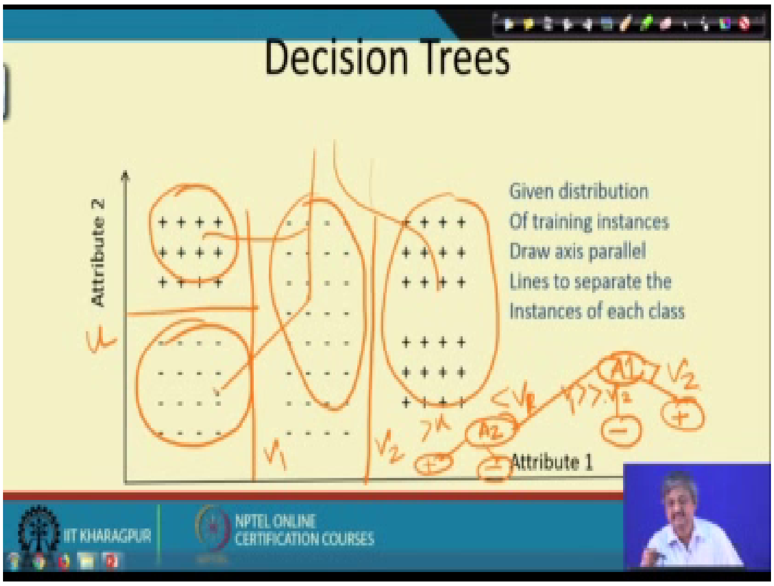
Data Mining
Prof. Pabitra Mitra
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 10
Decision Tree - III

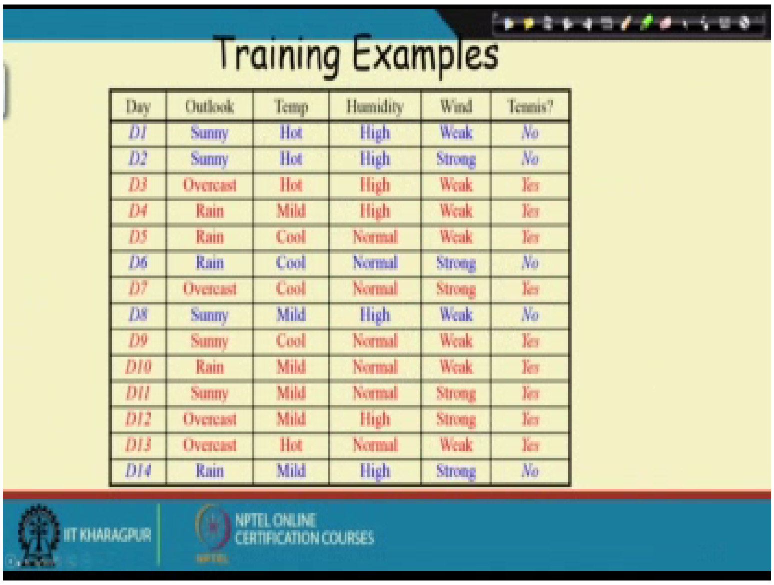
We continue our discussion on the Decision Tree.

(Refer Slide Time: 00:32)



So, to recapitulate we have a training set. If we look at the examples, we have a training set of we have a training set where I have some examples and their corresponding class levels yes or no.

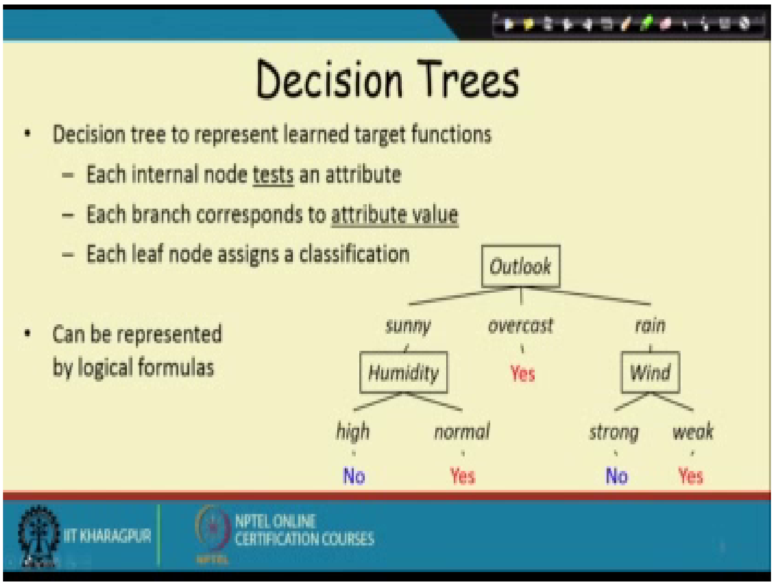
(Refer Slide Time: 00:44)



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

(Refer Slide Time: 00:50)



So, I want to construct a decision tree of a particular form, where each node is an attribute and each branch is a value of the attribute. The leaf nodes correspond to class.

