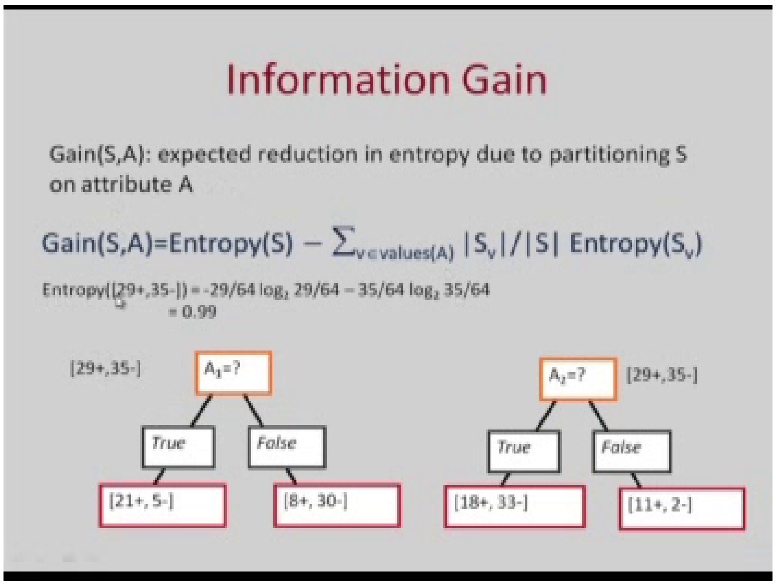


Information Gain
1. When to stop
- Which attribute to split on?
split gives smallest
$\text{Gain}(S,A) = \text{Entropy}(S) - \sum_{v\in \text{Values}(A)} \frac{|S_v|}{|S|} E$
S
A
T
F
$|S_1| = \frac{1}{3}|S|$
$|S_2| = \frac{2}{3}|S|$

Information gain of a sample S with respect to an attribute A is defined to be, this is the original entropy and the new entropy of the system if you split on attribute A. Suppose, if you split on attribute A and you get two different values of A, then for every value of A this is general for all the multi values. So for v included in all the values that A takes, the entropy for a particular class this is the fraction so S v by S is the fraction of examples that have value or which the attribute has the value v. So, this is S v by S entropy of S v.

So what does it mean? Suppose, you have the number of training examples at this node of the decision tree and you split on attribute A and A has two values true or false, and you have S 1 here S 2 here, Suppose, fraction of S 1 equal to size of S 1 equal to one third of S and size of S 2 is two third of S. Then S 1 has an entropy S 2 has an entropy and the resultant entropy is one third in to the entropy of S 1 plus two third into entropy of S 2. And information gain is the original entropy of S minus the resulting entropy.

(Refer Slide Time: 21:27)



Information Gain

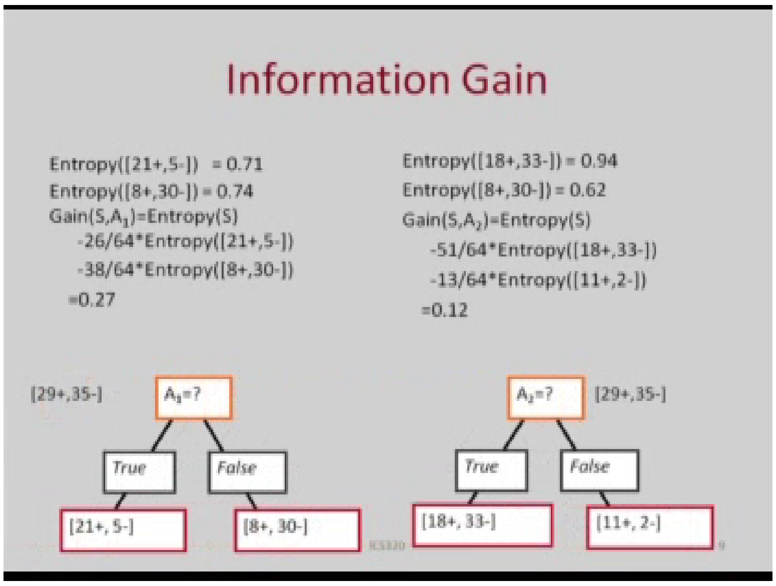
Gain(S,A): expected reduction in entropy due to partitioning S on attribute A

$$\text{Gain}(S,A)=\text{Entropy}(S) - \sum_{v\in \text{values}(A)} |S_v|/|S| \text{ Entropy}(S_v)$$

Entropy([29+,35-]) = -29/64 $\log_2$ 29/64 – 35/64 $\log_2$ 35/64
= 0.99

In this particular case that we had seen that if we split on attribute A 1 and if you split on attribute A 2, which one we should prefer, if we use this measure of information gain. So entropy of S, S is the original sample. In this case S has 29 positive and 35 negative examples and if you do the computation entropy is 0.99. Now we have to find out what is the information gain if you use A 1 and information gain if you use A 2.

