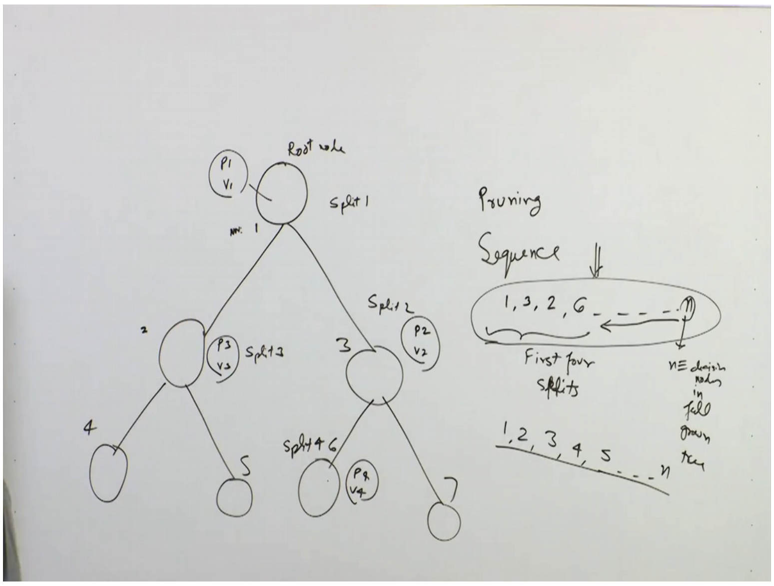
Business Analytics & Data Mining Modeling Using R
Dr. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology, Roorkee

Lecture – 43
Pruning Process- Part II

Welcome to the course business analytics and data mining modeling using R. So, in the previous lecture, we were discussing classification trees, in particular, we were doing an exercise in R for the same. So, we did some modeling using the promotional offers data set. So, we talked about the way we did a modelling, there especially, the pruning part.

So, we were specifically focusing on the pruning part and there; when we try to prune back the full grown tree to a label where it does not over fit the data or fit the noise the way, we followed the pruning process that was you know a sequence of pruning was as per the node number ordering and it was not the nested sequence, right. So, we talked about a bit about this in previous lecture where we discussed that if this is our root node.

(Refer Slide Time: 01:16)



And in this root node we will have a predictor one and value one. So, predictor value combination based on which the split would be performed. So, some observation will fall in this part other observation will fall in this part. Similarly, for next split, we have to see that whether on this node or this node you know where the reduction performing you know for the these nodes where the optimum split mole reduction in impurity is going to take place.

So, let us say; the next you know impurity reduction high impurity reduction happens in this particular node. So, let us say, this is happens at variable P 2 and V 2 right. So, this is going to be about a split 1 this is split 2, then after the split is perform some observation will go to this side other observation will go to this part. Now, again for next split will have to check between these 3 which on you know which particular node and which particular predictor value combination will improve the impurity further, right improve the impurity the improvement reduction and impurity would be highest.

So, let us say now that here at this node the reduction in is impurity is highest, then this is let us say the predictor value combination for the same is here. So, this is split 3 right now. So, here again we will have some observation that will go into this part some observation will go into this part right now for next split. Now, among these 4 nodes will have to check which one is giving the most reduction in impurity let us say this is the split this is the node and we have a P 4 V 4 and predictor value combination and it will be split 4.

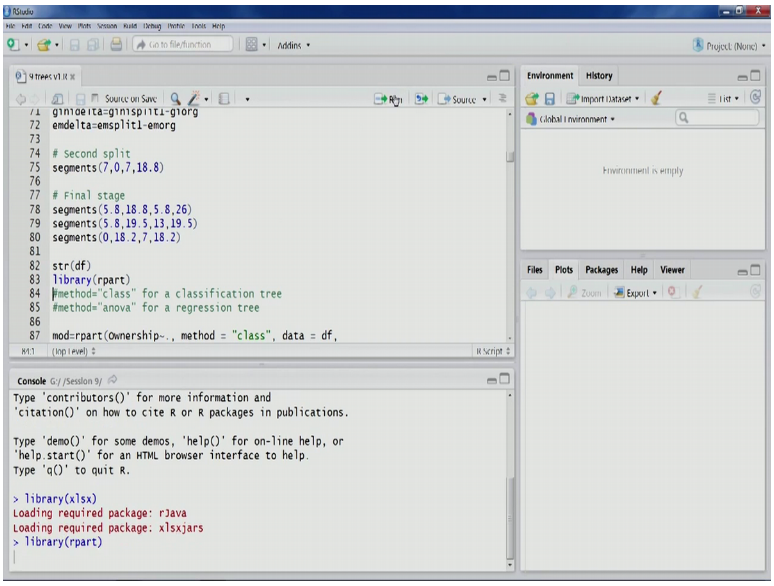
So, the pruning sequence. So, from this we wanted to derive the pruning sequence. So, we look at the pruning sequence, it is going to be this node, right. So, if it is node number one. So, if we follow the unique you know node numbers that ordering that we discussed in the previous lecture this is going to be node number one this is going to be 2, this is going to be 3, then 4, 5, 6, 7.

So, our pruning sequences first node number 1, then the second split happened at node number 3, then it happened at node number 2, then it happened at node number 6, right. So, 4 first 4 splits in this is example, if we look at first 4 splits. So, they happen in this order. So, when we prune back the full grown tree to a certain level will have to follow this splitting pattern. right ah. So, let us say last know if there are n number of splits ah; that means, actually this is going to be n number of this is going to be equal to the decision nodes decision number of decision nodes in full grown tree.

So, therefore, we have to when we start pruning the full grown tree back to the desired levels will start deleting the you know least important splits; that means, splits which have done a least amount of reduction in impurity. So, probably we will start from here and go our way back to the higher up to level. So, that we get to a point where the error on validation data is minimized. So, essentially the exercise that we had performed in the previous lecture the pruning that we had that we were following was based on this.

