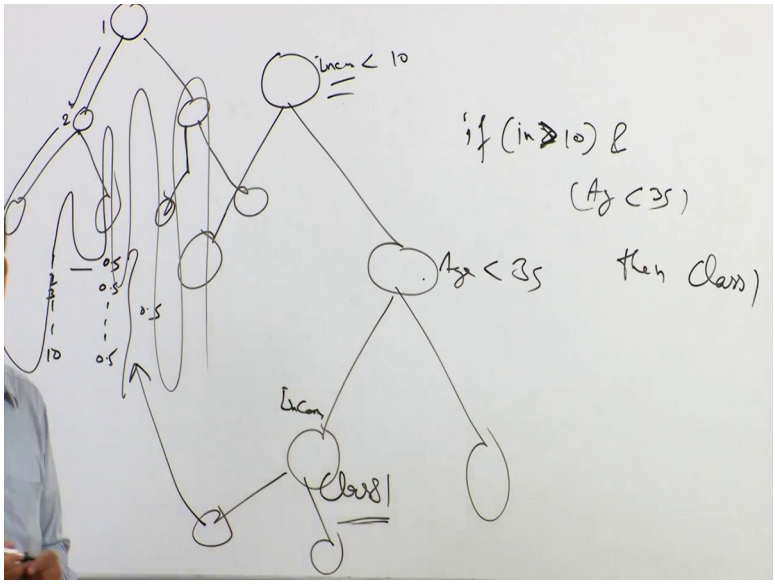
Business Analytics & Data Mining Modeling Using R
Dr. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology, Roorkee

Lecture – 45
Regression Trees

Welcome to the course business analytics and data mining modelling using R. So, in the previous few lectures we have completed our discussion on classification trees. So, in this particular we will move on to regression trees. So, before that let us discuss few more points about classification trees. So, out of classification trees we get some simplified classification rules. So, each terminal node or leaf node that, we get in our final tree model that is equivalent to a classification rules.

(Refer Slide Time: 00:59)



So, for any particular tree we can always we can always formulate based on the final tree model right based on the final tree model we can always formulate the classification rules for example, let us say this is income less than something 10, then this is let us say age less than 35 then in this fashion.

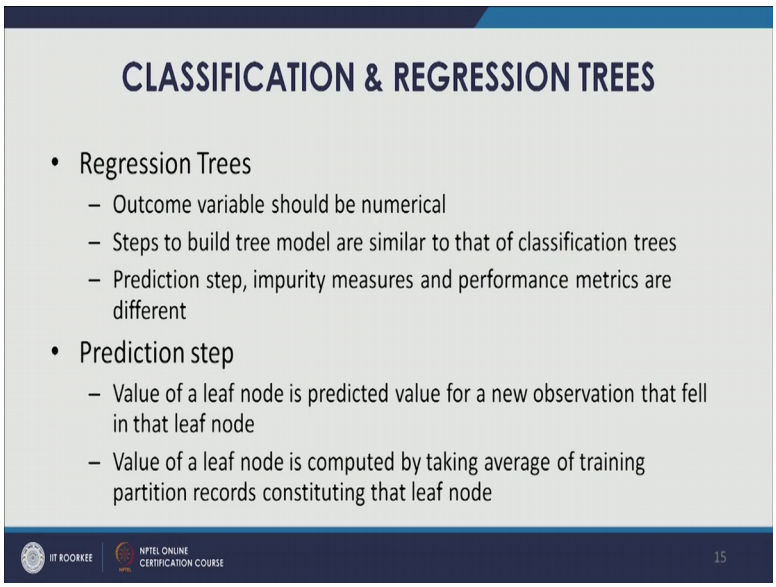
So, classification rule is going to be like if income less than 10 and age less than 35 then you know age less than 35, then let us say this is re if it is you know let us say greater if less than goes this side and greater than all the points greater than this go this side then it will come here and then age less than 35 will come here. So, observation will fall here let

us say this is class 1 so you can say and class 1 so these kind of simplified classification rules we can get out of a tree model so that that gives us the ease of implementation where these rules are very easy to understand these classification rules very easy to understand and easy to implement.

Now when we have build a you know when we finally, build you know select a build and select a final tree model then all the terminal nodes as we taught what they will represent some classification rules. Now these classification rules can be simplified further as you might have as we have seen in previous lectures in different exercise that we have done the same variable may might re occur again at some level.

So, let us say income comes here again right. So, therefore, there are going to be two conditions on income, two conditions based on income variable in your classification rule right. So, in those situations we might look at those values and we can simplify the rules further right similarly some rules you know out of all the rules that we might have based on you know number of leaf nodes we can further identify the redundant rules. So, if one rule is applied the second rule which might be kind of you know sub rule of a particular rule so those rules can be identified so those redundant rules can be identified and removed from the list

(Refer Slide Time: 03:44)



Now, let us just start our discussion on regression trees. So, regression trees outcome variables should be numerical. So, classification trees because that was for the

classification task so typically we were looking to classify looking to predict the class of a new observation. Now in this case regression trees the outcome variables would be numerical so we would look to predict the value of a new observations.

So, as far as steps to build tree models tree model are concerned they are quite similar to that of classification trees few differences are there for example, prediction step so that is of course, going to be different from the classification step right and the impurity measures they are going to be certainly different in the prediction task and the performance metrics are also different.

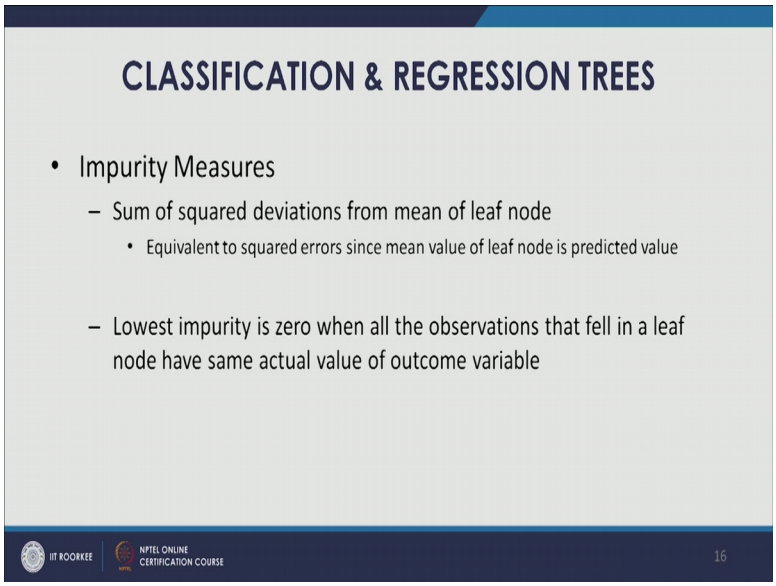
So, let us discuss these differences one by one so, first one prediction steps. So, a value of a leaf node is a predicted value for a new observation that fell in that leaf node. So, in a regression tree once when we want to predict the value of a new observation so again just like the classification trees that particular observation will also be dropped down from the root of root node of the tree and it will keep following a particular sequence and finally, it will reach to a terminal node.

So, the value of that terminal or leaf node is going to be the predicted value of that particular new observation. Now how the value of a leaf node is decided in a regression tree as we know that in case of classification tree that is the majority voting that is taken and based on that class assigned. In case of regression tree we compute average of all training partition records which fall in that particular same you know terminal node.

