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

Lecture - 57
Artificial Neural Network-Part V

Welcome to the course Business Analytics and Data Mining Modeling Using R. So, in previous few lectures we have been discussing Artificial Neural Networks. Specifically in the previous lecture we were doing an exercise in R using our used cars data set and there we discussed different steps that we executed related to a variable transformation and normalization that were required and then the formula and so will again restart that exercise and we will do our modelling and discuss some of the important issues that we faced in neural network.

So, before going into R studio let us discuss few important issues in modeling exercise. So, one of the in one of the key issues that we typically a face in neural network modeling is over fitting; it is more likely that model will over fit the data in a neural network modeling scenario. So, how do we overcome this situation? So, typically error on a validation and test partition would be large in comparison to training partition. So, first we need to detect whether a neural network is over fitting.

So, typically when we build a when we train a neural network model then we can check the performance on a training partition itself and then performance on validation and test partition and we would see that error is quite you know quite small in comparison a quite small for training partition in comparison to that off in a validation and testing partition.

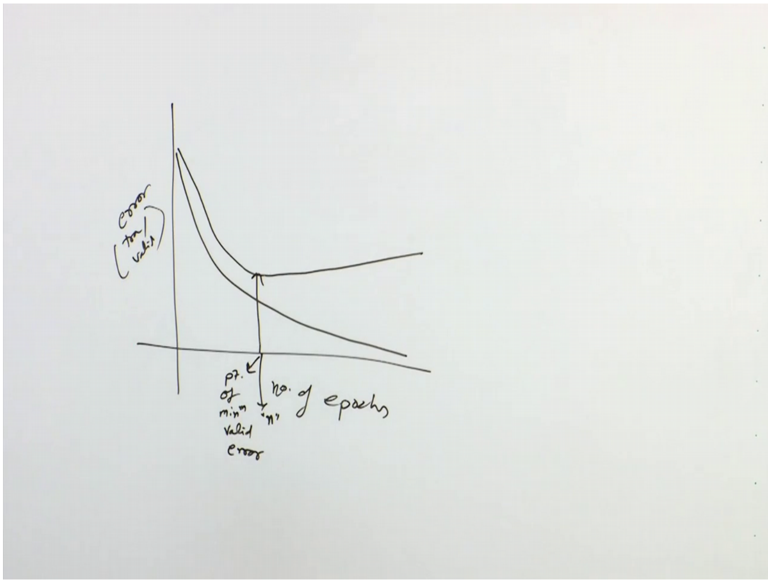
So, that is how will know that probably the neural network is over fitting to the data. Therefore, we need to limit the training or learning process of neural network. So, that this over fitting could be avoided, because as we have been talking about the objective in data mining modeling in prediction classification and other types of tasks, other types of data mining task. The objective is that our model should be performing well in new data.

So, a few things that can be tried out is one limit the number of epochs. So, number of times and number of sieves through the data; that we have to do in neural network learning process, training process that can be limited. So, that network you know, so that

that was just fits the key information that is there in predictors and not to start fitting to the noise.

Now, another approach could be a plot of validation error versus number of epochs that could be used to find out the best number of epochs, so for training. So, we can always look for point of a minimum validation error; something similar that we have been doing in previous few techniques as well where in we used to it create this kind of plot. So, in this case it is going to be number of epochs on x axis and error it could be you know for training and validation. So typically this kind of plot we have been you know generating in other techniques also.

(Refer Slide Time: 03:21)



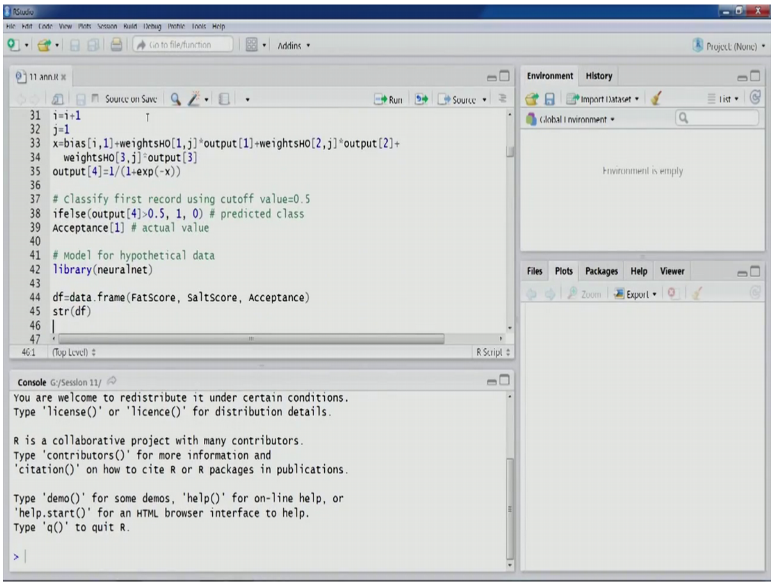
So, typically for any data mining technique typically the training error will plot curve per training error will go like this, and the as we keep on as we keep on training our model the error will keep on decreasing and it will reach to 0. However, for new data that is validation partitions the error will keep on decreasing, but the at one time at one point it would be minimum and then again it will start increasing. So, this is the point that we are looking for.

So, this is the minimum validation error point; point of minimum validation error that we are looking for. So, we would like to stop the learning process of our neural network at this point. So, we would like to stop here and this particular, so the network which I have been trade which have been trained up to this point these many number of epochs right.

So, if this is n, these many number of epochs that would probably perform better on new data.

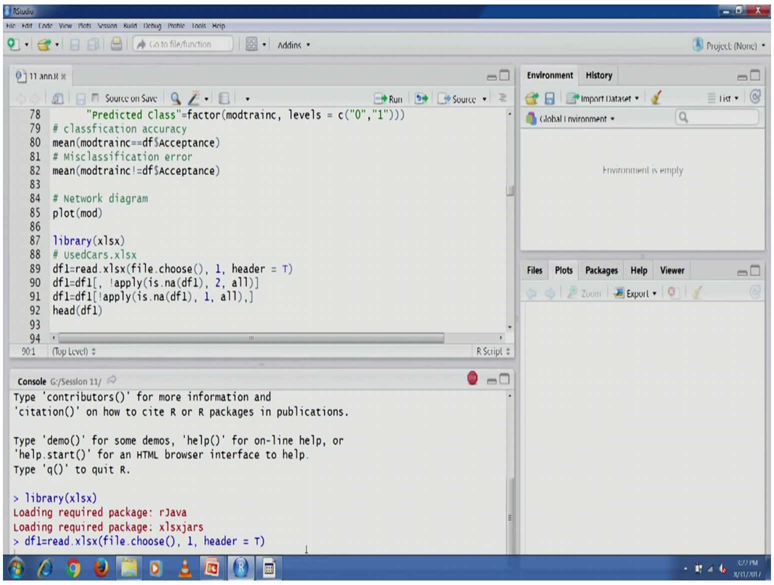
So, however, of course, this you know this criteria whether we have to take number of epochs here or some other parameter related to training process or learning process of neural network would actually depend on the implementation of neural network in the software. So, we will see what kind of plot we require to find out this point of minimum validation error in our case a using R. So, let us go back to R studio environment; so the data set that we were using in that in the previous lecture.

(Refer Slide Time: 05:32)



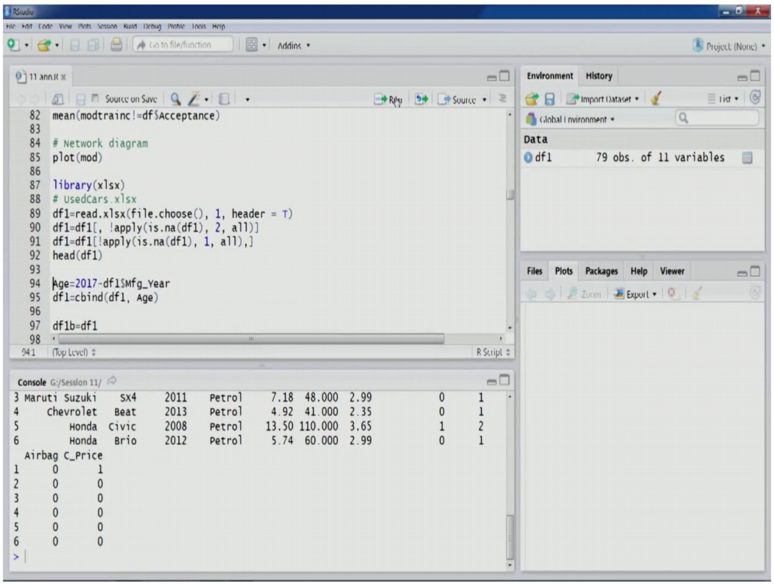
Used cars the last exercise that we were doing so let us, so let us load this library xlsx. So, let us import this data set used cars.