(Refer Slide Time: 22:00)



Information Gain

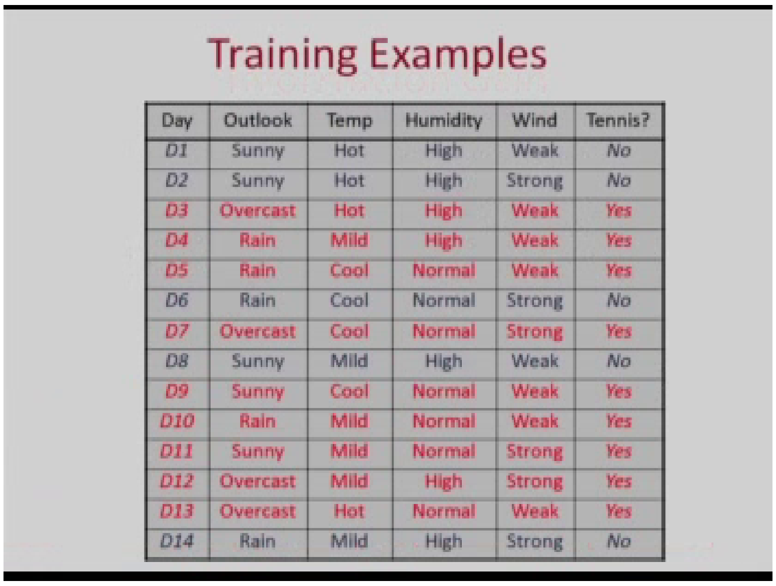
Entropy([21+,5-]) = 0.71
Entropy([8+,30-]) = 0.74
Gain(S,A_1)=Entropy(S)
-26/64*Entropy([21+,5-])
-38/64*Entropy([8+,30-])
=0.27

Entropy([18+,33-]) = 0.94
Entropy([8+,30-]) = 0.62
Gain(S,A_2)=Entropy(S)
-51/64*Entropy([18+,33-])
-13/64*Entropy([11+,2-])
=0.12

[29+,35-]
A_1=?
True
False
[21+, 5-]
[8+, 30-]
A_2=?
[29+,35-]
True
False
[18+, 33-]
[11+, 2-]

And this is worked out here. In the case of A 1 the entropy of S 1 is 0.71, entropy of S 2 is 0.74, and gain S A 1 can be computed as using the formula and not working it out but you can follow the slide information gain is 0.27. Whereas, if you use the attribute A 2 first splitting, the entropy of S 1 is 0.94, entropy of S 2 is 0.62 and the gain is 0.12. So where is the gain highest? The gain is highest for A 1 compared A 2. According to this measure we will use A 1 for splitting rather than A 2.

(Refer Slide Time: 22:29)



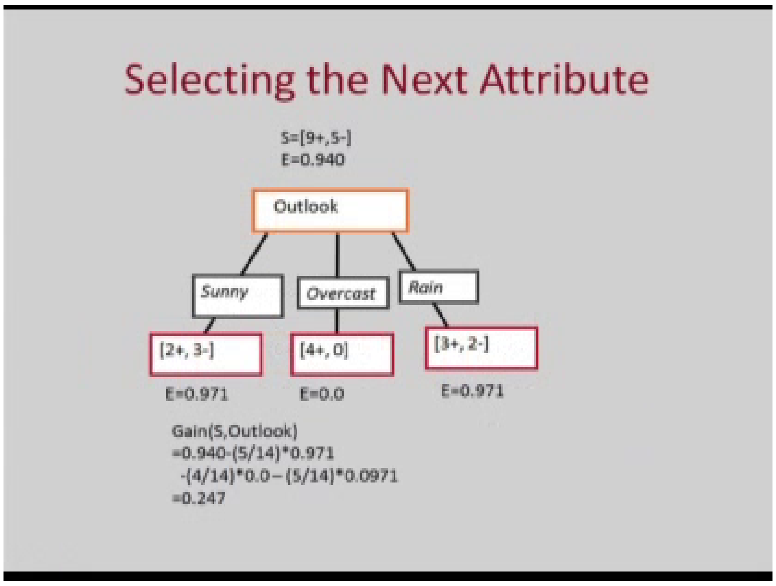
Training Examples

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

So, A 1 get selected because it has higher information gain, that is the reduction in entropy is more for A 1. You want to reduce the entropy because you want smaller decision trees. Let us look at an example.

