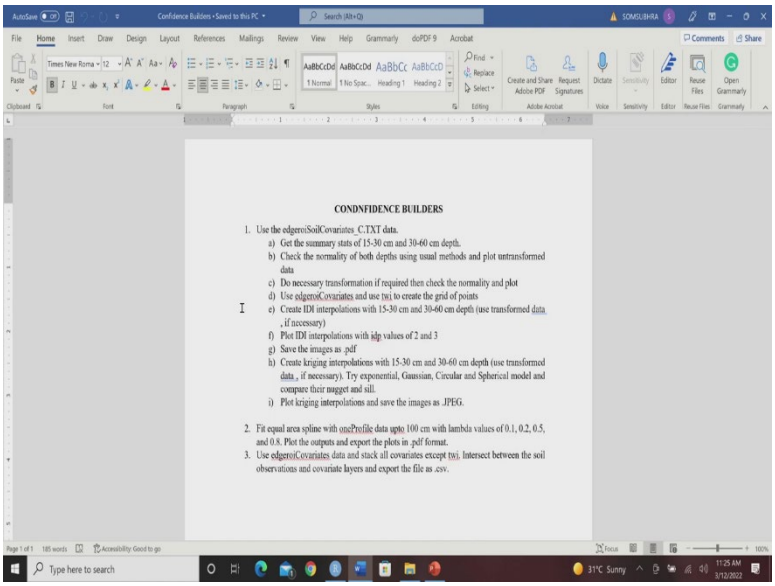
Machine Learning for Soil and Crop Management
Professor Somsubhra Chakraborty
Agricultural and Food Engineering Department
Indian Institute of Technology, Kharagpur
Lecture 54
Digital Soil Mapping with Continuous Variables (Contd.)

Welcome friends to this fourth lecture of week 11 of NPTEL online certification course of Machine Learning for Soil and Crop Management. And in this week, we are dealing with Digital Soil Mapping with Continuous Variables. And in our previous lectures, we have already seen how to deal with different types of… exploratory data analysis, basic GIS operation, basic geo statistical operation using R and also we have seen how to use R for producing the Kriging interpolation, IDI interpolation, all this. And I will be sharing all these codes with these annotations in the forum, but at the same time, I also have some practice set for you for boosting your confidence. So, that you can execute these codes for solving this question and feel more confident.

(Refer Slide Time: 01:23)



So, let me show you this is the these confidence builders which I have developed from you. So, you got you are going to use these edgeroiSoilCovariates underscore C dot txt data. Once you run all the codes which I have already shared, you can be able to get these edgeroiSoilCovariates underscore C dot txt data and then the first question is get the summary stats of 15 to 30 centimetre and 30 to 60 centimetre depth check the normality of both depths using usual methods and plot and transform data do necessary transformation if required, then check the normality and plot then edgeroiSoilCovariates and use twi to create

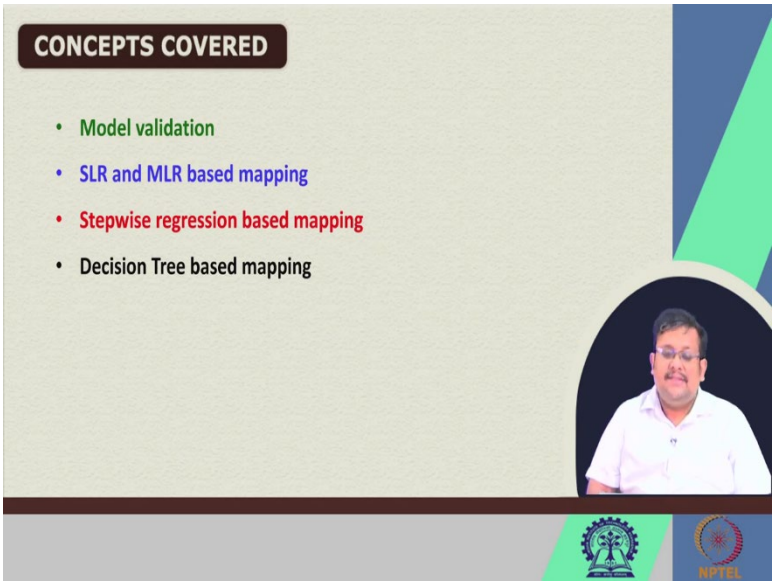
the grid of points and then create idea interpolation with 15 to 30 centimetre and 30 to 60 centimetre depth, use transform data if necessary.

And plot idea interpolation with idp values of 2 and 3 and save the image as dot pdf create Kriging interpolation with 15 to 30 and 30 to 60-centimetre depth use transform data if necessary, try exponential Gaussian circular spherical model and compare their nugget and sill. Plot Kriging interpolation and save the images as dot JPEG.

And the second question is fit equal area spline with oneProfile data up to 100 centimetre with lambda values of 0.1, 0.2, 0.5 and 0.8 plot the outputs and export the plots in dot pdf format. And then third question is used edgeroiCovariates data and stack all the covariates except twi intercept between the soil observation and covariate layers and export the file as dot csv.

So, whatever we have covered so far, you will see these you know, these confidence builders are basically showing an extract. So, you know, if you go through these confidence builders you will be able to perform all the tasks which we have already seen. So, please go through these questions, the problem set and solve it and once you solve it, then you will feel more confident about this DSM model application using R.

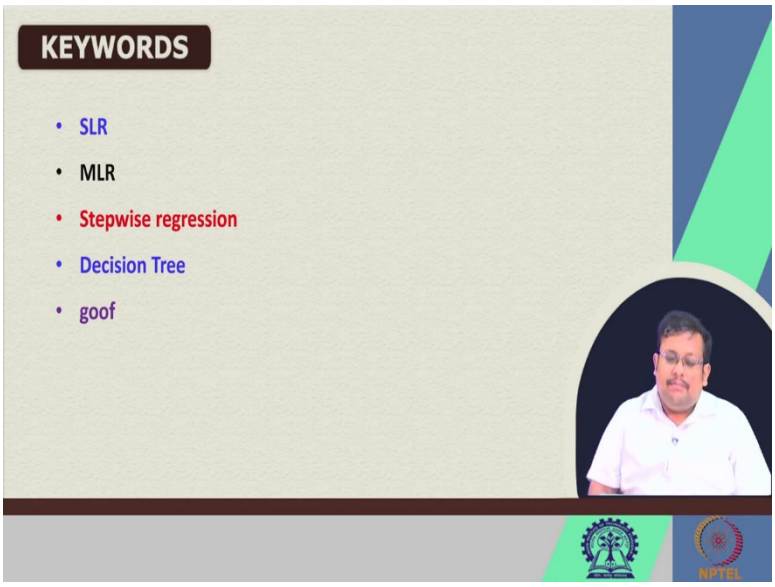
(Refer Slide Time: 03:51)



Now, in this lecture we are going to cover these concepts, we are going to cover the model validation, how to do the model validation. We are going to start with the Simple Linear Regression, Multiple Linear Regression. And I am going to show you how to do the mapping based on this Simple Linear Regression and Multiple Linear Regression. Then, we are going

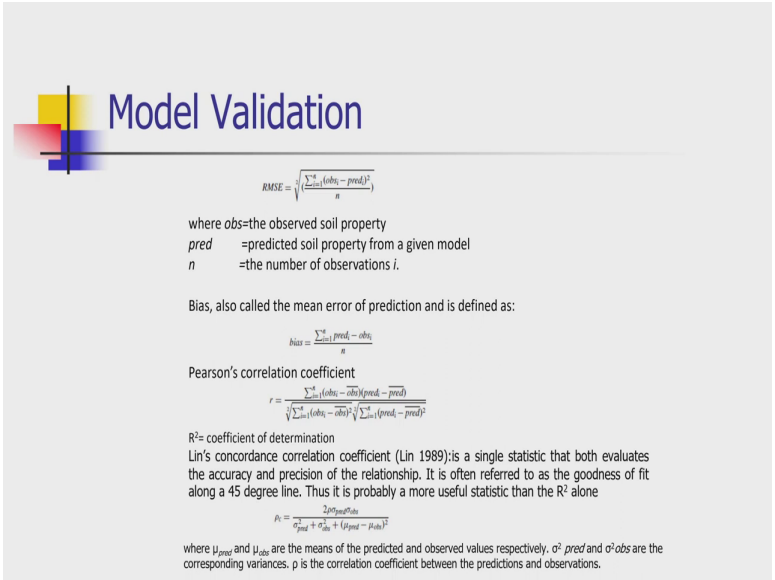
to see the stepwise regression-based mapping and then finally, we are going to talk about that decision tree-based mapping. So, decision tree is also known as the classification regression tree when it is a classification problem, then we call it a classification tree and when it is a regression problem, then we call it a regression tree.

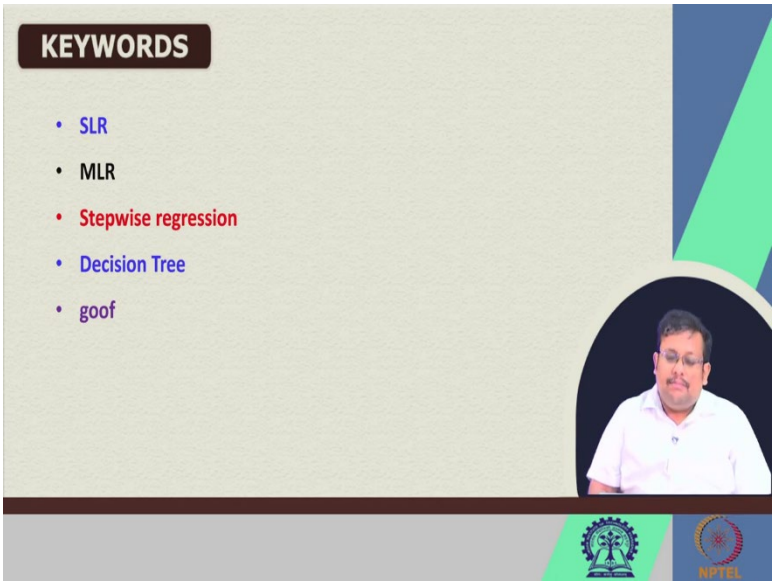
(Refer Slide Time: 04:32)



So, the keywords which we are going to discuss are SLR, MLR, Stepwise regression, Decision Tree and goof. Goof is a function which we are going to discuss.

(Refer Slide Time: 04:44)



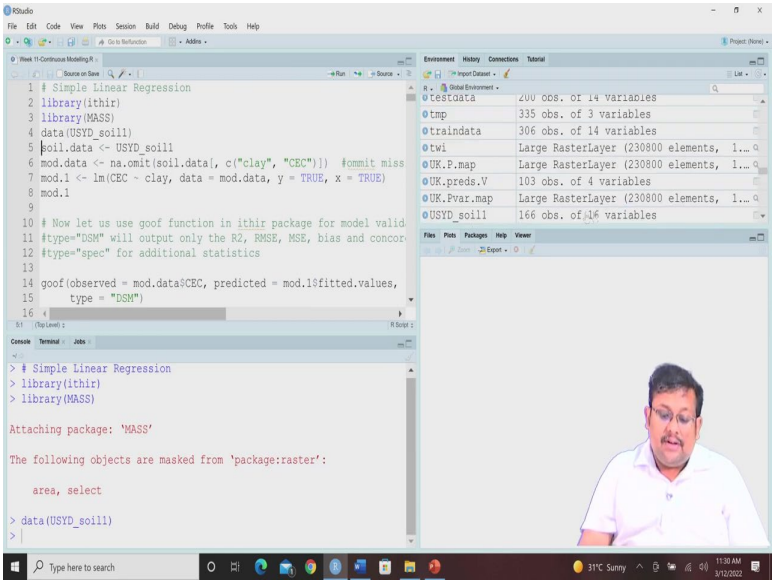


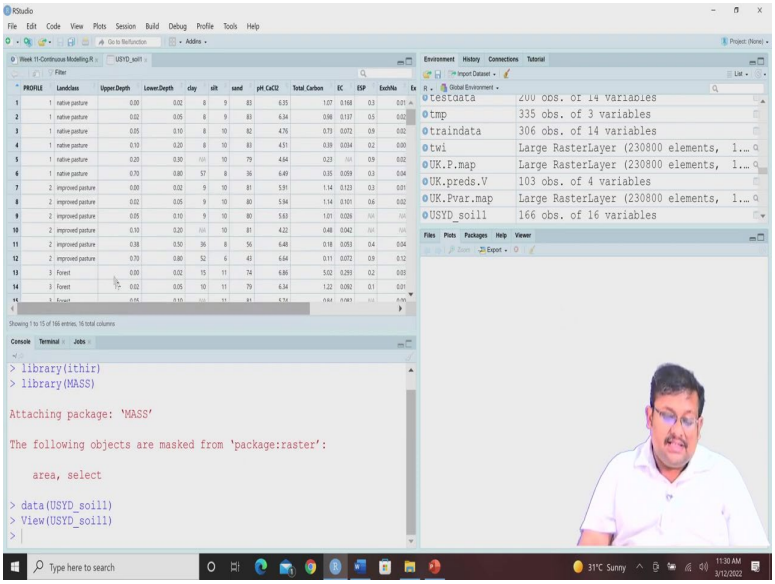
So, remember this goof function we are going to use for goodness of fit… for identifying the goodness of fit statistics. So, the short form of this goof is, the short form is goodness of fit is goof and there are different types of model indicators which we are going to use you know about RMSE bias is basically the mean error.

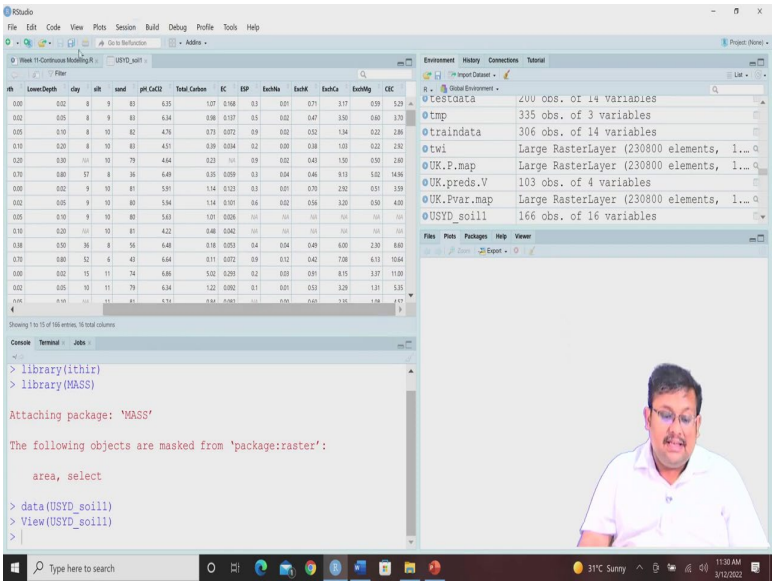
So, it is predicted minus observed values for all the observation divided by n. So, an IBL model should have 0 bias and so, Pearson correlation coefficient we have already discussed, another model coefficient we are going to discuss is the Lin's concordance correlation coefficient or CCC. So, it is a single statistic that both evaluates the accuracy and precision of the relationship, it is often referred to as the goodness of fit along a 45-degree line. Thus, it is probably a more useful statistic than the R square alone.

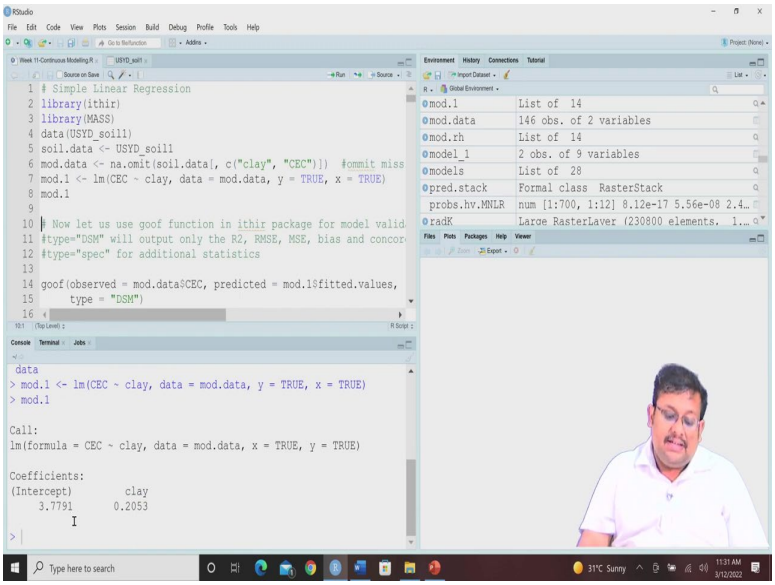
So, this is the formula this is the Lin's concordance correlation coefficient and this is how you calculate it. So, 2 rho sigma pred and sigma observation then sigma pred square, plus sigma observation square plus mu pred minus mu observations whole square, so were these mu pred and mu observed are the measures of the predicted and observed values respectively and this sigma squared pred and sigma square OBS are the corresponding variances of the predicted values and observed values and rho is the correlation coefficient between the predictors and the observations.

(Refer Slide Time: 06:34)









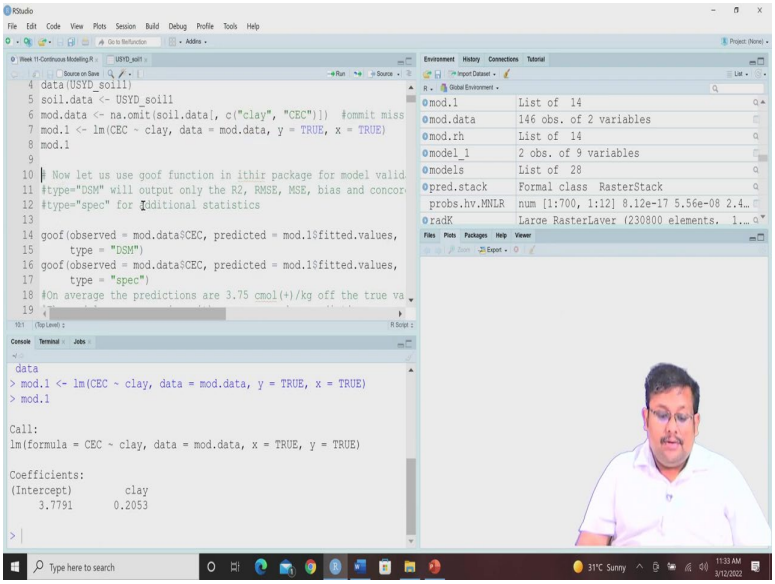
Now, let us go ahead and see how to execute this thing in R. So, we are going to start with the simple linear regression, simple linear regression again we are going to call the library ithir then library mass, the mass package we have already installed before then the data USYD soil 1 we are going to call it.