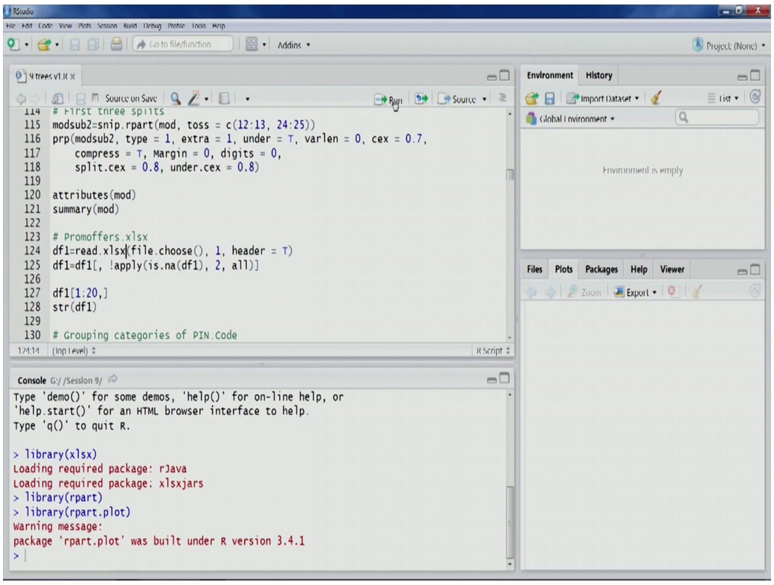
So, we just looked at the road node numbers and you know pruning was based on this. So, we are following the sequence in the increasing order as per the node numbers the optimal way of pruning that we want to follow is this one. So, today we will do an exercise in R, wherein, we will follow this particular pruning sequence and then let will understand few of the, you know few more points using a particular exercise in R. So, let us start. So, first let us load this particular package x plus x. So, let us go down. So, all these things we have already done.

(Refer Slide Time: 06:16)



In previous lectures, let us load this program package as well we would be requiring this R part and one more package we would be requiring this one as well R part dot plot. Now let us move to our data set. So, promo offers dot x l s x is the file. So, we would like to import it here in R environment.

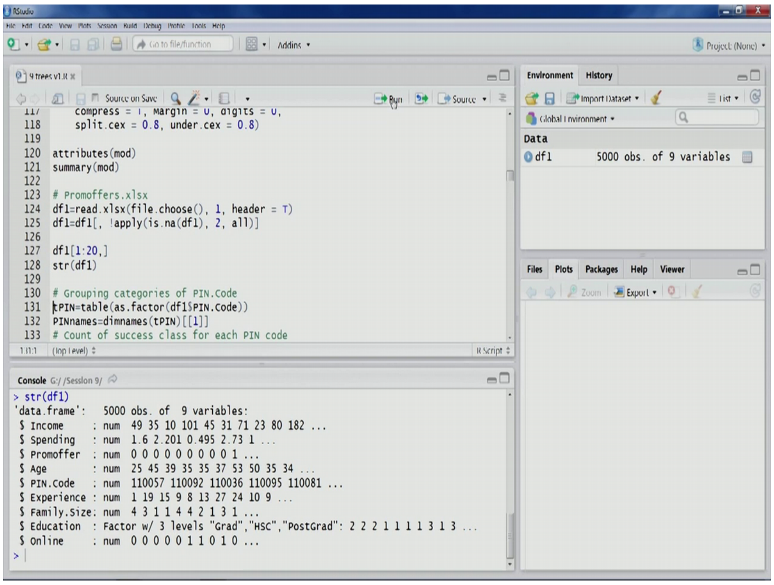
(Refer Slide Time: 06:36)



So, let us perform this. So, it will take some time because it has this particular data set has 5000 observations. So, it will take slightly more time that we have been doing for other datasets smaller datasets.

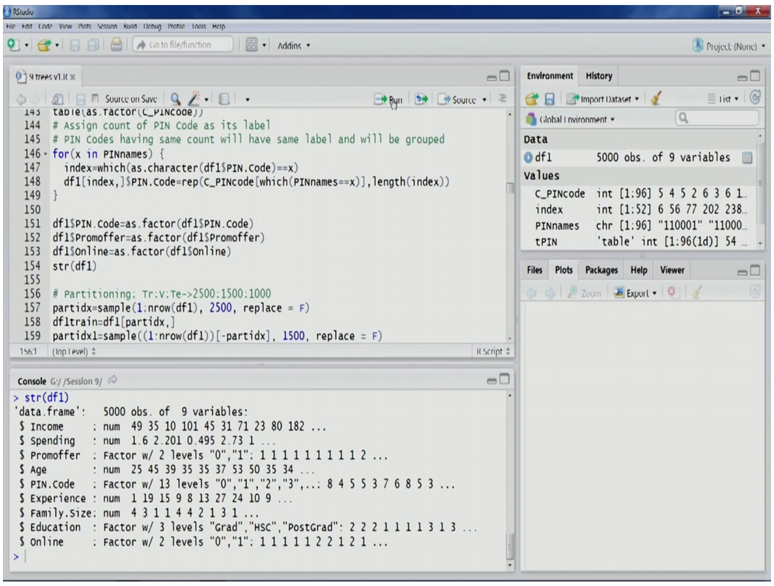
So, once this particular data set is loaded we will go through some of the steps that we had performed in the previous lecture and once that is done. So, once the pruning specific steps start, then we will discuss what we have covered here. So, you can see all the observation 5000 observations of 9 variables all of them are loaded in R environment. Now let us move any columns structure these are the variables.

(Refer Slide Time: 07:38)



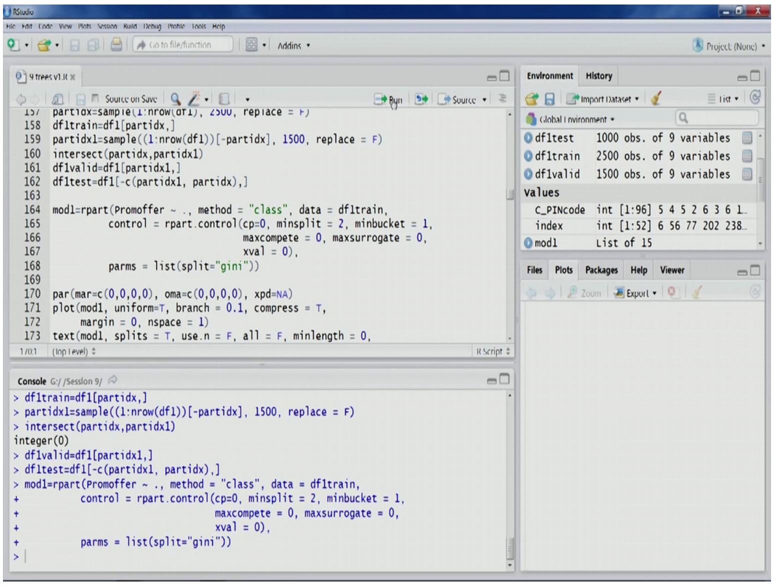
So, some of the steps will have to quickly go through for example, we did grouping of categories. So, we will have to perform this again. So, that we are able to reach to the same point. So, let us go through this code we have already discussed this part before. So, we are just going through this. So, that we are able to create the; so, this is the now our data frame is ready all the variables are in the appropriate you know types data types numerical and factors.