(Refer Slide Time: 05:44)



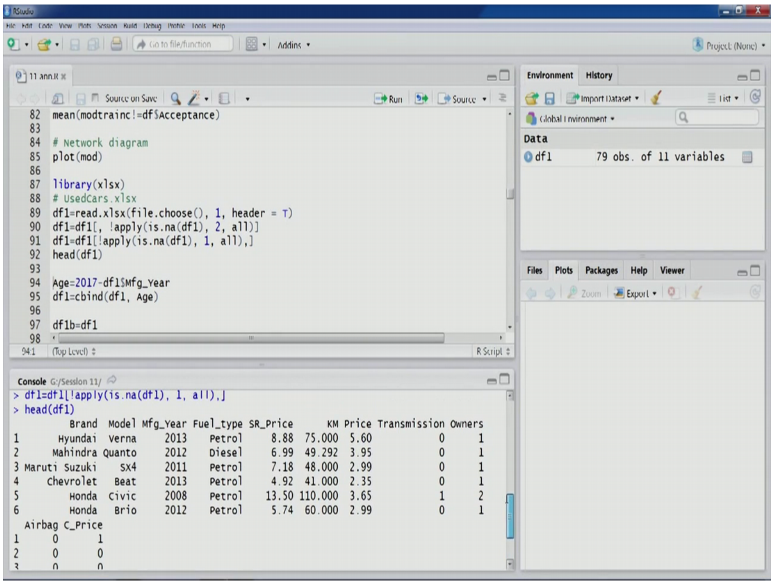
So, small data set about the used cars. So the task is to as you can see environment section 79 observations, 11 variables. So, the task is to a build a model for predicting the used cars price. Let us move na columns, na rows.

(Refer Slide Time: 06:16)



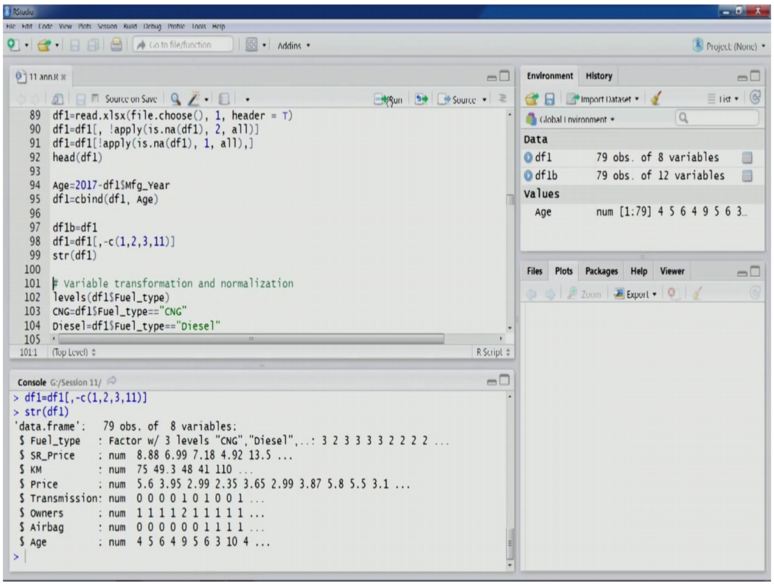
So, we are already familiar with this data set these are the variables as used in the previous lecture as well.

(Refer Slide Time: 06:19)



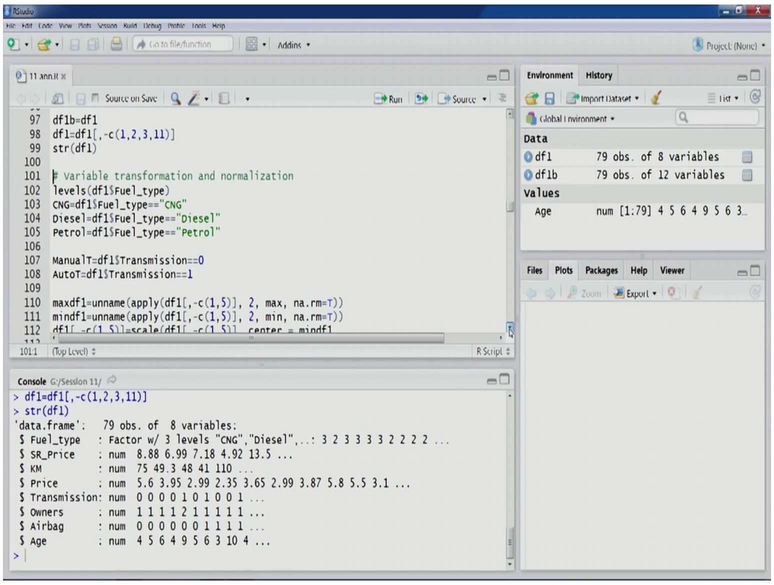
Let us create a is added to the data frame; let us take a backup, will exclude the variables that we do not want to you know take forward for our modelling. So, these are the variables now.

(Refer Slide Time: 06:34)



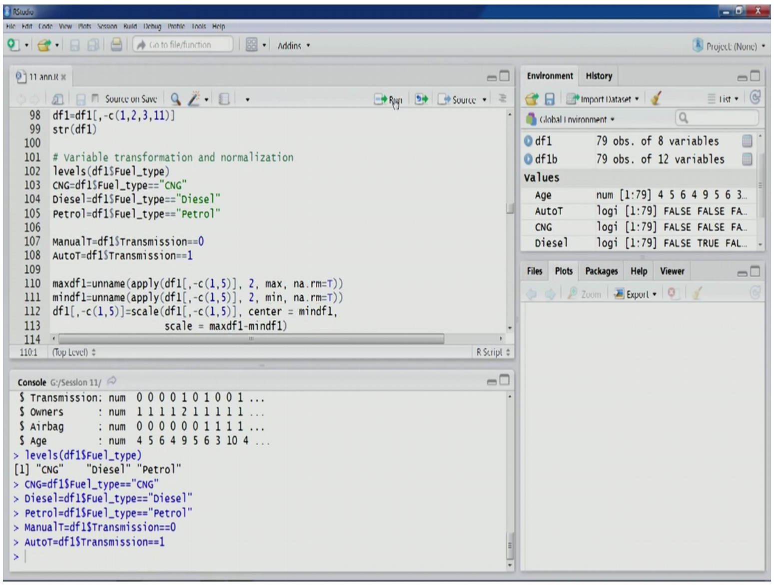
Now, as we did in previous lecture, let us go through a the transformation process.

(Refer Slide Time: 06:39)



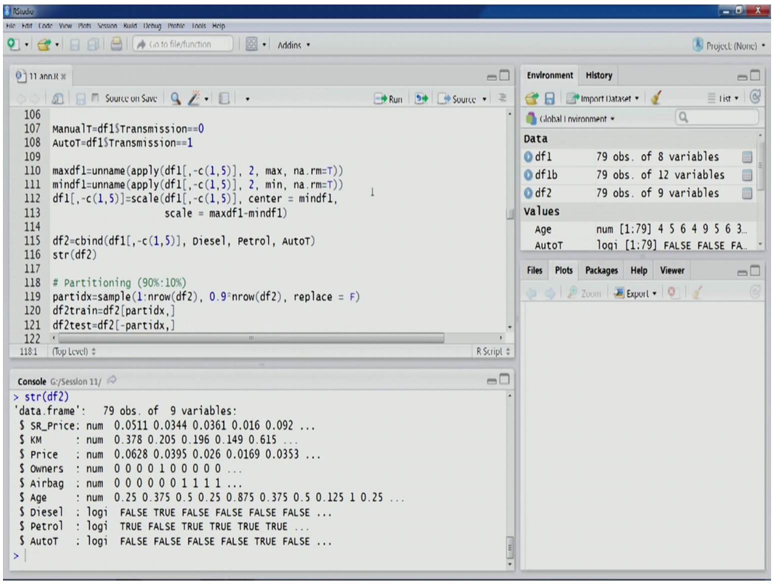
So, this part we have discussed before in previous lecture. So, we will just quickly go through this, a scaling of the numeric variables.

(Refer Slide Time: 06:49)



And then we will create a data frame that would finally, be used for our modeling exercise.