So, value of a leaf node is computed by taking average of training partition records constituting that leaf node. So, when we build that tree using the training partition points so the training partition record which are going to be part of that particular leaf node right which will you know fell to drop down to that particular leaf node.

So, average of you know those training partition observation would actually be the predicted value for that leaf node and any new observation that falls that when dropped down from the tree if it is falls down to that particular leaf node the predicted value for that observation is going to be the value of that leaf node. So, this is difference in the prediction steps otherwise in terms of building the tree the steps are very much similar to what we have discussed for classification trees.

(Refer Slide Time: 06:43)



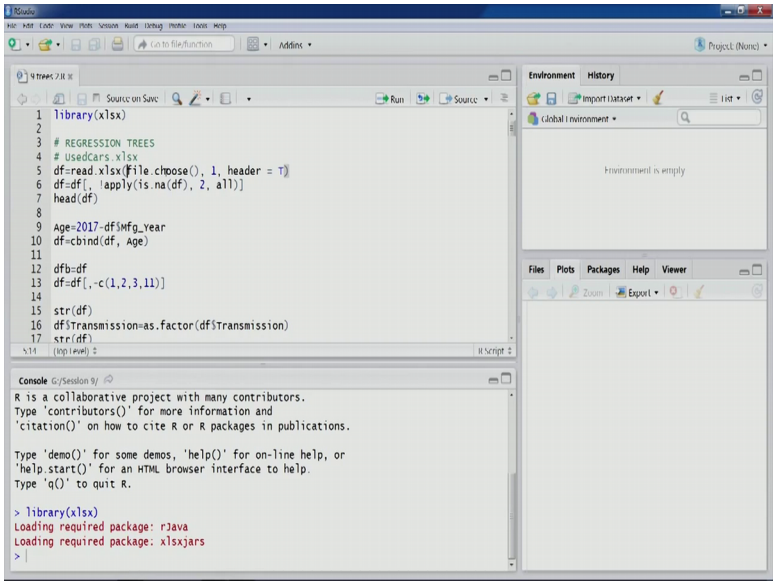
Now let us discuss impurity measures so impurity measures that we have used for classification tree were gini and entropy. Now we are in regression trees the impurity measure is different here we use sum of a square deviations from mean of a leaf node. So, once we reach to a leaf node right for all the observations that are there all the observations the of training partition record training partition that are part of that particular leaf node right during the rebuilding process.

So, me so their deviation from the mean value so their deviations from the mean value that actually becomes the metric impurity measure. So, sum of a square deviation from mean of leaf node. So, this is quite equivalent to squared errors since mean value of leaf node right the training partition records and their mean value is also taken as the predicted value right. So, therefore, the deviations are from the actual value deviation are between actual and predicted values so therefore, this is cannot this is also equivalent to squared errors

Now, lowest impurity is 0 and this will occur when all the observations that fell in a leaf node have same actual value of outcome variable. So, all the observations that fell in a particular leaf node right. Let us say there are 10 observation 1, 2, 3 up to 10 and if they have same value let us say 0.5 then mean will also be 0.5 and therefore the deviation would become 0.

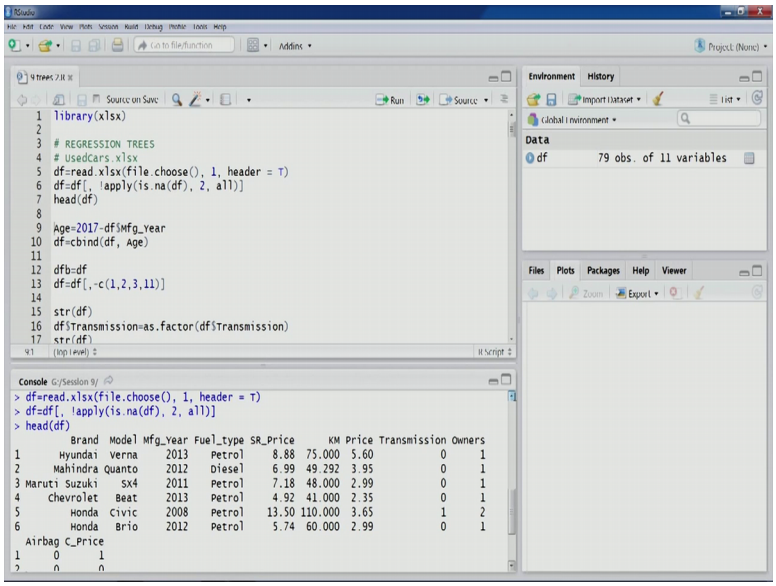
So, lowest impurity would be recorded when all the observations in a particular leaf node in a particular leaf node they have you know they have same actual value right. Now before discussing further on (Refer Time: 08:54). So, let us do a regression tree exercise using R. So, let us go back to our R studio environment now this is the code for regression tree the script for regression trees let us load this.

(Refer Slide Time: 09:12)



So, for regression trees we are going to use the chars data set that we have used in previous technique some of devious techniques as well. So, let us import this particular data set you can see 79 observations of 11 variables. Let us to move any columns and let us look at first 6 observations.

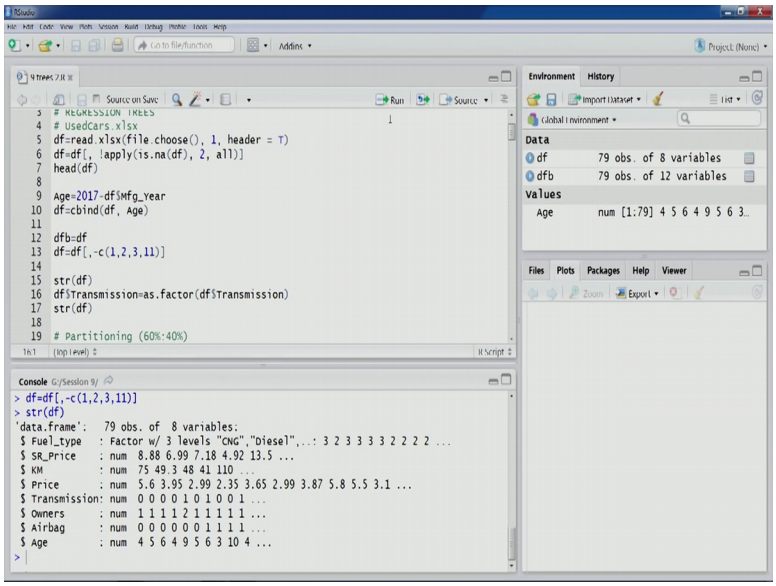
(Refer Slide Time: 09:38)



So, you are already familiar with this particular data set you can see brand model manufacturing year, fuel type S R price that is showroom price, K M price, transmissions owners air bags and c price.

So, as we have been doing for this particular data set and for previous techniques as well let us compute this age variable out of manufacturing here. So, this is age variable let us add this to the data frame. Now let us take up backup of the data frame and which since we are not interested in first few columns and also see underscore price. So, we will get rid of them.

