


Machine Learning for Engineering and Science Applications
Doctor Ganapathy Krishnamurthy
Department of Engineering Design
Indian Institute of Technology Madras
Binary Decision trees

Hello and welcome back, in this video we will look at binary decision trees. Introduction to basically binary classification trees.

(Refer Slide Time: 0:23)

Introduction to Decision Trees

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



So we will look at this data set basically each row is a data point. It is collected over 14 days, so there are about 14 data points, it is just to understand this data set. there are 4 features, I can see here Outlook, temperature, humidity and wind. And you can consider this our training data and based on this 4 features and data points provided the decision has to be made regarding whether to play tennis or not, right?

So you can see all the Yes the decision yes is marked red and the decision no's are marked in black. So we have 4 features about 14 data points and based on these features we have to decide whether to play tennis or not. So how do we go about doing that using decision trees is what we are going to look in this video.

The idea is to again (1:17) this to make a decision as to whether to play tennis or not based on the features provided. the way you go about doing that is to split the entire data based on the features provided eventually leading to a decision as to whether to play tennis, okay. So we start at a node this is what is called a node which contains all the data set, okay.

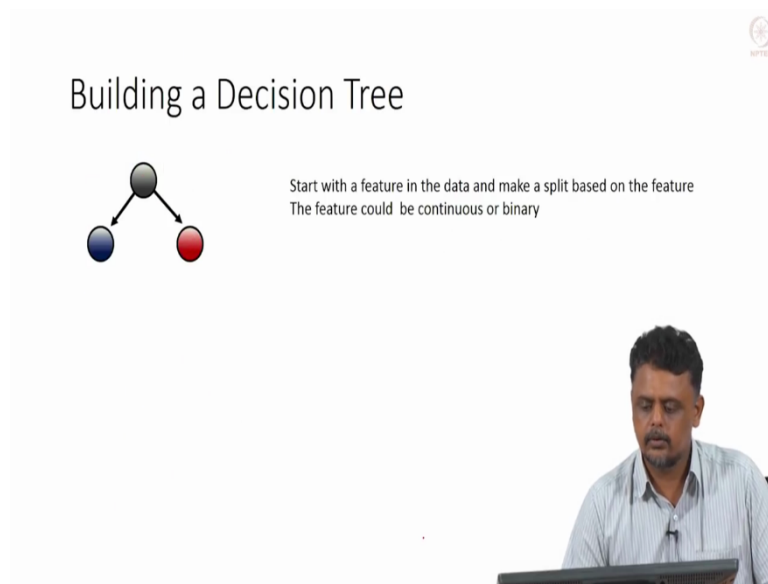
And then we do this check whether the temperature is greater than or equal to mild. Just recall the temperature has mild, cool and hot, go back and check here. So there is hot, mild and cool are the 3 values that the temperature can take, categorical values. So the temperature is greater than or equal to mild, we decide to play tennis otherwise we decide no, okay. So that () (2:09).

And we don't have to stop with one question, we can go ahead and split further, so we can check whether the humidity is normal or not. If humidity is normal then we play tennis or if otherwise we don't to play tennis, okay. Now remember that at each node what is basically once we start at this node with all the data, okay. And as we split the data based on this based on this condition temperature we will take all the data points in a dataset where the temperature is greater than or equal to mild and assign it to this node.

And if all the data points where the temperature is cool we will assign to this node, right? This is what we want to do. And from here on again among this data points that we have assigned to this node we will check if the humidity is normal or not, if it is Yes then we will assign it to this node here. Otherwise if the humidity is not normal then we go there, okay.

So this is the route now, right here this is the route of the tree that you have created and the leaves the nodes here are the leaves. So the idea is that the actually go down this tree or if you go up this tree which depending on how we interpret it. The leaves would expect to have all the data that has similar decision. So expect all the data that we decide to play tennis to accumulate in this node and all the data where we decide not to play tennis will be accumulate in the other node, okay. So that is the idea behind building this tree. So how do we go about doing that?