(Refer Slide Time: 07:01)

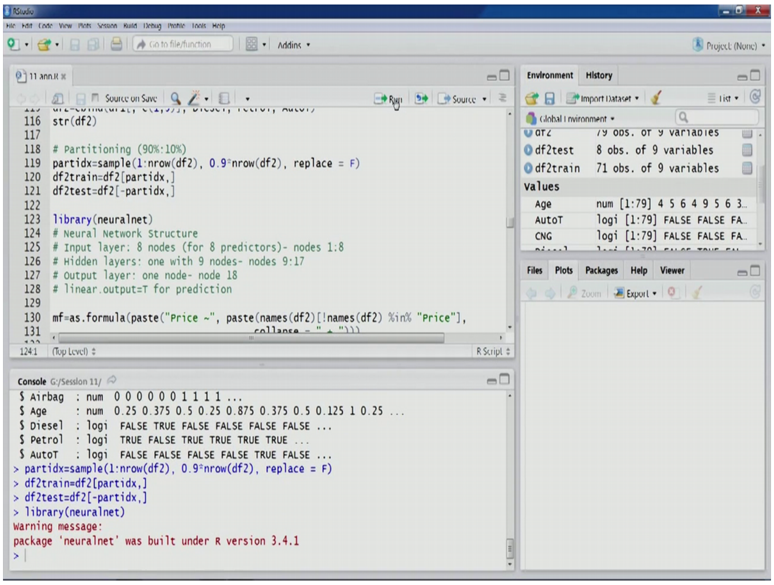


So, this is the data frame df 2 and you would see these are the variables price is our outcome variable and others are predictors and you would see all the values for all the variables they are in 0 to 1 scale. Now we will do our portioning, so 90 percent and 10 percent, 90 percent for training partition because this is small data set. So, we would like to use you know higher percentage of observation for our training partition.

So, let us do our partitioning. So, only 10 percent of the observation that is about 8 observation would be there in test partition and the remaining 71 are going to be used in the training partition.

Now, as we have discussed neural net is the package that we require that we have been using for our neural network model. So, it has been loaded.

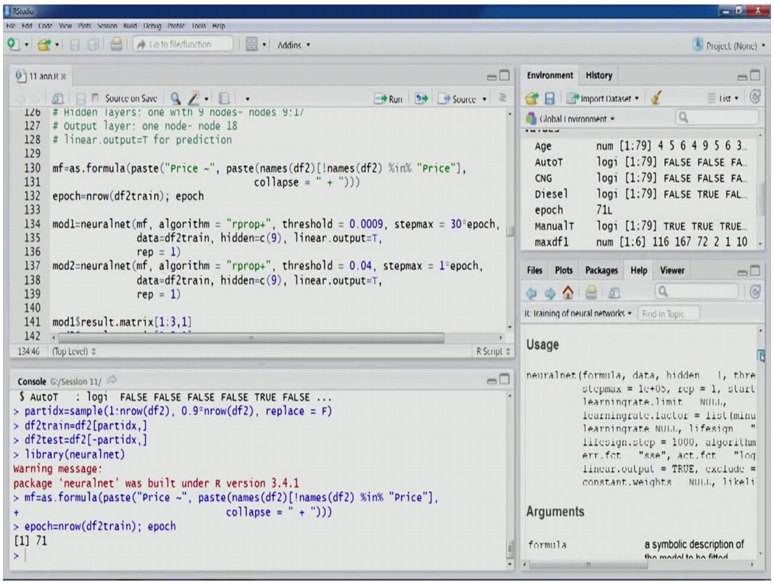
(Refer Slide Time: 07:49)



We have already discussed the neural network structure that we would be following for this particular exercise.

So, we talked about that there are 9 variables, so one being the outcome variable. So, we will have 8 predictors effectively. So, therefore, 8 nodes in the input layer and then the you know will take typically we take one hidden layer and one more you know we can always do experimentation with the number of, you know hidden layers and also number of nodes that are there. So, we will for our illustration purpose we will take just 9 nodes a one more than the number of predictors here and the output layer just one node that is for the our outcome variable.

(Refer Slide Time: 08:25)

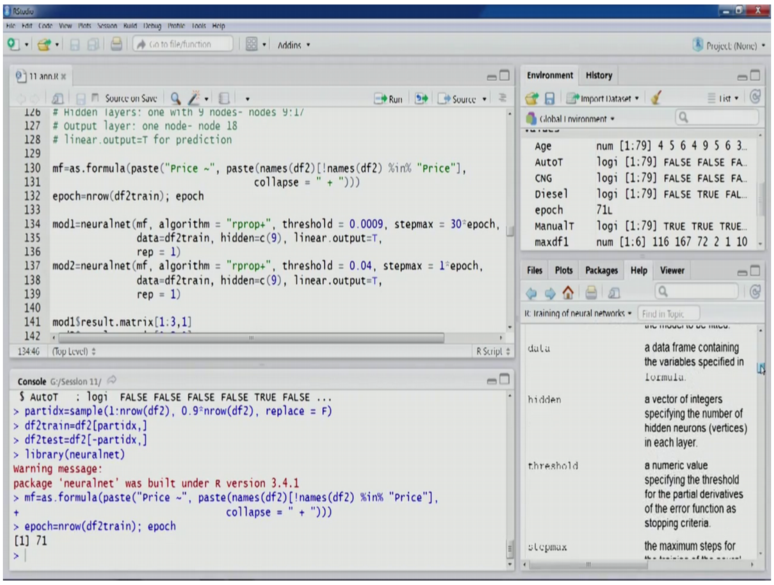


Now, this particular argument linear dot output has to be true for a prediction model. So, let us create the formula. So, you can you can use as dot formula function and here the price being the outcome variable. So, all other variable will be collapsed using plus sign and will get the destring of all the predictors. So, this will be our formula let us create this.

Now a number of epochs so as we have discussed this particular thing. So, 1 epoch means you know all the observations iterations of all the observations. So, let us compute this (Refer Time: 09:13) df 2 train certain number of observation training partitions. So, that would be a 1 epoch. So, 1 epoch will have 71 runs through the network, 71 observation that are there in the training partition.

Now, in the model modeling you would see that now there typically we required we are required to do you know you know higher level of experimentation in a neural network modelling, because there are so many things that can be changed; for example number of hidden layers, nodes that are there in hidden layers and a few other things that we will discuss for example, threshold value. So, what is threshold value here? So, quite similar concept, so we have been using in previous technique for example, we look at the neural net network you know function in the help page will get down here and we will see that what is threshold a numeric value is specifying, the threshold for the partial derivatives of the error function as a stopping criteria.

(Refer Slide Time: 09:58)



So, this is the in this the implementation of neural network that we are using here neural net function this the main stopping criteria that is used is threshold value; that is you know rate of change of you know that error given the error function. So, error function typically that is by default that is used is SSE. So, the rate of change in that particular function SSE is going to be used here as a stopping criteria, right.

So for example in this plot when we talk about the number of epochs and this plot so probably instead of the epochs we would like to create a plot error versus you know threshold, because threshold is the stopping criteria that is that is as per the implementation of this neural network function. However, you can see another argument here these stepmax. So, that is also this is also something that we have discussed in previous lecture that the maximum number of (Refer Time: 11:12) steps that would be taken to a train the neural network. So, you can see 30 epochs we are taking, so in typically the implementation neural net implementation the function that we are using.

So, the stepmax is given a bigger number a large number by which the neural network good converge and so typically this is given a large number and this becomes the last result for stop stopping the learning of learning of neural network. However, it should you know if if the neural network is not able to converge even within this number then will get will get some error in this particular appear on this particular function.

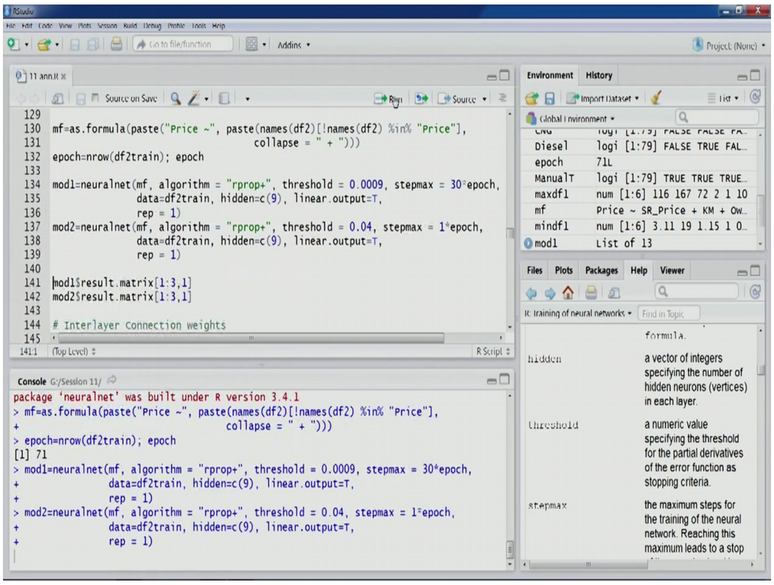
So, therefore stepmax is typically used as just in a large enough number and it is the threshold value which decides the you know convergent. So, of course, if we have a smaller if we have a smaller threshold value we would be requiring a higher much higher value of stepmax because if we you know our neural network model might not converge might not reach the optimum and if we have a higher threshold value we will require less number of less number of steps in stepmax argument. So, this experimentation we can always perform. So, you can see you know two models we have written two lines of code for your calling neural net function twice and two models we have written.