So, if a example comes and if we check against the values of the each node, it will follow one of the branch and then we will check another attribute it will follow another branch, till it reaches the leaf. And every leaf is associated with a class in this case only 2 yes and no, and I will classify that example as belonging to that class. So, now, the question is how do we construct the decision tree; that means, which attribute should I check first and then which attribute, then which attribute where do I stop all these are part of the decision tree construction. I will explain this with an sort of geometric example.

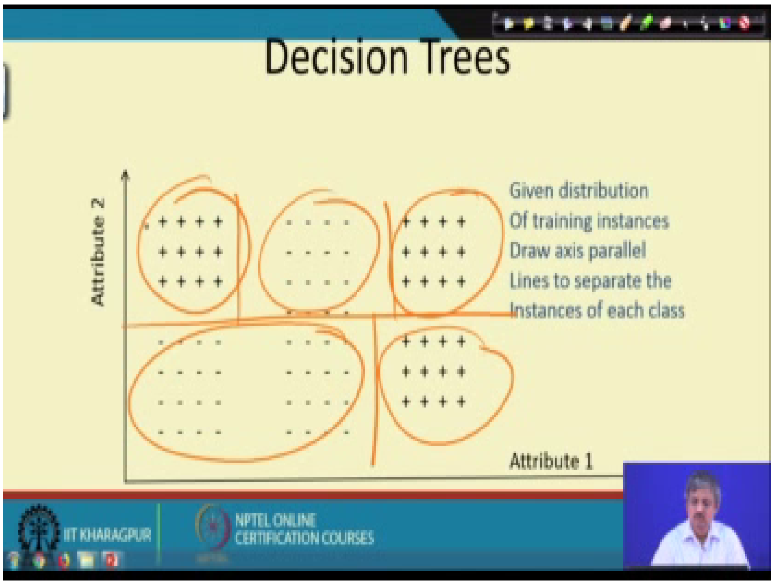
So, as I have told you before any instance can be described as a vector consisting of that actual values. So, if for example, we have 2 attributes, I can represent it by A 2 a every instance as A 2 dimensional vector. So, in this figure what I have is, I have a set of points in 2 dimension which are belonging to S class I mark them as plus and no class I mark them as minus. And what I do is that I construct a decision tree of this particular form where first I check attribute, I check the value of attribute one and suppose it is greater than some value, I say it is a leaf belonging to class.

And then I check if it is greater than some other value, and less than this value it is this region minus class and if it is less than this certain. So, I if I call this as V 1 V 2, if it is greater than V 2, if it is greater than V 2, I say at this class between V 1 and V 2 between sorry the other way round it should be; between V 1 and V 2 this class and less than V 2 sorry less than V 1 I have messed up completely.

So, less than V 1 it is this part; if it is this part then I again check the value of attribute 2 because whereas, this region and this region contains only point from one class, I can infer that belong to that class, it is a mixed. So, I again check the value up attribute A 2 and let us say if it is above a value of u I check, if it is greater than u or less than u. If it is greater than u I call it as plus if it is less than you I call it as minus. So, the point to be noted is that you can visualize any decision tree as a set of axis parallel cuts, which split up the points training set points into small small regions.

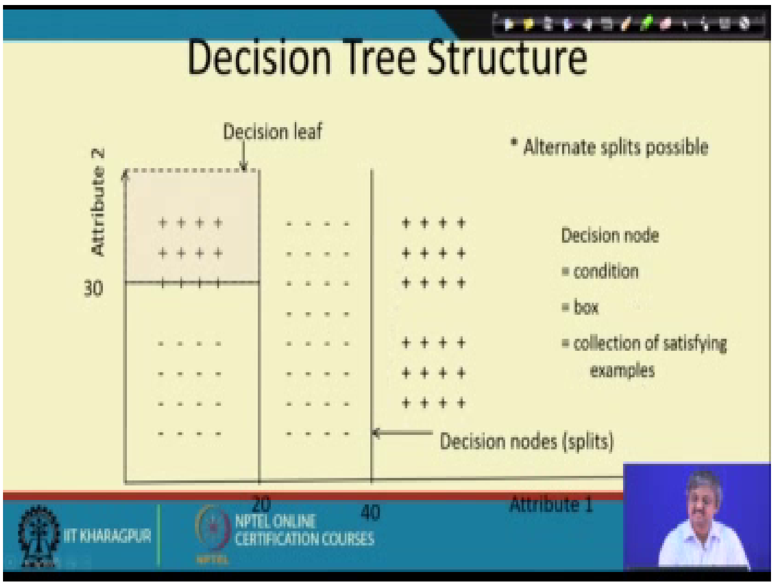
Finally leafs, they will be one of the region which I do not split further. So, this is a leaf because I do not split it further, this is a leaf I do not split, this is a leaf and this is a leaf, I do not split them further. And what I want is the points note that again if a new point x if I push down to that tree, I will just check these cuts and it will be falling in either in this region or this region or this region or this region a point will fall, I will check the values of attribute and decide. And if this regions are pure; that means, content points from a single class only I have a good decision tree. So, I could have as well drawn another decision tree say like this.