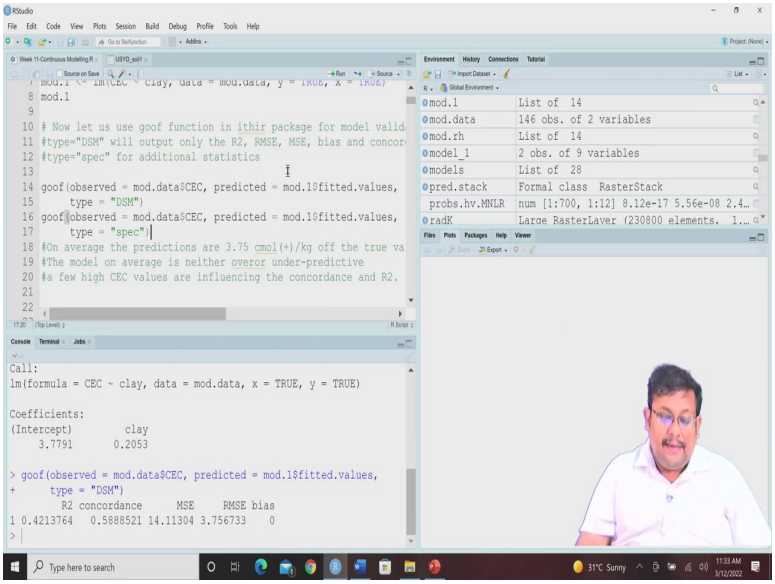
So, USYD soil 1 you know 166 observations 16 variables, we have previously seen these data profile, land class, upper depth, lower depth, clay, silt sand, pH CaCl2 total carbon EC, ESP, then extendable sodium, extendable potassium, extendable calcium, extendable magnesium and mineral magnesium and CEC, we know it already.

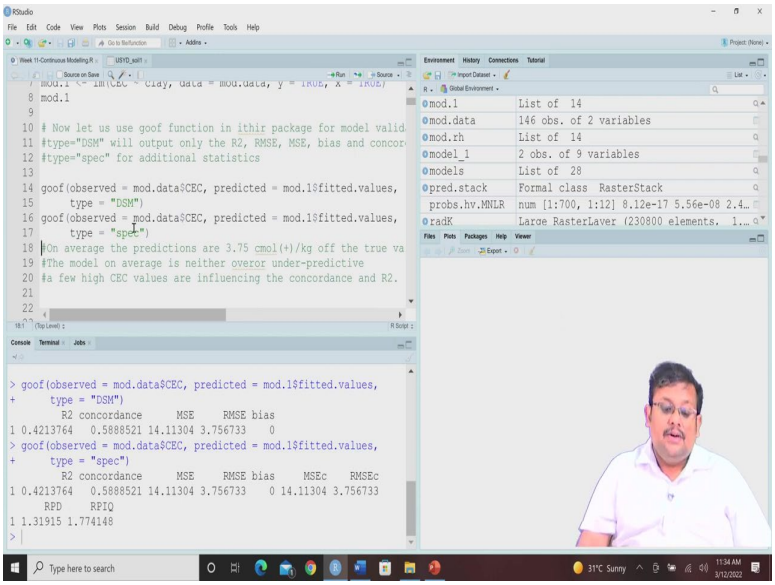
So, here we are going to first the omit the missing values. So, the missing values for omitting the missing values we are going to use these na dot omit function and we are going to keep only the clay and CEC because now, we are interested to predict the CEC based on the clay content. So, we are going to keep this only clay and CEC and now we are going to fit the linear model for fitting the linear model the function is lm and here we are targeting the cation exchange capacity by clay.

So, we are putting this sign to indicate that CEC is the target whereas clay is the predicted and our data is mod dot data. So, because mod dot data is basically the USYD soil 1 data after removing the missing values and so, let us run these and let us see what is the what are the model parameters. So, you can see the model parameters the intercept you are getting 3.77 and the slope of the clay is 0.20. So, from there you can have an idea about the model accuracy.

(Refer Slide Time: 08:30)







Now, we are going to use this goof function in ithir package for model validation. Now, when we use the goof function, there is a specific argument where we can either type DSM or we can use the spec. So, if we use the DSM then only it will produce 4 to 5 different… model indicators like R square, RMSE and then and also the R square, RMSE, RPD and bias but in case of you are using the spec as a you know in the type argument then you will have more and more results.

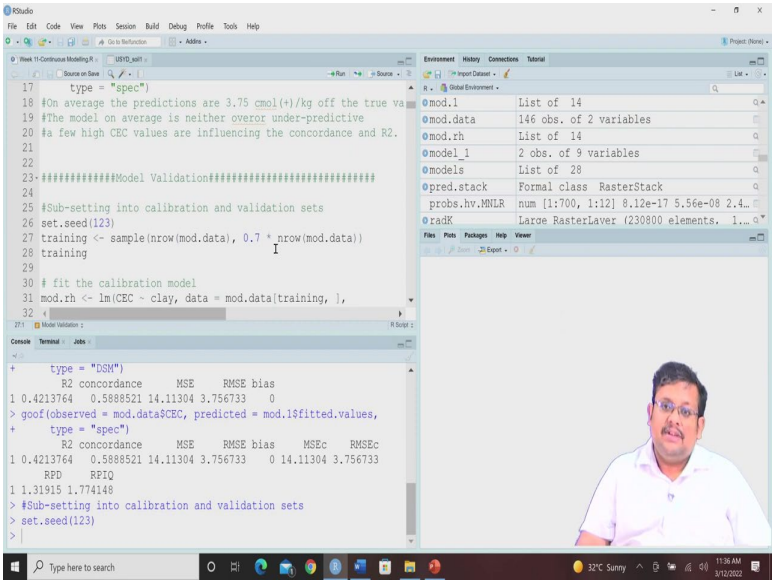
So, if you are using again the DSM you will get R square, RMSE, MSE and bias for concur and concordance correlation statistics, but in case of spec you will be getting the additional statistics. So, let us see how this looks like. So, we are going to use this goof function our observed is mod dot data and we are specifying CEC our predicted values our mod fitted values and that type is DSM.

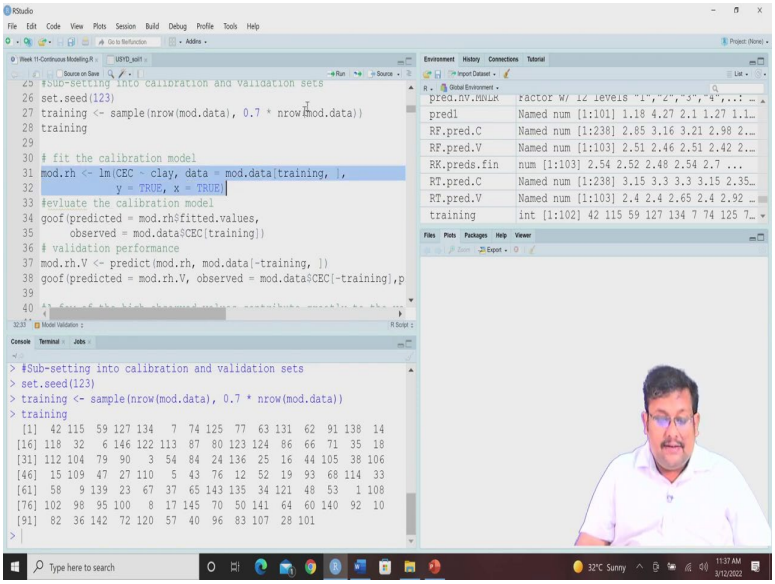
So, the… so, let us run this. So, you can see R square is 0.42 and we are getting the concordance correlation coefficient 0.58 MSE 14.11 RMSE 3.75 and bias 0 and if we use the spec, so you can see here we are using the DSM in the type argument, but if we want to use the spec, you will see that it will give you more statistical output. So, R square concurred MSE, RMSE, bias MSE calibration RMSE calibration RPD and RPIQ.

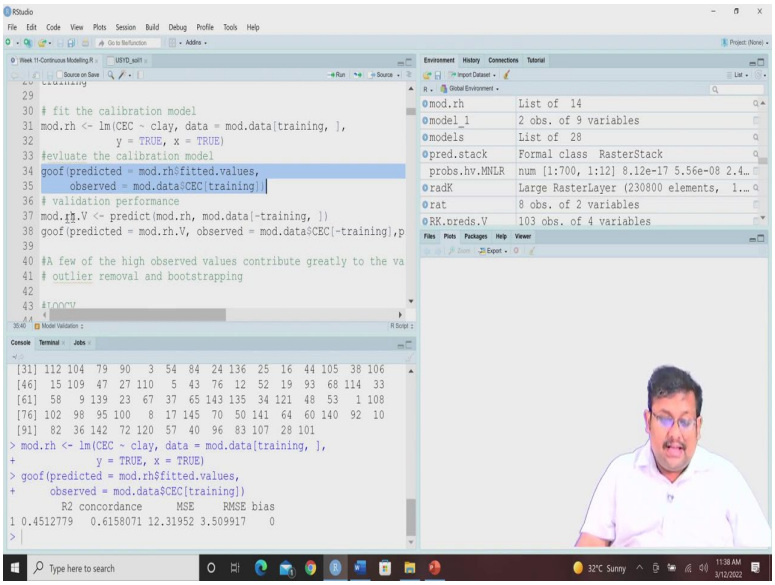
Now, since we have not divided the data set into calibration validation, so whole data set has been already killed you know, they have they have calculated they have considered R… have considered that the whole dataset as calibration as well as validation file. So, you will see both calibration and validation will give you the same results. So, here the MSE of the validation is 14.11 also MSE of calibration is 14.11. Then RMSE 3.75 here also you can get 3.75 RPD 1.31 and RPIQ 1.77.

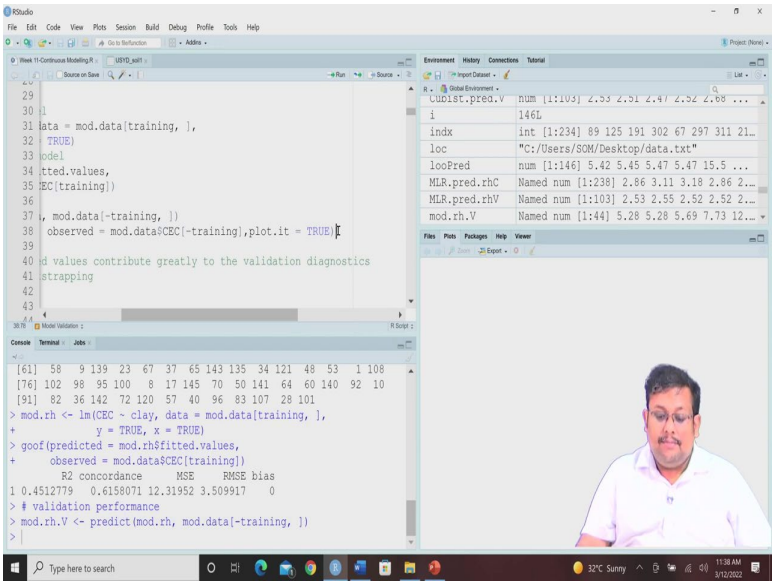
So, from this result we can see that the on average the prediction of R 3.75 centimole per kg of to the true value and the model on average is neither over or undefeated, if you highs CEC values are influencing the concordance and R square. So, this is the interpretation from this from this simple linear regression.

(Refer Slide Time: 11:40)









Now, so here we did not do any kind of model validation, but if we want to do the model validation separately, so for that we need to divide the data into calibration and validation states. So, for calibration and validation set identification, we are going to randomly divide the data set into calibration file and validation file. So, for that we are going to use the set seed comment.

So, set seed 123. So, set seed 123 basically when we use the set seed, set dot seed 123 it is these values specifies the initial value of the random number seed. So, here we are using these 123, that means 123 is set as the random number value and the main point of using the seed is to be able to reproduce a particular sequence of random numbers.

So, if you want to change the sequence of random numbers, you can use other seeds also. So, here we are using these 123 as a sequence for producing these random number seed and then you can see we are separating the data, we are first selecting the data called training data and training data we are we are we are separating the data 70 percent of the total data. So, for separating the 70 percent of the total data, we just have to multiply with 0.7, 0.7 means 70 percent with the number of rows number of rows means the number of samples.

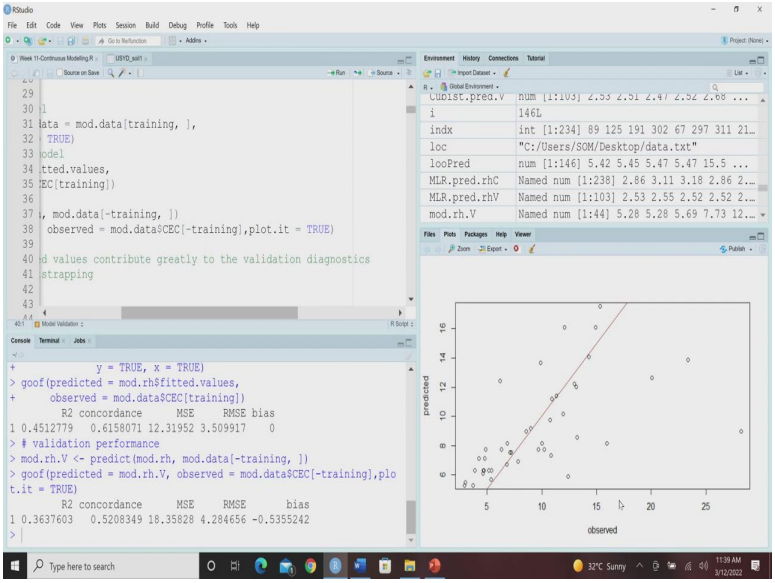
So, when we multiply the number of samples with 0.7 that will give you that number of training data. So, we are getting that and then we are we want to see how this training data will look like. So, here you can see in that training data set these are the observations so 40 second observation, 115th observation, 59th observation, 127th observation, 134. So, these are selected very randomly, so okay.

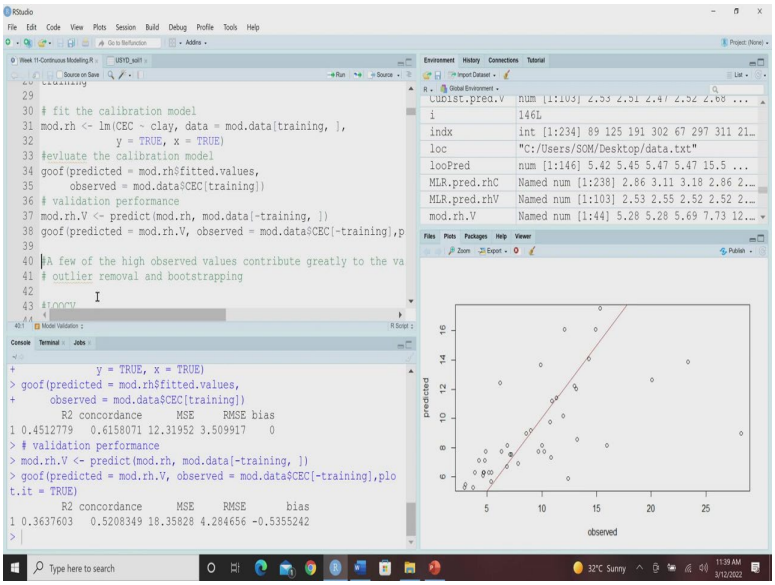
Now, once we have divided the world, we have selected the calibration model, let us fit the calibration model using this linear model. So, here again, the CEC is our target clay is our predictor, our data is mod dot data. And we are focusing now only with the training dataset for all the columns. So, all the variables are included. So, let us run it first.

And then we will let us run these goof function for these predicted values and missing values and you can see this is the model calibration statistics. So, we are getting R square concordance MSE RMSE bias. Now, we are going to see the validation performance. So, for validation performance again, same thing, except we are using minus training.

So, minus training means if we have the total number of samples, and if we subtract the number of training samples on or the specific training samples, then we will be left with only 30 percent of the data that is the testing samples. So, these testing samples or the validation samples we are going to use and then we are going to predict for this testing samples and the goof function we are going to use, now… you see one thing, that the argument of goofy is you have to give predicted values and observed values that is it. So, here we are giving the predicted values which we have already predicted in our previous step, observed values is already you know, this mod dot data minus training. And then you want to plot that also. So, if you want to plot then you can use this plot is true.

(Refer Slide Time: 15:21)

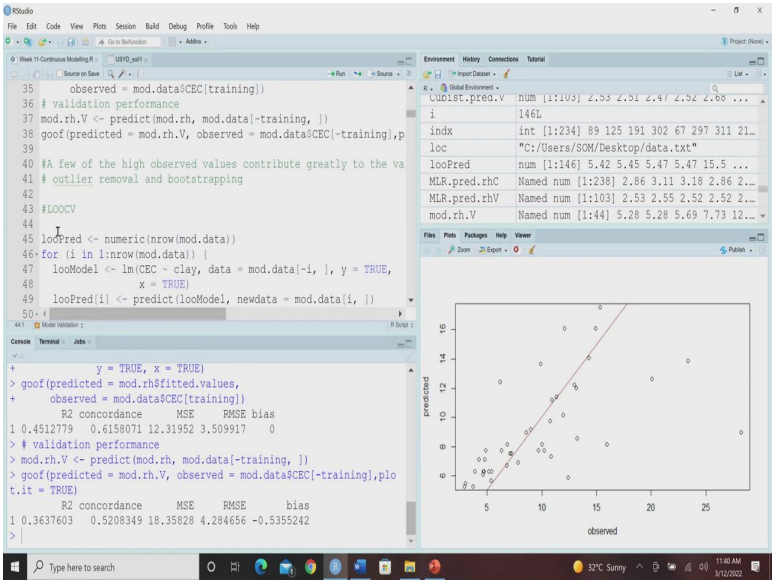


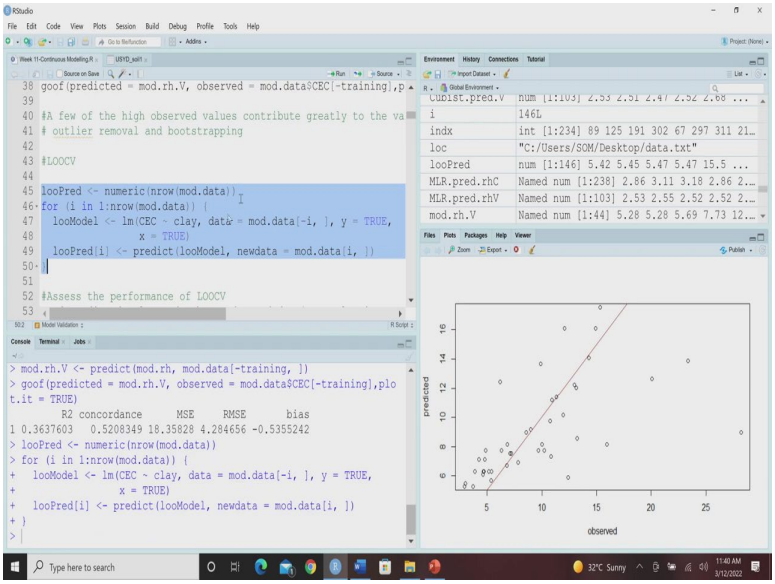


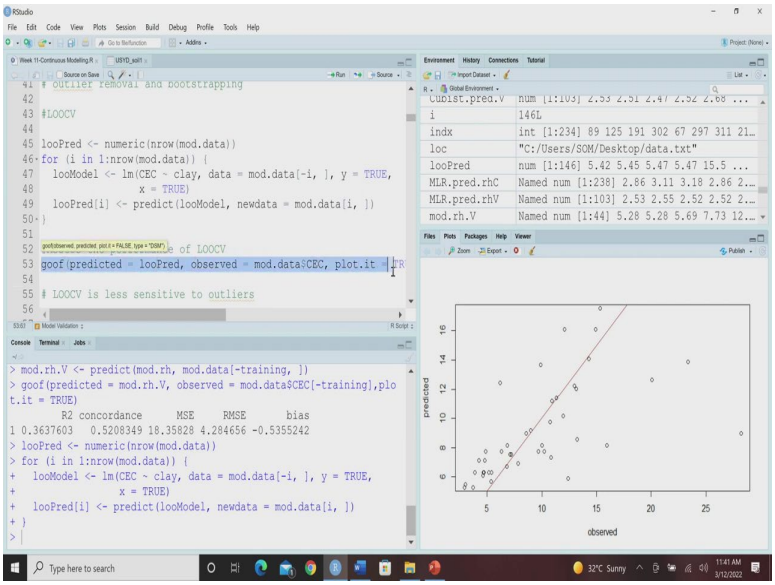
So, here you can see this is the model validation results R squared 0.36 concordance 0.52 and then MSE is 18.35 RMSE 4.28 bias is minus 0.53. And this is a predicted versus observed results. So, you can see from this calibration statistics and validation statistics is that a few of this high and also from this plot that a few of these high observed values contribute greatly. So, here here here, so these high observed values contribute greatly to the validation diagnostics.