(Refer Slide Time: 10:17)

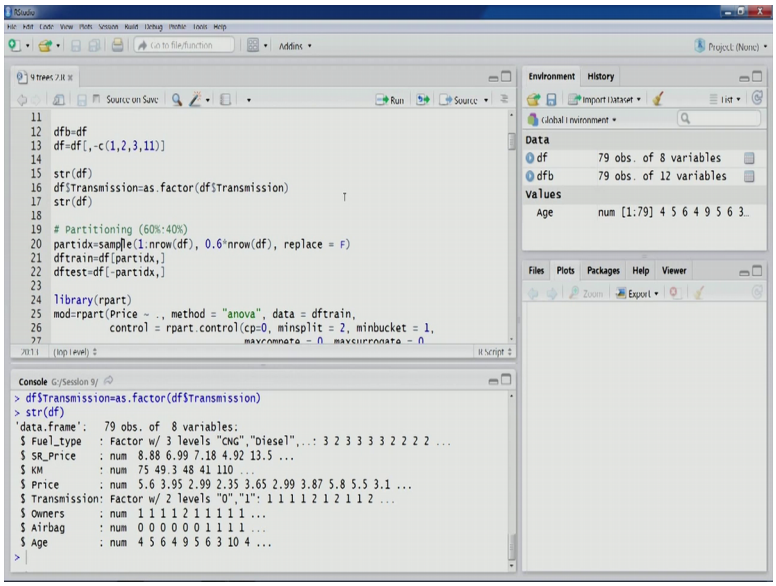


So now, what we will have is these many variables so just 8 variables, 79 observation, so, you can see fuel type is appropriately specified as factor variable 3 levels right c n g diesel and petrol. The others S R price, K M price then we have transmission this should actually be a factor variable because in, but since it is a numeric code format.

So, therefore, we will have to change it to a factor variable as we know that any variable and if categorical variable if it is having values in text format like c n g, diesel, petrol. So, it will automatically be stored as factor variable in R environment now for transmission we will have to do it and other variables are fine owners airway airbag and age they are numeric variable.

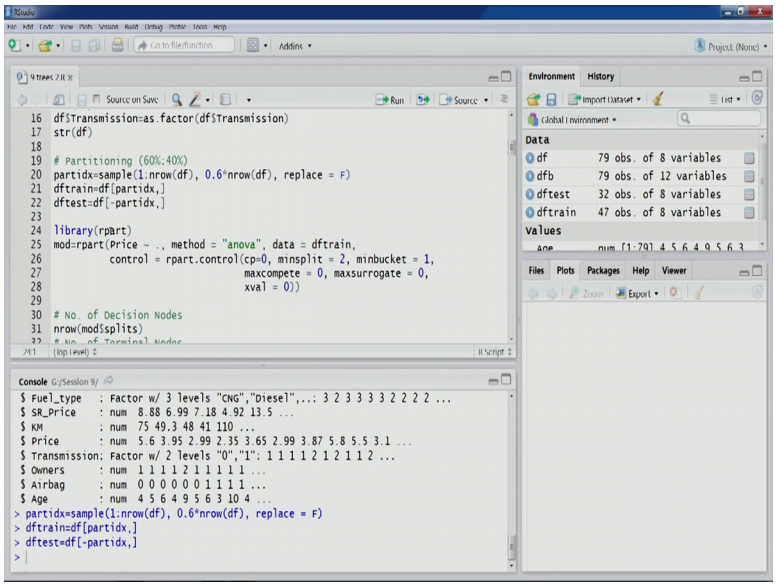
So, let us just change this one transmission as dot factor and you would see in the new output our structure function transmission is also changed a factor with two levels. Now let us perform the partitioning.

(Refer Slide Time: 11:25)



So, we will go for 60 percent, 40 percent, 60 percent for training partition and 40 percent for test partition. So, let us create these two partitions here once this is done we can go ahead and build our model.

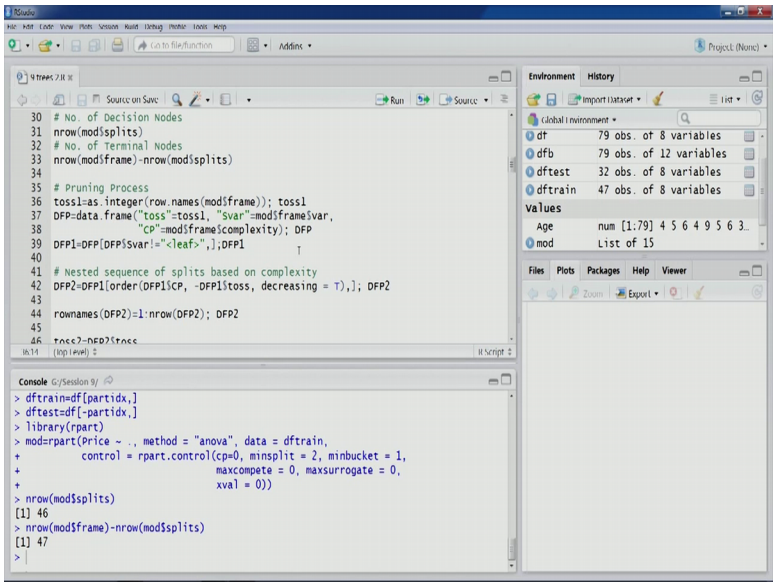
(Refer Slide Time: 11:42)



So, again the same package same function that we are going to use r part. So, let us load this particular library this particular package. Now, if we look at the r part function the arguments are quite similar to what we have used in classification tree exercises, but one difference now method has changed. So, method has changed to ANOVA.

So, for regression trees we have to specify method as ANOVA and for classification trees the method was class now other things are quite similar right. So, other things do not change you can see x well value is specified as 0, by default is it is 10; 10 as we have talked about in previous lectures. Because we want to first develop the will the full grown tree and therefore, we would like to have x well value as 0, and also c p value at 0 because we would like to for the same region we would like to build the full grown tree.

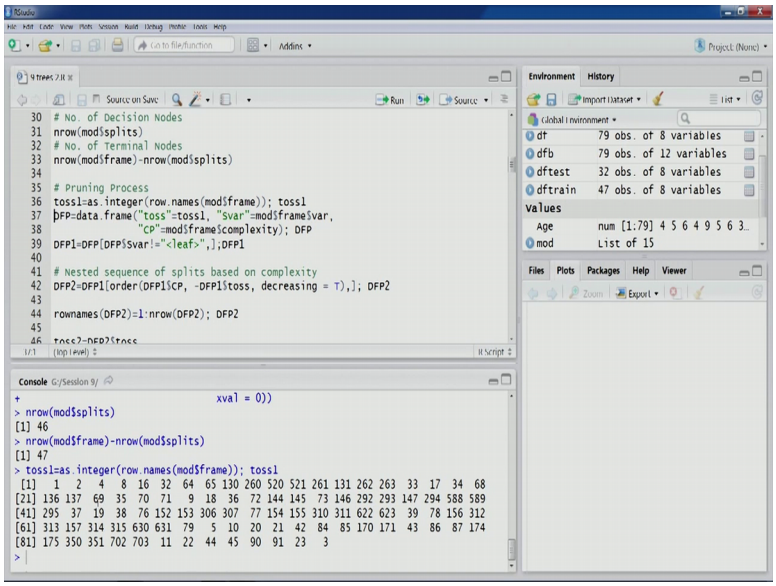
(Refer Slide Time: 12:50)



So, let us run this and the model is built now let us look at the number of decision nodes 46 in this case number of terminal nodes 47 and so these are the number of nodes in the full grown tree model. Now what we will do we will straight away we will move to the a pruning process the pruning process as we talked outward is of course, similar to what we did in classification trees.

So, first we will record the node numbering so as we have talked about that node numbering is in a particular order in r and that you can understand from the frame attribute of r part object so, let us compute this.