So, in a first one as you can see first is mf that is the formula for our neural network modeling, then algorithm we are we are we are using rprop plus; resilient propagation plus algorithm here and you would see threshold value is quite as small here 0.0009 and stepmax is 30 epochs. So, this threshold value is small threshold value you know is because that we are using a large stepmax. So, of course, we expect that our model would converge and we will get the you know will we can use the model then.

The second code you can see neural net function first two arguments are same, but the threshold value you can see that it is quite high 0.04. So, the model would converge quite quickly and you would see therefore, the stepmax value is also smaller just a 1 epoch. So, you know you know, so when we specify particular threshold value we have to take care that stepmax value is good enough, adequate enough to you know. So, that the model is convert model the conversion takes place.

So, let us run this model, so you can see other arguments are similar data training partition df 2 train number of nodes in the hidden layer just 1 hidden layer 9 nodes. Learning output you can see it you know true that is for prediction and there is another argument left that is 1. So, will see how rep is used in a later in the lecture. So, let us run this code so you will get mod 1 and so, this has this has quickly converged. Now let us look at mod 2.

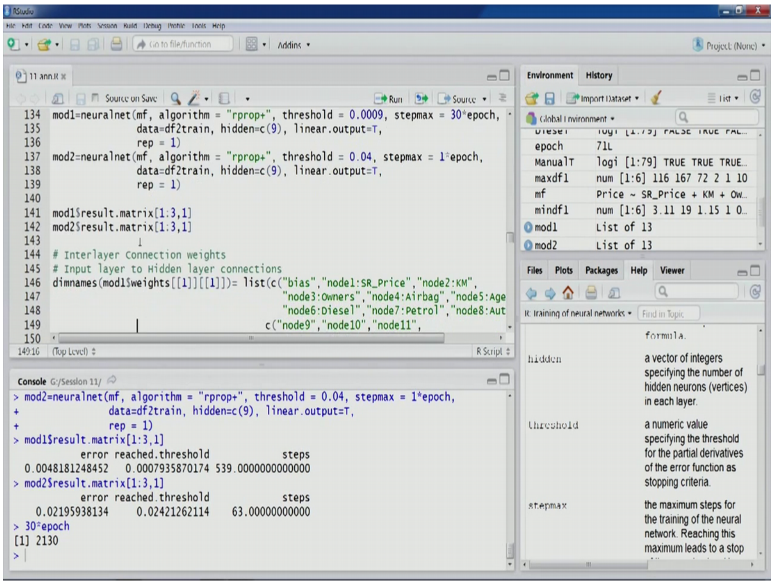
(Refer Slide Time: 14:34)



So, this has also converged. Let us look at the errors of these two models you can see that mod 1 the error is smaller than a mod 2 this is expected because the threshold value was higher for mod 2 therefore, less training and therefore, more error. And you can see these threshold is also you can see clearly the difference you can see the 0.00079 that is 0.0008, almost 0.0008 that is the reached threshold value which is you know smaller than the threshold value that we have given here 0.0009 and then we can see mod 2 the value threshold value the 0.04 and we it is has it stopped that 0.024.

So, if we look at the number of steps that were required to reach the threshold it can see just 63 steps which are less than the number of steps that we have given, but the way we have been giving the the way we have initializing the stepmax value in our call to this function has been you know quite close based on some experimentation so that the number of steps that are required are quite close to these stepmax that we are specifying. To be on safer side we can always try much larger stepmax value however, because of the experimentation you would see which 1 epoch would have in this case you can see in the in the number of observation train partition is 71.

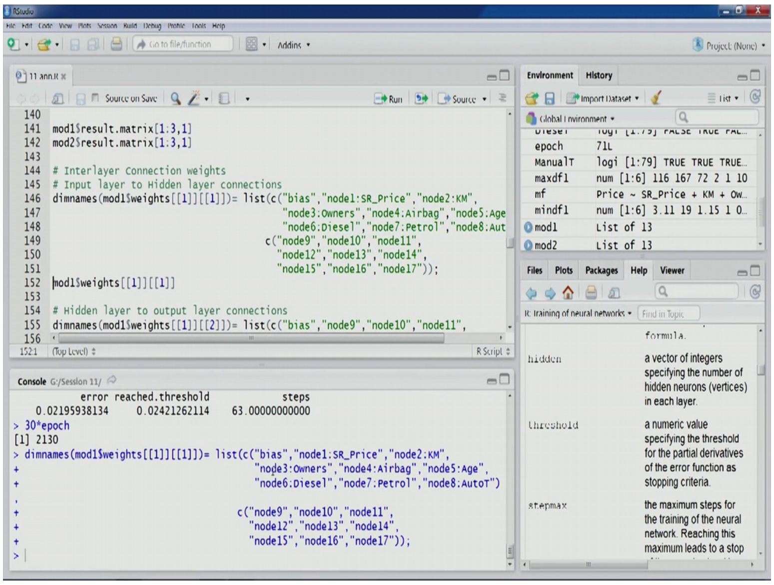
(Refer Slide Time: 16:06)



So, less than 71 that is 63 steps were required, but if we look at the mod first model, model 1 539 steps were required, but if we look at the 30 epoch value it is much larger 2130. So, within 30 epochs we are allowed 2130 steps however, only 539 were required. If we run the model again probably it might take more number of steps or less number of steps so that is you know how we can always you know do the experimentation with threshold and it stepmax.

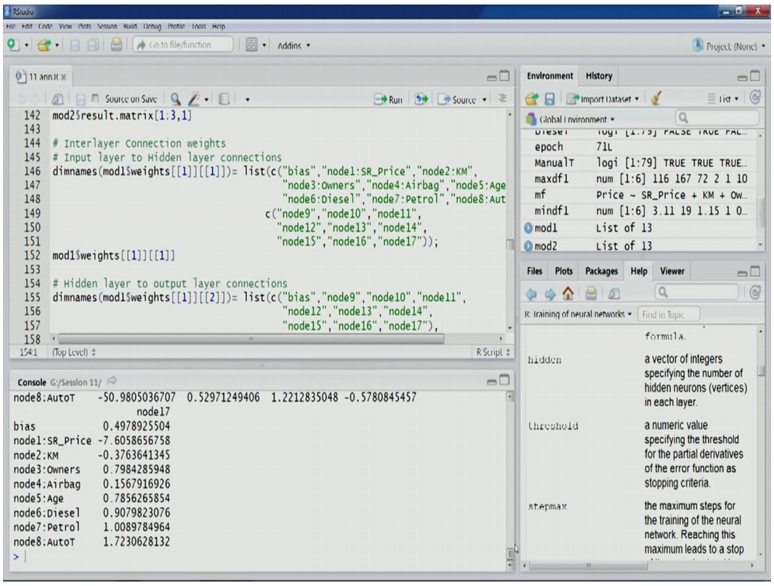
So, now let us look at the some of the details; for example, interlayer connection weights just like we did for our previous you know you know model that you know in a previous lecture that for fat and salt cheese sampling model fat and salt score were the predictors. So, similarly we can create you know we can you can see that.

(Refer Slide Time: 17:18)



We are renaming the row names, row names and column names. So, we are changing the dimension name.