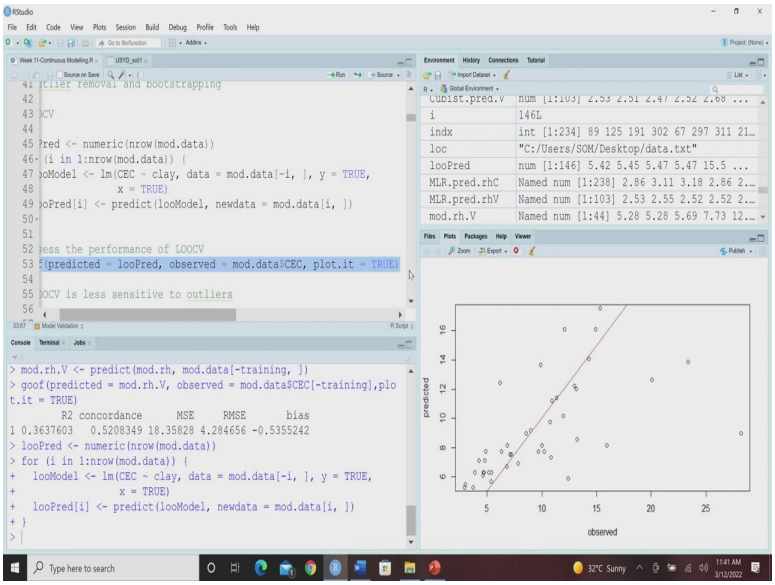
So, whatever we are getting here 0.36 they may be so, this low comparatively low validation score could be due to these high values. These are the validation samples these are the 30 percent validation samples. So, how to deal with this problem, so they may be some outliers. So, you can see there some unex, abnormal higher values or outliers. So, what are the two ways to deal with this type of problem either you can go with the outlier removal or you can go with the bootstrapping, both the methods.

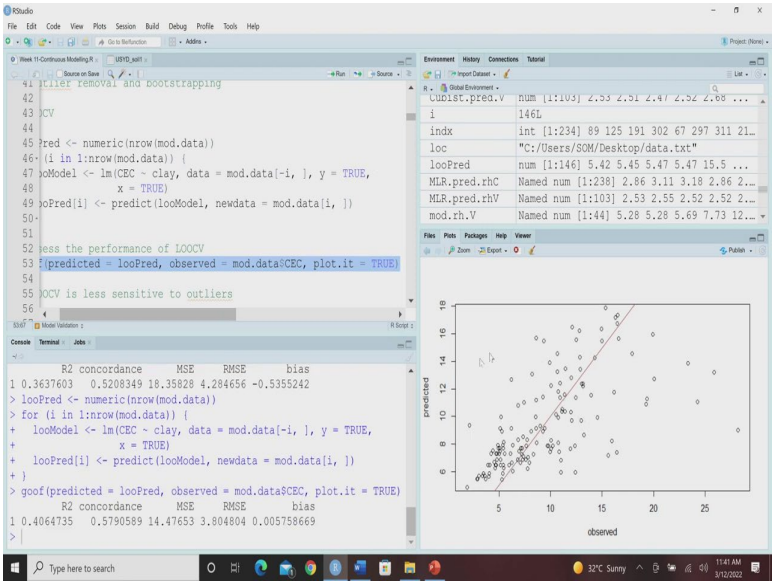
(Refer Slide Time: 16:40)









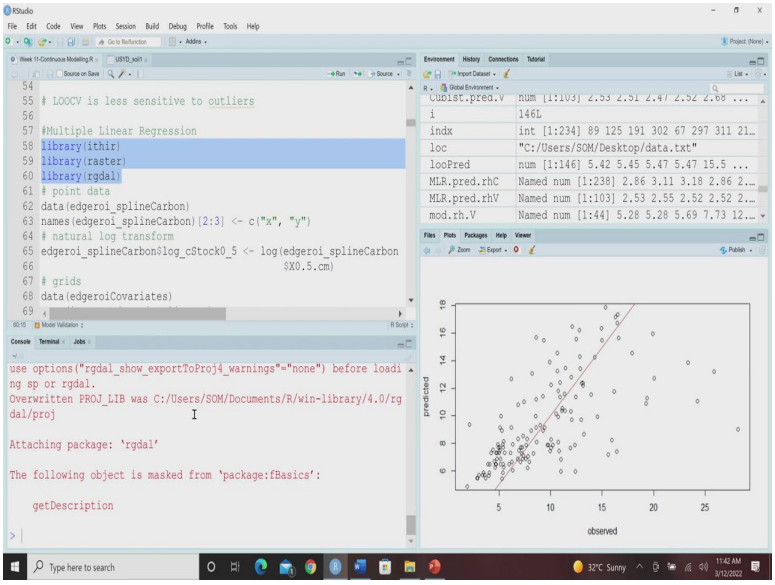


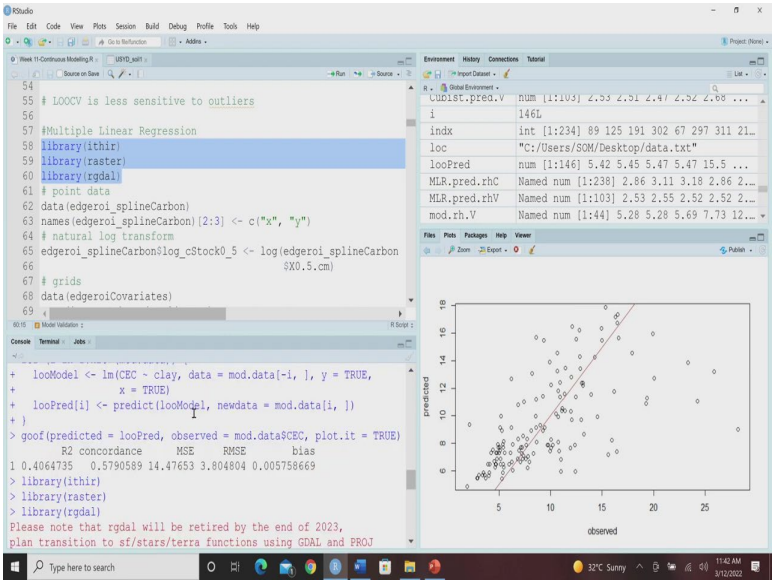
Now, guys you have seen that how we have done the you know model fitting without dividing the data into calibration validation. Now, you have seen how we can do that using the separate training set and testing set. Now, what about doing this model validation using leave one out cross validation. So, this scrip stands for leave one out cross validation. So, you will repeat this step in the steps in loop for each and individual sample and we get the leave one out cross validation prediction parameters.

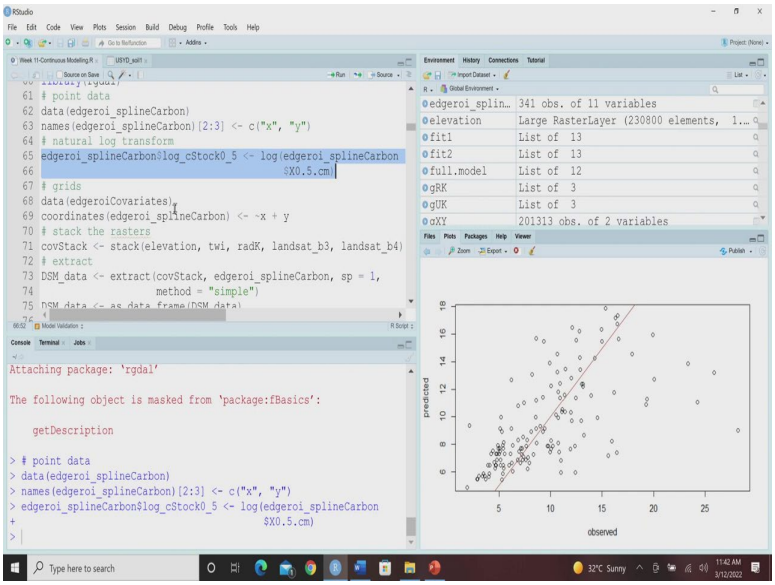
So, I am going to run the script. So, the predicted values for this leave one out cross validation has been already selected has been already executed and we have stored now, again we are using this goof function our predicted is the LW a looPred value observed is mod dot data and specific to CEC and then we are also want to plot it.

So, here are all the 166 observations we have seen and then you can see this is the leave one out cross validation results. So, here you can see the results 0.40, 0.57 is a concordance MSE is 14.47, RMSE is 3.80 and then bias is 0.05. So, remember that that leave one out cross validation is less sensitive to outlets. So, you can use this leave one out cross validation. So, guys, we are done with the simple linear regression, let us go with the multiple linear regression.

(Refer Slide Time: 18:21)

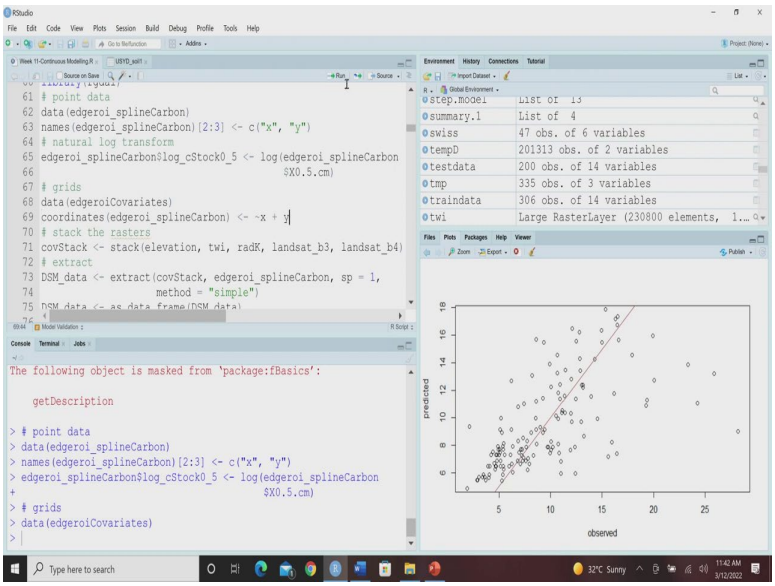


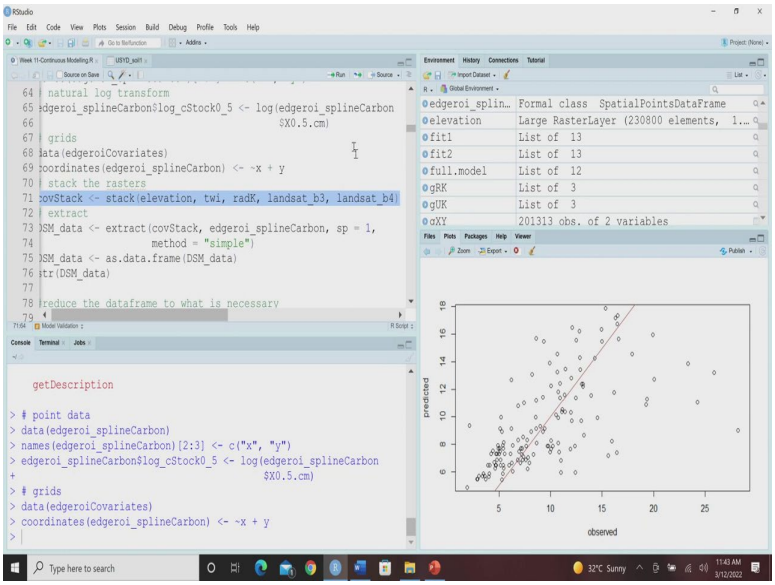


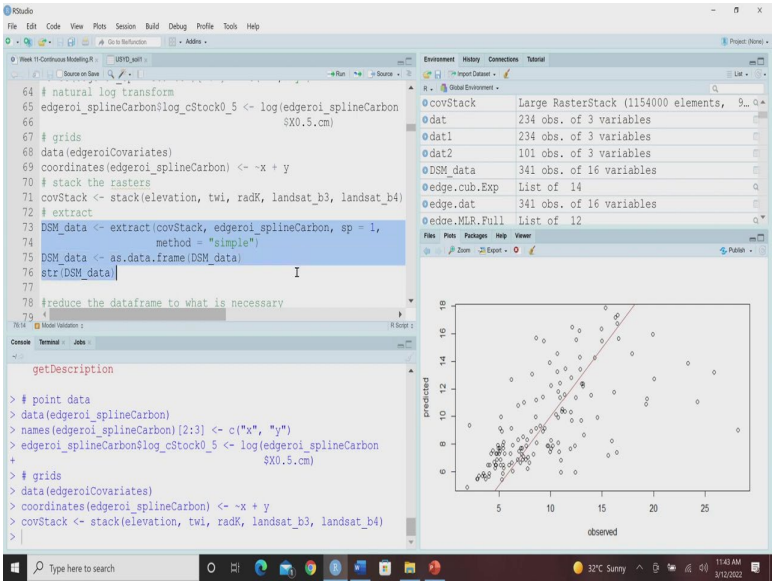


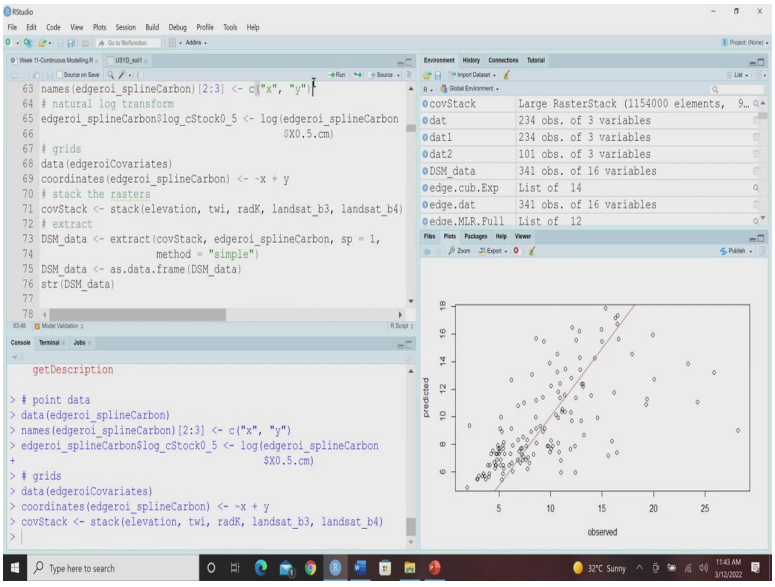
So, again, we are going to run this library, ither library, raster library, rgdl let us call this point data that is that is so, we have already uploaded these packages. Now, let us download the data edgeroi underscore splineCarbon names and then we instruct R that the second and third column are x, y or location, then we do the natural log transformation of the 0 to 5 centimetre organic carbon data, then we need to take the grids.