(Refer Slide Time: 13:30)

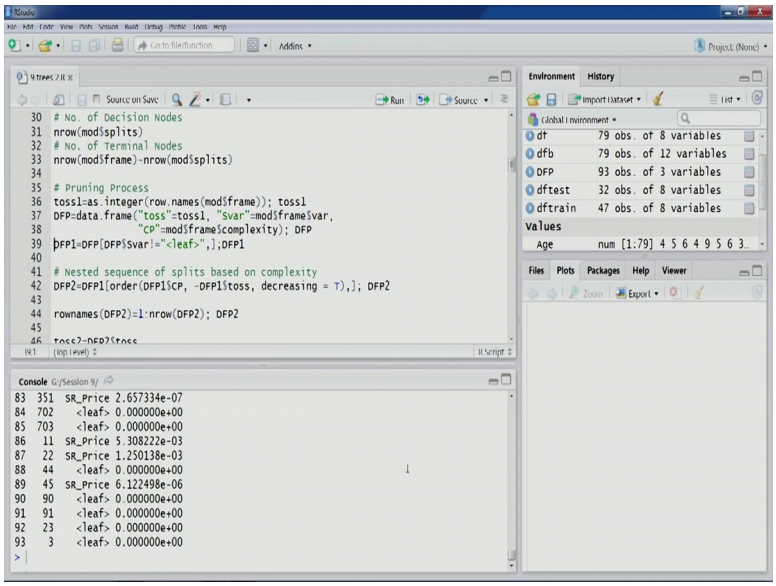


So, these are the node number as you can see in as we have explained in previous lectures as well. So, node numbering in R environment typically happens in this fashion let us this is let suppose this is our tree. So, node numbering typically happens in this fashion 1 and it will go like this 2 and this fashion and then again it will move back once that tree full level is achieved it will move back and move back in this fashion the node numbering is recorded for a particular tree. So, as you can see 1, 2, 4, 8.

So, row numbers 1, 2, 4 it will be in the left part of the tree. So, 1, 2, 4, 8, 16, 32, 64 will keep on going there, but 64 it ends there then we will have to go back to 65 the nearest you know node and then 65 then we again go down 130 and in that fashion then we go to 260 to 520, 521 then again we will have to go back so in this fashion the node numbering happens as we saw in the here in the board.

So, as we have as we have done in classification trees we will create this data frame. So, first column would record these node numbers then we will record the variables that have been used for a split and then we will record the complexity for all those all these nodes.

(Refer Slide Time: 15:00)



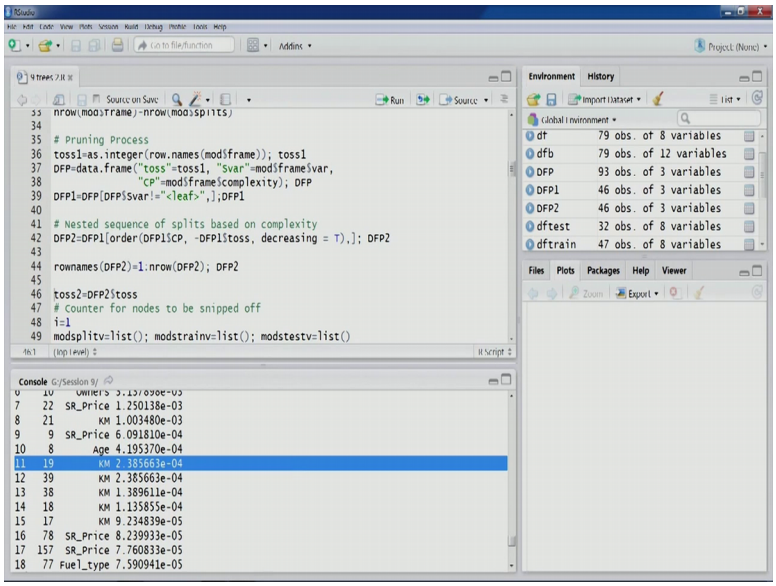
So, you can see 93 total number of nodes in the tree are 93 and the same number of node rows are present in this particular data frame. So, as we have done in case of classification trees we will first get rid of the leaf nodes. So, the code is quite similar actually same.

So, let us get rid of these nodes and you would see in the D F P 1 we have just 46 observations which are nothing which are same as the number of decision nodes that we had that we have in the tree you can see 46 were the number of decision nodes and the same number of observation you can see here in D F P 1.

So, now in D F P 1 we have just the decision nodes leave leaf nodes have been those rows have been removed. Now next nested sequence office splits based on complexity so as we did finally, for the classification trees and that first ordering would be based on complexity values and then we will like to further order the node numbers. If a particular number of nodes are having same value same complexity value than for those group of you know those group of rows we would like to order them based on their node numbering.

So, let us execute this and we will get the sequence let us also change the row names you can see 46 are the total number of rows now as you can see so 46 decision nodes. Now these if you look at the complexity value this has been sorted so, this is decreasing order.

(Refer Slide Time: 17:00)

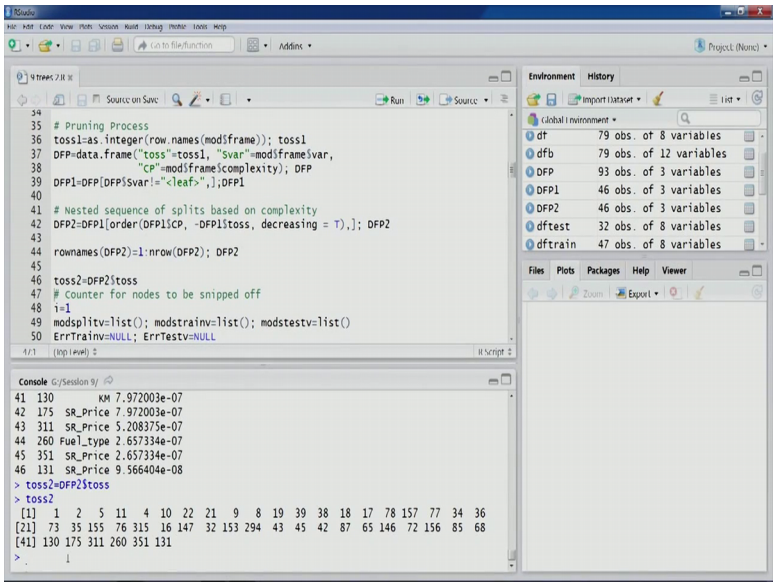


So, higher complexity value first in the root node then root node number 2, and number 5, 11 and then four so you would see in this fashion then as we move down we can again even for regression tree we can see that some of the rows are having the same complexity values you can see here this is same complexity value for 19 and 39 right. So, there again as we have done for classification trees we would first like to remove 39 and then 19. So, therefore, it is at it was sorted same for classification trees, same for regression trees

Now, we can find out similar types of groups of rows here which have the same complexity where here once again you can see same complexity value. So, the sequence has to be based on node numbers because we would like to have a smaller sub tree. So, the nodes with higher number would be should be pruned first.

Similarly as we go down we can find many more nodes having same complexity value. So, therefore ordering has to be accordingly pruning has to be accordingly here also you can find that so everything seems to be appropriately ordered here now once this is done. Now we can record another task to argument which will have the desired sequence of nodes to be pruned off so we can have a look at the toss two again. So, this is probably the this is this the desired sequence of nodes to be pruned off.