(Refer Slide Time: 06:00)



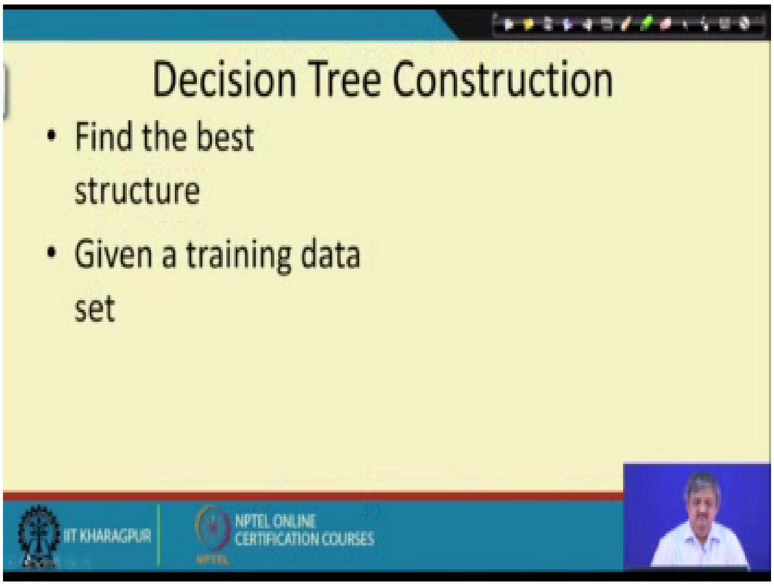
I could have maybe first split on this, and then I could have split each of the region further on this and then on this and then on this, then also I get pure leaf classes. If we if you see it will be a slightly different decision tree; if we actually draw this tree you will see the previous tree was actually smaller cell one than this tree, it has less number of checks if conditions. So, I would prefer this tree. Now, I ask you the question that you tell me. So, this is what I have drawn.

(Refer Slide Time: 06:59)



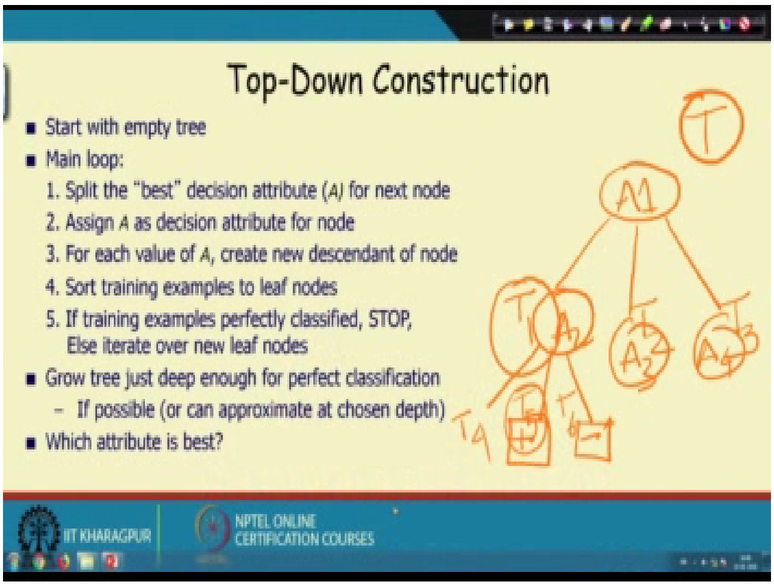
It so, this is my tree this is my decision leaf, note these values every decision nodes correspond to a condition a particular box, and a set up samples training samples which satisfy those conditions and fall in this box.

(Refer Slide Time: 07:20)



So this is my problem given a training set find the best structure.