(Refer Slide Time: 08:00)



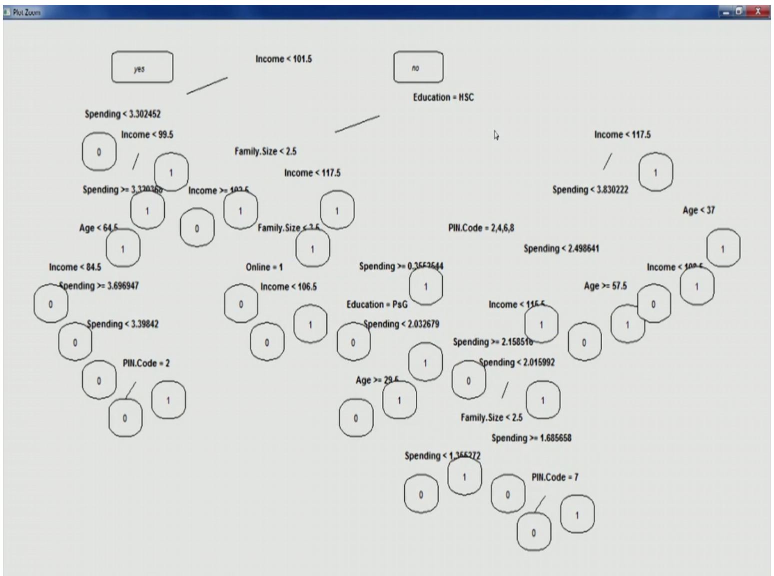
Now, let us do the partitioning already discussed these steps as well. Now let us build the full grown tree. So, this is the code that we had used before.

(Refer Slide Time: 08:19)



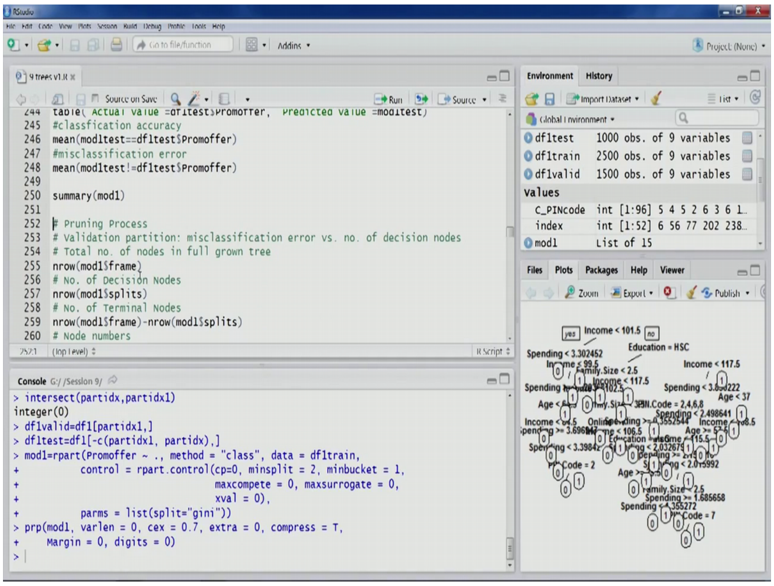
So, you can see here as we have discussed before that x well a value is 0, right, by default this is 10. So, that is reserved for task validation. So, just we want to pull the full grown trees let us plot this.

(Refer Slide Time: 08:44)



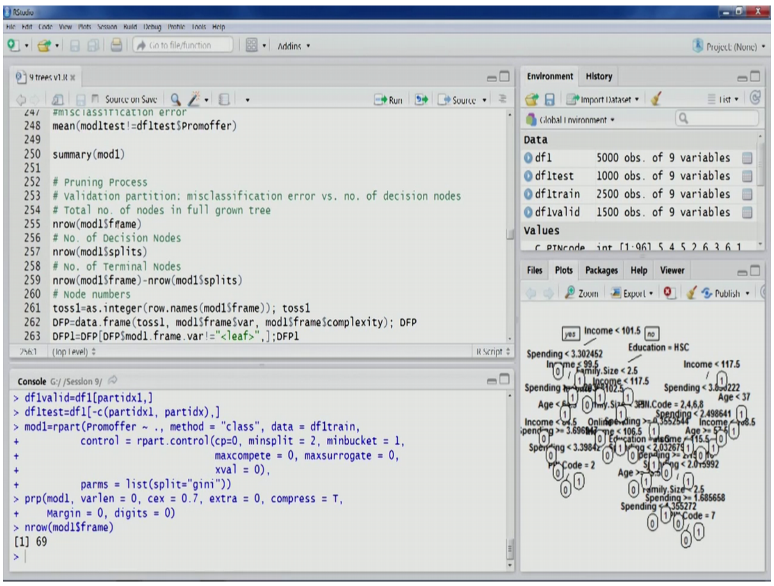
So, this is our full grown tree you can see quite messy as we had seen, it in the previous lecture as well. So, now, let us move to the point where we wanted to where we want to discuss further. So, the split variable and value combination this particular table we have already discussed we have gone through this.

(Refer Slide Time: 09:11)



So, we will not do this again performance of full grown tree we have gone through that. So, let us come back to the pruning process where as I discussed we followed a different pattern you know different pattern for pruning. Now we will follow the actual pattern the desired pattern for based on complexity.

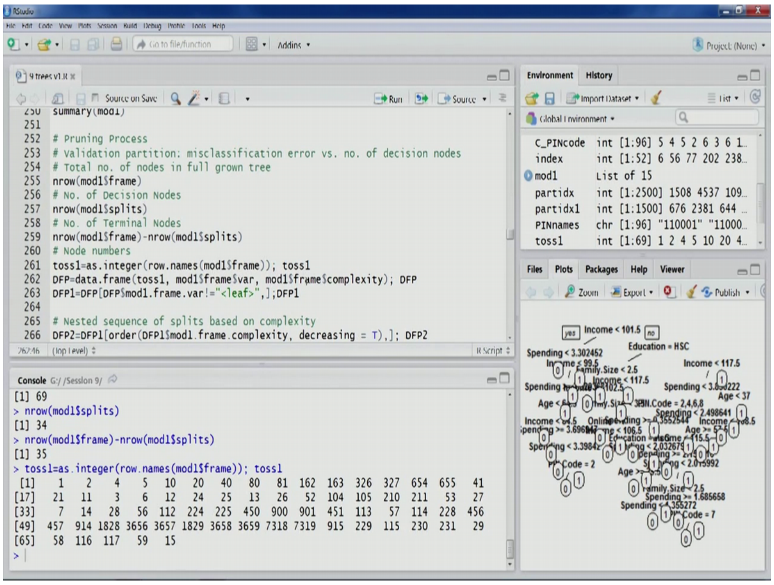
(Refer Slide Time: 08:36)



So, a pruning process let us look at the number of total nodes in this particular tree as you can see number of total node number of total nodes 69 and 34 in the 34 decision nodes and 35 terminal nodes. So, node numbering; so, you would see now certain steps that we had performed you will see differences now toss one is the argument that we want to compute at this point which we would be passing on to this snip r part function now toss one.

