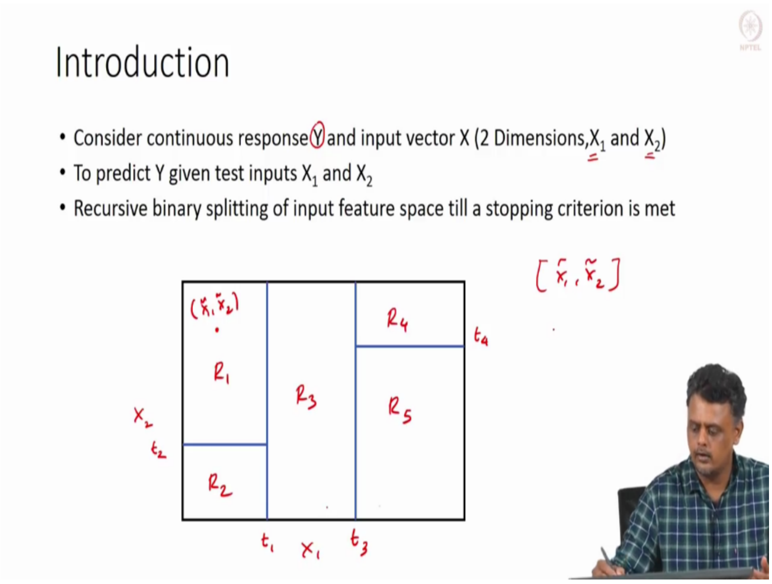
Machine Learning for Engineering and Science Applications
Doctor Ganapathy Krishnamurthy
Department of Engineering Design
Indian Institute of Technology Madras
Binary Regression Trees

Hello and welcome back. In this video we will look at binary regression trees. The material in this Uh video inspired by the textbook "elements of statistical learning" by Tibshirani.

(Refer Slide Time: 0:42)



So binary trees try to solve the regression problem by dividing your input feature space recursively into widgets and assigning a constant value to the output if the input features fall into that particular reason, okay. So let's consider this example where we have a continuous response Y and an input factor X with 2 dimensions. In this case we will refer to them as X1 and X2.

And of course the task is to predict Y given test inputs X1 and X2. So assume that you have given a training data and of course we will see how recursive binary splitting of input space give you the desired results, okay. So let's start with the feature space X1, X2 there are 2 dimensions, right? So we recursively split by choosing a threshold along X1, so we will call that t1, okay.

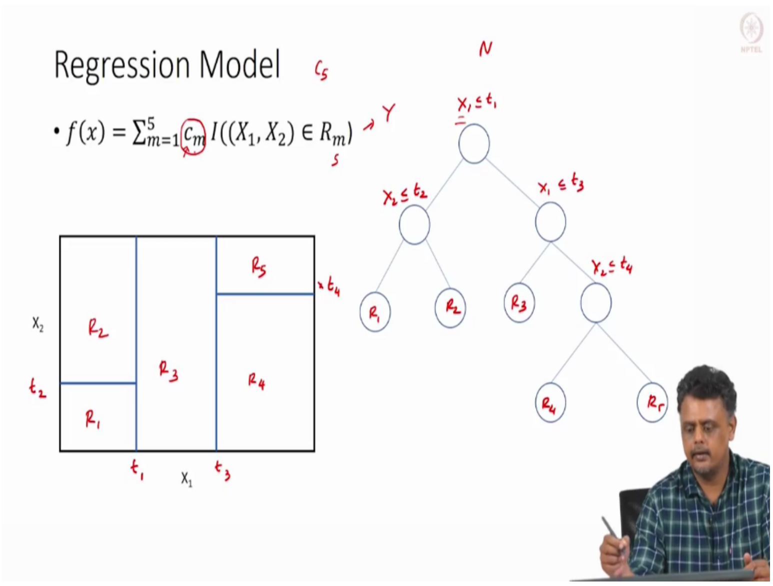
So now once we choose this threshold t1, we see that the input feature space is now divided into 2 regions, one to left of the blue like and one to the right. We can do a further split, so we take the region on the left and we can do a further split by choosing a threshold along X2 call that t2, right? And we go to the right again, here where we split this again by take that by Uh considering threshold t3 along X1.

Because then we look at again once when we choose t3 then this region is split into 2, so we choose the region on the right and then once again we can split this region into 2 by choosing another threshold t4, okay. So we get about, if we go we will get R1, R2, R3, R4 and R5, right? So what I have not specified so far is, how do we determine when to stop, right? So how far do we go? How often do we keep splitting?

So every time we land up with 2 regions after each split and we and each of those regions we split further into 2 and we can keep doing that till a specified point or till a specified criteria is met, okay. Now once we have these regions, how do we determine the output, right? For new test data.

So how do we determine the output? Let's say we have test data some X1 X2 some value is given we are going to put tilda there so that, I am confused. So is X1 tilda X2 tilda fall in region X R1, right? You can block that, so this is X1 tilde and X2 tilde this is your region. R1 into which it falls then how do you determine what Y would be, okay. So that's what we are going to look at now.