(Refer Slide Time: 18:35)



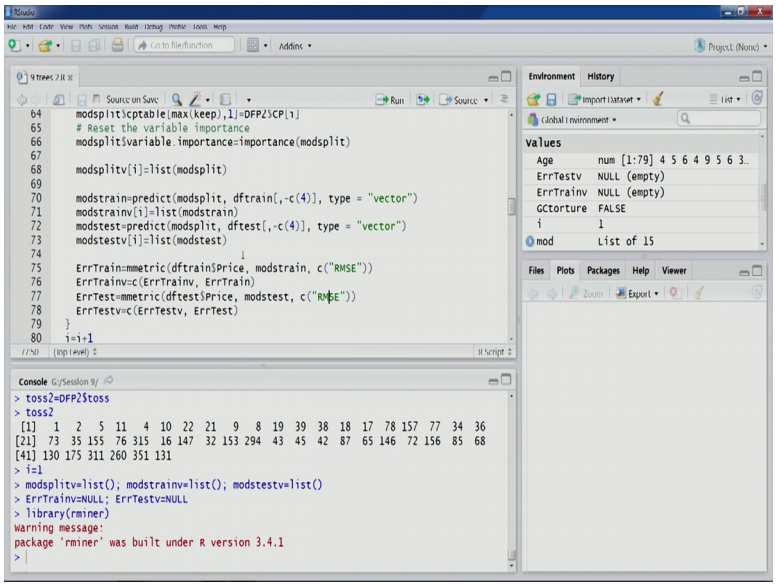
So, you can see the pruning will start from the last node that is this one 131 this would be pruned first then 351 260 and then 311. So, in this fashion pruning good pruning or full grown tree would have. However, in the code that we are going to run we will build all such models.

So, all models with you know no decision nodes with one node that is this one first one then 2 nodes that is first and second with 3 node there is 1, 2 and 5 then 4 node that is 1 and 1 2 2, 5, 11. So, in this fashion we will keep one building our model. So, let us start executing our code for this for the same.

So, let scene is like i then you can see these 3 variables which are being used to either store the record the different models or to record the scoring of what is scoring for different partitions. So, let us initialize them and because this is our regression tree. So, the metric that we are going to use so earlier in the classification trees we were computing the misclassification error and we looked at the misclassification error to identify to find out the minimum error tree and also best prune tree.

In this case we will look at we will use the r m s e value. So, we are going to use this particular package R minor. So, let us load this.

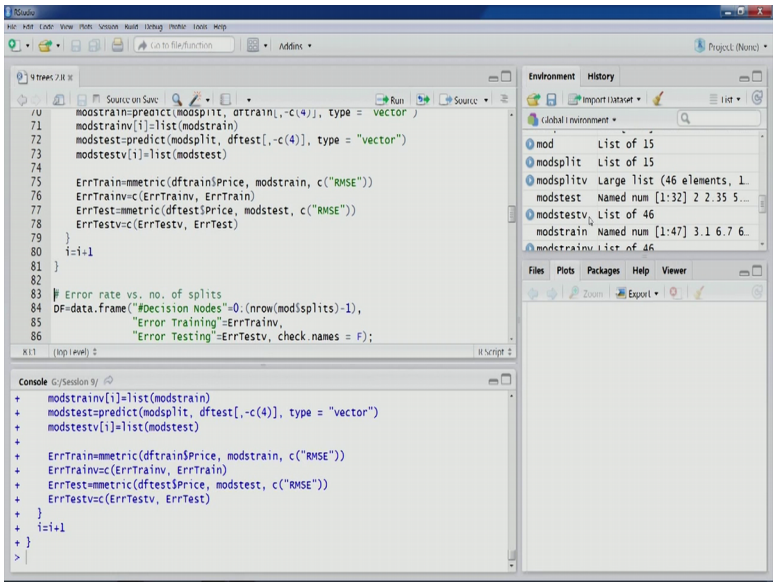
(Refer Slide Time: 20:07)



So, that will be computing the r m s e value within the loop for each model and then that is going to be used for finding the minimum and best prune tree the code is quite similar to what we had for the classification tree with one change. Let us so, you can see a predict function now you would see type is vector because we would like to predict the values right instead of plus and in computing the error for different partition and for different models can see r m s e value is being computed right. So, these are few noticeable changes in regression trees in comparison to what we did in classification trees.

So, let us run this loop, but before running this let us load the importance function and that is part of this code this function let us so once this is loaded we can go back to the loop and run it.

(Refer Slide Time: 21:12)



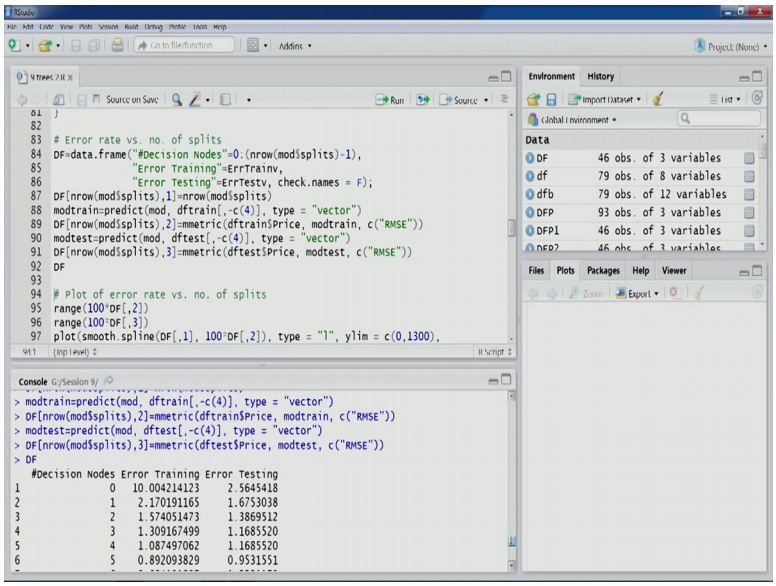
So, you can see here in the environment section you can see model split v large list for 3 6 elements, we had 46 decision nodes. So, the same number of models have been developed right and scoring also test partition and training partition also. So, depending on the number of observations so you can see list of a 46 here and list of 46 up all 46 models we have the scoring right.

So, once this is done as we did from the classification trees in the last you know model that we developed in classification trees will get this data frame having a number of decision nodes in the same fashion as we did last time for the classification trees and trees and then the error train v and a error test v. So, let us say create this data frame

Now, this will have the all rows starting from 0 decision node to the last one decision node and for that we will have to for to include all model with all decision nodes will have to bring the full grown tree information details. So, that we are doing now.

So, last row we are getting manually you can see here that we are scoring off the training partition using the full grown tree model and then that is added to the data frame and then we are scoring the test partition and we are adding this information to the data frame. Now we have the data frame now you can see we had 46 decision tree, and we have 47 models and their performance.