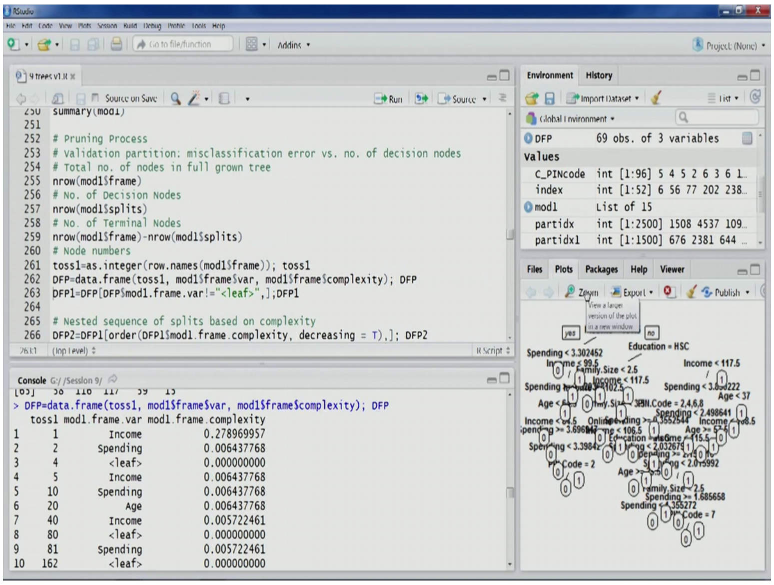
So, as we have discussed that r part object it has a flame attribute and within that frame attribute it has the row numbers. So, this we have discussed in previous lecture. So, we will get the row numbers will convert it into integer vector. So, that we will have the these numbers unique node numbers ah, but the ordering is not at for the desired order. So, now, we will constrain now we will create this data frame where we have the these node numbers in toss 1 and we will also have the variables write the variables involved at different nodes. So, whether the decision nodes are leaf nodes for leaf node it would just mention leaf as we have seen in tables in the previous lecture.

(Refer Slide Time: 11:54)



Now, for each node, we will also have complexity value which is also still stored in the frame attribute and within the frame we have this complexity variable. So, it would be stored there. So, let us create this data frame you can see here let us scroll through this particular data frame, as you can see first column is toss one which is nothing, but unique you know node numbering with respect to rows.

(Refer Slide Time: 11:24)

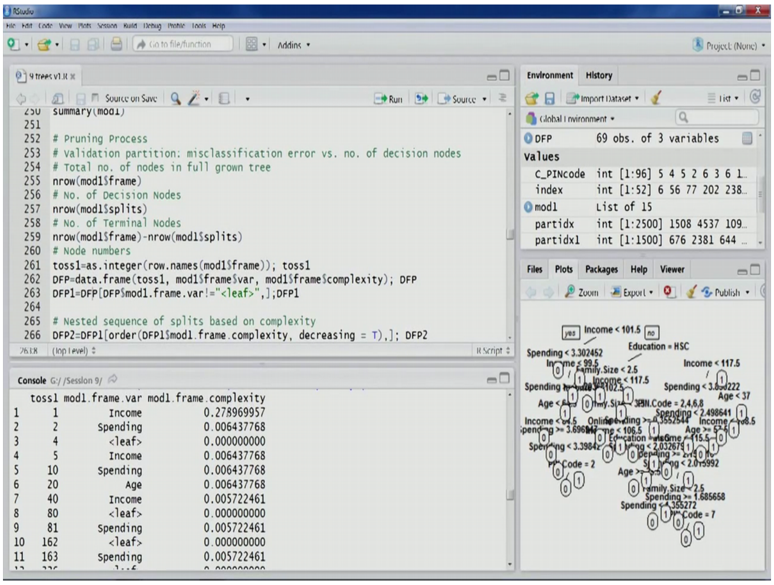


The ordering of these we have already discussed in the previous class, the previous lecture that it follow this sequence node numbering we actually discussedn it by showing the node numbers.

So, 1, then 2 and then 4, 8 in this fashion these numberings are going to be there. Now once this data frame is there. Now you would look, you can see that in the second column, you can see the variables that are involved here and the involved variable have the you know income is spending, then leaf for each of these nodes whether what predictor was used if it was a decision node what was the splitting variable for that decision node and what was leaf node the leaf, it just means the it mentions that this that particular node is a leaf or terminal node the corresponding complexity value the complexity parameter concept that we talked about that is used to control the size of the tree.

So, the corresponding complexity value for that particular node is also mentioned. So, this is the value at which point the tree will collapse so, based on this. So, we can perform our pruning so that will give us the you know that will control the our tree size and will give us the you know best prune tree and minimum error tree. So, we will do this. So, before that we will like to order this particular data frame.

(Refer Slide Time: 12:54)



In decreasing order of complexity values right. So, the starting nodes from where the first split and then say onwards other splits happen. So, the starting nodes will have higher complexity values, right here the complexity value would be much higher that is why this was the split 1 number 1 here the complexity value would be after this.

So, that is why it was split number 2 followed by you know split number three and spirit number four. So, we would like to order this particular data frame by complexity values and once we order this particular data frame that we just saw by complexity values will also get the this sequence, right because this was the first split and the complexity value will be higher for this, right. So, this would be first then this was the second split complexity value for this particular node is going to be the second one after this. So, it will come here.

So, once we order this particular data frame which is having complexity values for each node we will get this. So, let us do this execute this code. So, you would see that before ordering first we are trying to remove the leaf node. So, we do not want to have a leaf node at this point you would see there are many leaf nodes here. So, we would like to remove the leaf node because the pruning is basically driven by the decision nodes, right. So, once we remove the leaf nodes from this data frame, we will get the new one this is the one were.

So, this is the one this is the new data frame that we can see DFP 1, we have 34 observation which is equal to the number of decision nodes that are there this is this these particular numbers, we have already seen in the previous output as you can see 34 decision nodes and 35 terminal nodes. So, once we remove the leaf nodes the number of you know observation that are there in the new data frame the 34 equal to the number of as you can see in the environment section 34 equal to the number of decision nodes.