The slide shows as set of training examples or yesterday in the last class we had introduced a decision tree to decide whether the proponent once to play tennis given the different parameters of the day; outlook, temperature, humidity, wind, etcetera, and this is the data set. In this data set if you want to decide at the route node which attribute we want to split on, let us humidity and wind at two possibilities. The gain for humidity is 0.151, gain for wind is 0.048. So we will prefer to split on humidity rather than a wind.

(Refer Slide Time: 24:00)

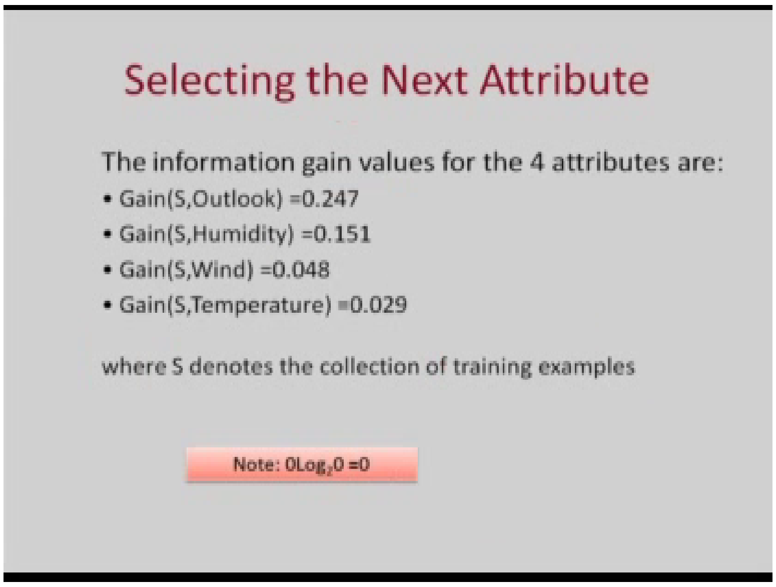


Selecting the Next Attribute

Gain(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.0971
=0.247

Similarly, we look at the third attribute call outlook. For outlook the gain is 0.247, it is worked out in the slide you can now look at it. So, outlook is preferred to either humidity or wind. Among this three outlook seems to be the boast promising.

(Refer Slide Time: 24:21)



Selecting the Next Attribute

The information gain values for the 4 attributes are:

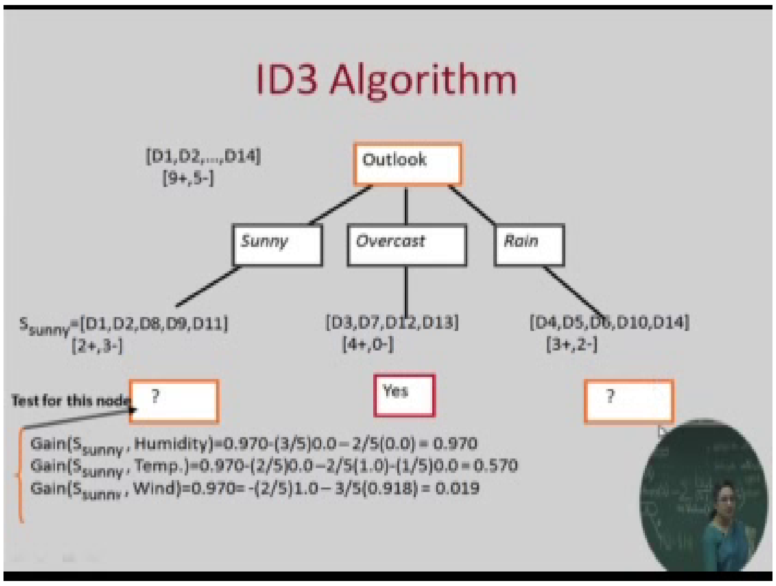
- Gain(S,Outlook) =0.247
- Gain(S,Humidity) =0.151
- Gain(S,Wind) =0.048
- Gain(S,Temperature) =0.029

where S denotes the collection of training examples

Note: $0\text{Log}_2 0 = 0$

If you do it for the all four attributes also do it for temperature. Temperature has a gain of 0.029. This slide show for that training example the gain for various attributes and we see the gain for outlook is the highest. And so if you using this measure of information gain we will use outlook as the route test for the decision tree.