(Refer Slide Time: 19:02)




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
natural log transform
edgeroi_splineCarbon$log_cStock0_5 <- log(edgeroi_splineCarbon
$X0.5.cm)
# grids
data(edgeroiCovariates)
coordinates(edgeroi_splineCarbon) <- ~x + y
# stack the rasters
covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
# extract
DSM_data <- extract(covStack, edgeroi_splineCarbon, sp = 1,
method = "simple")
DSM_data <- as.data.frame(DSM_data)
str(DSM_data)
#reduce the dataframe to what is necessary
getDescription
> # point data
> data(edgeroi_splineCarbon)
> names(edgeroi_splineCarbon)[2:3] <- c("x", "y")
> edgeroi_splineCarbon$log_cStock0_5 <- log(edgeroi_splineCarbon
+ $X0.5.cm)
> # grids
> data(edgeroiCovariates)
> coordinates(edgeroi_splineCarbon) <- ~x + y
edgeroi_splin... Formal class SpatialPointsDataFrame
elevation Large RasterLayer (230800 elements, 1...
fit1 List of 13
fit2 List of 13
full.model List of 12
gRK List of 3
gUK List of 3
oXY 201313 obs. of 2 variables
predicted
observed
Type here to search
32°C Sunny
11:43 AM
3/12/2022

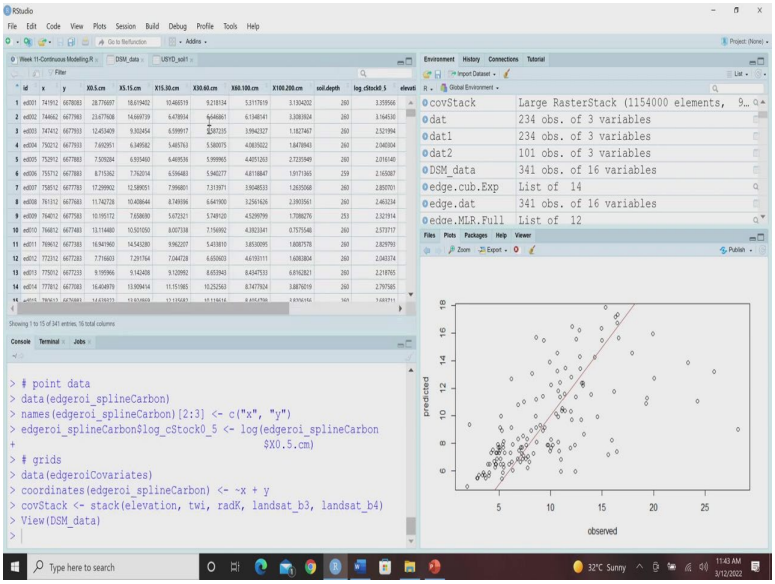

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
natural log transform
edgeroi_splineCarbon$log_cStock0_5 <- log(edgeroi_splineCarbon
$X0.5.cm)
# grids
data(edgeroiCovariates)
coordinates(edgeroi_splineCarbon) <- ~x + y
# stack the rasters
covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
# extract
DSM_data <- extract(covStack, edgeroi_splineCarbon, sp = 1,
method = "simple")
DSM_data <- as.data.frame(DSM_data)
str(DSM_data)
#reduce the dataframe to what is necessary
getDescription
> # point data
> data(edgeroi_splineCarbon)
> names(edgeroi_splineCarbon)[2:3] <- c("x", "y")
> edgeroi_splineCarbon$log_cStock0_5 <- log(edgeroi_splineCarbon
+ $X0.5.cm)
> # grids
> data(edgeroiCovariates)
> coordinates(edgeroi_splineCarbon) <- ~x + y
> covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
covStack Large RasterStack (1154000 elements, 9...
dat 234 obs. of 3 variables
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 16 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
predicted
observed
Type here to search
32°C Sunny
11:43 AM
3/12/2022

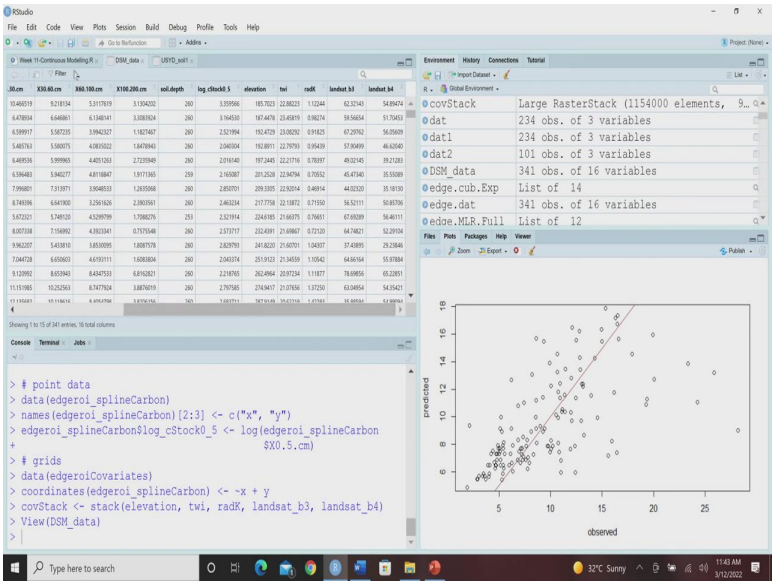

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
names(edgeroi_splineCarbon)[2:3] <- c("x", "y")
# natural log transform
edgeroi_splineCarbon$log_cStock0_5 <- log(edgeroi_splineCarbon
$X0.5.cm)
# grids
data(edgeroiCovariates)
coordinates(edgeroi_splineCarbon) <- ~x + y
# stack the rasters
covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
# extract
DSM_data <- extract(covStack, edgeroi_splineCarbon, sp = 1,
method = "simple")
DSM_data <- as.data.frame(DSM_data)
str(DSM_data)
getDescription
> # point data
> data(edgeroi_splineCarbon)
> names(edgeroi_splineCarbon)[2:3] <- c("x", "y")
> edgeroi_splineCarbon$log_cStock0_5 <- log(edgeroi_splineCarbon
+ $X0.5.cm)
> # grids
> data(edgeroiCovariates)
> coordinates(edgeroi_splineCarbon) <- ~x + y
> covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
covStack Large RasterStack (1154000 elements, 9...
dat 234 obs. of 3 variables
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 16 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
predicted
observed
Type here to search
32°C Sunny
11:43 AM
3/12/2022

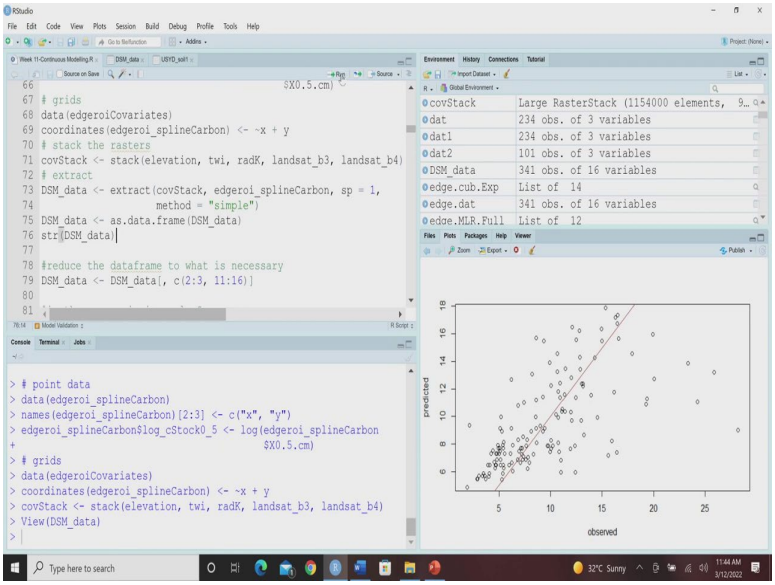
So, for the grids we are going to download the edgeroi covariates we have already seen how to do that then we want to have the coordinate we want to instruct R that these x and y are basically the coordinates. So, we run that now, you can see here you know now it has been considered as a special points data frame it is not a simple data frame right now, these edgeroi underscore splineCarbon dataset.

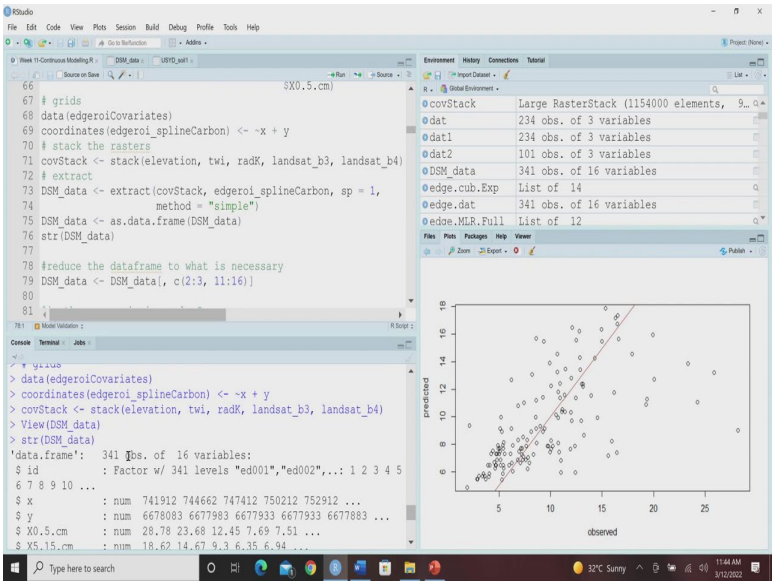
Now, we are doing the stacking of the raster using the stack function. So, again just like previously we are doing the stacking and then we are extracting based on the edgeroi splineCarbon data using simple method. We have already executed this code previously in our previous in our previous lectures. So, we are going to use this… stacked covariate and then we are going to extract the values from the stack covariates for all the five covariates using simple method and then we are creating the whole data frame that is called DSM underscore data.

(Refer Slide Time: 20:10)



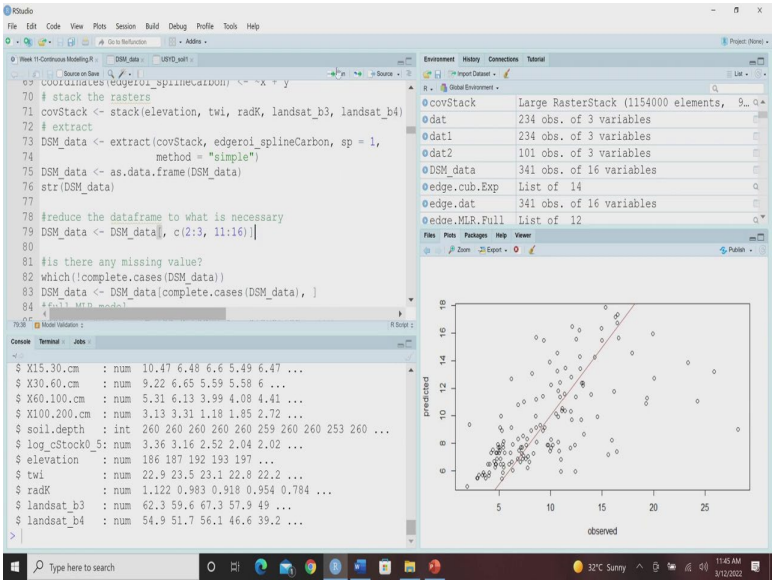

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
> # point data
> data(edgeroi_splineCarbon)
> names(edgeroi_splineCarbon)[2:3] <- c("x", "y")
> edgeroi_splineCarbon$log_cStock0_5 <- log(edgeroi_splineCarbon
+ $X0.5.cm)
> # grids
> data(edgeroiCovariates)
> coordinates(edgeroi_splineCarbon) <- ~x + y
> covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
> View(DSM_data)
covStack Large RasterStack (1154000 elements, 9...
dat 234 obs. of 3 variables
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 16 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
predicted
observed
Type here to search
32°C Sunny
11:43 AM
3/12/2022

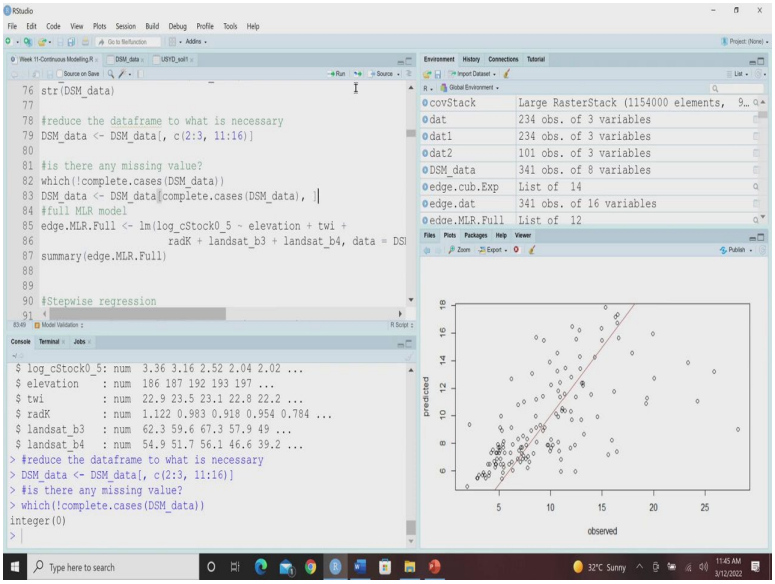

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
66 $X0.5.cm)
67 # grids
68 data(edgeroiCovariates)
69 coordinates(edgeroi_splineCarbon) <- ~x + y
70 # stack the rasters
71 covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
72 # extract
73 DSM_data <- extract(covStack, edgeroi_splineCarbon, sp = 1,
74 method = "simple")
75 DSM_data <- as.data.frame(DSM_data)
76 str(DSM_data)
77
78 #reduce the dataframe to what is necessary
79 DSM_data <- DSM_data[, c(2:3, 11:16)]
80
> # point data
> data(edgeroi_splineCarbon)
> names(edgeroi_splineCarbon)[2:3] <- c("x", "y")
> edgeroi_splineCarbon$log_cStock0_5 <- log(edgeroi_splineCarbon
+ $X0.5.cm)
> # grids
> data(edgeroiCovariates)
> coordinates(edgeroi_splineCarbon) <- ~x + y
> covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
> View(DSM_data)
covStack Large RasterStack (1154000 elements, 9...
dat 234 obs. of 3 variables
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 16 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
predicted
observed
Type here to search
32°C Sunny
11:44 AM
3/12/2022

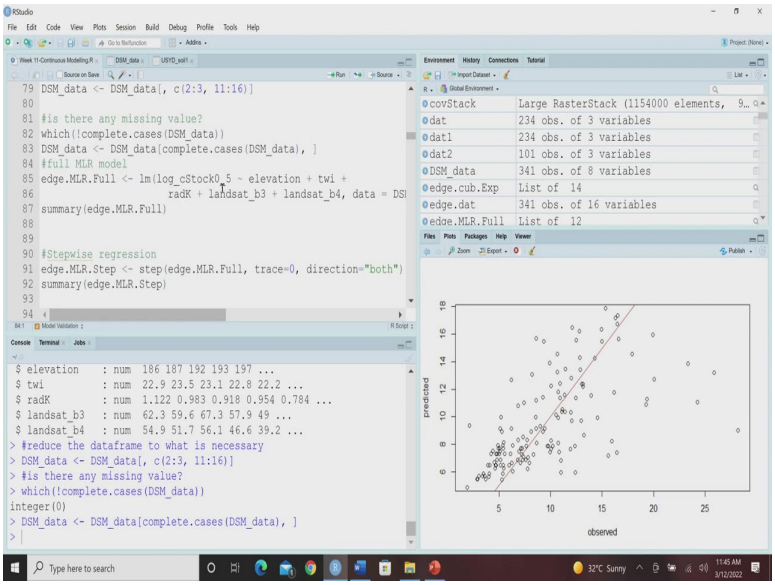

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
66 $X0.5.cm)
67 # grids
68 data(edgeroiCovariates)
69 coordinates(edgeroi_splineCarbon) <- ~x + y
70 # stack the rasters
71 covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
72 # extract
73 DSM_data <- extract(covStack, edgeroi_splineCarbon, sp = 1,
74 method = "simple")
75 DSM_data <- as.data.frame(DSM_data)
76 str(DSM_data)
77
78 #reduce the dataframe to what is necessary
79 DSM_data <- DSM_data[, c(2:3, 11:16)]
80
> data(edgeroiCovariates)
> coordinates(edgeroi_splineCarbon) <- ~x + y
> covStack <- stack(elevation, twi, radK, landsat_b3, landsat_b4)
> View(DSM_data)
> str(DSM_data)
'data.frame': 341 obs. of 16 variables:
$ id : Factor w/ 341 levels "ed001","ed002",..: 1 2 3 4 5
6 7 8 9 10 ...
$ x : num 741912 744662 747412 750212 752912 ...
$ y : num 6678083 6677983 6677933 6677933 6677883 ...
$ X0.5.cm : num 28.78 23.68 12.45 7.69 7.51 ...
$ X5.15.cm : num 18.62 14.67 9.3 6.35 6.94 ...
covStack Large RasterStack (1154000 elements, 9...
dat 234 obs. of 3 variables
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 16 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
predicted
observed
Type here to search
32°C Sunny
11:44 AM
3/12/2022

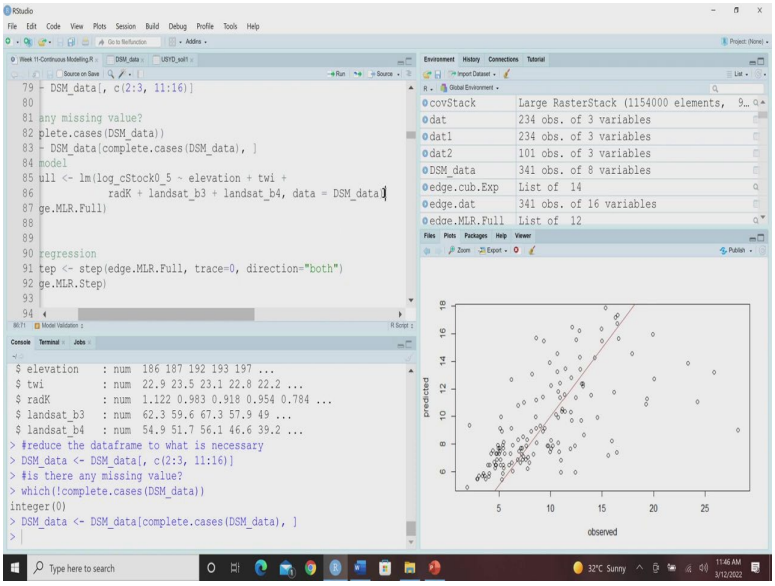
So, this DSM underscore data. So, you can see 341 observation with 16 variables where we have all the value all the variables as well as the all the covariates. So, once our data is ready, now, next step is to see the structure of the data. So, you can see this is the structure of the data, we have these 341 observations of 16 variables.

(Refer Slide Time: 20:37)







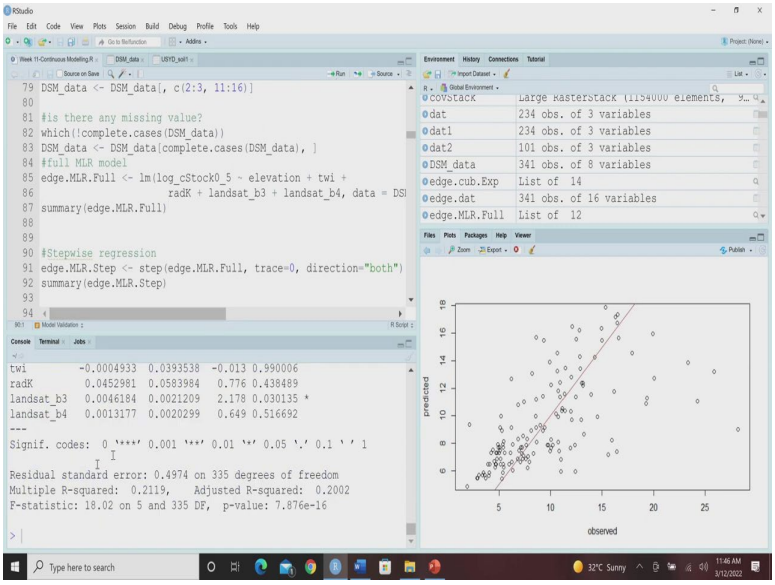


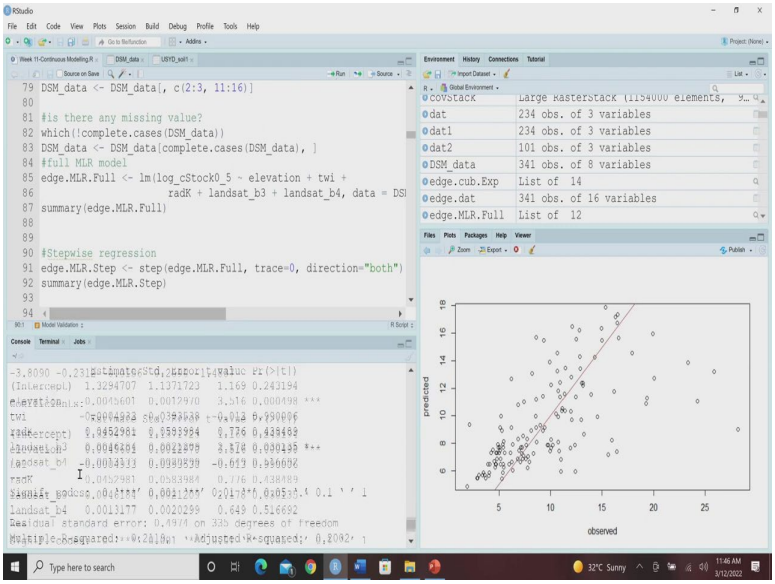
Now, suppose we want to deal with only logarithmic… log converted 0 to 5-centimetre data and only with the covariates. So, we are going to specify the columns, which we are going to keep for further calculations. So, we are going to use another data set called DSM underscore data. So, that keeps all the observation but keeping only the location and context that is x and y and from 11 to 16.