Now once this is done we can order as we talked about we want to obtain the nested sequence of splits based on complex tree. So, we will order this data frame based on the complexity values and once we order this will get the desired nested sequence, right. So, let us execute this code now you would see that ordering has been done and if we scroll back to see this table now the first you know you know first is entry is income variable. So, this is the split number one and the complexity value is there the second split is also having the same complexity values, right we will discuss this further what happens if we the same complexity values are there, then why you know income was the first split and education was the second split ah. So, considering that what happens when this is the scenario?

So, family size and then third mode the third spirit is based on this having the third highest complexity value. So, in this fashion you can see that complexity values are decreasing. So, this is this is how our trees when we develop the full go grown tree. So, this is how this sequence determines how the splits are going to take place and how the tree is going to be built. So, once we start deciding about pruning you know pruning this full grown tree this is the process that we have to take and therefore, the earlier one that we did in previous lecture is not the desired process now you would see because we have sorted this particular data frame the row numbers have changed you can see these were the original row numbers we present in the original data frame DFP 1.

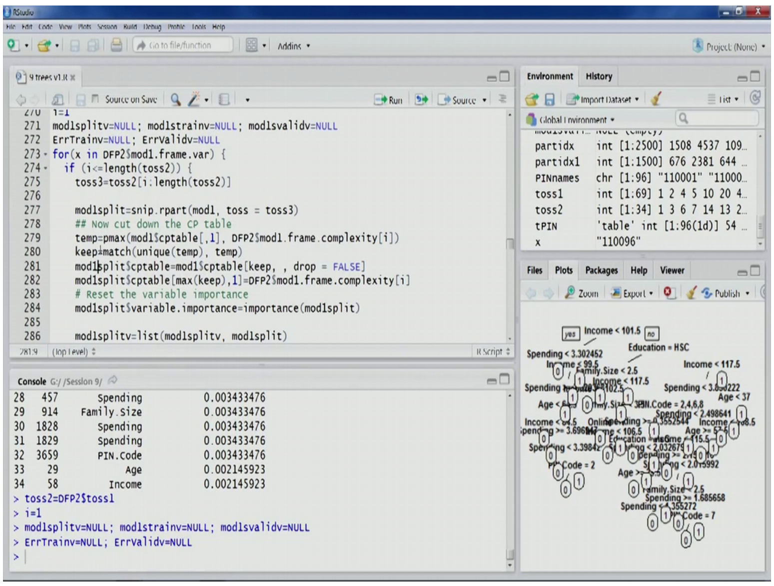
Now one sorting of that has been performed the row numbers are still same. So, we would like to change these row numbers to reflect the now DFP 2, let us look at the table like again you can see now the row numbers row numbers are also sorted. So, 1 to 34; 34 decision nodes, right. So, once this is done, now we can start calculating our toss argument that we have to pass on to the snip dot r part function. So, toss the 2 argument can is simple nothing, but in the data frame that we have just you know created the P 2; the first you know variable toss one that is going to be this argument.

So, let us create this toss two now what we are going to do is we are going to start our pruning process and as we did in the last lecture after every pruning, we used to record the model and we used to apply that model to a score on training, you know partition and other partition validation another part validation partition training and validation partition. So, that later on we can compare the error rates right.

So, the same thing will follow here what we did in the last lecture, but now this time with the actual pruning sequence the nested training sequence. So, counter for nodes to be sniffed off I and once, then we have this mod one split v the same wherever that we use in the last lecture this is going to you know store all the mod variables then you know mod 1 train v, these variables are going to store the other things that we will see this course right mod one mod one train. So, it will its score it will have the that return value of predictor.

So, it is scored variable right list ah. So, let us initialize them, then we will have these two vector 0 train v and other valid v. So, these two actors are the important for our plotting and to identify where the error on validation partition is minimizing. So, let us initialize these two variables now you can see as we discussed in the previous lecture the loop is running for all the variables. So, the in the this in this particular case you can see we are running this loop for all the decision nodes that are there right DFP 2 in this particular column.

(Refer Slide Time: 19:34)

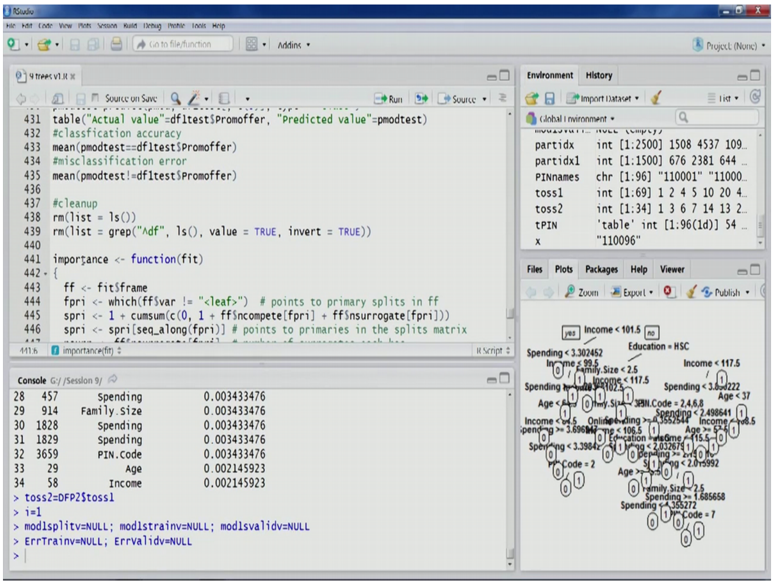


Now we have only the decision node you can see again that in environment section DFP 2 has just 34 observation that are the number of decision nodes in this particular tree. So, we will run this loop for the number of decision nodes. So, some of the checks that we had done in the previous lecture; that code that we were eliminating the leaf now we do not need to perform because we are dealing with only decision nodes. So, the if if section, you would see that I; we are comparing with the length of the toss two that is the total number of nodes and then because we would be pruning a node by node. So, we are starting this protocol process from i that is one to the full to the final node number that is the last one.

So, first we start by you know pruning all nodes, then you know from node number 2 do the last one node number 2, in the sequence not node number 2 actually node number 2 in sequence as a stored in toss 2. So, to show you the toss 2 values you can see toss 2 1 to 34. So, 136. So, node number are actually unique node numbers are 136. So, first we start by you know first we start by sniffing all the nodes, then we start by sniffing from this particular node to the remaining nodes then we start from this particular node that is 6 to the remaining nodes in this fashion we will start and then a snip dot for part function is being called for the for every time the loop is run and you are recording a few more we are correcting few more things.

For example, CP table; once we create and you know once we do this sniffing, we will get the new model new sub tree model. So, therefore, we need to correct the CP table and there. So, the code for the same is there then one CP table is corrected will also have to correct the variable importance code for that is also there right. So, for this we are using an importance function which is nothing, but taken from the source code of you know prune dot r part function. So, there they have written this importance function.