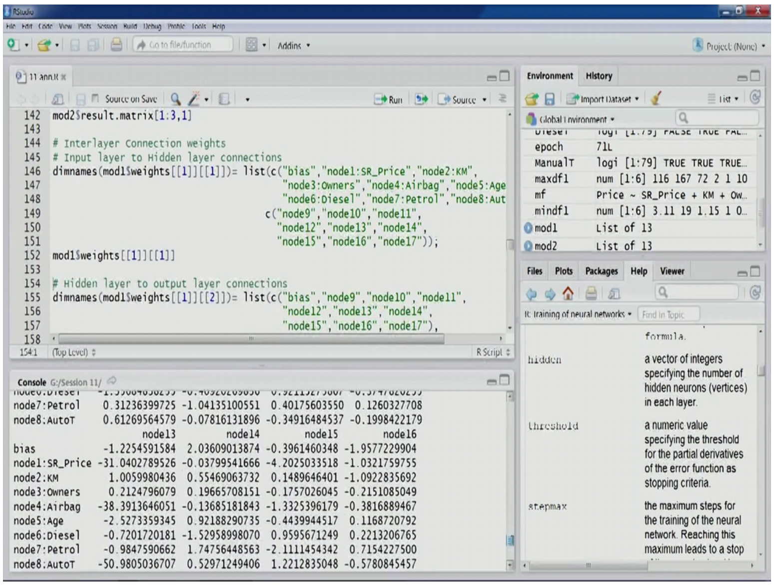
(Refer Slide Time: 17:22)



So, in this case you can see this is the, so this is the value you can see here, the interconnection interlayer connection weights.

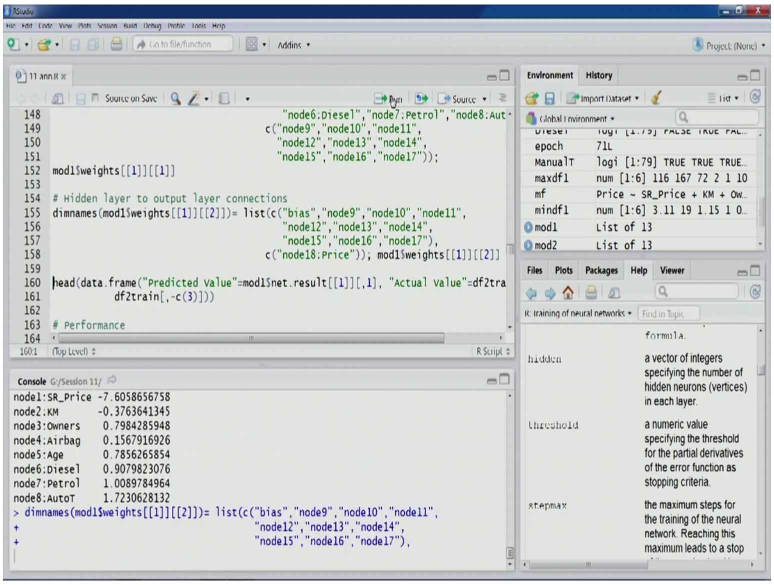
So, this is between input layer, 2 hidden layer. So, these are input layer to hidden layer connection. So, you can see bias values first row, second value is first node that is where that is corresponding to this predictor SR price, and the weight values, then for KM, then for owners, and its connection to the different nodes in the hidden layer node 9 10 11 22 up to 17 and the a corresponding weights; so that is here.

(Refer Slide Time: 17:55)



Now, a hidden layer to output layer connection also we can see. So, you can see here again in this particular code I rename the dimensions row and column. So, you can see here since we had just 1 output node price we can see node 18 price in the column and all other you know in there are row side we have bias and others node 9 to node 17 or the hidden nodes, hidden layer nodes and you can see the connection rates and bias value. So, this is this was these values are called the model 1.

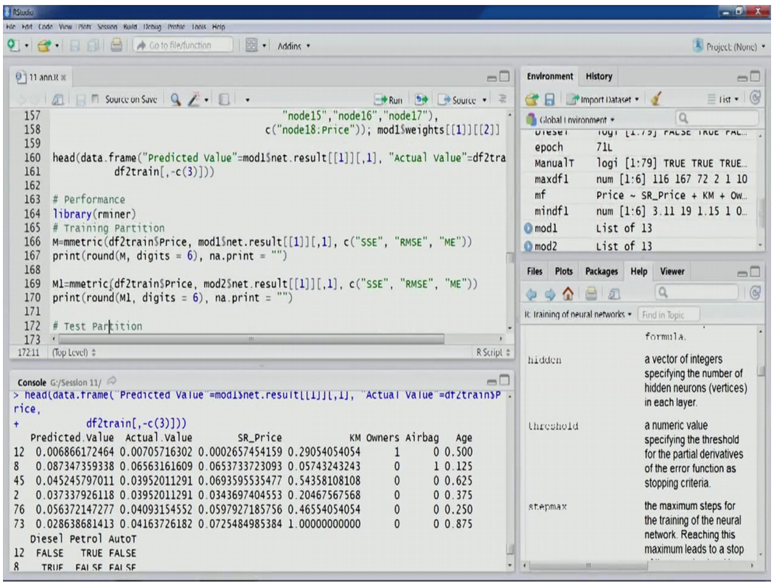
(Refer Slide Time: 18:01)



Now, we are interested in looking at the results the predicted value the actual value other things. We can run this particular code. So, I have created data frame of predicted value. So, the result would be captured here, in that result element of this mod 1 an actual value, and the remaining of the predictors in the training partition.

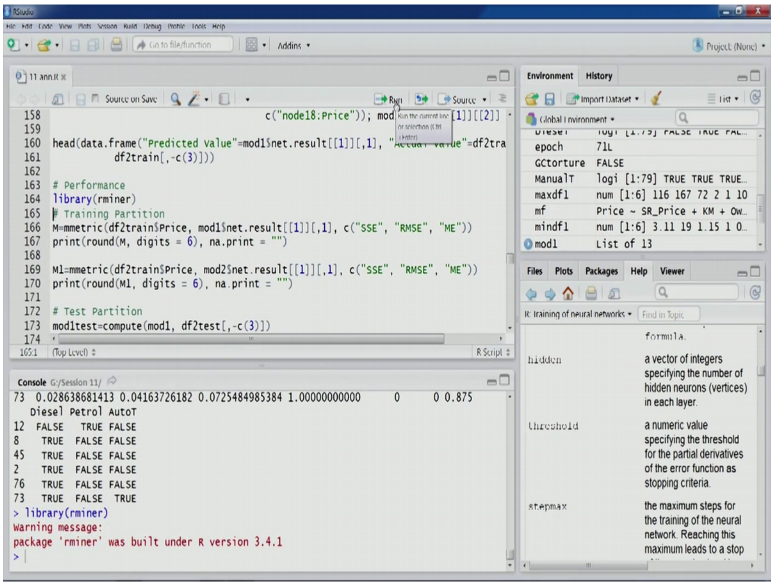
So, you can see these are first 6 observations. So, you can see predicted value and actual value. So, you can see in most of the cases the predicted value is quite close to the actual value.

(Refer Slide Time: 19:14)



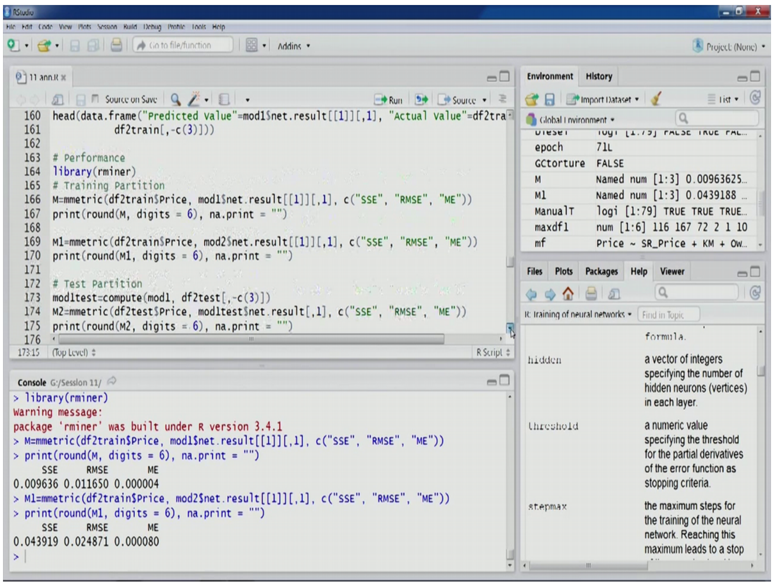
So now what we can do is let us look at the performance. So, we will use this package rminer and then we will compute some of these matrix a SSE, RMSE and ME. So, let us compute these values.