(Refer Slide Time: 07:25)



I will follow this top down construction rule for doing this, what we do is start with an empty tree and then I define something called a best attribute, I will later explain what a best attribute means. So, I choose which attribute is the best that I choose as my leaf say

A 1 is the best attribute I choose as my leaf and I make branches for each different value of A1 I make a branch. So, again I will come back to another question that you might have already have is that suppose A 1 is continuous valued A 1 has a range of values say temperature or something then, how do I decide the branches. For the moment I will come back to that question later for the moment assumes that A 1 has only a discrete set of values in the domain. So, now, this is a decision attribute. So, if we look at the training examples, you say I call it as T, some of the examples will fall follow this branch satisfy this condition some examples will satisfy this condition, and some examples will satisfy this condition.

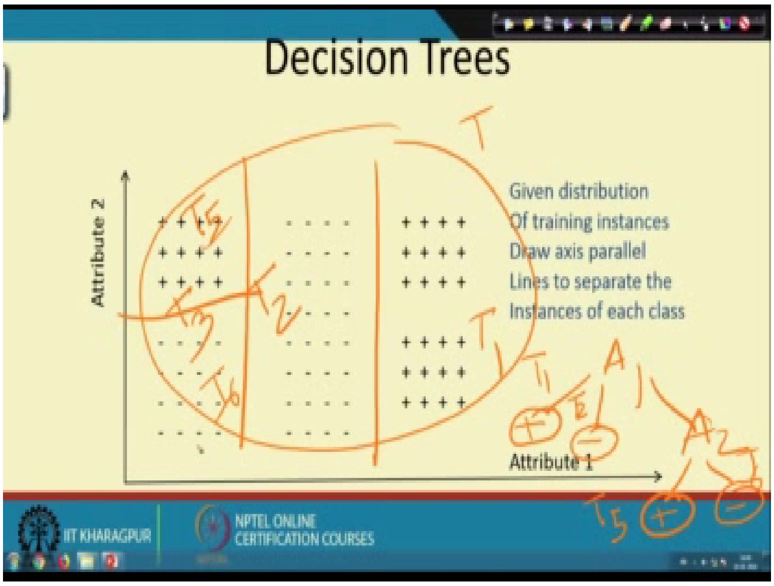
Now, as if I recursively carry on this. So, what I do, as if I start building a new tree here, but not using this T examples, I decide what is the next best node to split, again what is best and I am defining later. So, I find out an attribute which will be split this T 1 in the best possible way. To give you a hint this best possible way corresponds to the way the attribute which sort of discriminates 2 classes yes or no classes as much as possible, maybe A 2 and for here maybe it is A 3, and from here maybe it is A 4 or something.

And then this examples I split further and maybe I get T 5 T 4 set of examples, T 5 set of example, note that T 4 plus T 5 plus T 6 will give me T 1, similarly T 1 plus T 2 plus T 3 will give me all that (Refer Time: 10:24) examples T they are getting distributed among the nodes. Now if a particular branch all or if it this set of example T 5, belong to a single class I do not split it further and make this as a leaf. And whatever is the class of T 5 I put that class here. Similarly I check a node if it is pure; that means, all the points in T 6 are from a single class, I make it a leaf I do not split further say they are minus class I make it like this otherwise I keep on splitting.

Till all the leaf nodes all the examples belong to a single class. So, till I get, till I reach a leaf node and. So, I grow that tree and then I stop. So, if I do it you can visualize that I will get perfect classification, all the examples training examples will be perfectly classified. I am not yet commenting on what about future examples a unknown examples. So, this algorithm is clear it is a top down algorithm initially consider the entire training set choose an attribute which best splits it say it splits into T 1, T 3 and T 2 examples, each T 1 again recursively split further using what is the best example considering T 1 only.

So, if we actually look at the previous example, it will become clear. So, what I do is that. So, initially my entire training set is T. So, initially this entire set is my T.