(Refer Slide Time: 3:54)

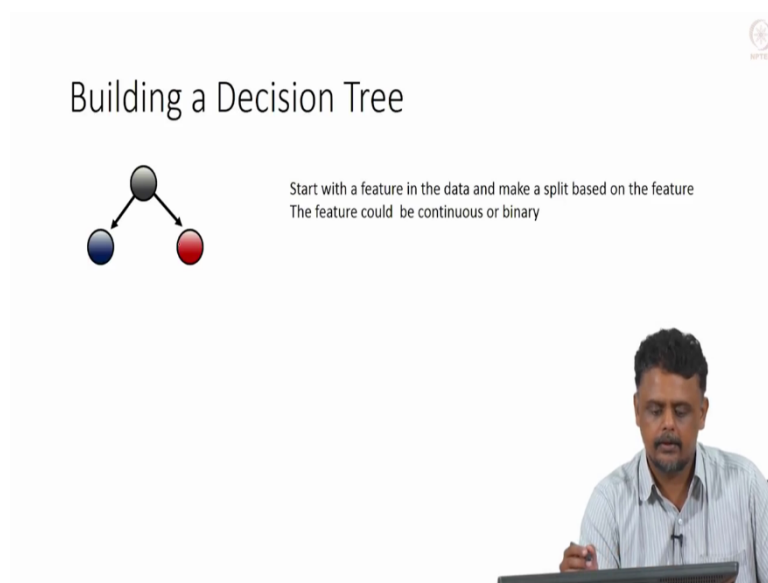


The slide is titled "Building a Decision Tree". It features a small diagram of a decision tree with a root node (grey) and two child nodes (blue and red). To the right of the diagram, the text reads: "Start with a feature in the data and make a split based on the feature. The feature could be continuous or binary". In the bottom right corner, there is a video inset of a man in a light blue shirt speaking.

So just to summarize, we start with the feature in the dataset and make a split based on the feature, okay. So the split could be continuous or binary or categorical. So the split is based on some condition, okay. We will see what that condition is later on but just for the sake a few slides we will assume that we have a way of figuring out which feature to choose and how to make the split, okay.

So we choose a feature and we make a split on the data into 2, so leading to 2 different node, so we assign the split data into each of these nodes, right?

(Refer Slide Time: 4:35)



This slide is identical to the one above, titled "Building a Decision Tree". It shows a decision tree diagram with a root node and two child nodes, and the text: "Start with a feature in the data and make a split based on the feature. The feature could be continuous or binary". A video inset of the same speaker is present in the bottom right corner.

And we continue splitting the data based on the features, so we can choose another feature here again understanding that there is a way to choose your feature and make further to splits, okay.

(Refer Slide Time 4:53)

Until:

- Leaf node(s) are pure—only one class remains
- A maximum depth is reached
- A performance metric is achieved

So we can keep doing that, we keep splitting until we come to a point where all the leaf nodes are pure, so these are the leaf nodes, right? These are the leaf nodes and the sense they contain only one class. So all the data in those leaf nodes contain only one class or when we reach a maximum depth, okay. So we decide that we will only go up to depth of 3 or 4, okay.

So then we stopped after the depth of 3 or we can evaluate performance metric upon which based on which we can stop the free from growing further, okay. So this is how we build the tree and this is using the training data but how do we test it? So in the sense that when the tested data arrives, the pass it through the tree and we will expect it to take a path through the tree and end up in one of the leaf nodes.


And each of those leaf nodes correspond to a class and that class will be assigned to the tested dataset, okay. So that is how we would go about building a binary decision tree, okay.

(Refer Slide Time: 6:01)

How to Make the Split

- Classification Error ✓
- Entropy
- Gini Index

Information Gain



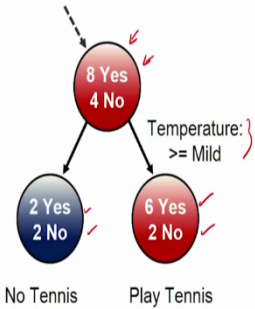
So then here comes the question how do we make the split? Okay. So usually the split is based on something called information gain, okay. We look at what that is again but we can also use classification error and typically these 2 would correspond to what is called the information gained the entropy and the Gini index. What we would call the information gained criteria?

We can also do this split based on classification error. We will look at some examples of how we would go about doing that, okay.