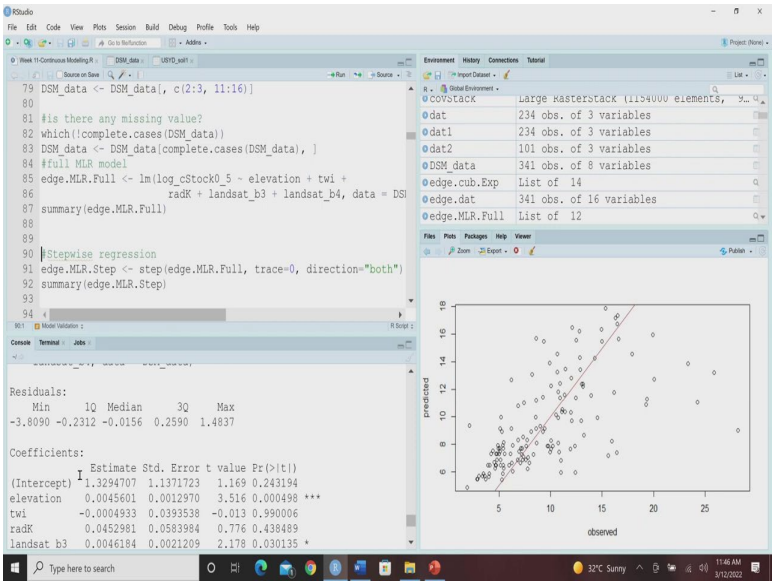
So, you know 11 To 16 stands for these five variable covariates as well as the log converted carbon stock. So, let us run this. So, this reduced data set has been already developed, then we want to keep only the complete cases. So, we want to inquire R that whether there is any missing values or not. So, when we use the exclamatory sign that means it opposite. So, we want to see whether there any missing values in the DSM data or not. So, R returns integer 0 that means there is no missing value.

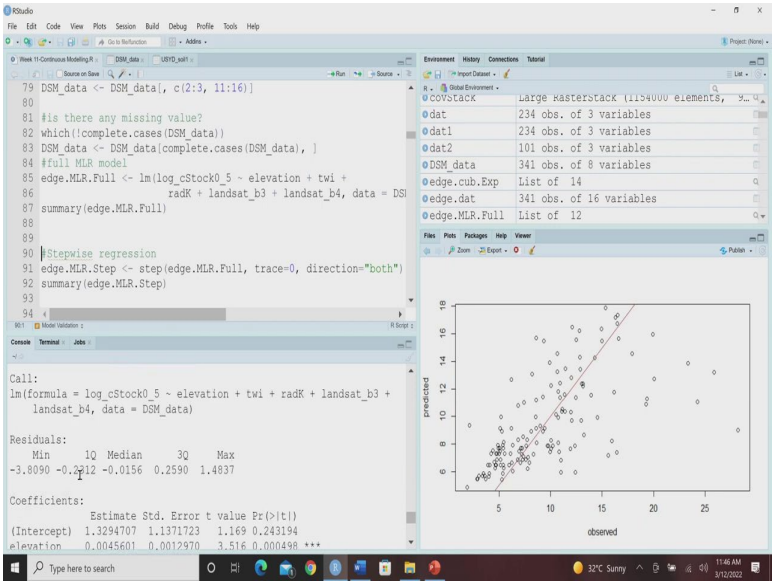
So, anyway, we want to keep all these complete values complete data set in the DSM data and then we are going to keep these DSM complete cases of the DSM data for subsequent modelling. Now, in the full model you can see we are going to use this linear model again just like the simple linear regression. So, here our target is logarithmic converted stock, 0 to 5 centimetre and our predictors are elevation twi, radk, landset b3, landset b4 and our data is the DSM data, let us run it, let us see how it looks like.

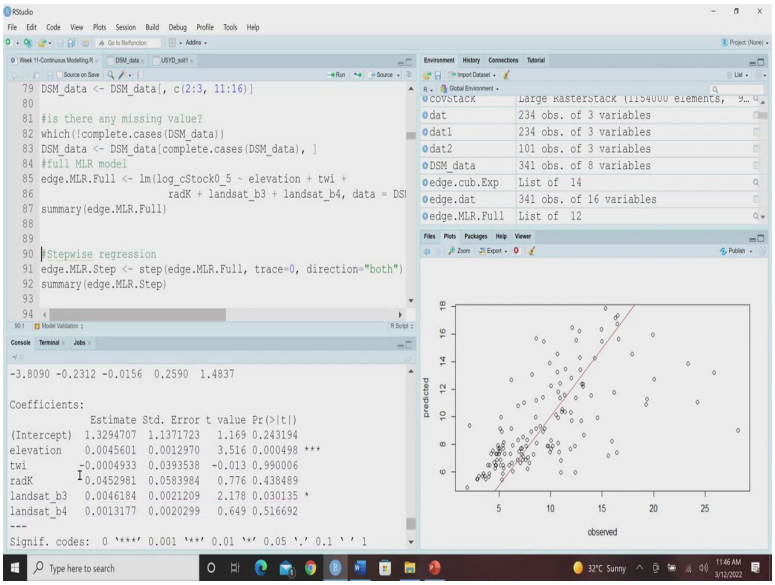
(Refer Slide Time: 22:25)



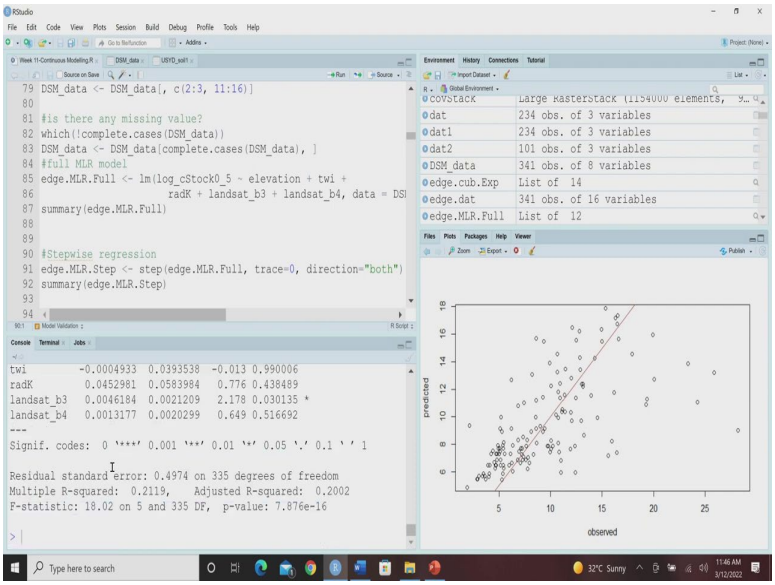



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
DSM_data <- DSM_data[, c(2:3, 11:16)]
#is there any missing value?
which(!complete.cases(DSM_data))
DSM_data <- DSM_data[complete.cases(DSM_data),]
#full MLR model
edge.MLR.Full <- lm(log_cStock0_5 ~ elevation + twi +
radK + landsat_b3 + landsat_b4, data = DSI
summary(edge.MLR.Full)
#Stepwise regression
edge.MLR.Step <- step(edge.MLR.Full, trace=0, direction="both")
summary(edge.MLR.Step)
Residuals:
Min 1Q Median 3Q Max
-3.8090 -0.2312 -0.0156 0.2590 1.4837
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.3294707 1.1371723 1.169 0.243194
elevation 0.0045601 0.0012970 3.516 0.000498 ***
twi -0.0004933 0.0393538 -0.013 0.990006
radK 0.0452981 0.0583984 0.776 0.438489
landsat_b3 0.0046184 0.0021209 2.178 0.030135 *
Environment History Connections Tutorial
covStack Large RasterStack (1154000 elements, 9...
dat 234 obs. of 3 variables
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 8 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
predicted
observed
Type here to search
32°C Sunny
11:46 AM
3/12/2022


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
DSM_data <- DSM_data[, c(2:3, 11:16)]
#is there any missing value?
which(!complete.cases(DSM_data))
DSM_data <- DSM_data[complete.cases(DSM_data),]
#full MLR model
edge.MLR.Full <- lm(log_cStock0_5 ~ elevation + twi +
radK + landsat_b3 + landsat_b4, data = DSI
summary(edge.MLR.Full)
#Stepwise regression
edge.MLR.Step <- step(edge.MLR.Full, trace=0, direction="both")
summary(edge.MLR.Step)
Call:
lm(formula = log_cStock0_5 ~ elevation + twi + radK + landsat_b3 +
landsat_b4, data = DSM_data)
Residuals:
Min 1Q Median 3Q Max
-3.8090 -0.2312 -0.0156 0.2590 1.4837
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.3294707 1.1371723 1.169 0.243194
elevation 0.0045601 0.0012970 3.516 0.000498 ***
Environment History Connections Tutorial
covStack Large RasterStack (1154000 elements, 9...
dat 234 obs. of 3 variables
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 8 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
predicted
observed
Type here to search
32°C Sunny
11:46 AM
3/12/2022

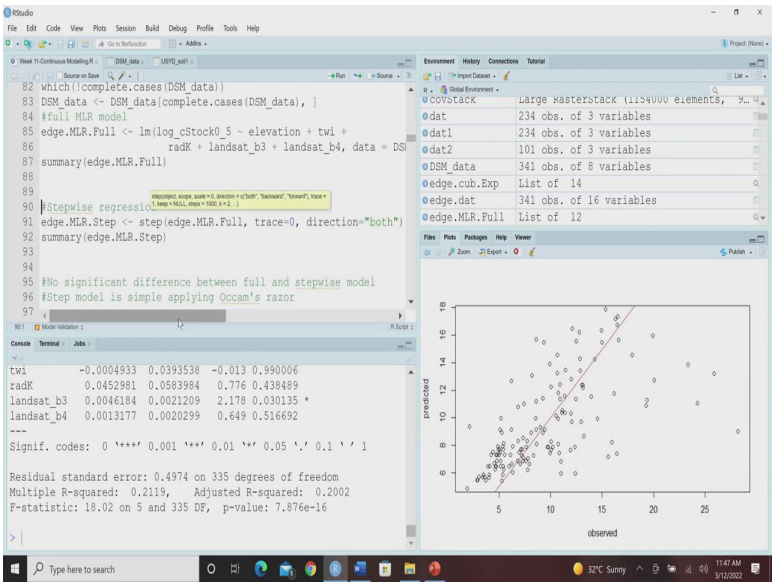

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
DSM_data <- DSM_data[, c(2:3, 11:16)]
#is there any missing value?
which(!complete.cases(DSM_data))
DSM_data <- DSM_data[complete.cases(DSM_data),]
#full MLR model
edge.MLR.Full <- lm(log_cStock0_5 ~ elevation + twi +
radK + landsat_b3 + landsat_b4, data = DSI
summary(edge.MLR.Full)
#Stepwise regression
edge.MLR.Step <- step(edge.MLR.Full, trace=0, direction="both")
summary(edge.MLR.Step)
-3.8090 -0.2312 -0.0156 0.2590 1.4837
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.3294707 1.1371723 1.169 0.243194
elevation 0.0045601 0.0012970 3.516 0.000498 ***
twi -0.0004933 0.0393538 -0.013 0.990006
radK 0.0452981 0.0583984 0.776 0.438489
landsat_b3 0.0046184 0.0021209 2.178 0.030135 *
landsat_b4 0.0013177 0.0020299 0.649 0.516692

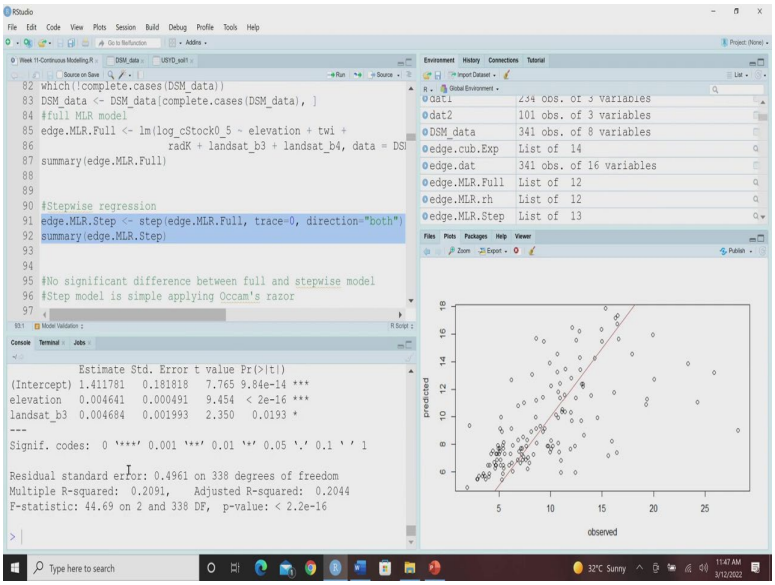
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Environment History Connections Tutorial
covStack Large RasterStack (1154000 elements, 9...
dat 234 obs. of 3 variables
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 8 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
predicted
observed
Type here to search
32°C Sunny
11:46 AM
3/12/2022

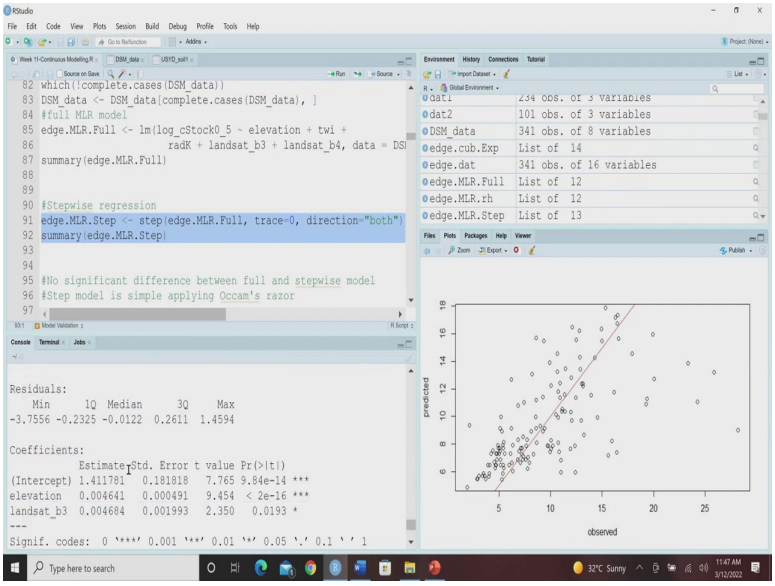


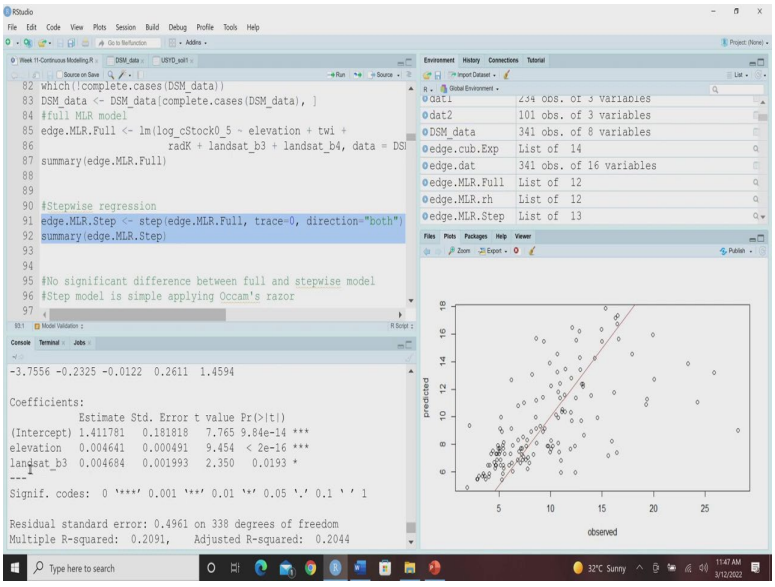
Now, the summary of the full model, you can see this is the summary of the full model, where you can have minimum value maximum value, then first quartile median and third quartile, you can have the coefficient of intercepts and other variables also their standard error their t values and you can have the multiple R square and adjusted R square and their P values also. So, you can get all the information using this.