(Refer Slide Time: 21:53)



And we are directly using the same source code here in this our exercise because this particular function is not available for us to you know call you know is not part of the r part library, once we load they are they do not have access to this function this is called internally within r part. So, that is why we have to get that source code here and to be able and then we are using this particular here.

So, we will have to now create this function here. So, that will do here. So, the you and you would see that in the environment section function this importance function has been created. So, we will not go into the detail of this particular function this function is actually being called to once we create the sub tree model, we would like to we would like to change the variable importance accordingly right in the sub tree model. So, the same thing is being done by calling this function. So, then we are storing we keep on storing these you know; all these all these sub tree models then we score them off score the training partition and the validation partition as we did in the last lecture and then we are storing the error rates for training partition and validation partition and this is the entry look counter.

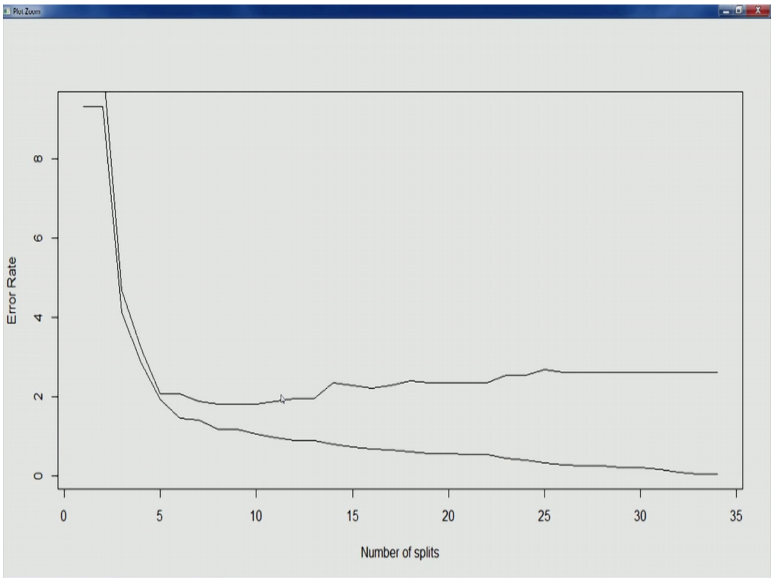
So, let us execute this code. So, it is done. Now you would see that one thing I would like to point out here that those we have been storing the models, but we cannot access all of them you can see this is quite large list and given 3 MB and just having two elements. So, these this R part object that we were trying to store in list there you know the size is quite big. So, therefore, it has not stored all the all the you know all the R part or model sub tree models and therefore, only two are there. So, ah, but; however, we are interested in only the error rates.

So, let us create this data frame like we did in the previous lecture. So, now, let us look at these values 4 for decision nodes in this ordering sequence and you can see either training and validation. So, now, when the for the first decision node this is the training error and validation error you can see that training error is slightly lower than the validations you know error when we start and as we perform second split then again the both are same.

So, there is not no not much decrease in error after second split then third you would see that further the error has significantly decreased for the training as well as for the validation. So, in this fashion if you as we did in the last lecture if you scroll down this these are rates the second column that is error rate for the training part it will keep on decreasing till it becomes 0, right till it become 0 or close to 0 right and in the in the validation partition you would see that error will keep on decreasing till one point and after that it will start in you know increasing.

So, you can see that this is this is the point where the error is minimum, right. So, this is the point where the error is minimum and then after this particular point it will it will hold up to for some more nodes and then it will start increasing that it keeps on increasing. So, with this now we can go to you know we can also we can create this plot to visualize the same information then information that we saw in table.

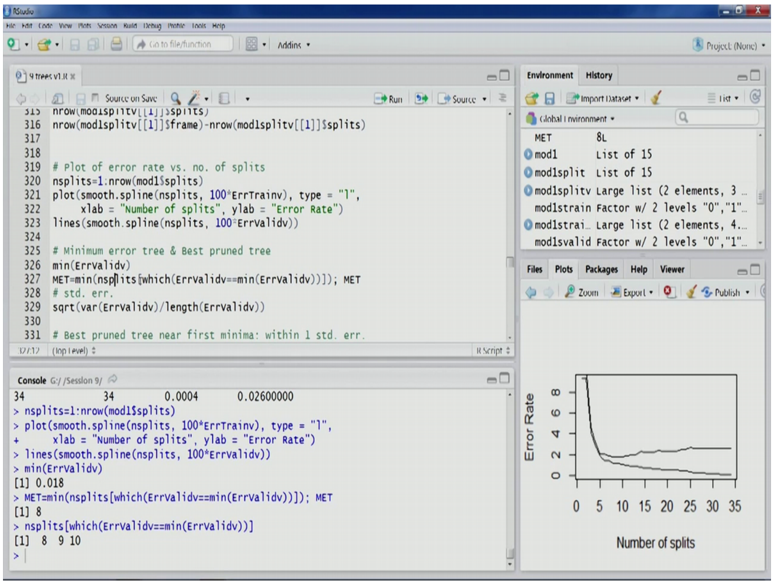
(Refer Slide Time: 25:45)



So, this is the plot that we had seen in previous lecture as well now with the correct pruning sequence you can see that the plot which is this plot this is you know the this particular is for the validation data the lower plot the upper plot is for the validation data and the lower a plot is for the training partition. So, for the training partition you would see that the error you know keeps on decreasing till it becomes 0 for the validation part you would see the error keeps on decreasing up to some point and after that it will start you know it will start increasing right.

So, probably here we need to in this particular zone we need to find out the point with minimum a territory like we did in the last lecture and then within one standard deviation we will have to find out the best tree. So, let us look at this value minimum error tree is this, this is the value which we already saw in the table then let us look at the particular number of decision nodes corresponding to this error value error 8 decision nodes minimum tree is can we obtain at 8 decision nodes if we look at the graph again. So, 8 decision node would be somewhere around here. So, probably this is this particular straight link straightening line you see. So, all these you know nodes they are nothing, but representing the, you know minimum error on validation partition. So, whether they are 8, 9 or 10.