(Refer Slide Time: 3:39)



Here this is the model determines Y. So what this formula says is that you consider the region in which X1 and X2 fall in to, okay. So let's say it belongs to region 5, right? Here I is an indicator function, so it returns true for the region into which X1 and X2 falls into and what it says is, you just assign a constant value c of m, so let's say c of 5 and that constant value is written as the output.

So that will be the same output for all X1X2 that fall into that particular region even the test data as well as the training data, okay. So let's see how we can fit this into a tree like structure, so that it becomes more obvious. So we start with all the data points, so we have capital N Datapoint, let's say. And we will consider the first threshold at the top note here, so each of the circles is note.

So we consider with the top note, let's say X1 less than or equal to t1, so this is the threshold t1 here, okay. And it splits into 2 like we saw in the previous slide and on the left-hand side we will look at X2 less than or equal to t2 (())(5:06) t2 giving rise to R1 and R2 I might have to reverse things here. So this will be R1 R2, okay. Okay. So we take the region on the right-hand side.

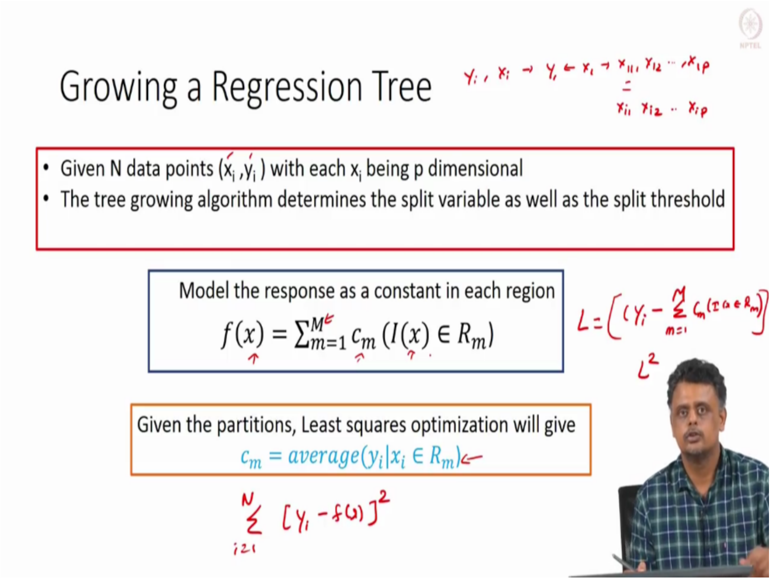
So X1 greater than t1, so that region we split again into 2 by considering X1 less than or equal to t3, so we will get R3 here which corresponds to this region, okay. And on this , so that gives you 2 regions one to the left of t3 one to the right of t3 and the right region we once again split by considering X2 less than or equal to t4, this is t4, okay. So this will be R4 and R5.

Let's with consistent with how we, if this is true then it is R1, if it is greater than, then it is R2 which is correct, so that's the order in which we have divided the input feature space into regions. So now we see how we can fit this recursive splitting into a binary tree, so next step is to see how to actually blow this tree. So what I mean by growing this tree is.

Determining how we choose this features to split, so in this case I just chose to split with X1 in the beginning but what is to stop you from using X2, let's say, okay. So how do you determine which feature to choose and in fact the threshold also, this t1 how do you split based on t1? why is t1 so special? The 2nd problem to solve of course is to figure out what this cm are?

In this case we have seen subscript m I just said some constants that we assign to any input features that land in that particular region Rm. So how do we determine cm 90 number of partitions, okay? So that is the problem that the tree growing algorithm solves, okay.

(Refer Slide Time: 7:34)



So to grow a regression tree Uh we will consider this problem where we have N Datapoint is xI, yI Uh the notation here is slightly differs from the previous slide and just to make simpler. So but in this case you have to consider the full problem, so you have N data points xIyI with each xI being p dimensional. What I mean is that, so you have yI is the output corresponding to input xI.

But each of the xI are p dimensional in the sense, so let's say x1, so y1 is the output corresponding to x1 and x1 itself is p dimensional in the sense x11, x12 so on and so forth up to x1p, okay. So the input vector p dimensional, so in the previous case you are looking at 2 dimension, right? So here it's p dimensional mode general case. So the tree growing algorithm determines this split variable, which of these should be split?

So is it x1I12 or if you write this down xi1 xi2 up to xip, right? So we have to determine which of these features we have to choose, is it 1, 2 or 3? Or the pth feature and what is the split point itself? For in that feature, in this case to put in the context of the previous slide what should be the value of the t's that we use as thresholds, right? So the basic model is, is to model the response that is the output wise as a constant in each region.