(Refer Slide Time: 19:23)



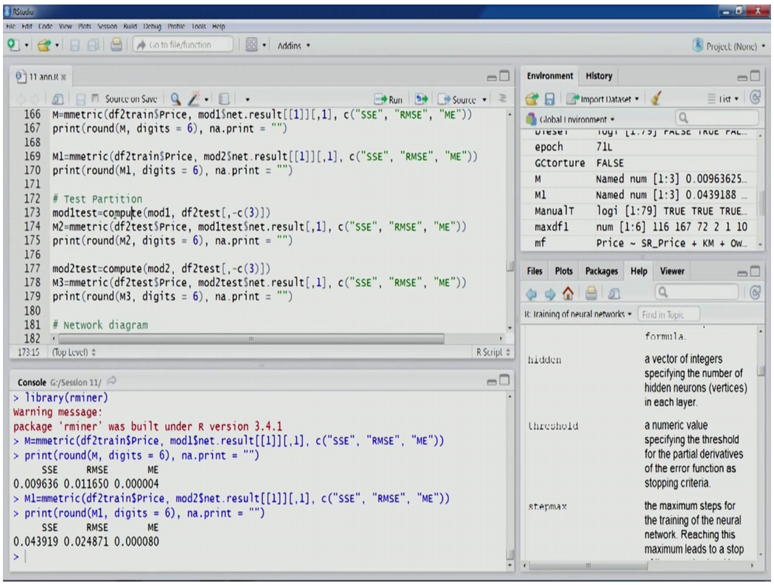
So, the first one is for the training partition. So, as you can see here.

(Refer Slide Time: 19:34)



So, these are the value for the training partition and then we will compute the values for the for the second model. So this was for the first model then let us compute for the second model. So, these are the values for second model. So, you can see that RMSE value is smaller in first model in comparison with the second model because over training has happened in case of first model. Now, what we will do will look at the performance of these two models on test partition.

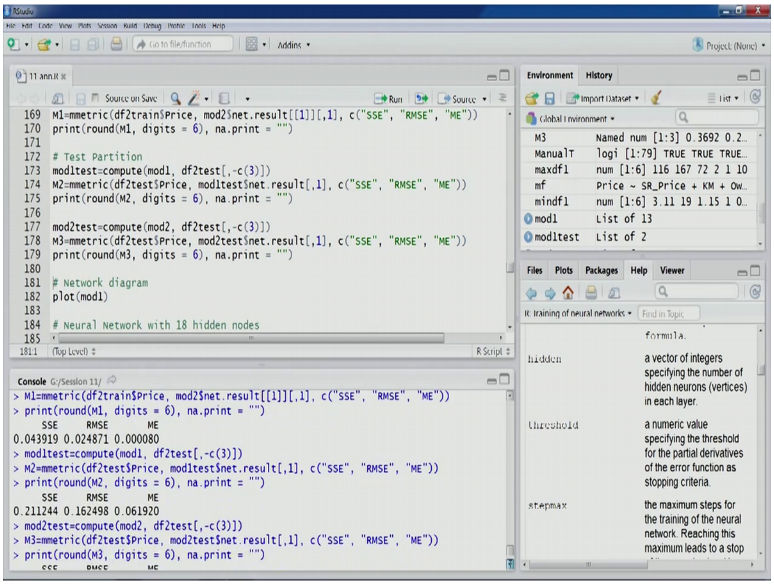
(Refer Slide Time: 20:12)



So, for this we have till now in other techniques we have been using predict function to score the test partition, but in this case this particular package neural net and we have compute function to is called the test partition observation. So, we will use compute and then other arguments are quite similar first model object and then the test partitions will score this will compute these metrics SSE, RMSE, ME and then. So, this is with respect to the first model you would see that RMSE value was 0.01 for training partition, but it is 0.16 in the test partition.

So, from here we can say that the model has over fitted to the data; so over fitted to the noise. So we can see that the error on test partition is 10 times more than that of in the partition. So this is for the model 1 and let us look at the performance using model 2. So, value is 0.21 in the for the test partition if we look at the value for the model 2.024.

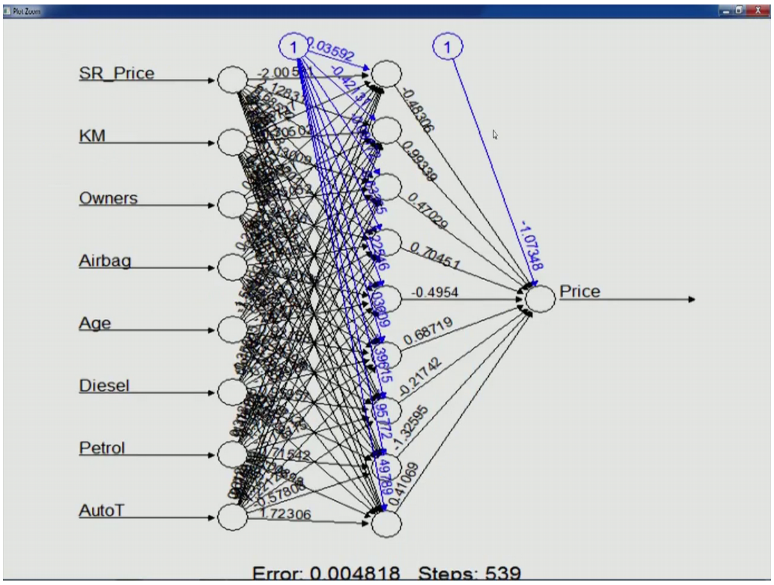
(Refer Slide Time: 21:27)



So, even in this case you would see that the performance of the model is quite poor even though less training has happened just 1 epoch just a you know I think about 60, about 60 observations we can look at the previous results epoch value our epoch this converge a number of steps that we had seen 63, so just 63 steps and so even after that the second model also seems to have over fitted to the data or it might be under trained. So, there are there could be two scenario either because the only 63 steps were used.

So, more likely scenario is this that this particular models are under trained and because of that its performance on test partition is poor however, in case of first model it seems to be over trained and because of that its performance is poor on test data. So one model, model 1 is over trained and the model 2 is under trained and that is very that can be seen from the number of steps that have been used and the threshold value that were used in these two cases. So, now let us look at the diagram plot model 1 so look at this.

(Refer Slide Time: 23:05)

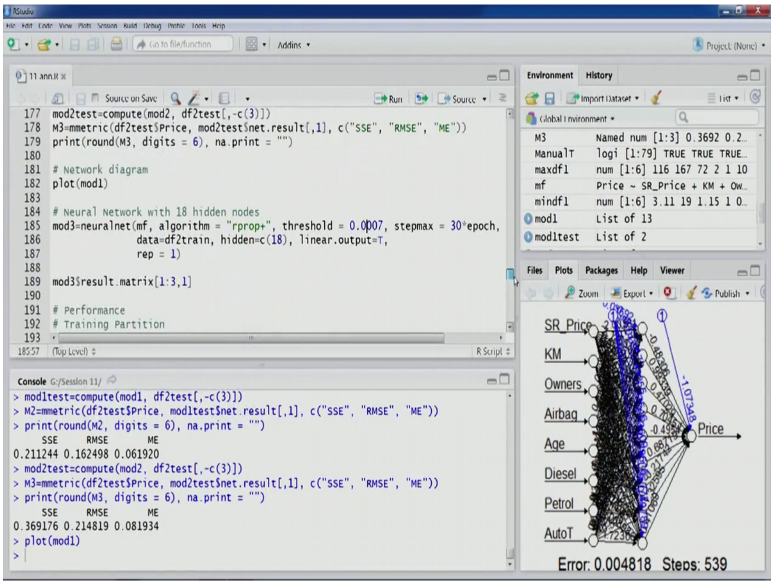


So, for model one this is the this is the network diagram that we have. So, you can see here all 8 predictor SR price, KM up to automatic transmission AutoT and you can see different connecting arrows from input layer nodes to hidden layer nodes and you can also see the bias node, bias values you know bias node and they connect connected to the hidden layer nodes on the bias values and then from hidden layer nodes connections are there to output layer node and then there is another bias value bias node, right.

So, this is what we have, so you can see that. So now so as I talked about that two models; so one model is over fitting and the second model is under fitting. So, what else can be done? So, there are as I talked about a number of experimentation can be performed in neural network modeling. So, next is whether we can change the number of hidden layer nodes. So let us see what happens if we increase the number of hidden layer nodes.

So, since we have already you know build one model, model one 1 had 9 hidden 9 hidden nodes in the in the layer so I still it was over fitting. So therefore we expect if we create a model with 18 hidden nodes. So, this one is also going to this one also this one is also going to over fit to the data or fit to the noise. So, we look at the this particular model 3 that we are going to create look at the arguments. So, threshold value you can see now this transition much is smaller. So earlier one was 0.0009 that we had a specified 0.0009 yes.