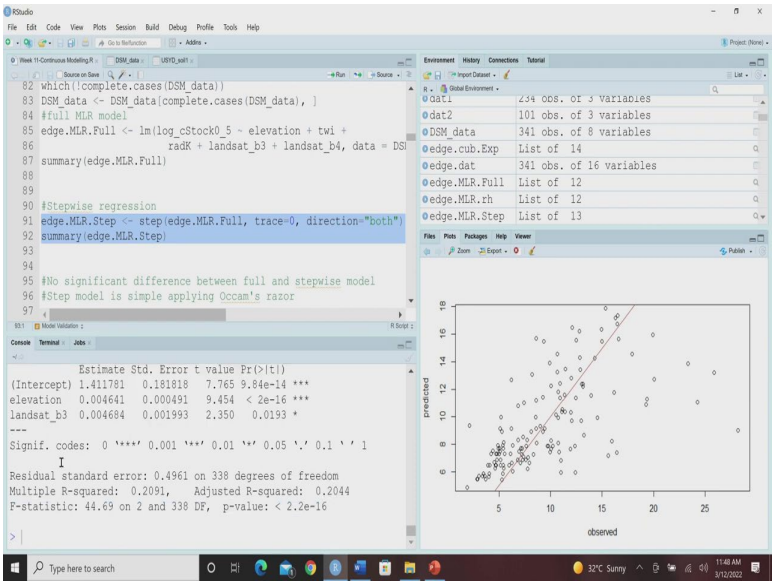
(Refer Slide Time: 22:50)

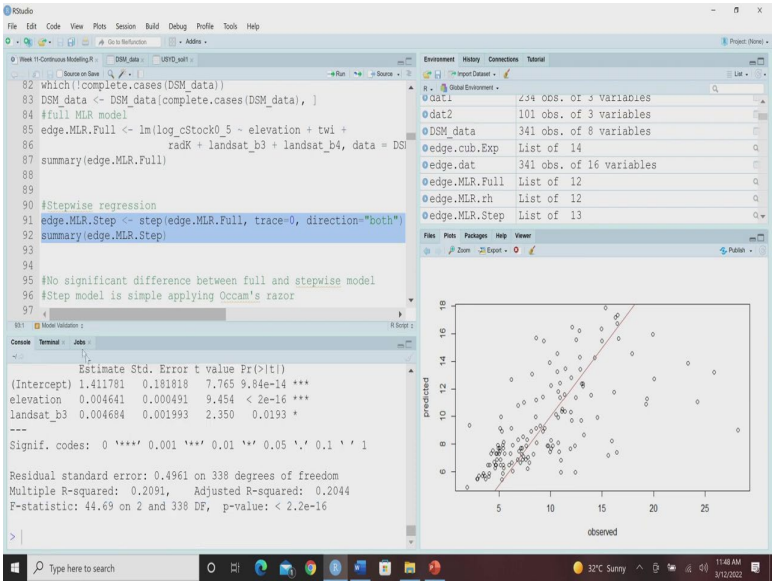



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
82 which(!complete.cases(DSM_data))
83 DSM_data <- DSM_data[complete.cases(DSM_data),]
84 #full MLR model
85 edge.MLR.Full <- lm(log_cStock0_5 ~ elevation + twi +
86 radK + landsat_b3 + landsat_b4, data = DSI
87 summary(edge.MLR.Full)
90 #Stepwise regression
91 edge.MLR.Step <- step(edge.MLR.Full, trace=0, direction="both")
92 summary(edge.MLR.Step)
95 #No significant difference between full and stepwise model
96 #Step model is simple applying Occam's razor
Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.411781 0.181818 7.765 9.84e-14 ***
elevation 0.004641 0.000491 9.454 < 2e-16 ***
landsat_b3 0.004684 0.001993 2.350 0.0193 *
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.4961 on 338 degrees of freedom
Multiple R-squared: 0.2091, Adjusted R-squared: 0.2044
F-statistic: 44.69 on 2 and 338 DF, p-value: < 2.2e-16
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 8 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
edge.MLR.rh List of 12
edge.MLR.Step List of 13
predicted
observed
Type here to search
32°C Sunny
11:47 AM
3/12/2022


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
82 which(!complete.cases(DSM_data))
83 DSM_data <- DSM_data[complete.cases(DSM_data),]
84 #full MLR model
85 edge.MLR.Full <- lm(log_cStock0_5 ~ elevation + twi +
86 radK + landsat_b3 + landsat_b4, data = DSI
87 summary(edge.MLR.Full)
90 #Stepwise regression
91 edge.MLR.Step <- step(edge.MLR.Full, trace=0, direction="both")
92 summary(edge.MLR.Step)
95 #No significant difference between full and stepwise model
96 #Step model is simple applying Occam's razor
Residuals:
Min 1Q Median 3Q Max
-3.7556 -0.2325 -0.0122 0.2611 1.4594
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.411781 0.181818 7.765 9.84e-14 ***
elevation 0.004641 0.000491 9.454 < 2e-16 ***
landsat_b3 0.004684 0.001993 2.350 0.0193 *
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 8 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
edge.MLR.rh List of 12
edge.MLR.Step List of 13
predicted
observed
Type here to search
32°C Sunny
11:47 AM
3/12/2022


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
82 which(!complete.cases(DSM_data))
83 DSM_data <- DSM_data[complete.cases(DSM_data),]
84 #full MLR model
85 edge.MLR.Full <- lm(log_cStock0_5 ~ elevation + twi +
86 radK + landsat_b3 + landsat_b4, data = DSI
87 summary(edge.MLR.Full)
90 #Stepwise regression
91 edge.MLR.Step <- step(edge.MLR.Full, trace=0, direction="both")
92 summary(edge.MLR.Step)
95 #No significant difference between full and stepwise model
96 #Step model is simple applying Occam's razor
-3.7556 -0.2325 -0.0122 0.2611 1.4594
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.411781 0.181818 7.765 9.84e-14 ***
elevation 0.004641 0.000491 9.454 < 2e-16 ***
landsat_b3 0.004684 0.001993 2.350 0.0193 *
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.4961 on 338 degrees of freedom
Multiple R-squared: 0.2091, Adjusted R-squared: 0.2044
dat1 234 obs. of 3 variables
dat2 101 obs. of 3 variables
DSM_data 341 obs. of 8 variables
edge.cub.Exp List of 14
edge.dat 341 obs. of 16 variables
edge.MLR.Full List of 12
edge.MLR.rh List of 12
edge.MLR.Step List of 13
predicted
observed
Type here to search
32°C Sunny
11:47 AM
3/12/2022





Now, here in the multiple linear regression model, we have used all the model parameters, but there is another regression method that is called stepwise regression method. So, in the stepwise regression method, it moves step by step and tries to see whether addition or removal of any variable can significantly affect the model outcome or not.

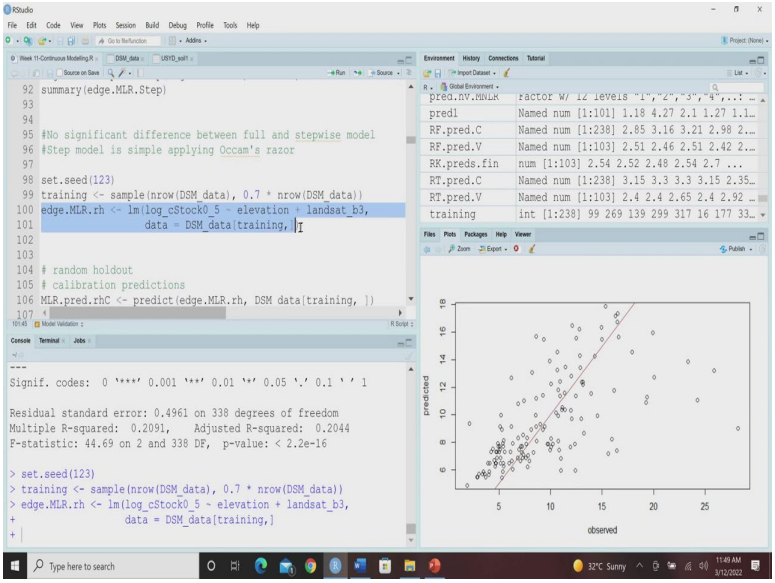
And based on that, it can select the most parsimonious model or simplified model than that of the full multi multiple linear model which keep all the variables. So, here we are going to use the step function and then we are going to use these edge dot MLR dot full models. So, here we have already used these edge dot MLR dot full model and then we want to use it in the both direction.

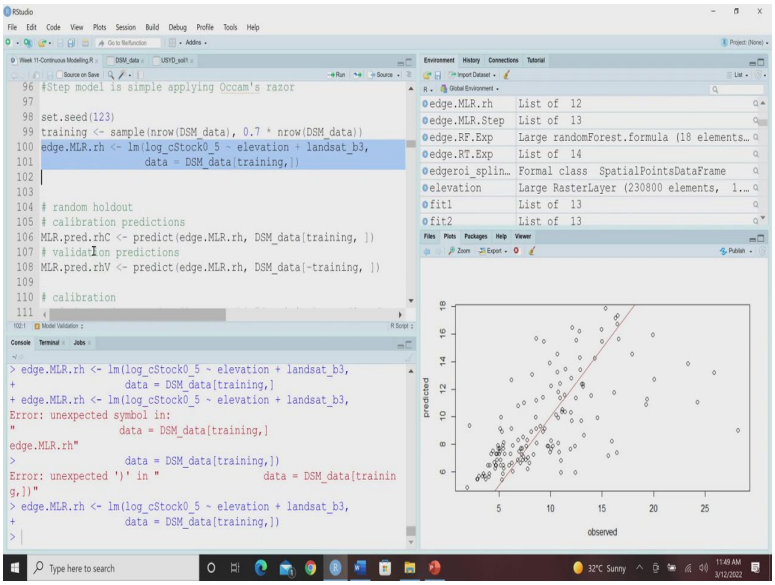
So, the stepwise regression, let us keep this stepwise regression. And then we will see that the stepwise regression will give you this output from this output we can see except you know,

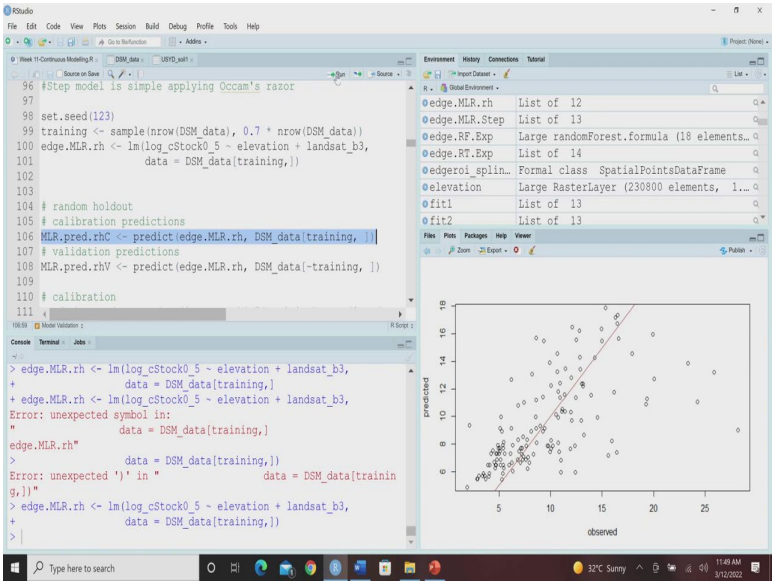
instead of all the covariates the stepwise regression kept only the two covariates like elevation and landset b3 which are giving the significant you know impact which has significant impact on the organic carbon.

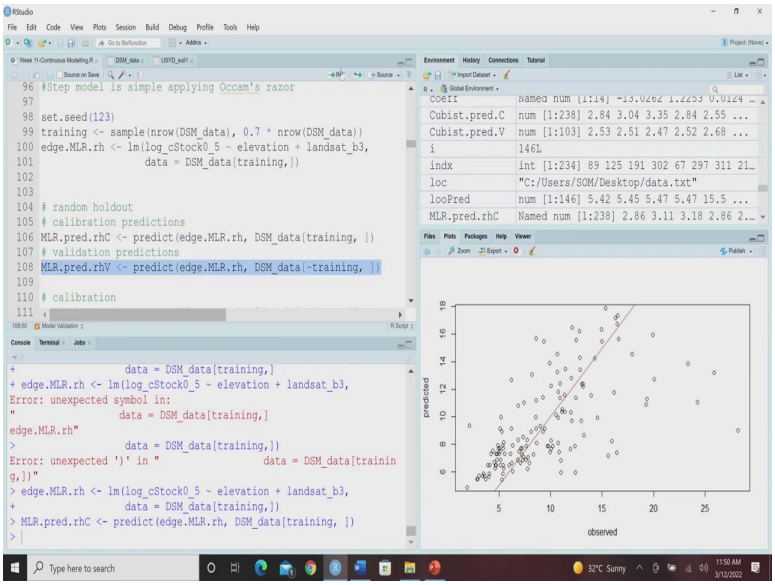
So, the stepwise regression will simplify the model in earlier, in the full model we have kept all the five variables, but in case of stepwise regression, we are getting less number of variables by this individual step by step basis they have selected these two variables. So, you can see here the results from this to… from the full model and the stepwise model. In case of stepwise model, we are getting the adjusted R squared values 0.20 however, in the full model we are getting the adjusted outskirt R squared is again same 0.20. So, we cannot see any difference between the full model In the stepwise model, but step model is more simpler than full model.

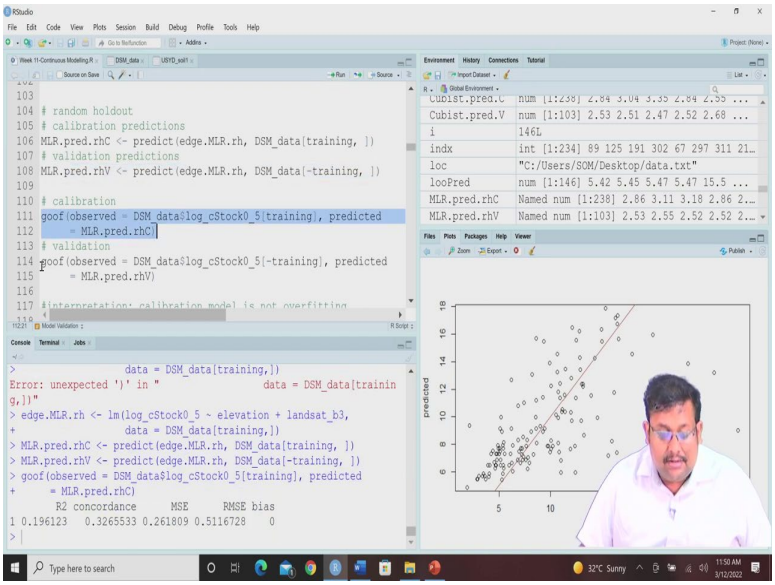
(Refer Slide Time: 25:07)

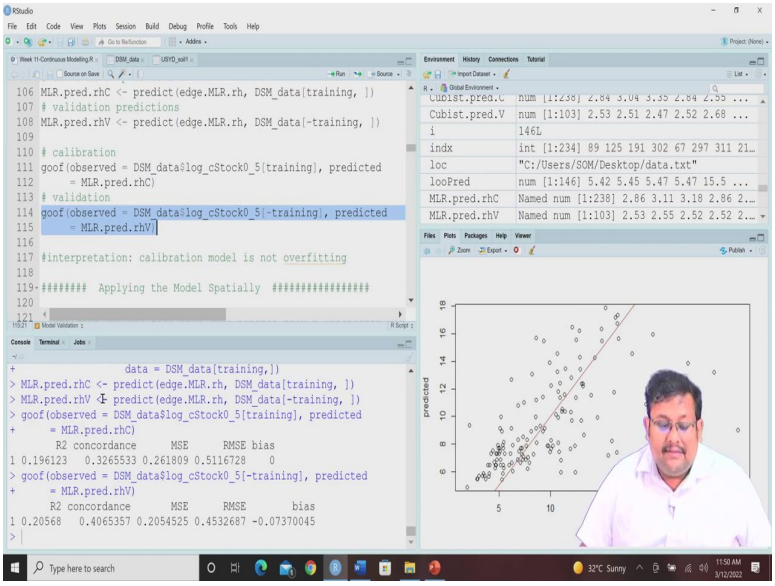



#Step model is simple applying Occam's razor
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
edge.MLR.rh <- lm(log_cStock0_5 ~ elevation + landsat_b3,
data = DSM_data[training,])
# random holdout
# calibration predictions
MLR.pred.rhC <- predict(edge.MLR.rh, DSM_data[training,])
# validation predictions
MLR.pred.rhV <- predict(edge.MLR.rh, DSM_data[-training,])
# calibration
edge.MLR.rh List of 12
edge.MLR.Step List of 13
edge.RF.Exp Large randomForest.formula (18 elements...
edge.RT.Exp List of 14
edgeroi_splin... Formal class SpatialPointsDataFrame
elevation Large RasterLayer (230800 elements, 1....
fit1 List of 13
fit2 List of 13
predicted
observed
Type here to search
32°C Sunny
11:49 AM
3/12/2022


#Step model is simple applying Occam's razor
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
edge.MLR.rh <- lm(log_cStock0_5 ~ elevation + landsat_b3,
data = DSM_data[training,])
# random holdout
# calibration predictions
MLR.pred.rhC <- predict(edge.MLR.rh, DSM_data[training,])
# validation predictions
MLR.pred.rhV <- predict(edge.MLR.rh, DSM_data[-training,])
# calibration
edge.MLR.rh List of 12
edge.MLR.Step List of 13
edge.RF.Exp Large randomForest.formula (18 elements...
edge.RT.Exp List of 14
edgeroi_splin... Formal class SpatialPointsDataFrame
elevation Large RasterLayer (230800 elements, 1....
fit1 List of 13
fit2 List of 13
predicted
observed
Type here to search
32°C Sunny
11:49 AM
3/12/2022


#Step model is simple applying Occam's razor
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
edge.MLR.rh <- lm(log_cStock0_5 ~ elevation + landsat_b3,
data = DSM_data[training,])
# random holdout
# calibration predictions
MLR.pred.rhC <- predict(edge.MLR.rh, DSM_data[training,])
# validation predictions
MLR.pred.rhV <- predict(edge.MLR.rh, DSM_data[-training,])
# calibration
Cubist.pred.C num [1:238] 2.84 3.04 3.35 2.84 2.55 ...
Cubist.pred.V num [1:103] 2.53 2.51 2.47 2.52 2.68 ...
i 146L
indx int [1:234] 89 125 191 302 67 297 311 21...
loc "C:/Users/SOM/Desktop/data.txt"
looPred num [1:146] 5.42 5.45 5.47 5.47 15.5 ...
MLR.pred.rhC Named num [1:238] 2.86 3.11 3.18 2.86 2...
predicted
observed
Type here to search
32°C Sunny
11:50 AM
3/12/2022



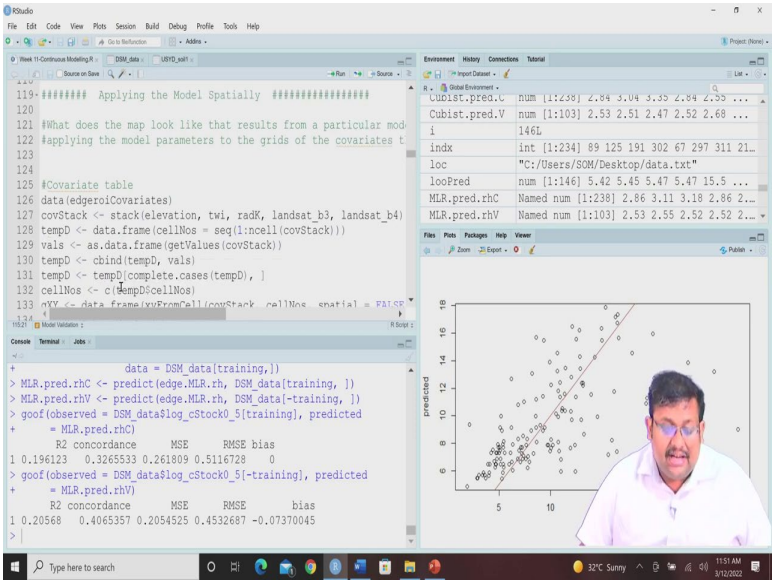


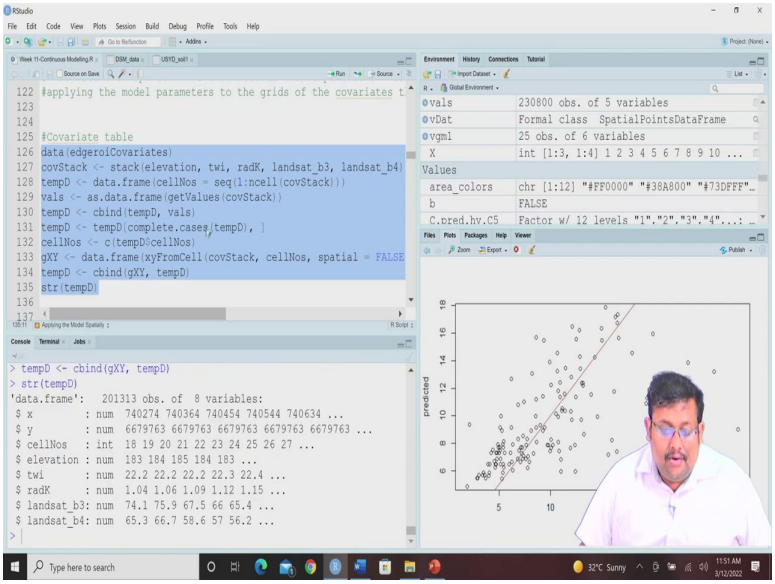
So, according to the Occam's Razor principle, we already we will select the step model then the full model. So, next step is to map based on, next step is to develop the separate training samples by selecting 70 percent of the data, so again we are setting the seed 123, then we are selecting the training samples that is 0 you know 70 percent of the data and then we are fitting this you know model now, here you can see instead of all the five variables, we are keeping only the two variables elevation and landset b3 based on our stepwise model you know output we are keeping only these two and just a minute sorry guys.