(Refer Slide Time: 12:32)

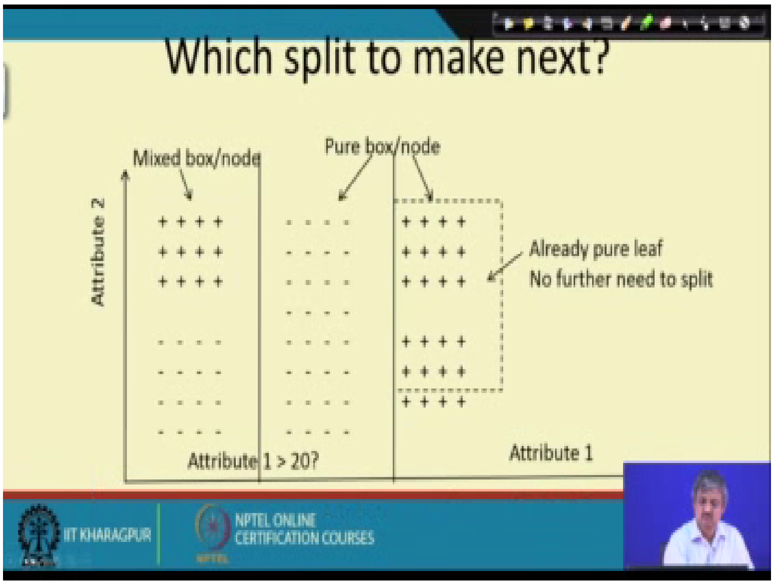


So, now, if I split along A 1 into 2 3 branches actually, then this is my T 1, this is my T 2 and this is my T 3 they get split. So, I have 3 branches. So, now, T 1 T 2 they are pure they need not be split further. So, they are if I call this T 1 this is T 2 their leafs.

This is a T 1 is a plus leaf T 2 is a minus leaf, T 3 is not yet pure. So, I have to split it further. But now considering T 3 you look A 1 attribute one is no longer the best attribute to split along considering only T 3 considering entire T A 1 as the split considering only T 3 A 2 is a special split, attribute 2 is a better split. So, I split it further using A 2 into say 2 groups. So, one of the groups become plus and one of the groups becomes minus plus. So, let me call it as T 5 and this as T 6. So, this is my T 5 and this is my T 3 gets split into T 5 T 6 and then we stop. So, this is the algorithm.

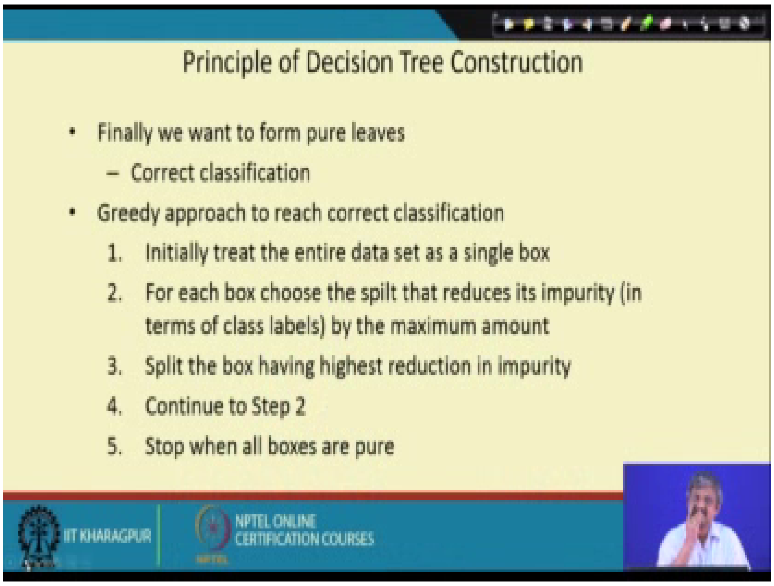
So, let me now sorry this is the algorithm. So, let me now formally write this down. So, I repeat again find the best attribute split on it, and on each of the branch again find the best till you get a pure leaf, now to answer the next question what is a best attribute.

(Refer Slide Time: 14:49)



You think a bit when you built a tree we consider the first split, it is better to split along attribute one why because this attribute produces as child directly pure classes it produces a child a pure class is. So, this purity of class can be sort of used as to decide what is a best attribute. So, I can use this. So, let me show it. So, this is the principle what I had I have already explained.

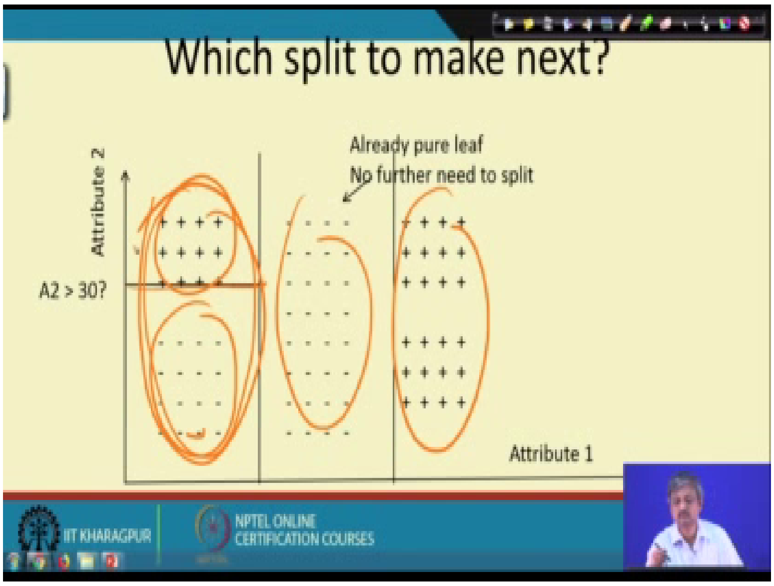
(Refer Slide Time: 15:45)