(Refer Slide Time: 27:16)

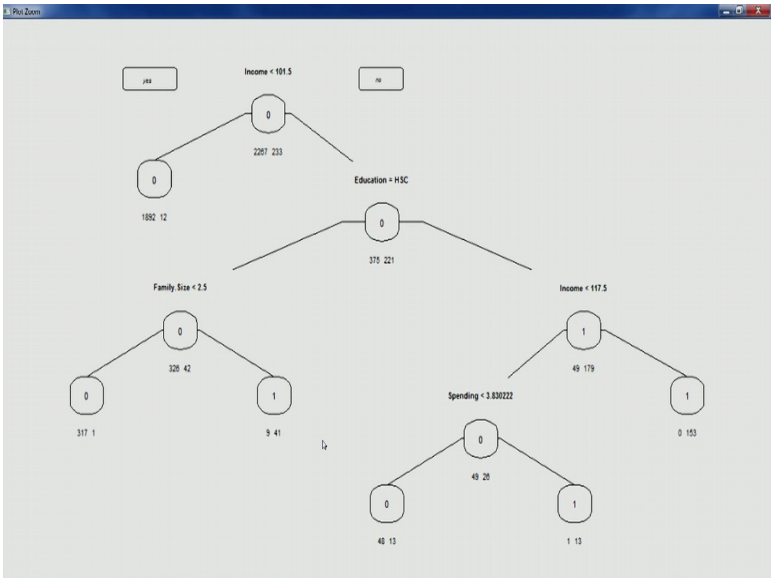


So, we can look at these values, if you are interested how many how many of these decision nodes are having the same or having the same number of same error minimum error 8, 9 and 10. So, the sub tree models with decision nodes 8 decision nodes and 9 decision nodes and 10 decision nodes; all three of them having the same number of same amount same amount of error on validation partition, but; however, we will just use the smallest tree here and then we look at the standard error of you know off error rate.

So, this is the value; now we will look at the range where we need to find the now best prune tree. So, the best from tree should be having value less than this particular value error like we did in the last lecture and should be greater than the error that we saw for the this one minimum error tree, all right. So, this is the code for the same. So, this part we have already discussed. So, you can see best prune tree is now coming at 5. So, if you want to confirm this, we can go back to the level table and we can see that node 8, this is the point where the minimum error tree is there, now within that range, we can see this particular tree is giving us the best prune trees this is within one standard deviation of minimum error tree. So, this part we have already discussed.

Now, once this is done once we have identified then we can go ahead and create our min best prune tree model. So, this is how again BPT we would like to contain we would like to contain these many number of design nodes. So, we can generate about toss three and then call this sniff R part and we will have the best prune tree. Now let us plot it. So, this is our best prune tree let us look at this particular plot.

(Refer Slide Time: 29:21)

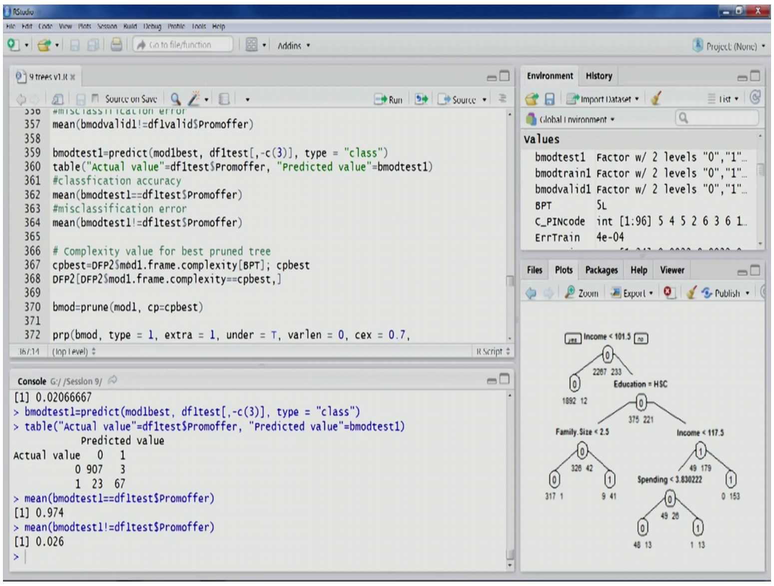


Now this particular best prune tree now the earlier one which we did in the previous lecture because we are following the order of you know shorter order of a node numbering. So, we were getting the balance tree. Now, we get the right tree would see that this is not balanced first income then education and then that sequence the it is it split sequence the optimized split sequence is being followed in this particular example, right. So, this is the best prune tree that we can have you can see 1, 2, 3, 4, 5 decision nodes are there you can see important variables of course, it is income education. So, income education families are spending.

So, all of them figuring here; now we can check the performance of this particular tree on different partitions; so, you can see the performance 98.56, then on validation 90.9 close number, then on test 97.4, this is also close. So, performance is quite good. So, there could be another approach to follow this process that we discuss in the previous lecture as well based on complex tree value.

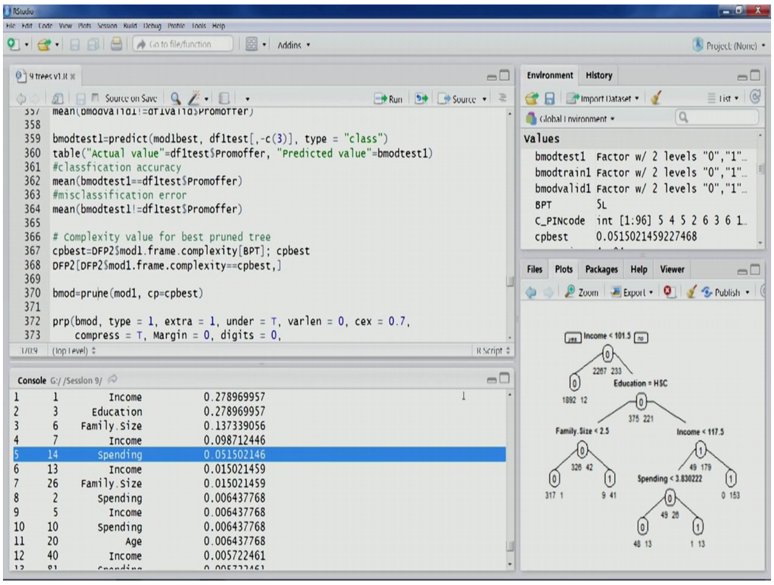
So, we use the complex tree value for example, we have identified the best prune tree now following the you know actual order that is the split order. So, in that we can find the appropriate you know complexity value because we have this pruned function this which we use in the previous lecture which takes the CP value and cuts the prunes the tree inter based on that CP value; however, we will understand some of the problems with this particular function.

(Refer Slide Time: 31:08)



For example, let us find out the complexity value for the best prune tree that we have just identified which was the tree with 5 decision nodes. So, CP best is this is the corresponding complexity value and this you can see, we you can see toss 1 is 15, right and so, this will using this particular value. So, we can go back to the table and find out how many number of nodes are there here ah. So, let us look at let us look at that table. So, if we look at the value that we just saw their 0.0515. So, you can see this is the value 0.0515 and we can see that toss 1.