(Refer Slide Time: 6:39)

Making the split- Example

c - classes



Classification Error Equation

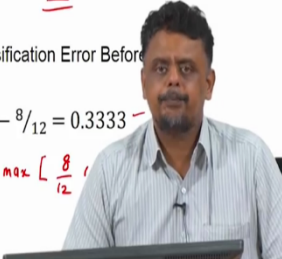
$$E(t) = 1 - \max_c [p(c)]$$

proportion of data belonging to class c

Classification Error Before

$$1 - \frac{8}{12} = 0.3333$$

max [8/12]



So let's say we are at this particular node where we have 8 data points for which the outcome is yes that is to play tennis and for data points where the outcome is to not to play tennis that is no, okay. So then decide to make a split based on the temperature, okay. Greater than or equal to mild, so which results in 6 data points for Yes to play tennis and 2 data points for no and this is equally distributed 2 for yes and no respectively, okay.

So we still haven't decided how to make the split and what features to choose but we will see as we go on how to do or go about doing that. In this particular example is just to illustrate why using the classification error is a bad idea. So this is the equation for the classification error $1 - \max_c p(c)$ where c is the class.

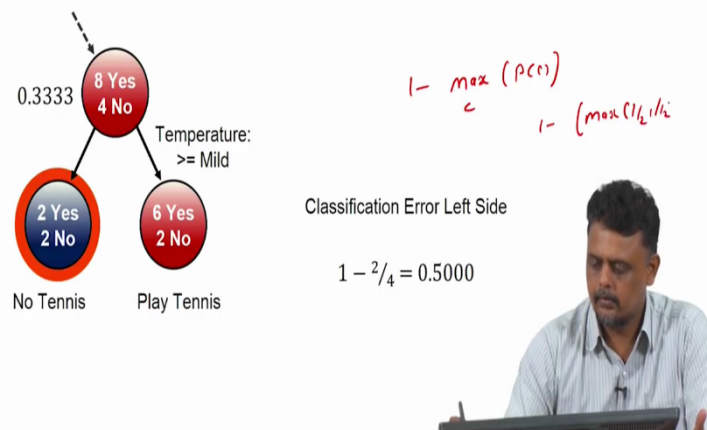
c denotes the class, so number of classes can be more than 2, so typically for a binary classification there were 2 classes but there could be multiple classes also for this kind of algorithm. So $1 - \max_c p(c)$ is what we would like to evaluate as the classification error, okay. So let's see if we look at this particular note here.

So we have $1 - \max_c p(c)$, so we have 8 out of 12 for decision yes and 4 out of 12 for decision no, so which leads to $1 - \frac{8}{12}$ as the classification error, right? So this $p(c)$ is basically the proportion of data belonging to a class, belonging to class c . So the $p(c)$ is just a proportion of data belonging to class c , so then $1 - \max_c p(c)$ means that we have to calculate.

Let's say in this we have 2 classes, if we have N classes there will be N number here and we had choose the maximum of that and $1 - \max_c p(c)$ will give you a classification error in this case it is 0.33, right? So the question is how are we actually deciding on the feature and how are we actually do the split? So in this case we will see how that is calculated.

(Refer Slide Time: 9:09)

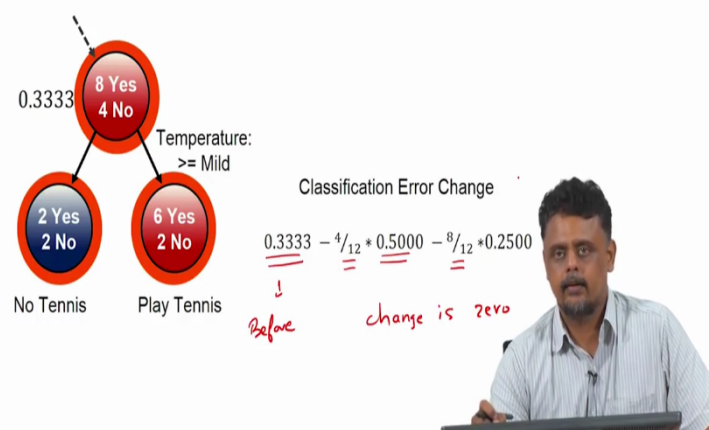
Making the Split- Classification Error



So what we are doing is then we have to look at the left and the right node. So we are looking at a binary decision tree, so usually there is only a 2-way split, okay. So if you look at the left then the classification error again the formula $1 - \max_c p(c)$ will give you $1 - \max(0.5, 0.5)$. I will give you 0.5, okay.