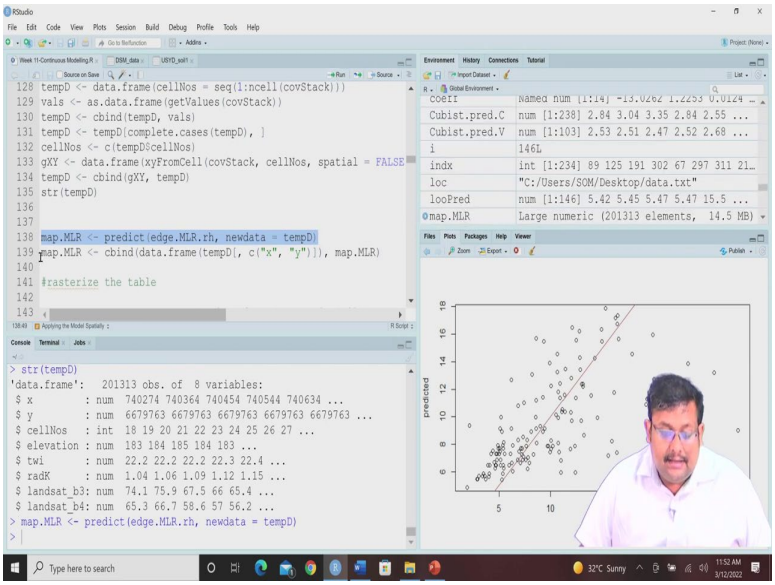
So, then we can also do the random holdout. So, based on the, this calibration data set, we can develop this you know, we can we can predict based on this calibration data set, but at the same time we can predict based on the validation data set. So, validation data set is you can see here this is minus training. So, once you do that, then you can do the goodness of his calculation based on both calibration and also based on variation also separately.

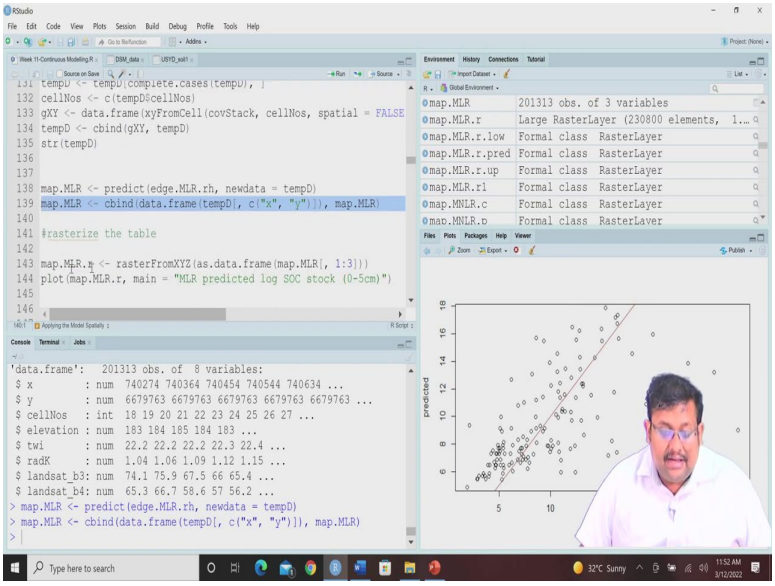
So, you can you can you can compare their results and you can see from this result is the calibration R squared 0.19 or almost 0.20 validation R squared is also 0.20 RMSE 0.51 here 0.45 so, we can see there is no the difference, a significant difference between the calibration model statistics and validation model statistics. So, we can infer that there is no you know, there is no overfitting because in case of overfitting this calibration model performs over optimistically than the validation model.

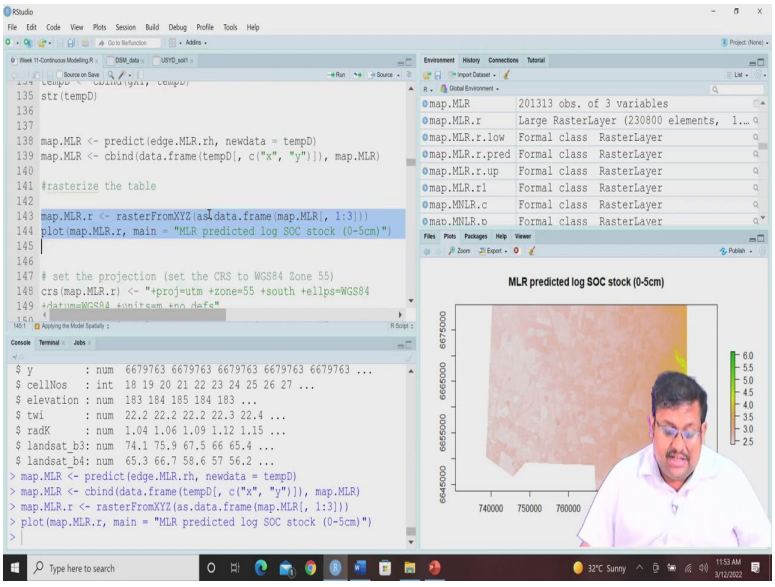
(Refer Slide Time: 27:18)






RStudio
tempD <- data.frame(cellNos = seq(1:ncell(covStack)))
vals <- as.data.frame(getValues(covStack))
tempD <- cbind(tempD, vals)
tempD <- tempD[complete.cases(tempD),]
cellNos <- c(tempD$cellNos)
gXY <- data.frame(xyFromCell(covStack, cellNos, spatial = FALSE
tempD <- cbind(gXY, tempD)
str(tempD)
map.MLR <- predict(edge.MLR.rh, newdata = tempD)
map.MLR <- cbind(data.frame(tempD[, c("x", "y")]), map.MLR)
#rasterize the table
> str(tempD)
'data.frame': 201313 obs. of 8 variables:
$ x : num 740274 740364 740454 740544 740634 ...
$ y : num 6679763 6679763 6679763 6679763 6679763 ...
$ cellNos : int 18 19 20 21 22 23 24 25 26 27 ...
$ elevation : num 183 184 185 184 183 ...
$ twi : num 22.2 22.2 22.2 22.3 22.4 ...
$ radK : num 1.04 1.06 1.09 1.12 1.15 ...
$ landsat_b3: num 74.1 75.9 67.5 66 65.4 ...
$ landsat_b4: num 65.3 66.7 58.6 57 56.2 ...
> map.MLR <- predict(edge.MLR.rh, newdata = tempD)
Coeff Named num [1:14] -13.0262 1.2253 0.0124 ...
Cubist.pred.C num [1:238] 2.84 3.04 3.35 2.84 2.55 ...
Cubist.pred.V num [1:103] 2.53 2.51 2.47 2.52 2.68 ...
i 146L
indx int [1:234] 89 125 191 302 67 297 311 21...
loc "C:/Users/SOM/Desktop/data.txt"
looPred num [1:146] 5.42 5.45 5.47 5.47 15.5 ...
map.MLR Large numeric (201313 elements, 14.5 MB)
predicted
Type here to search
32°C Sunny
11:52 AM
3/12/2022


RStudio
tempD <- tempD[complete.cases(tempD),]
cellNos <- c(tempD$cellNos)
gXY <- data.frame(xyFromCell(covStack, cellNos, spatial = FALSE
tempD <- cbind(gXY, tempD)
str(tempD)
map.MLR <- predict(edge.MLR.rh, newdata = tempD)
map.MLR <- cbind(data.frame(tempD[, c("x", "y")]), map.MLR)
#rasterize the table
map.MLR.r <- rasterFromXYZ(as.data.frame(map.MLR[, 1:3]))
plot(map.MLR.r, main = "MLR predicted log SOC stock (0-5cm)")
'data.frame': 201313 obs. of 8 variables:
$ x : num 740274 740364 740454 740544 740634 ...
$ y : num 6679763 6679763 6679763 6679763 6679763 ...
$ cellNos : int 18 19 20 21 22 23 24 25 26 27 ...
$ elevation : num 183 184 185 184 183 ...
$ twi : num 22.2 22.2 22.2 22.3 22.4 ...
$ radK : num 1.04 1.06 1.09 1.12 1.15 ...
$ landsat_b3: num 74.1 75.9 67.5 66 65.4 ...
$ landsat_b4: num 65.3 66.7 58.6 57 56.2 ...
> map.MLR <- predict(edge.MLR.rh, newdata = tempD)
> map.MLR <- cbind(data.frame(tempD[, c("x", "y")]), map.MLR)
map.MLR 201313 obs. of 3 variables
map.MLR.r Large RasterLayer (230800 elements, 1....
map.MLR.r.low Formal class RasterLayer
map.MLR.r.pred Formal class RasterLayer
map.MLR.r.up Formal class RasterLayer
map.MLR.r1 Formal class RasterLayer
map.MNLR.c Formal class RasterLayer
map.MNLR.p Formal class RasterLayer
predicted
Type here to search
32°C Sunny
11:52 AM
3/12/2022

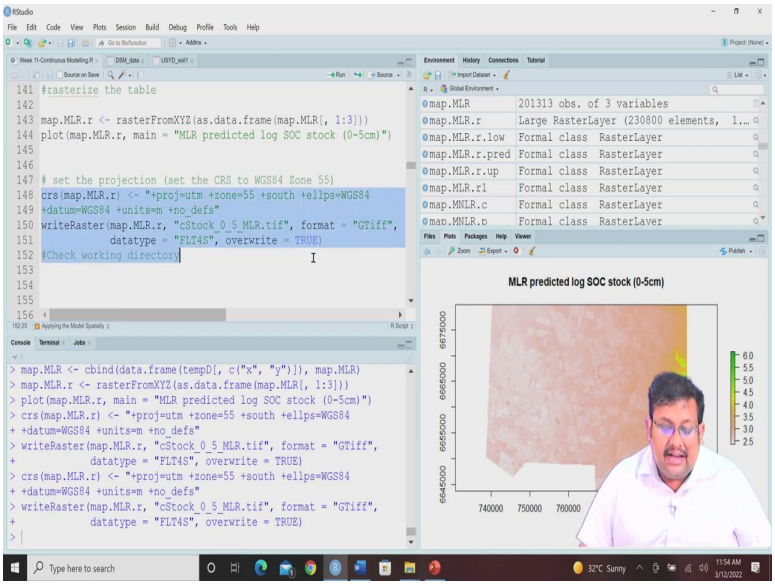

RStudio
str(tempD)
map.MLR <- predict(edge.MLR.rh, newdata = tempD)
map.MLR <- cbind(data.frame(tempD[, c("x", "y")]), map.MLR)
#rasterize the table
map.MLR.r <- rasterFromXYZ(as.data.frame(map.MLR[, 1:3]))
plot(map.MLR.r, main = "MLR predicted log SOC stock (0-5cm)")
# set the projection (set the CRS to WGS84 Zone 55)
crs(map.MLR.r) <- "+proj=utm +zone=55 +south +ellps=WGS84
+datum=WGS84 +units=m +no_defs"
$ y : num 6679763 6679763 6679763 6679763 6679763 ...
$ cellNos : int 18 19 20 21 22 23 24 25 26 27 ...
$ elevation : num 183 184 185 184 183 ...
$ twi : num 22.2 22.2 22.2 22.3 22.4 ...
$ radK : num 1.04 1.06 1.09 1.12 1.15 ...
$ landsat_b3: num 74.1 75.9 67.5 66 65.4 ...
$ landsat_b4: num 65.3 66.7 58.6 57 56.2 ...
> map.MLR <- predict(edge.MLR.rh, newdata = tempD)
> map.MLR <- cbind(data.frame(tempD[, c("x", "y")]), map.MLR)
> map.MLR.r <- rasterFromXYZ(as.data.frame(map.MLR[, 1:3]))
> plot(map.MLR.r, main = "MLR predicted log SOC stock (0-5cm)")
map.MLR 201313 obs. of 3 variables
map.MLR.r Large RasterLayer (230800 elements, 1....
map.MLR.r.low Formal class RasterLayer
map.MLR.r.pred Formal class RasterLayer
map.MLR.r.up Formal class RasterLayer
map.MLR.r1 Formal class RasterLayer
map.MNLR.c Formal class RasterLayer
map.MNLR.p Formal class RasterLayer
MLR predicted log SOC stock (0-5cm)
6675000
6665000
6655000
6645000
740000
750000
760000
6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
Type here to search
32°C Sunny
11:53 AM
3/12/2022

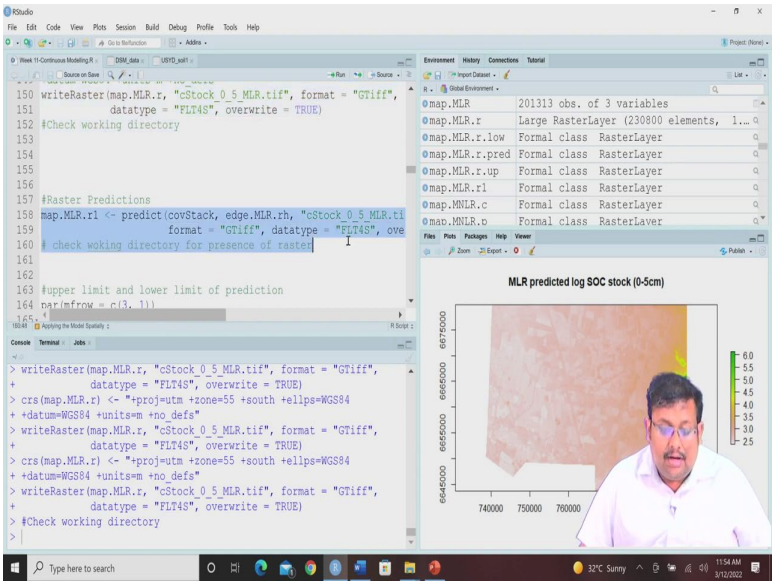
Now, if we want to if we want to apply this model for producing the map specially, then again, we have to run the script again for downloading the data using stacking of the data and then creating the data frame, selecting the cells for which we had the values and extracting those values and extracting their location. And so, once we have it this you know this is the temporary file which we have created with the, their location.

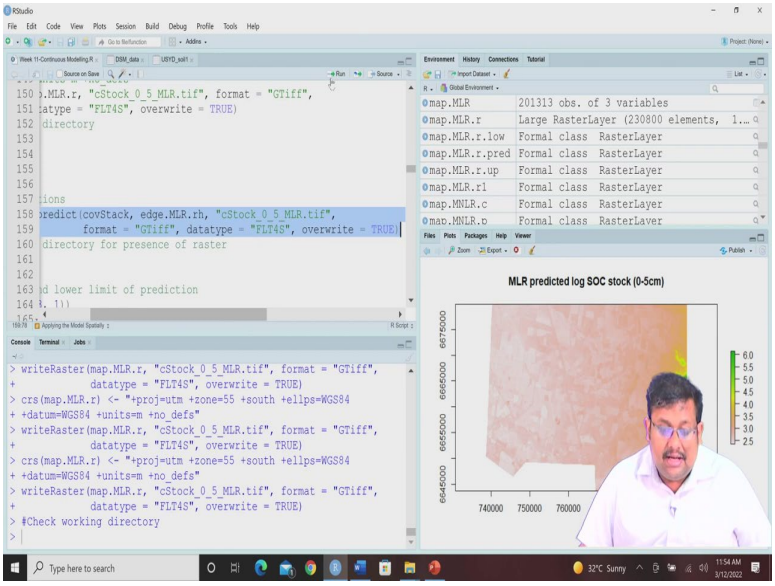
So, g x y is basically you know the location x and y and the temporary data file you can see here the cell numbers for which we have this location and their elevation and twi, radk, landset b3, landset b4 values and then we want to map based on this MLR model and then so, then columbine, this dataset.

And then finally, we are going to raster from the x, y, z from this data set and then we are going to plot so, this is the MLR predicted logarithmic SOC stock for 0 to 5 centimetre. So, this is how guys you can produce the map using the full model or stepwise model using this and also you can check the model performance based on the you know the model performance by either simple holdout or random holdout and also by leave one out cross validation.

(Refer Slide Time: 29:18)



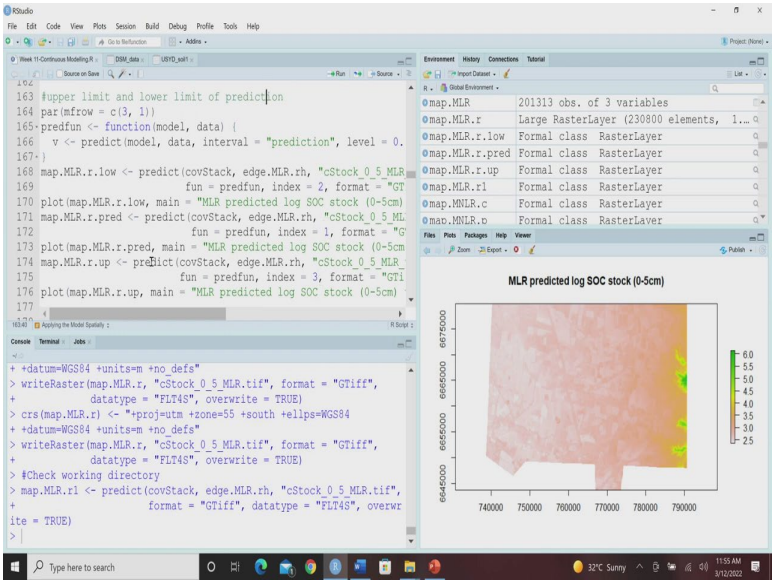


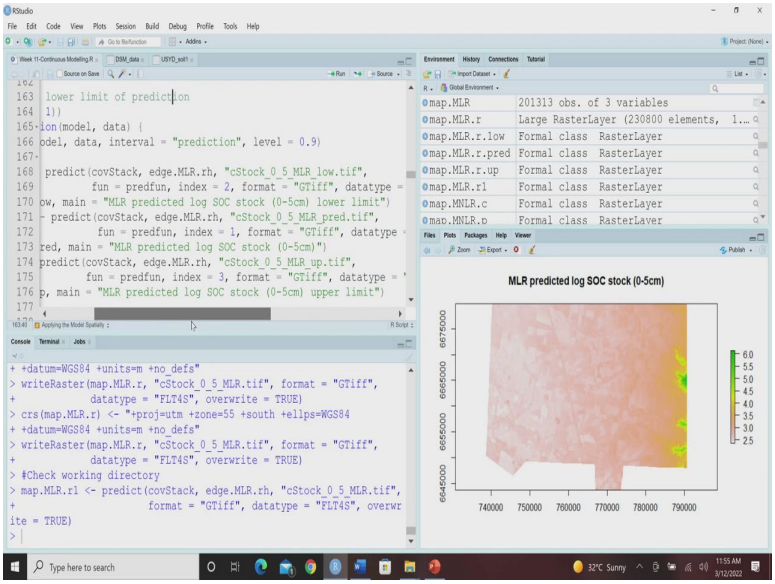


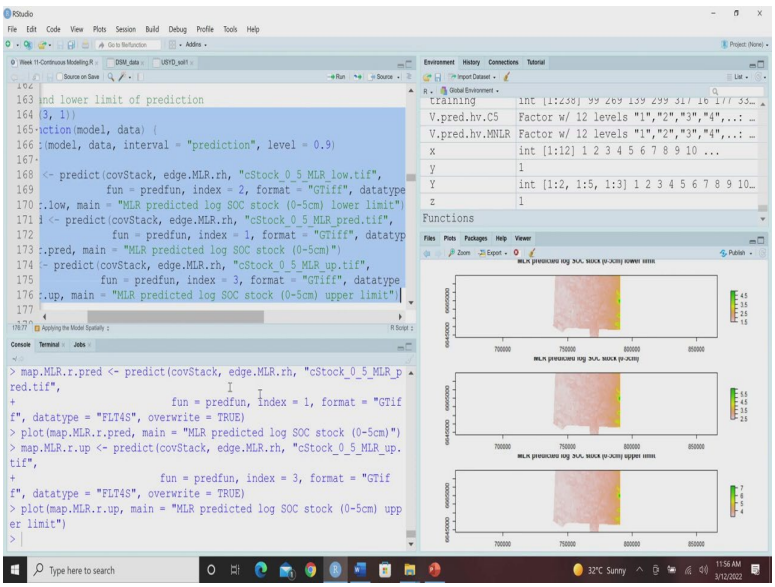
Now, once we have produced this project, you know produce this map next, we are going to set the projections, coordinate reference system as well as you know they are so here you can see we are setting the coordinate reference system as UTM zone 55 south with the WGS 84. And we can we can create you know we can we can write the raster also which will be saved in our working directory.