(Refer Slide Time: 31:55)

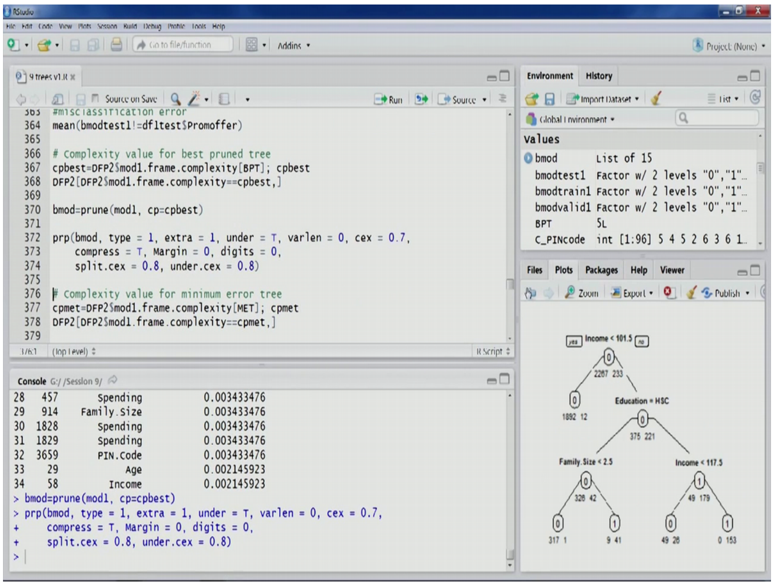


So, 1, 2, 3, 4, 5; so, this is also 5. So, the same you know corresponding tree is there, but; however, it might. So, happen that. So, now, we are discussing the problems that would be there with the prune function now the previous few values sometimes if we run the same model previous few values might also have the same complexity value in that case the tree with the smaller size would be selected by in this fashion. So, if we do the you know pruning using the complexity values, even though we have identified you know followed that passes minimum error tree and within one standard deviation best prune tree.

And now instead of the number of decision nodes we use the complexity value to prune this tree you know the previous you know nodes they also had the not in you know they also had the same complexity value. So, the pruning will happen will happen at that level. So, it is might with the tree size might reduce from 5 to 3 or 2 something in some scenarios in some runs and even in this data itself we do again the same thing, we do it again, then probably because of the sampling and the different observation that are going to be selected in the training partition and therefore, the different model that could be there because of the limitation on the sample size that we have even though this is larger data set.

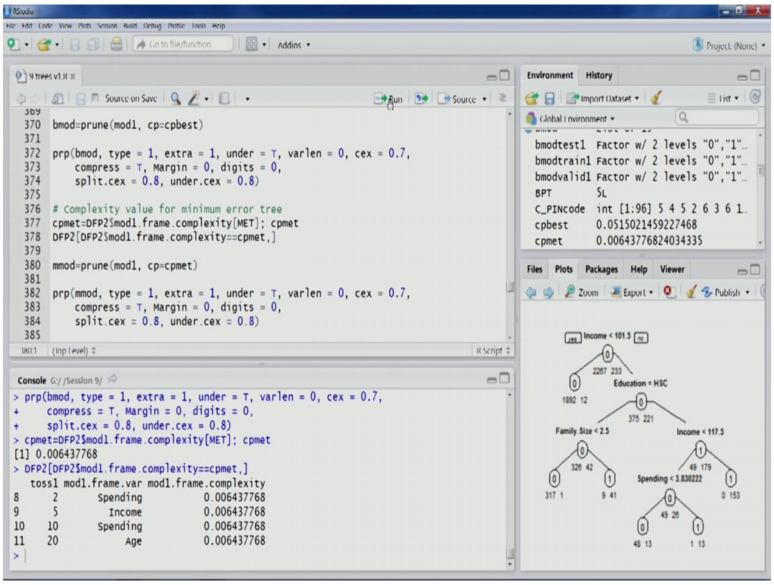
So, we can get different prune tree using this particular prune function. So, in this particular case it comes out to be the same. So, we can use prune function, we pass on the full grown tree model mod one and then the pruning value till the point where we would like to prune it. So, we can see this.

(Refer Slide Time: 33:43)



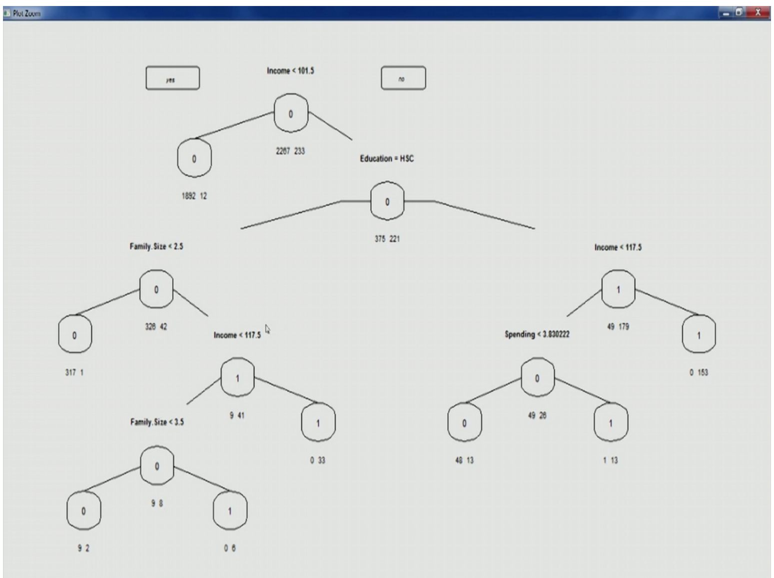
So, this particular tree; 1, 2, 3; you can see 4 nodes are there and here we had 5 nodes. Now in this particular internal processing that happens in prune function one more node they spend they spending one it has been removed off. So, that is the tree that we will have if we follow that complexity value right. So, the tree will collapse at that value collapse at this value right and only 4 particular decision nodes would be there. Now further we can we can we can compare this particular case with the minimum error tree. So, we can plot the minimum error tree as well.

(Refer Slide Time: 34:30)



So, this is going to be the corresponding complexity value. So, this is up following the prune process prune function process. So, you would see just. So, these are the nodes you can see this is the value. So, we look at the, we prune it. So, this is the model that we get. So, you can see minimum error tree model is much bigger, even if we follow the prune function right.

(Refer Slide Time: 34:56)



You can see 1, 2, 3, 4, 5, 6, 7 nodes; there right. So, we saw that size when the prune sequence the last one is also removed off. So, we get the 8 size, if you are interested in looking at other things; for example, CP table and other things. So, this per got aspect, we have already discussed, right. So, with this we stop here and in the next lecture, we will start our discussion on regression trees.

Thank you.