So, this concept of purity of the split how pure a split leads to plus finally, you want a absolutely pure leaf that is my criteria of best attribute.

So, what is this criteria? The criteria is if you again look carefully here is that for example, this one initially this segment was impure it contained both and splitting this led to more purity of the child.

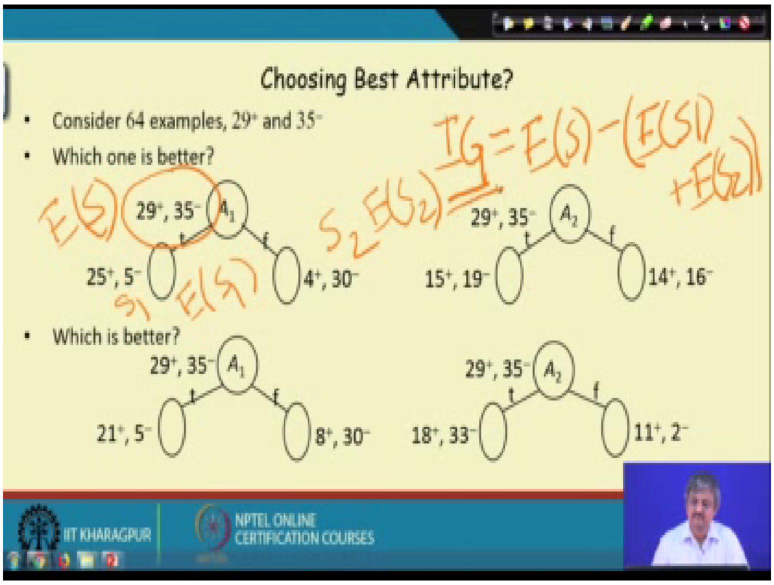
(Refer Slide Time: 16:20)



So, every split similarly here this was T 3 splitting led to more purity of these and these and this is still impure. So, every split should led to increase in purity of the resultant boxes. So, compared to the parent the child’s are purer.

How to quantify this? I will quantify this using a function called the entropy; you know entropy is nothing but a measure of the purity or order liners’ or something. So, I will do this.

(Refer Slide Time: 17:34)

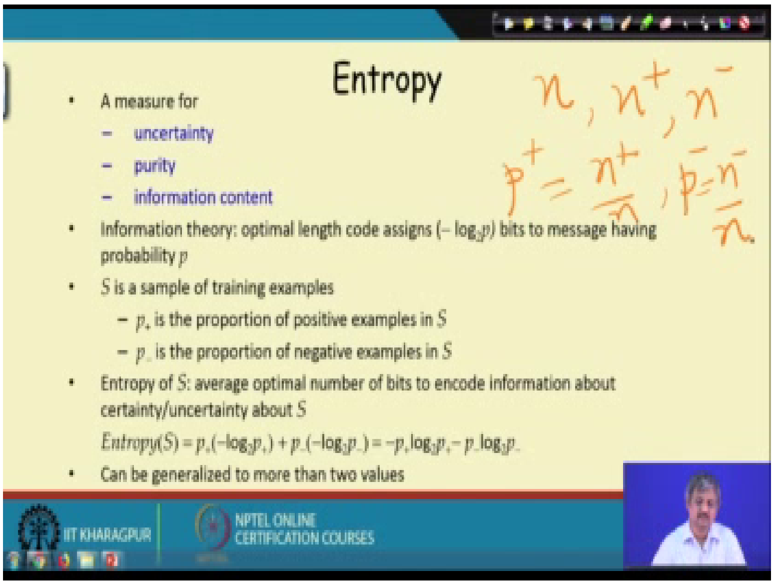


So, let me give an example to do this, suppose your initial training example initial training set contain 64 examples, and of them 29 are S class and 35 are no class.