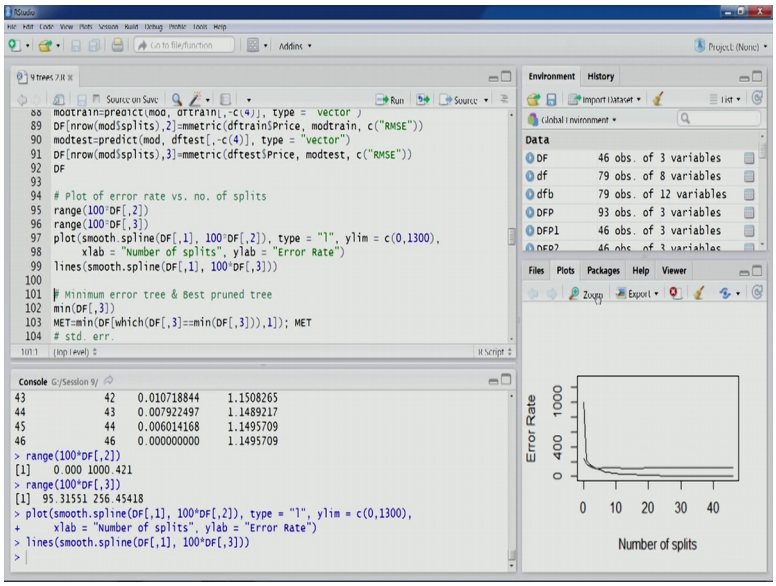
(Refer Slide Time: 23:05)



So, first model is model with 0 decision nodes that is just rude node acting as a terminal node and the corresponding error values also you can we can see then as we move down we can see that models with different number of decision nodes. Now for the training partitioned you can see training error it is highest when we have 0 decision node and it keeps on decreasing and till when we have the 46 decision nodes it reaches 0 right.

So, this are finally, decreases to 0 for the testing partition you can see that it starts from this particular value 2.56, that keeps on decreasing and it decreases I think to this value this seems to be the minimum value in this particular row and then again it starts increasing right.

(Refer Slide Time: 24:17)

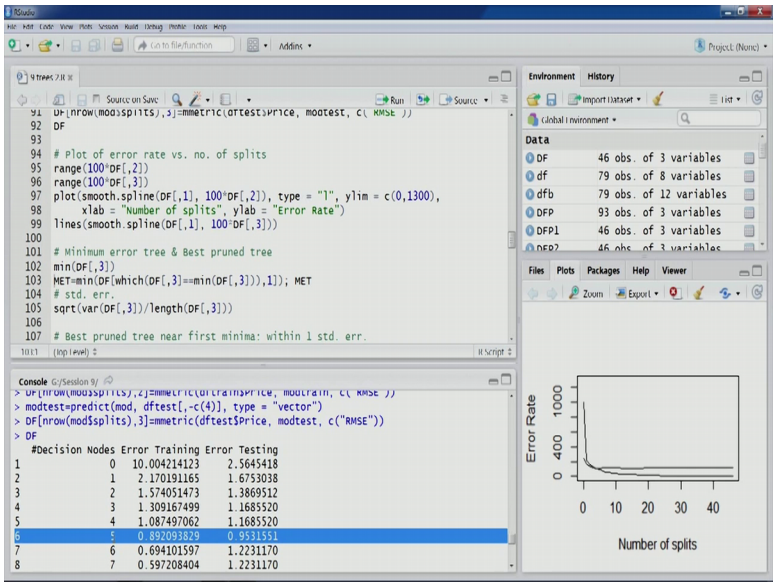


So, once we have this particular table this particular data frame now let us go ahead and plot the curve for error rate for these two partition training and the test 1000 and here 256 so within 0 to one thousand we will have these two plots.

Now, we would see some differences with respect to regression tree so you can see that training partition it starts with a quite high error then it keeps on decreasing until it becomes 0 for the validation partition again it comes down to a minimum level and then it starts increasing right.

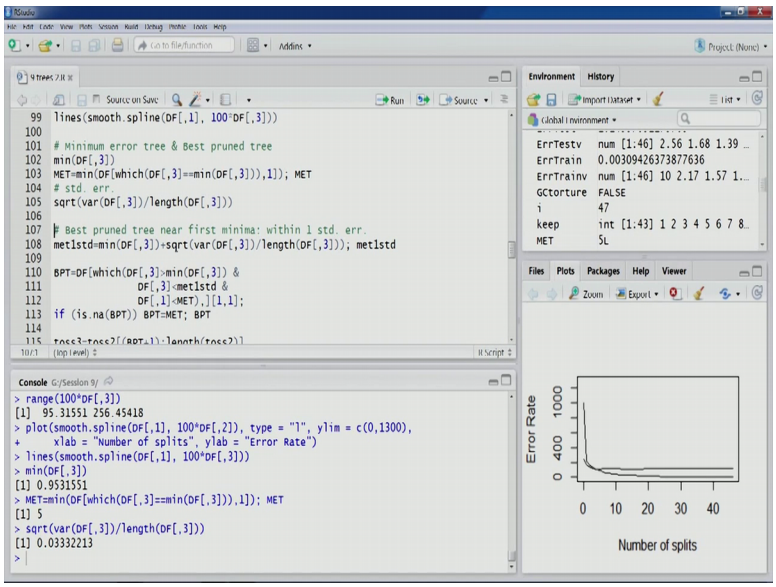
So, so from this we need to find out the point where the error on r m s e value is minimum for the validation partition. So, so this can be done using this particular code I which be used for class field entry as well let us find out the minimum error on validation partition using this column third column. So, this is the value the same that we identified in the label in the table right.

(Refer Slide Time: 25:18)



So, this is the value you can see number of decision or corresponding no originals are 5. So, with 5 decision node we get this minimum value right same thing would, should come in this code 5 you can see that has been recorded. Look at the let us look at the standard error. So, this is the standard error to find out the best prune tree within this within one standard

(Refer Slide Time: 25:42)

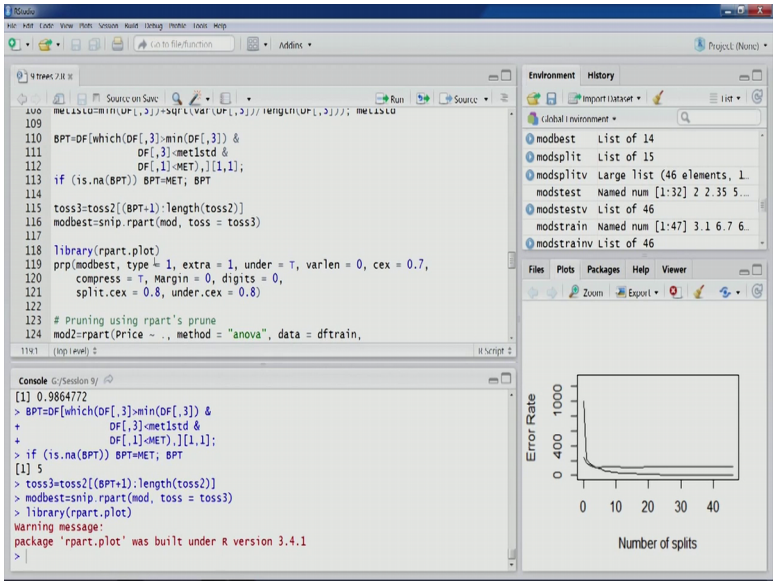


So, we will let us compute this particular value. So, the error should be less than point nine eight, but as we have seen in the table let us look at the go back to the table you can

see my 0.95 and as we go upwards in the table we do not see any any row where the value is within that 0.98 you know that less than that so within one standard deviation of this particular value 0.9531.