(Refer Slide Time: 24:51)

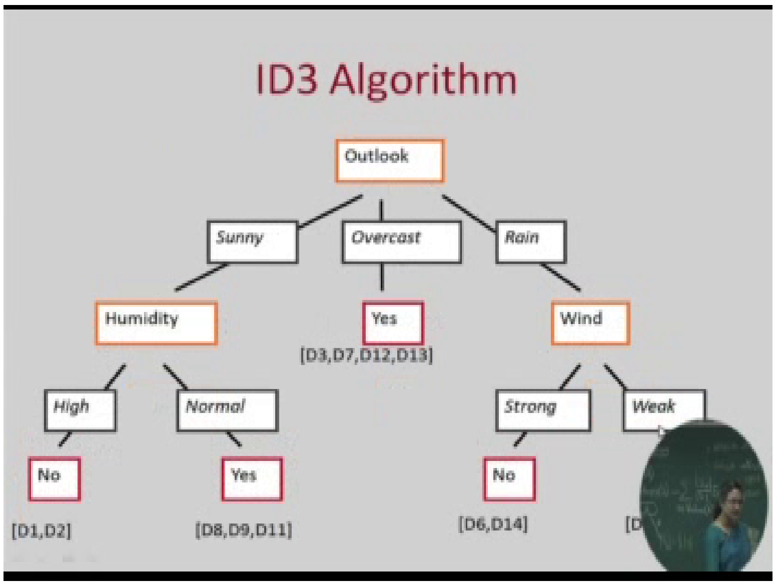

ID3 Algorithm

Gain(S_{sunny}, Humidity)=0.970-(3/5)0.0 – 2/5(0.0) = 0.970
Gain(S_{sunny}, Temp.)=0.970-(2/5)0.0 –2/5(1.0)-(1/5)0.0 = 0.570
Gain(S_{sunny}, Wind)=0.970= -(2/5)1.0 – 3/5(0.918) = 0.019

So, this ID3 algorithm given by (Refer Time: 24:54) it will select outlook has the route on this 15 training example. Then outlook has three different values, and then you take outlook equal to sunny. For outlook equal to sunny again you have to find out which node to split on. Since, outlook has already being used up we have a choice of three different attributes; humidity, temperature, and wind. And this slide shows the computation of the gain of these three attributes.

We see the gain of humidity is 0.97, for temperature 0.57, and for wind 0.01. We will use humidity as the test in this node. For outlook equal to over cast we see all the examples here a positive, so we will make it a leaf node. And outlook equal to rainy there are 3 positive and 2 negative examples. Again, we have to do the computation to decide which attribute to split on.

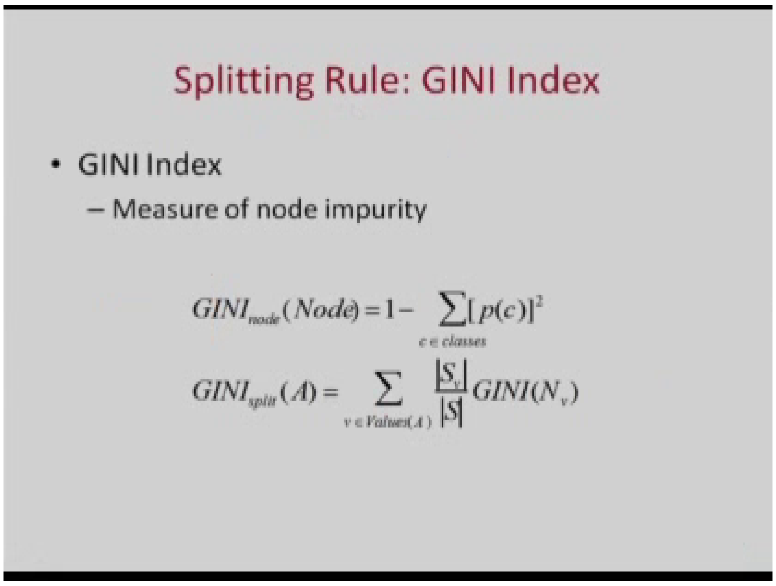
(Refer Slide Time: 25:53)



ID3 Algorithm

So if you grow the full decision tree this is what we will get and you can work it out.