Suppose using A 1 I take a value of A 1 is a binary valued attribute which takes on a value true or false out of this 64 I see how many examples have A 1 equal to true, let Us say thirty examples have A 1 equal to 2 and remaining 34 examples have A 1 equal to false. out of these 30 examples 25 are plus 5 are minus yes and no and here 4 are plus 30 are minus whereas, if we split along using attribute A 2 this is the picture this is the split. So, which split is better? Naturally this split is better because here there are a 50 50 mixture of both class, now the classes become purer here still they are mixed.

So, naturally this is a better thing. So, similarly here which one is better? So, what I do is that I define a measure called entropy to do this.

(Refer Slide Time: 19:19)



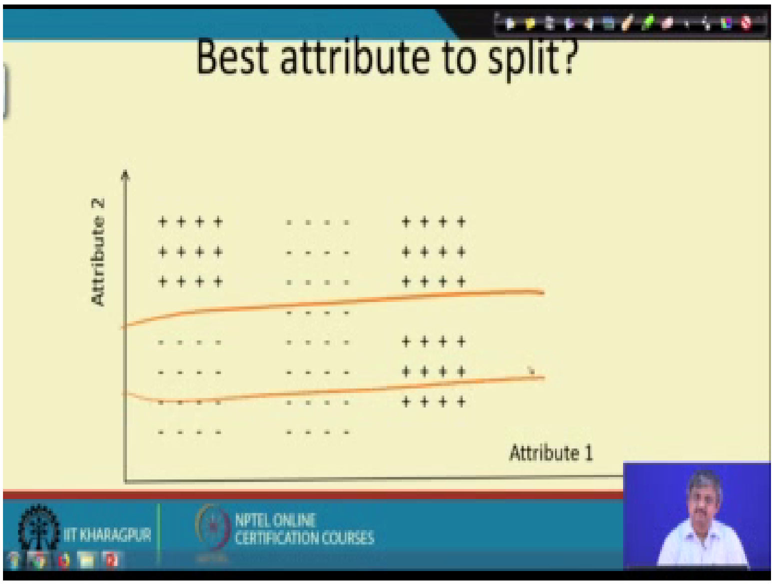
The measure is defined this way. Suppose S is my set of training points. So, in the previous case there are 64, S has a side 64. Now let us say p plus fraction of them are positive S examples S class and p minus are negative term. So, in other words I can easily find define it this way, suppose you have n examples of which n plus are positive, plus n minus a negative plus, then p plus is n plus by n and p minus is n minus by n. If you have more than 2 classes you can easily generalize it to fraction belonging to each class. So, it will be if you have 1 2 3 classes it will be p 1 p 2 and p 3. So, the entropy of this set of examples S is defined by this formula.

It is p plus into minus log of p plus note that p plus is a less than 1. So, log of p plus will be negative. So, I put a minus in front. So, p plus into log of p plus plus p minus into log of p minus, let us say log base 2 you can actually take any base. So, if you had more class say 1 2 3 plus then I will write p 1 into log of p 1 plus p 2 sorry p 1 into minus log of p 1, plus p 2 into minus log of p 2, plus p 3 into minus log of p 3. So, suppose this is that S is the entropy. So, now, what I can do is the following, I can say for this split.

So, this is my training set S, I can find its entropy. let me call it E S. It splits into 2 child's; let us say I call them set S 1 of these thirty examples, and S 2 of these 34 examples and similarly I can define entropy for this set S 1 and for this set S 2. I will say the information gain that you get by this split is the reduction in entropy as a result of this split. So, if I define information gain, I can define as entropy of the parent minus sum of entropy of the child's.

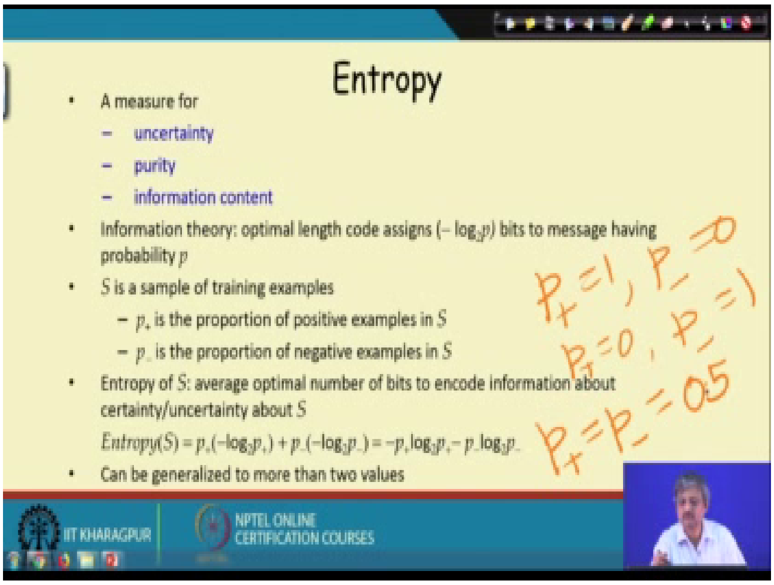
So, this is the total entropy of the child E S 1 plus E S 2. E S is the entropy of the parent, the reduction in entropy is the gain in information; information is kind of negative of entropy. Now what I will say is that my best split among this 4 is the one which leads to highest information gain or maximum reduction in entropy; so again if I come back to that illustrative example.