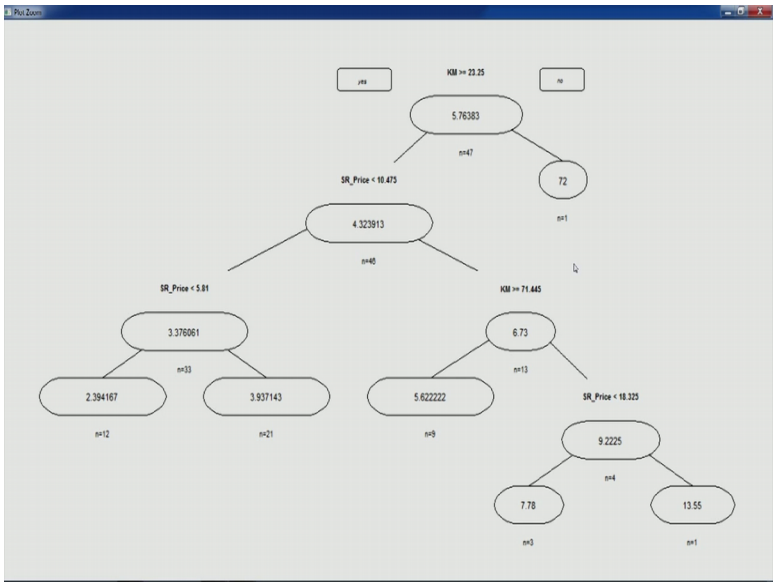
So, therefore, it seems so that best prune tree would also be same as minimum error tree. Let us find out by this code using this code let us run this you can see its 5. So, the best prune tree is also the same as minimum error tree. So, what we will do create the toss argument again so in this case we will like to remove we like to give just the five decision trees in the model that is prune tree model and snip of all the remaining nodes. So, let us create this we will get this particular and let us load this library r part dot plot. So, now we can plot this tree.

(Refer Slide Time: 27:06)



So, this is the tree that we have out of this regression tree exercise.

(Refer Slide Time: 27:16)



So, if we look at this tree the first root node there the first spilt predictor and split value combination is comes from K M so K M greater than 23.5 right. So, if yes then this side so all the observation with that a greater than this value they will come this side and then we get immediately we get the terminal node on the right side right child and fill then in this side and we have S R price and within S R price.

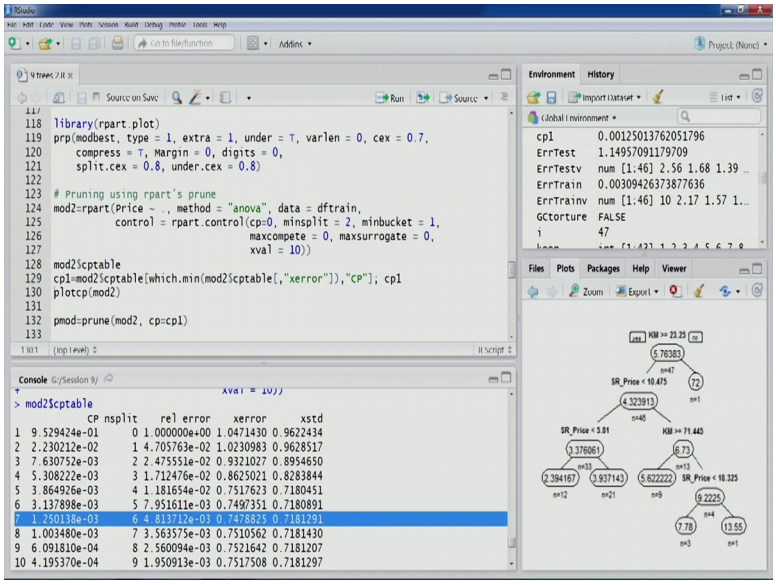
So, this should be less than 10.475 then of course, we go this side again we have S R price. So, S R price kilometre is seems to be the important topmost you know most important variable and then followed by S R price which comes at second level then again we see kilometre. So, then again we see S R price so you can see that we have 5 decision nodes and out of these five decision nodes S R price is occurring thrice and the K M is occurring twice.

So, K M and S R price out of all the variables that we had K M and S R price are eventually determining the this particular tree model. So, K M and S R price seems to be the most important variables for this prediction task as you can see at each terminal node as well and we have some value right. So, this value once for a new observation if you would like to classify it the observation has to be drop down from this particular tree and once it reaches to a particular leaf node these nodes last in our nodes leaf nodes so that the value of these leaf nodes is going to be the predicted value.

So, let us come back now the same exercise that we followed through you know by minimizing error on validation partition and for different more models with different number of decision nodes you know we can have alternative mechanism which is quite similar using R part prune function.

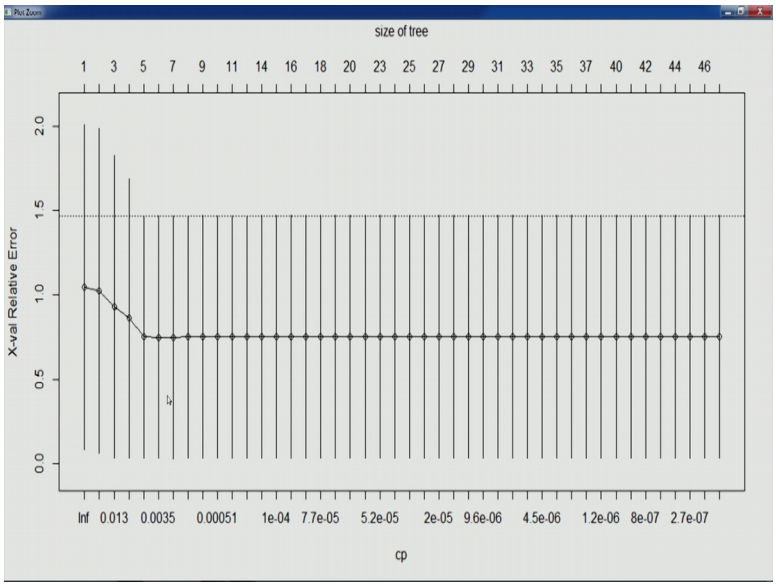
So, what we will do x value is now 10 as you can see and we will build this model once this is done. Let us look at the c p table. So, this is the t c p level that is part of this output. So, here we need to identify the row where x error is minimum right so that we can do using this particular code.

(Refer Slide Time: 30:15)



So, corresponding c p value would be recorded 0.01250 we look at the table we look at the table so I think this is the error that is being minimum value you can see from the value itself (Refer Time: 30:17) minimum value is 0.7478825 and then corresponding c p value we can also see. Now this value can be used to prune the tree. So, you can see number of splits 6 right so this can be used now we can also plot c p for this model and based on this as we did for classification tree.