(Refer Slide Time: 26:00)



Splitting Rule: GINI Index

- GINI Index
  - Measure of node impurity

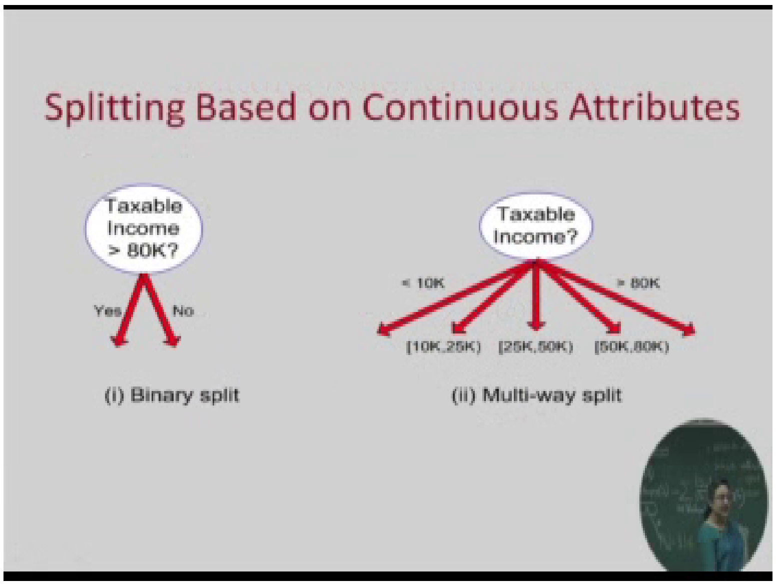
$$GINI_{node}(Node) = 1 - \sum_{c \in classes} [p(c)]^2$$

$$GINI_{split}(A) = \sum_{v \in Values(A)} \frac{|S_v|}{|S|} GINI(N_v)$$

Now, other than the information gain there are other measures of deciding the attribute for decision tree. One popular measure is Gini index. And Gini index is another measure of node impurity we will not going to the details, but just to tell you the gini index of a node is computed as 1 minus sigma of probability of c whole square. Where the c is the different classes and P c is the probability of the class or which can be estimated by the fraction of examples belonging to the class.

Once you have found the gini index of a node, you can do the gini index of a split for an attribute. So gini A is sigma again the fraction of training sigma over the values of the attribute the fraction belonging to that value gini of that node; so based on that, you can compute the gini index and gini indexes.

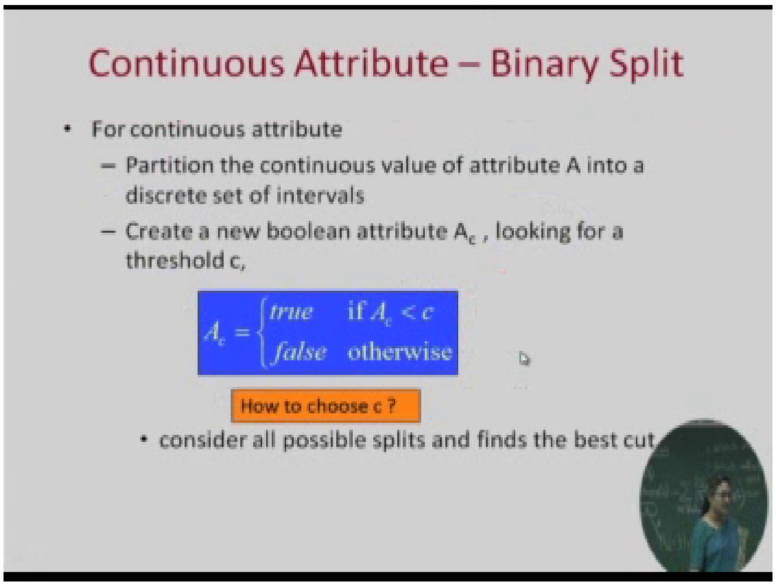
(Refer Slide Time: 27:00)



Splitting Based on Continuous Attributes

Another measure heuristic which can be used for decision tree; now what we have discusses is decision tree which has 2 or 3 or 4 values, that is nominal valued attributes. What if the training example contains an attribute which is really valued? So, if it is a real valued attribute what you can do is that you can split the attribute values in to two half's. For example, height you can say height less than 5, height greater than 5, you can divided into two half's. Or you can divide it into few discrete ranges and then you can grow that decision tree.