(Refer Slide Time: 9:41)

... Classification Error



And if you look at the classification error on the right-hand side then it's a max of one minus max of 6 by 8 and 2 by 8, so that is easily chosen you get 0.25. So what we typically want to look at is the change in the classification error. So we want something that would split that would give rise to the biggest decrease in classification error, so what we do is after the split we calculate the average of the classification error.

Here the weighted average corresponds to the weight average of the 2 nodes here, okay. So you can call this the parent node and these 2 like 2 child nodes. So you have to calculate the weighted average of these 2, so then how do we do that then we will see if you look at the classification error change. So this is the classification error before the split, okay.

Now we want to do it after the split. So the weighted average in this case, so 4 out of 12 data points went to the left node, so that weight is for over 12 times the classification error on the left node. The waiting for the right node is 8 over 12 that is a data points on the right node, 8 over 12 times 0.25 and you will see that if we do that then the we will see that if we do that then the classification the change is zero.

So this can happen when we use classification error as our metric to decide the split. So the general idea is to consider one feature added I'm and calculate the change in classification error for all of the based on split. So the idea is to consider each feature at a time and split the data based on the feature.

And calculate the change in the classification error and choose the feature that gives rise to the maximum change in the classification error. However it is possible that when we use the classification error as a metric then the change can sometimes be zero, so that's what , so that tends out to be a bad choice. So we will see why that is so in the later slide but to get over this we have 2 other metrics that we typically use that is the Gini index and the entropy.

(Refer Slide Time: 12:29)

Split based on Entropy

Temperature:
>= Mild

2 Yes
2 No

No Tennis

6 Yes
2 No

Play Tennis

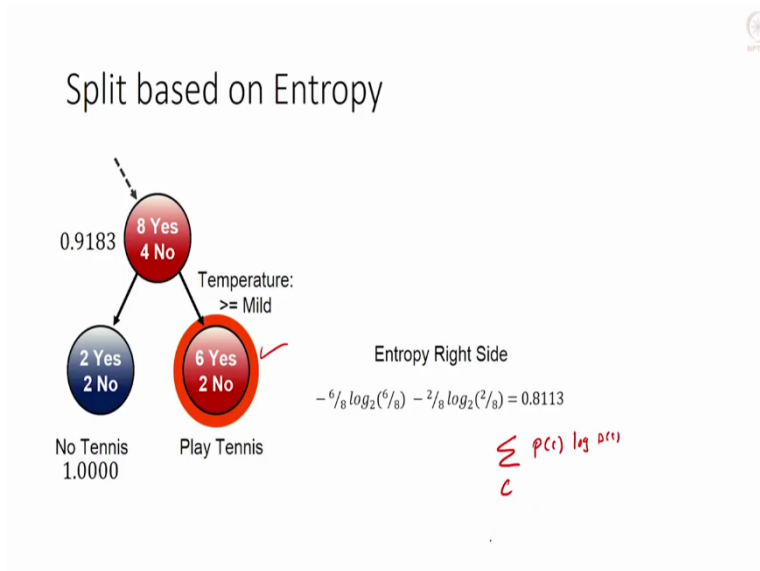
Entropy Equation

$$H(t) = - \sum_c p(c) \log_2[p(c)] //$$

So we will look at entropy first. So this formula must be factored to most of you. Summation over the classes $p \log p$ came to the base to (\log_2) (12:38) keep that in mind. So what we want to

do is, calculate the entropy before the split based on the feature and calculate the weighted entropy after the split based on the same feature and what the changes.

(Refer Slide Time: 12:58)



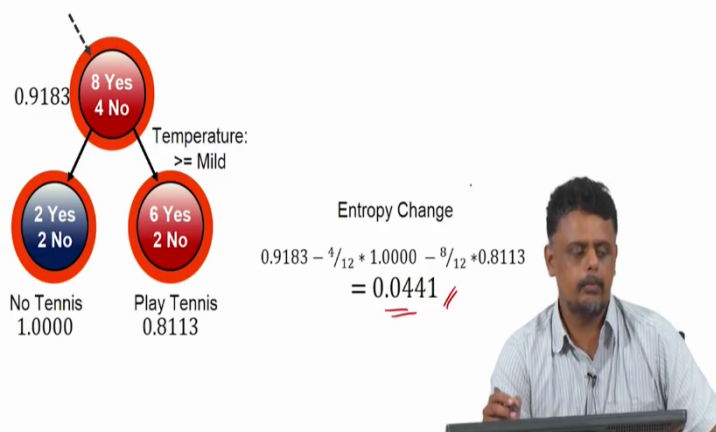
So let's see how that is done. So entropy before is basically there are 2 classes, so you have to sum over the classes the 2 classes are 8 over 12 or yes and no. So the p of c is just the proportion of data points corresponding to the class. So 8 data points out of 12 correspond to yes, so that p is 8 over 12 log 8 over 12 and the proportion of data points which correspond to the no class is 4.