(Refer Slide Time: 24:05)



So this splits leads to good reduction in entropy because they become purer whereas, this split is or maybe this kind of split does not lead to much reduction in entropy, because the childs are also impure. So, the previous split will be having a better score than this split. So, look at if you if you carefully examine some properties of this entropy you will you will notice one thing that, when p plus equal to p minus, there are equal number of points from both class entropy is the highest. When there are points from only one class pure S only p plus equal to 1, p minus equal to 0 or other way around p minus equal to 1, p plus equal to 0, then entropy is 0 actually entropy is 0.

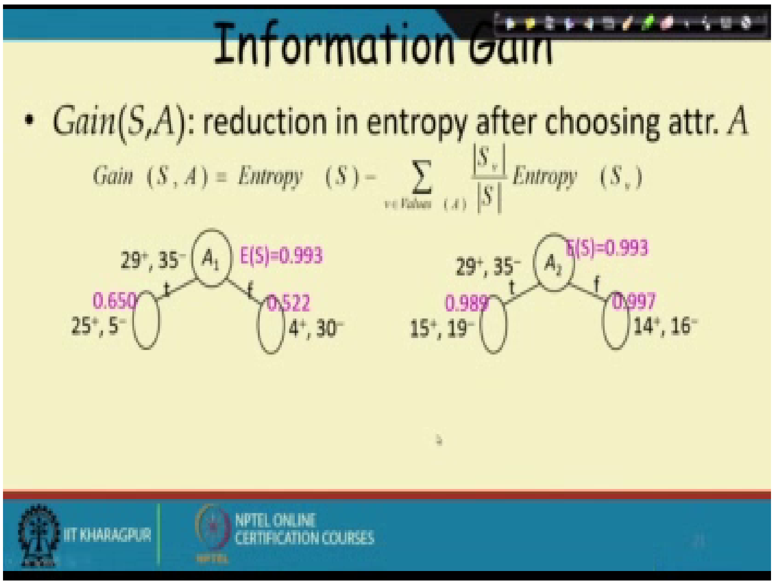
(Refer Slide Time: 25:45)



So, the purer the class; that means, if we have p plus equal to 1, p minus equal to 0 or p plus equal to 0, p minus equal to 1 pure class then we have high entropy sorry we have a low entropy they are pure less disorderliness, whereas, if you have p plus equal to p minus equal to 0.5 you have the highest entropy, you can work out the evaluate the expressions and check. So, my training algorithm is very simple, I just follow this rule I am giving you the description of E S, description of the alga I follow this rule recursively split on attributes first choose the best attribute on the entire training set then each of the branches individually what is the best attribute.

Where the best attribute is defined as the amount of information gained that split on that attribute leads to and then we stop, when there is a 0 entropy or pure leaf. So, that is my algorithm alright. So, I think that is clear. So, that is my algorithm. So, if we I have walked out I will upload the slide. So, I have walked out all the values I am not repeating them now you can check the values that we get.

(Refer Slide Time: 27:42)



Sometimes there is a slight modification to the entropy value or information gain value, it is I have mentioned entropy of the parent minus sum of the entropy of that your child entropy S 1 plus entropy S 2 it is sometimes.

Weighted sum of the entropy of the child where a larger child which has more examples will be given more; so fraction of examples of S which fall in the child phi. So, weighted by that ratio that is the entropy, sometimes this form is also used. So, what I will do is that I will use this if I do I will do the calculation. So, just to summarize my algorithm is like that, I will start with the entire training set see which attributes leads to best information gain, split on that. Now examples will be distributed across branches and on each branch again recursively is find out which is leads to highest information gain do that till each branch contain a 0 entropy or a pure set of examples, pure leaf then I stop.

So, in the next lecture I will give some more extensions of this on while dealing with not discrete values, but continuous attributes and on dealing with the training examples contain some noise and over fitting may happen, how to deal with that. We will continue that on the next lecture.

Thank you for this lecture.