(Refer Slide Time: 27:45)



Continuous Attribute – Binary Split

- For continuous attribute
  - Partition the continuous value of attribute A into a discrete set of intervals
  - Create a new boolean attribute A_c , looking for a threshold c,

$$A_c = \begin{cases} true & \text{if } A_c < c \\ false & \text{otherwise} \end{cases}$$

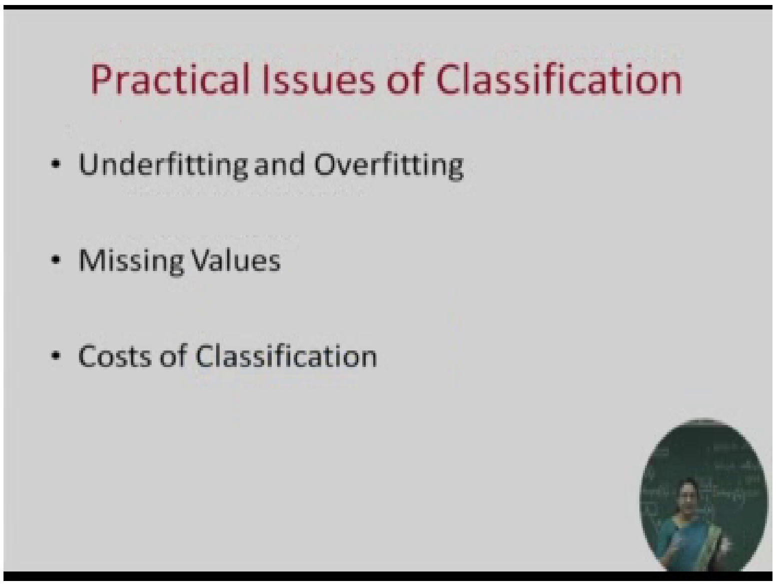
How to choose c ?

- consider all possible splits and finds the best cut

Now, suppose you want to divide the attribute into two ranges for a continuous attribute. You have to decide what is the value on which you will split, suppose there is the different heights are there and you want decide whether you want to split at 4 or 5 or 5.5 or 5.3 6.2 you have to decide where to split. Now for this also what you can do is that we can identify possible values for splitting and for each value that we split the range one we can find out where the information gain is maximum.

Of course, this is a computationally intensive and it would require sometime, but one can do it intelligently. So for continuous attribute one can do binaries split and in order to binary split if you want to do it optimally you find all possible split and find out where the information gain is highest.

(Refer Slide Time: 28:47)

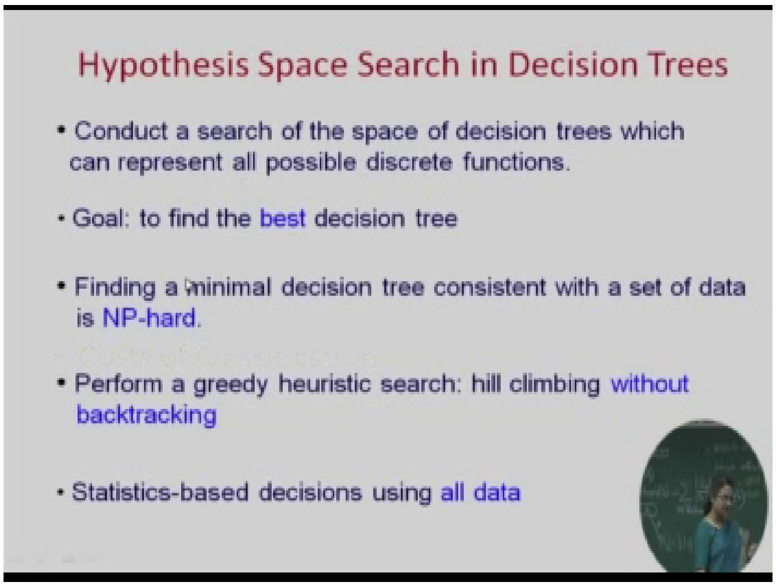


Practical Issues of Classification

- Underfitting and Overfitting
- Missing Values
- Costs of Classification

Now, we have covered the basic algorithm for decision trees. There are certain other things that you need to worry about when working with decision tress, whether decision trees or other running algorithms also under fitting and over fitting, missing values, costs of classification, etcetera which we will cover in the later class.

(Refer Slide Time: 29:06)



Hypothesis Space Search in Decision Trees

- Conduct a search of the space of decision trees which can represent all possible discrete functions.
- Goal: to find the best decision tree
- Finding a minimal decision tree consistent with a set of data is NP-hard.
- Perform a greedy heuristic search: hill climbing without backtracking
- Statistics-based decisions using all data

With this, we stop our lecture for this particular module and then will continue again in the next class.

Thank you.