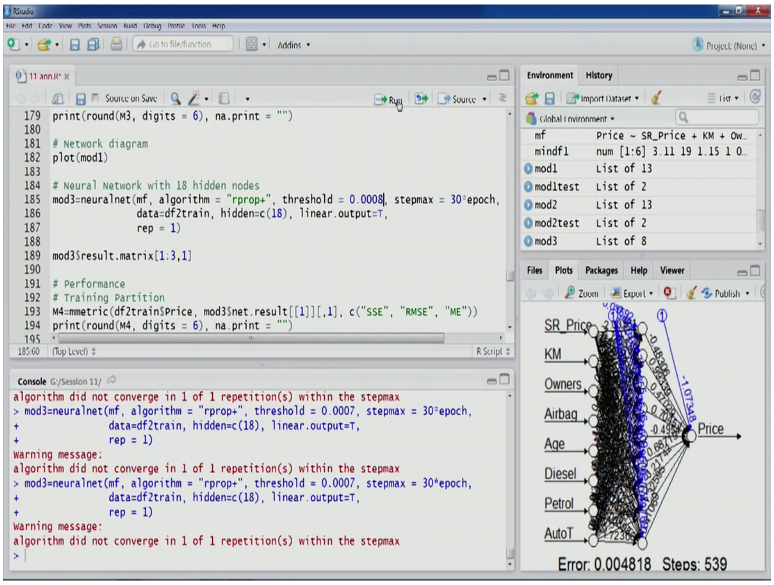
(Refer Slide Time: 24:50)



Now, you can see here 0.0007 and these steps are same this is because since we will have a more number of hidden layer node. So, of course, you know the model would converge even at a lower threshold value and still keeping the you know same number stepmax you know value.

So, we are having in, we are passing a quite tighter value for stepmax and within this you know value of stepmax; we are trying to you know get the highest possible threshold that can be used. So, of course, it will lead to over fitting. Let us run this so it did not converge. So, let us again run, not converged even this time. So let us one more time, not converging.

(Refer Slide Time: 25:30)



So, what we will do? We will increase threshold value to from 7 to 8 and we will run it again and you see that immediately it converged.

So, you can see even after you know increasing the number of hidden load no hidden layer nodes from 9 to 18 if we look at in terms of threshold value the earlier threshold value was 0.0009 and it what it what it converged and in this case it is just 0.0008. So, just you know one you know fourth decimal point one unit decreased there and we have increased the number of nodes and we have doubled the number of hidden layer nodes.

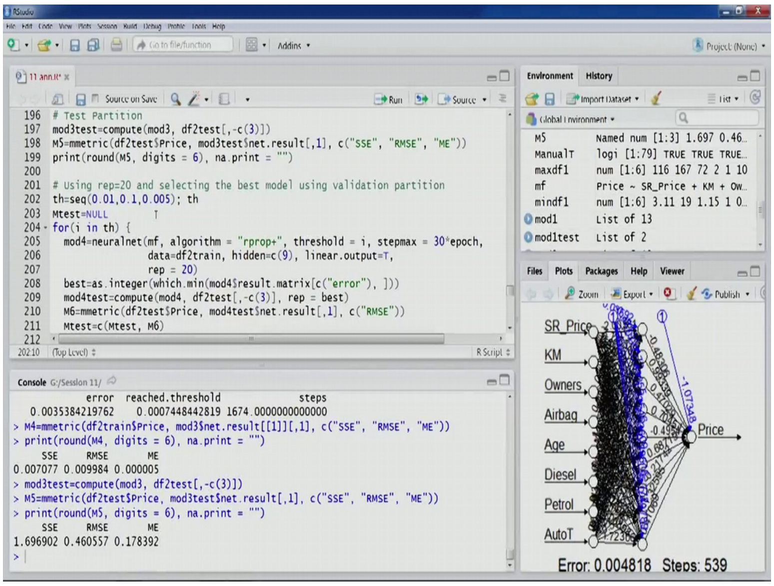
Now, we look at the error value now this is much lower. So, you can see 0.0035 we look at the earlier a value, error value in first model you can go back and you can see the other was 0.0048 and here what we have 0.0035. So, error is much less since threshold value is also the threshold value at which the model has converge is is lower and you can see the number of steps now more number of steps 1674 were required for this convergon to take place.

Now, let us look at the performance of this model on training partition and test partition. So, let us compute the same matrix. So, you can see now RMSE value has decreased further 0.0099 in this case. Now, let us look at the performance on test partition. So, again you can see that test partition the performance has become much worse. So, you can see that now the RMSE value is 0.46 and it is you know you know about it is much higher much times higher in comparison to previous case. So over fitting has increased if we compare this model 3 with model 1. So, we can see that performance on new data validation data is worsening and as reflected in the RMSE value.

So, what we need to do to find out the best model? So, what we will do now? We will build the model using certain such experimentation. So, what we are going to use the rep argument that we had kept as 1 in previous modeling we will make it 20. So, well run the same model 20 times and then pick best one out of those 20 runs.

Now, other things also will change. So, you can see here that what I am doing here in this particular for loop is step mac max is kept as 30 epochs that is the highest value and you would see that threshold value is now mentioned value i and I am running a for loop using i as a counter and we are going to build a number of models and we will test them on validation partition just like this graph. So, we will create this kind of plot. So, the threshold values, so we are going to do a performance we are going to do some experimentation with threshold value.

(Refer Slide Time: 29:10)



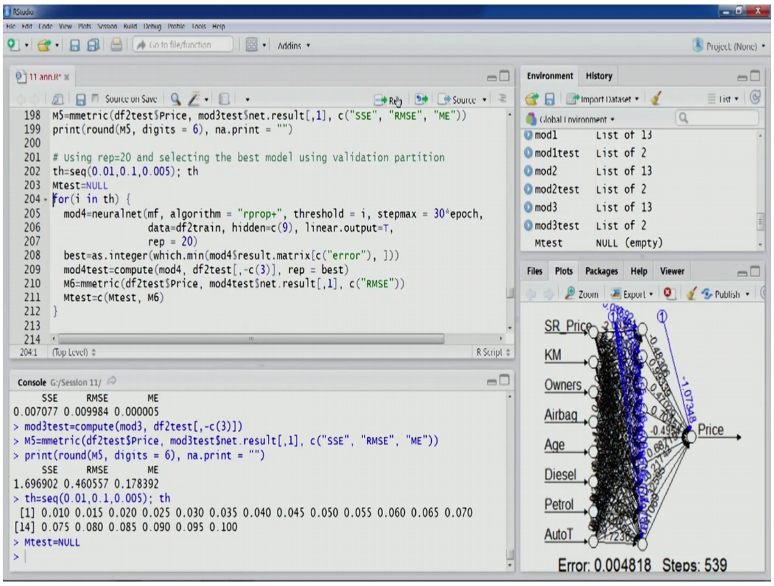
So, let us, these are the threshold value that we are going to use, so 19 of them, so starting from 0.01 and then 0.015 then 0.020 and up to 0.1.

So, we will create these 19 models and as you can see for each of these 19 threshold values we will create 20 models a repetition is 20 and the best one within that you know would be picked. So, essentially for each of these threshold value we would be picking the best model based on 20 runs. So, let us and then Mtest is the variable which where we would store the error value a error validation error value.

So, you can see in next few lines of code. So, will you can see here the best this one is being used to find out the the out of 20 runs which one is the best run, which one is the best model and once that is selected then it is being used to score the test partition as you can see here and once that is done we are computing the a matrix values RMSE mainly and then is storing it in this particular vector.

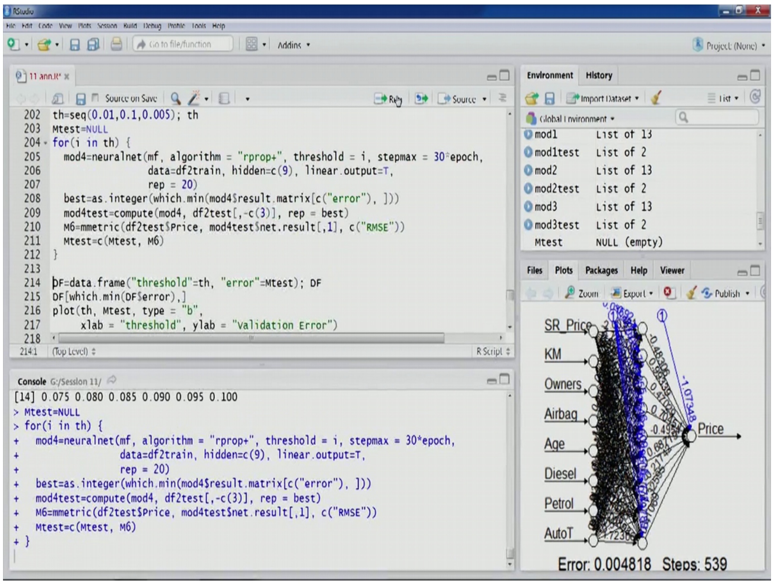
So, for all the 19 models we would be storing the this value. So, the plot that we are going to use is a going to be error on y axis validation error on y axis and threshold value on x axis. So, let us initialize Mtest, let us run this loop and you can also notice that number of hidden layer loads are 9.