So it's 4 over 12 log 4 12, so that gives rise to 0.91 then we go about calculating the entropy on the left which in this case minus 2 4 log 2 of 4 that is about 1, okay. And the entropy on the right-hand side again if you look at the right-hand side this is the one that's it is there. So there are 8 data points there 6 of them correspond to yes, 2 of them correspond to no.

Then we can just give you 6 over 8 logs 6 over 8 and 2 over 8 log 2 over 8, okay. So there is summation over the classes and so this is basically summation of the class's p of c log of p of c, okay. So there are 2 classes there are N classes there will be N terms in that sum, okay.

(Refer Slide Time: 14:13)

Split Based on Entropy



So we can do the same thing and we can calculate the entropy change as 0.0441. So this is one class which is the temperature. We would do the similar calculation for the other 3 classes other 3 features and we will see, we will calculate the same entropy change for all of them and choose the feature that gives rise to the maximum change. So there are many questions here because in this particular example I have chosen b of, all of them are category features are kind of categorical.

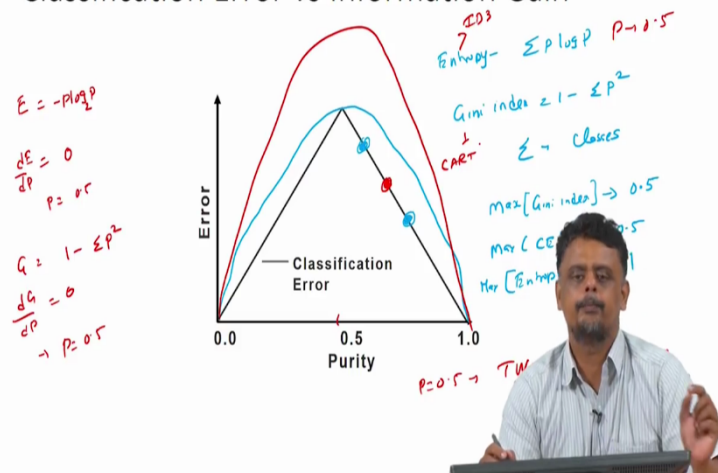
Since there are they take from a finite set of values. So it is either mild, so we can split based on anything, right? So it can be we can say less than hot and things like that. So to arrive at that answer then we have to evaluate this even for a particular feature. We might have to evaluate this cross entropy for different values of that feature.

So say if you have a continuous feature let's say you just have measured temperature in Fahrenheit or Celsius then we have set the right threshold and that temperature which gives rise to the largest entropy change, okay. And then we make a split this on that, so there are different ways of calculating that, alright.

So in our case we have a very nice categorical labels and we can always choose amongst them to see which gives rise to the maximum entropy change, okay. So this is typically how the split based on entropy works.

(Refer Slide Time: 15:54)

Classification Error vs Information Gain



So let's see why we had this problem with classification error wherein we saw that there is no change, okay. So that happens because let's say that you have your parent node had, you know some entropy here, okay. Had entropy value here, let's say the child node ended up having classification error there and there, okay. This is the possibility.

And we see that the weighted average can end up going setting with the entropy of the parent node which means that there will be no change and there will be no successive the tree won't be able to grow the tree any further. However if you look at your, the entropy formula which is $p \log p$ this is the entropy based split, okay. And we also have the Gini index which is 1 minus summation be squared.

Now remember the summation is over the classes, okay. Okay. So the maximum value of the Gini index can go it is 0.5 you can verify that. Same thing for the classification error the maximum value of the classification error I will call this CE is also 0.5. For the entropy, for 2 class problem, okay. For 2 class problem maximum entropy is 1, okay.

So if you plot all of them then for the Gini index it is kind of very concave too wiggly but you can say it is a much more smooth function then I drew you here. So you can plot that out it is squared 1 minus p squared you can see that how you can plot and if you want to look at the entropy it is something more like this because the value can be 2 1. We all reach their maximum value at p equal to 0.5 for a two class problem, okay.