So we determine which region X falls into and we have a constant which describes the output in that region, we assign that as the output, okay. We actually formulate this as a "Least square" problem. Let's say the number of, in the sense that if we know the number of partitions beforehand, so if we know the number of partitions beforehand then we can provide the least squares problem as.

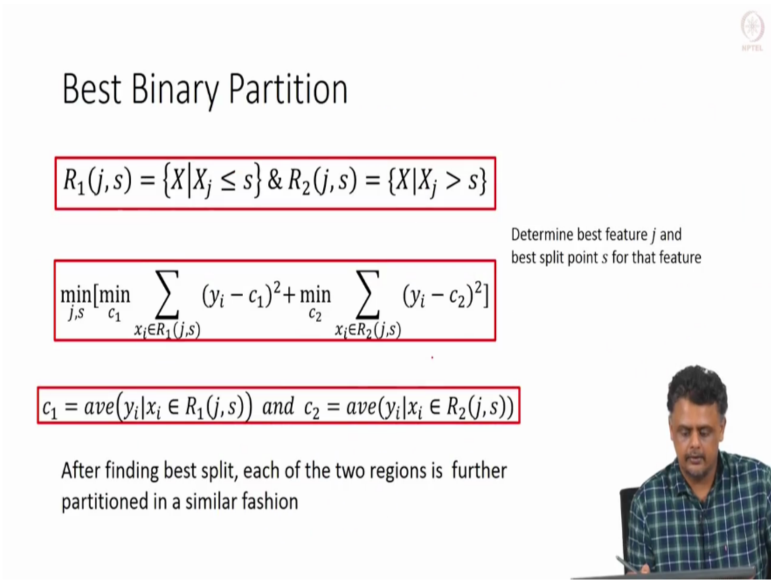
So, let’s say we will call this L. So this is L squared is what is your cos function, right? So I just went too far across, so I’ll have to write it down here. So let me rewrite it at the bottom. So we are looking at basically Yi minus f of x squared, right? Here f of x is given by this formula, okay. So here if we know the number of partitions m, alright. So we know that we are going to split the input region into m and we actually know about the m region R.

Then the cm’s are easily determined by solving this optimization problem. Now if you take the derivative with respect to the unknowns in the case cm then we have easily shown that cm’s are nothing but the average value of Yi in each of the regions, okay. So we know the output corresponding to every xi that falls into a particular region Rm.

And so with the training data, so we take a mean of those points that will be the output assign, okay. So if we know the number of partitions beforehand and if you post this as a least squares problem then the response that each of the partitions is nothing but the average of the response of the training data falling into the particular region, okay. So that’s the solution.

But now the problem is that we actually determining the number of partitions beforehand is computationally very difficult problem to solve because you can see that there are so many combinations that are possible as per the number of regions in order to optimize this particular least squares cross function. So BD strategy is adopted.

(Refer Slide Time: 11:49)



So how it is solved? So what we do is, we start off by solving the test where the recursive binary splitting comes into play. So we chose a particular feature and we determine the best split on that feature, so the best split point for that feature, okay. And what feature we choose and what are the best split points? Is determined by solving this optimization problem, okay.

So for the sake of charity let's say we have we have chosen feature j and its split point is s, so if it's a continuous variable you can think of s as a threshold like we had in the previous slides t1, t2, t3 and t4, okay. So j is your feature index and s is your split point or threshold for that particular feature. So if we split your entire based on that particular feature then we will get 2 splits 2 regions R1 and R2, okay.

And so then we can write down the last function for the optimization problem. So it has Uh the inner and outer optimization look, so the outer one is the one that determines the best j and the best s for that particular j, if you look at the inner optimization it is actually trying to figure out what C1 and C2 are? In the sense that, so we have and every node in the tree that we saw earlier.

We have all the data the input Uh training data and we decide that there are only going to be 2 partitions and what we have to estimate now is that, what is the best feature and what is the best split point for that partition, right? By optimize last function that's all we need to do. And it turns out that C1 and C2 like we saw earlier is nothing but the average of the points that fall or average outcome of the points that fall in that region.

And here is the average again outcome of the points that fall into R2, so that's the solution for the particular split, okay. So turns out by, we can determine j since this problem is easily solved, the inner loop is easily solved we can scan through all the features and find out the best split point and the lowest cost function for the split point and we choose the one which gives you the least cost function.

So once we split at a particular note then we are left with 2 other 2 regions and then we adopt the same procedure, we go to each one of the regions and then split them again into 2, okay. So the next question is where do we stop splitting? Can be just keep going on? Typically the way it is done is, the stopping criteria is not very well-defined.

So usually the tree is grown to predefined depth and then there is procedure called pruning which helps to bring down the depth of the tree. It is easy to see that as you increase the depth of the tree then it is easy to overfeed, right? Because you can always split your input space into smaller and smaller regions, so your training data will be fit perfectly of course your tested data there will be a lot of error in your response.

So in the other next videos we will look at how many trees are used for classification problems, thank you.