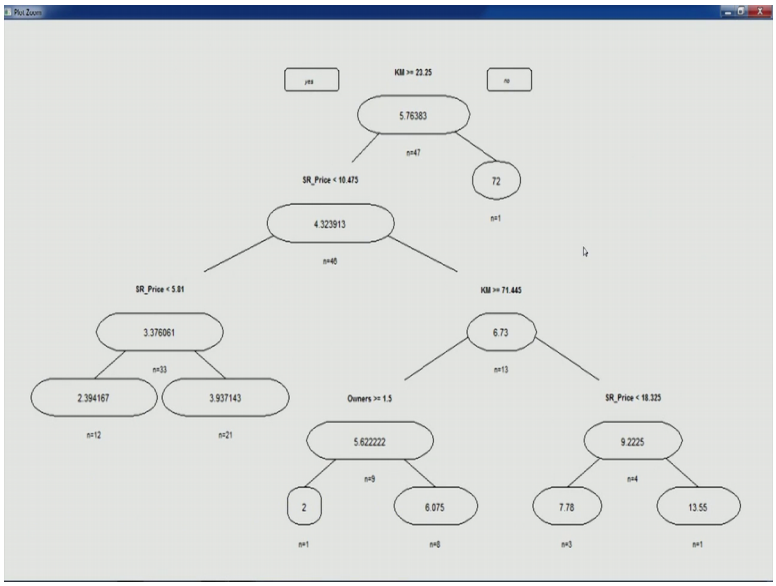
(Refer Slide Time: 30:46)



As we talked about that a similar approach wherein the first tree which is below this vertical line you know that can be used as the best prune tree; however, all the points are below this particular line. So, the model is not performing as well on the validation partition. One specific reason for the same is that we have just seventy nine observations in the full in the total full data set and out of that few are being used for the training partition and remaining what test partition. So, because of this smaller sample size this kind of result we are getting there.

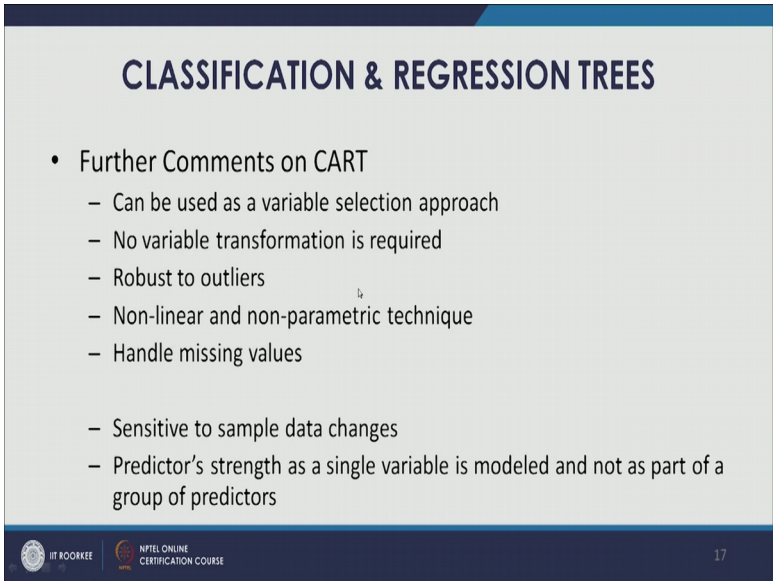
Now, even within this as we can see you know probably this particular point which is corresponding to 6 nodes can be the can be the prune tree as per this particular plot. So, we will use the value recorded in c p 1 and that is corresponding the minimum x error value to build a model and this is the model that we get.

(Refer Slide Time: 31:57)



If we look at the this particular model you would see that this is there is one more node in comparison to what we saw in our exercise owners is also there, but otherwise we can see that K M S R price this seem to be two important variables and then one extra node owners is also there. So, this is from the, this is this particular model is from using the by using be r parts prune. Now we can look at the number of node is a node 6 and 7. So, with this we completed our exercise in r for this.

(Refer Slide Time: 32:37)



Now let us discuss a few more comments a few more important things related to cart algorithm. So, some of the advantages of cart algorithm are that can be used as a variable selection approach no variable transformation is required you do not need to transform your variable and key derive new variables. Because the way tree is build using partitioning approach that recursive partitioning approach eliminates any requirement for variable transformation because the tree is essentially going to be the same because it is the midpoints values and the that are used or subsets in case of categorical variables that are used to create partition.

So, therefore, variable transformation in r require robust to outliers. So, because again recursive partitioning approach that does not rely on the specific values of outliers therefore, the model is going to remain too robust to outliers non-linear and nonparametric techniques. So, we did not make any assumption about the relationship between outcome variable and set of predictors right.

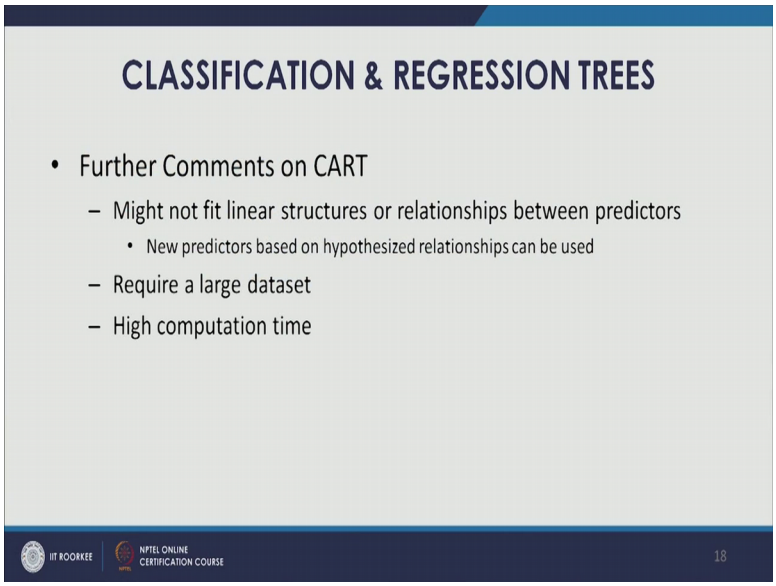
So, this is non-linear no parameter that we have used handle missing values. So, again for the same region because of the recursive partitioning approach and because we look to use mid values to find out the possible split points so, missing values can also be very well handled using this worker technique.

Now let us look at some of the problems of cart some of the disadvantages or issues with cart algorithm sensitive to sample data changes as we have seen in our classification tree exercises and regression tree exercise if the in the regression tree we had very small sample very small sample and we saw that the performance on validation partition was not good and in case of a classification trees we had large enough sample size; however, every time we use to run because of the different observation that become part of the training partition.

The tree model used to change full grown tree model used to change not even best prune tree a full grown full grown tree model used to change right and therefore, this particular technique is sensitive to sample data changes. Now if we look at the approach of cart you would see that this main approaches recursive partitioning and then pruning. So, a recursive partitioning approach in a way it captures the predictors strength as a single variable and that is actually model and not as part of a group of predictors.

So, modelling is does not consider the strength of a group of a set operators rather it relies on the predictor strength as a single variable right so that is that could be one drawback right. So, there could be some set of predictors which put together might give better performance using other techniques right so relies on strength of single variable.

(Refer Slide Time: 35:42)



Other comments on cart might not fit linear structures or relationships between predictors right so that is a one problem. So, if the typically it is understood that cart procedures if the partition because the recursive partitioning approach is used if the vertical and horizontal separation kind of scenario exists in a particular data set then the cart algorithm is to perform well.

However, if the partitioning is you know some diagonal line kind of a partition would be more suitable for the data then probably this technique is not going to give better results. So, in that case solution could be we can derive new predictors which actually you know which are actually based on the hypothesize relationship. If some diagonal line is better separator for observation then probably new variable can we derived to you know to express the same and that can be used when the cart models.

So, another problem another problem with cart algorithm is it requires large data set ah. So, robustness depends a lot on large data set and even then every run the changes could be there high computation time. So, because we are using high large data set recursive partitioning is there and pruning is there so much of sorting related exercises are to be a

part of this process. So, because of this it requires high computation time. So, with this we conclude our discussion on classification and regression trees in the next lecture we will start our discussion on logistic regression.

Thank you.