So the maximum entropy happens when in a particular node all the classes are equally distributed, okay. So then that what gives rise to maximum entropy and for a 2 class problem you can understand when that happens then p becomes 0.5, okay. That's a possibility.

So there is an easy way to figure out when the maximum happens. So for instance remembers that the entropy formula I will use that E is $p \log p$, okay. p think of p as a variables then you want to find out the maximum value of p then you just take the derivative with its minus $p \log p$ dp is 0, if we do that again this is base 2. If you do that you will figure out that happens when P equal to 0.5, okay.

Okay. You can do the same thing for a let's say 2 class problem for Gini index Gini index where I call this G is $1 - \text{summation } P^2$. Once again you can do the following dG or dP and set that equal to 0 you will get P equal to 0.5, okay. So that is very easy to calculate but the point is that if we use the classification error because it is kind of has this lineartrend when it is possible to end up with the same total change and classification error is zero and then there will be no way to grow the tree any further, okay.

And then if you use the other 2 entropy and Gini index then you can successfully split on the different features. If you use the entropy and entropy-based information gain then it is called the ID3 algorithm and if you use the Gini index it is called Cart CA R T. So this corresponds to ID3 and this one is CART. There are different algorithms for doing this.

So one way of looking at this is, this is a greedy search, alright. So at that particular node we are looking for the best split instead of considering the entire tree as a whole you just split based on the best top criteria available at that time, so it is like greedy algorithm.

(Refer Slide Time: 21:08)

Summary

- Until Stopping condition ✓
- At each node find best possible split of data based on attribute / feature
- Add leaf nodes
- Go back to first step

→ Overfit
→ Pruning trees

Impure leaf nodes → 8 → Yes
1 → No



And summarize, so based on a stopping condition, we looked at different stopping conditions one of them being the depth of the tree, okay. At each node you find the best possible split of data based on attribute or feature, okay. So you select every feature, threshold that feature, okay. Select the threshold based on that feature and calculate the information gain between the child nodes and the weighted average of the child nodes and the parent the node.

And whichever gives you the highest change in that criterion I you will do the split based on that feature and that threshold, okay. Once you have that split then you will assign the data to the leaf nodes, 2 leaf nodes or add 2 leaf nodes and you assign the split data to each of them and then again start from each one of those leaf nodes you again start doing this, okay.

So at test time to keep doing that till you reach a particular depth or if all the elements in the leaf nodes correspond to a particular class then you stop at that point, so going till that point, right? And for test whenever test the data comes in you just pass it through the trees, you start at the leaf node and you keep based on the criteria you use at that particular node, you will keep passing the data to successive child nodes till you get to a last leaf node which tells you the class, okay.

Now it is possible that at some of the leaf nodes might not be pure, okay. So you might have this impure leaf nodes, right? So in which case you just assign it to the, you do this majority voting. So y if one of the leaf nodes let's say 8 yes and 1 no for instance whenever the test data point and up in this particular node you will say yes, okay.

So that's a possibility. So that's what we have to do. This is a brief on how to use binary decision trees specifically for a classification problem. However we can also use decision trees for regression problem is, right? And we will see how we can go about doing that if you have a regression it's possible. There is one more aspect of this binary decision trees that they tend to overfit.

So if you keep growing the tree, so you keep adding leaf nodes or child nodes to every node that you have till you get maximum purity or in the sense till you get to leaf nodes which have only one class then what has happened is that you have grown a tree which fits your train data accurately, okay. And the test data might not generalize very well, okay. So then that's a problem with trees because you can always build a tree to overfit your data, if you enough depth, okay.

So to prevent that there are methods to what are called pruning trees. So once you have finished growing a tree then using the algorithm then you go back and see if you could have stopped earlier and then you throw away some of the leaf nodes and you stop, so you cut down on the height of the tree, so that is a possibility as far as in general decision trees are concerned, okay.

So in the next videos we will look at binary regression trees also and see how they work. Also give you a brief mathematical motivation to how do we define a cost function for this binary decision trees that is something also that we would like to look at and once we have done with the regression tree and as well as the cost function for the binary decision trees, we will see what are the other ways to address the overfitting problem.

So we will specifically look at bagging, boosting and we will also look at random forests, okay. Thank you.