(Refer Slide Time: 30:52)



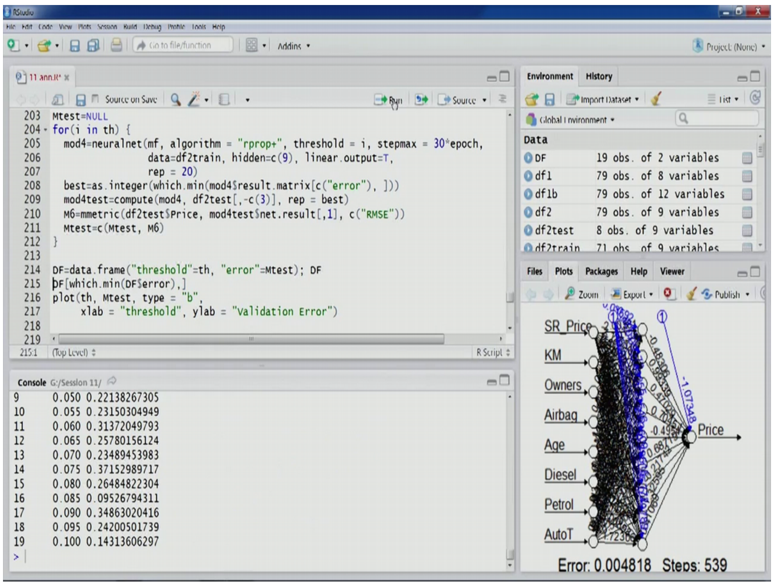
So, let us run this loop. So, it will take with time because we would be running 20 multiplied by 19 models and yes. So it has all of them have been computed.

(Refer Slide Time: 30:56)



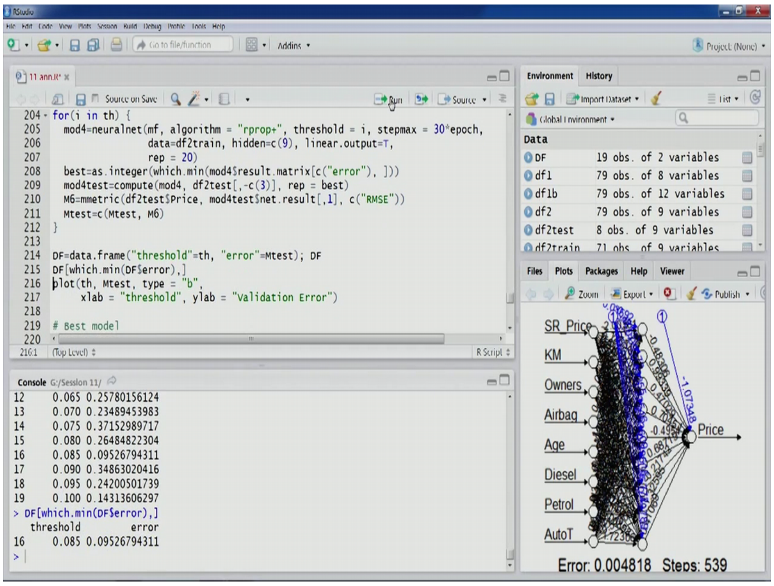
So, there were no problems of convergence as we saw in earlier cases where we had to reduce threshold value because stepmax is large enough to allow all the models for different threshold values to converge. So, once it has been computed we creating a data frame of threshold value and error values that have been computed.

(Refer Slide Time: 31:29)



So, you can see here, threshold value 0.01, 0.015 up to 0.1 and you can see the corresponding validation error that has been computed 0.33, then it drops to 0.17 and it keeps on dropping. So, there are swings if we look at this particular output. So, let us find out where the value is minimum? So, you can see 16 that is 0.085.

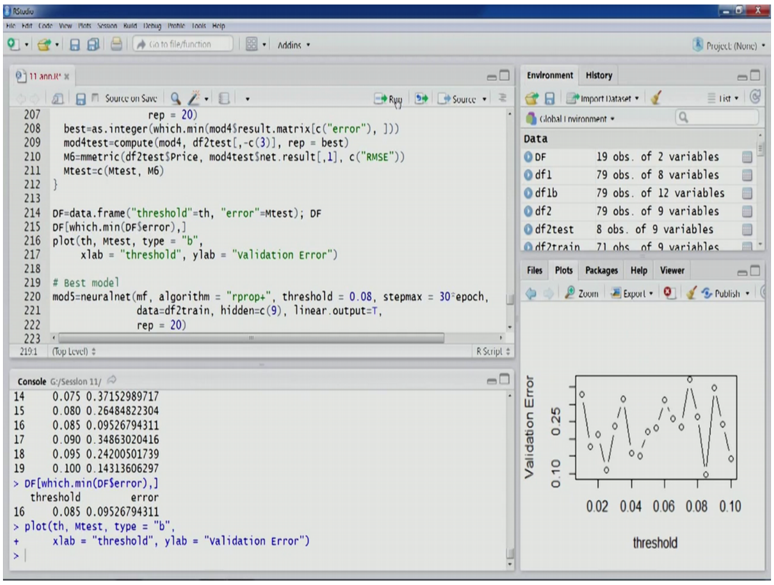
(Refer Slide Time: 31:55)



So, when the threshold value is 0.085 the error is minimum. However point of caution here that the data set that we are using is quite small however, a you know if you do start an experimentation with this loop you know you run it again then up is still the is still even after that if threshold value comes around that 0.08, 0.085, 0.09.

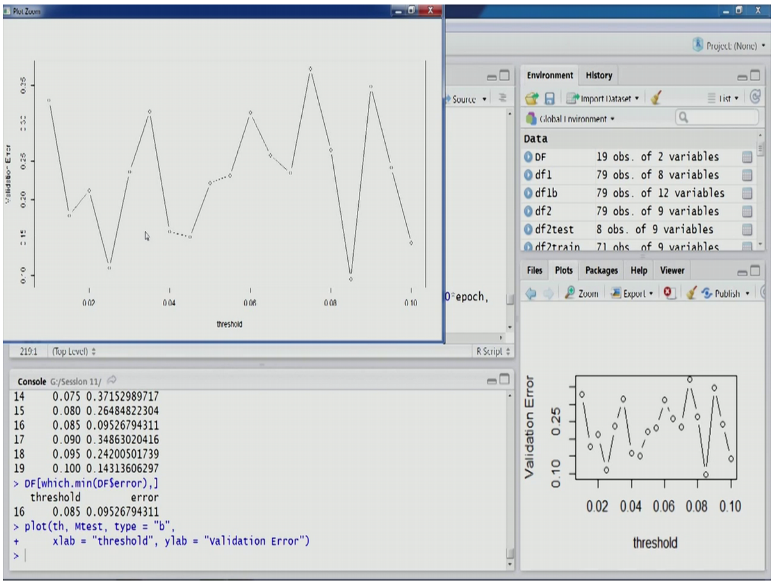
So, even after the smaller size, probably this one is the you know best threshold value to get the best model and once this is identified we can create our plot you can identify the same thing using this plot. Let us get this plot a error versus threshold value.

(Refer Slide Time: 32:40)



So, you can see here.

(Refer Slide Time: 32:43)



So, you can see that validation error because there are many you know you know you know swings here however, we had a larger test partition. So, these points would have been is smoothed out and we would have been seeing clear you know minimum point of minimum validation error just like this plot how because we have just 8 observation the rest partition. So, the plot is not that smooth. However, still we can identify 0.085 as the a minimum at the point of minimum validation error.

So, once this is known to us we can again build for our best model. So, as you can see that typically the best model as per these results is around 0.0885. And so we will stop here at this point. And in the next lecture we will build our model using this particular threshold value 0.085.

Thank you.