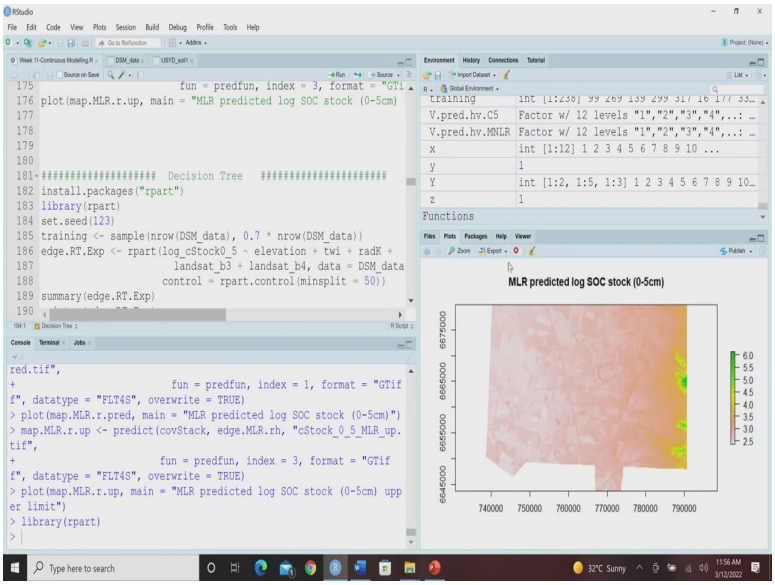
So, this is how we do this. And so, we can we can also do the rest of predictions So, we can produce this and you can, you can see that raster file will be created in your working folder and then the upper limit and the lower limit of the prediction. So, if we want to see the, what is the upper limit of your prediction, what is the lower limit of the prediction, you can also do that by running the script and you can see in the script, they are all basically the same, but we are changing the index argument from 2, 1 and 3.

(Refer Slide Time: 30:38)



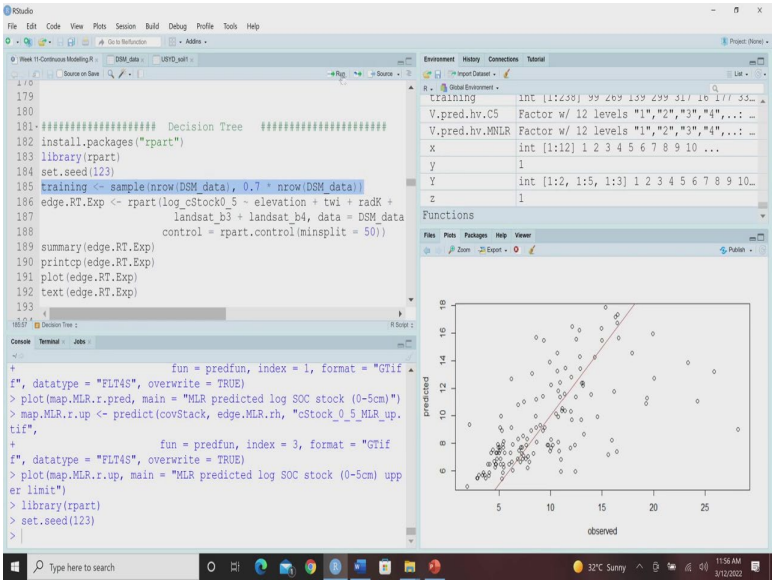


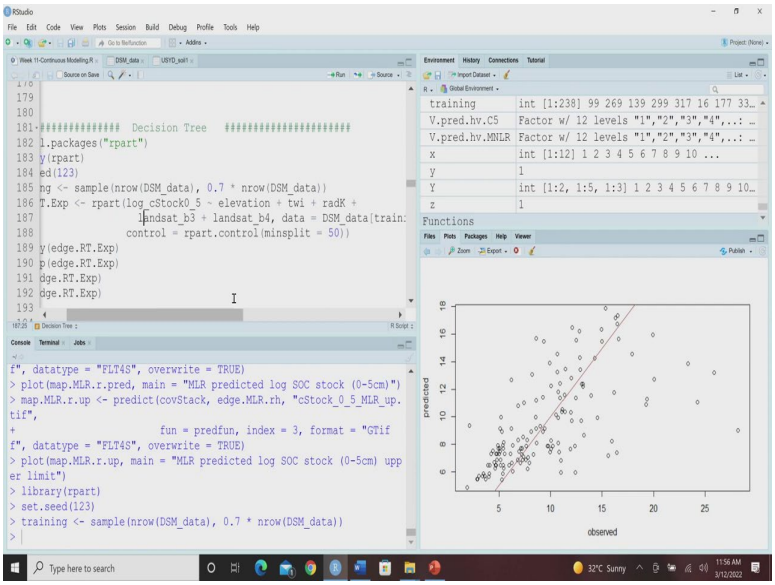


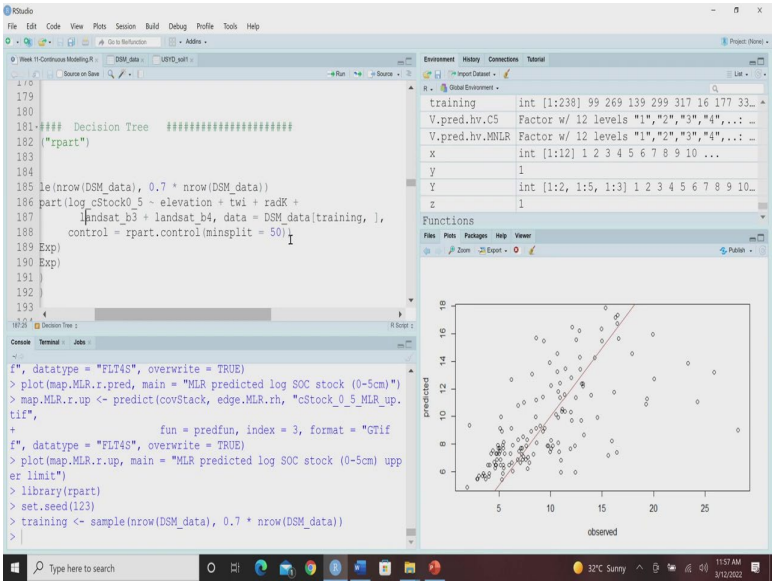


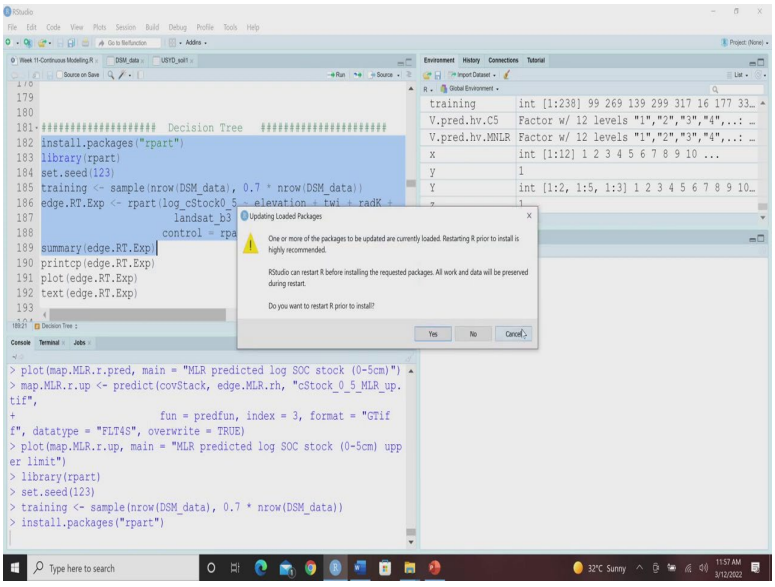
So, 2 1 and 3 one will give you the you know, the one will give you the predicted values and when the 2 and 3 will give you the lower limit and the upper limit of the prediction. So, let us run this script and you can see that 3 maps will be produced with the lower limit and original prediction and the upper limit. So, this is how you can produce the map there is for using the multiple linear regression. And for decision trees you know, you need to install this package called rpart and I have already installed so, I am not going to install it further.

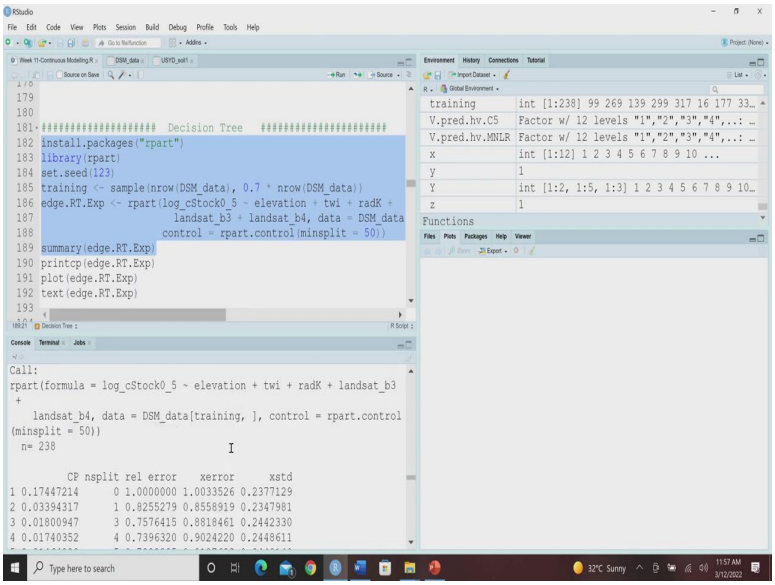
(Refer Slide Time: 31:28)




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Week 11-Continuous Modelling.R
DSM_data
USYD_soil1
Source on Save
Run
Source
Environment History Connections Tutorial
Import Dataset
Global Environment
training int [1:238] 99 269 139 299 317 16 177 33...
V.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",...: ...
V.pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",...: ...
x int [1:12] 1 2 3 4 5 6 7 8 9 10 ...
y 1
Y int [1:2, 1:5, 1:3] 1 2 3 4 5 6 7 8 9 10...
z 1
Functions
Files Plots Packages Help Viewer
Zoom
Export
Publish
predicted
observed
Console Terminal Jobs
f", datatype = "FLT4S", overwrite = TRUE)
> plot(map.MLR.r.pred, main = "MLR predicted log SOC stock (0-5cm)")
> map.MLR.r.up <- predict(covStack, edge.MLR.rh, "cStock_0_5_MLR_up.
tif",
+ fun = predfun, index = 3, format = "GTif
f", datatype = "FLT4S", overwrite = TRUE)
> plot(map.MLR.r.up, main = "MLR predicted log SOC stock (0-5cm) upp
er limit")
> library(rpart)
> set.seed(123)
> training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
>
Type here to search
32°C Sunny
11:56 AM
3/12/2022


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Environment History Connections Tutorial
training int [1:238] 99 269 139 299 317 16 177 33...
V.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",...: ...
V.pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",...: ...
x int [1:12] 1 2 3 4 5 6 7 8 9 10 ...
y 1
Y int [1:2, 1:5, 1:3] 1 2 3 4 5 6 7 8 9 10...
z 1
Functions
predicted
observed
Console Terminal Jobs
f", datatype = "FLT4S", overwrite = TRUE)
> plot(map.MLR.r.pred, main = "MLR predicted log SOC stock (0-5cm)")
> map.MLR.r.up <- predict(covStack, edge.MLR.rh, "cStock_0_5_MLR_up.
tif",
+ fun = predfun, index = 3, format = "GTif
f", datatype = "FLT4S", overwrite = TRUE)
> plot(map.MLR.r.up, main = "MLR predicted log SOC stock (0-5cm) upp
er limit")
> library(rpart)
> set.seed(123)
> training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
>
Type here to search
32°C Sunny
11:57 AM
3/12/2022


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Environment History Connections Tutorial
training int [1:238] 99 269 139 299 317 16 177 33...
V.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",...: ...
V.pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",...: ...
x int [1:12] 1 2 3 4 5 6 7 8 9 10 ...
y 1
Y int [1:2, 1:5, 1:3] 1 2 3 4 5 6 7 8 9 10...
install.packages("rpart")
library(rpart)
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
summary(edge.RT.Exp)
printcp(edge.RT.Exp)
plot(edge.RT.Exp)
text(edge.RT.Exp)
Updating Loaded Packages
One or more of the packages to be updated are currently loaded. Restarting R prior to install is highly recommended.
RStudio can restart R before installing the requested packages. All work and data will be preserved during restart.
Do you want to restart R prior to install?
Yes
No
Cancel
Console Terminal Jobs
> plot(map.MLR.r.pred, main = "MLR predicted log SOC stock (0-5cm)")
> map.MLR.r.up <- predict(covStack, edge.MLR.rh, "cStock_0_5_MLR_up.
tif",
+ fun = predfun, index = 3, format = "GTif
f", datatype = "FLT4S", overwrite = TRUE)
> plot(map.MLR.r.up, main = "MLR predicted log SOC stock (0-5cm) upp
er limit")
> library(rpart)
> set.seed(123)
> training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
> install.packages("rpart")
Type here to search
32°C Sunny
11:57 AM
3/12/2022



And then you can call this library rpart and then again we are setting the seed that is 123 and then we are selecting the training samples randomly 70 percent just like before and then we are going to use these rpart function here we are trying to predict the log converted 0 to 5 centimetre data organic carbon data with elevation twi, radk, landset b3, landset b4 our data is DSM data with the training dataset.

Here one thing you can see we are using a… an argument that is control, which is rpart dot control minimum split equal to 50. So, that means we are instructing r that you should not go for further splitting the node if the minimum sample is less than 50. So, in this way we can we can we can control the overfitting of the data because, if it is less than 50 You know, they are might be some kind of overfitting. So, you can play with the number we can change the number at the minimum number of samples to be to be further splitted.

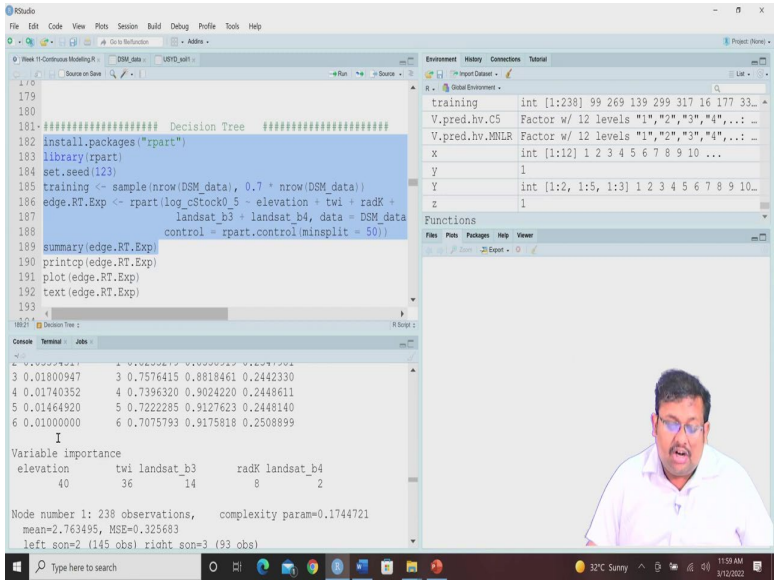
So, let us run this and you can see the summary when you let me just cancel it, so you can see the summary of the results also. So, it will give you the all the details, so this is the formula of and also you can see here the CP parameter also the number of split and also the relative error then x error and then x standard.

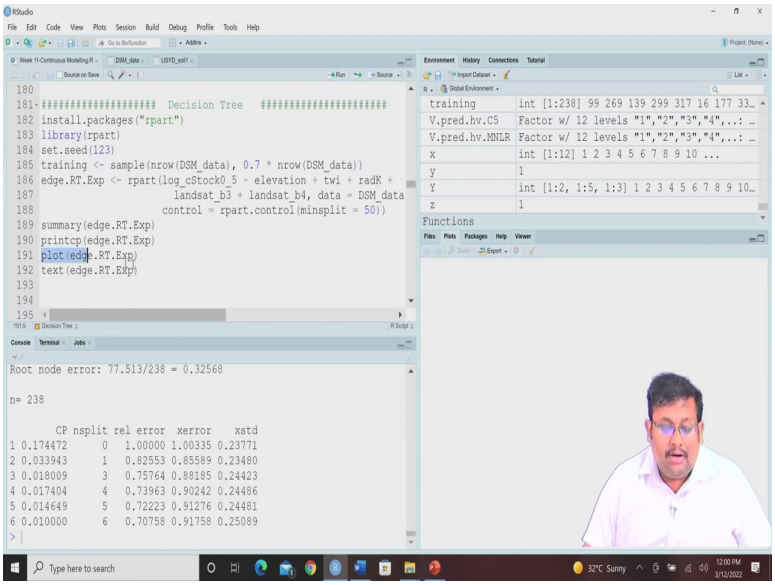
So, this x error is the cross-validation error, rpart basically has a built-in cross validation. So, there is a thumb rule that you select the CP value for which that minimize this X error, so this is one thumb rule another thumb rule is you can choose the lowest level, so you can see the lowest level, so you can see the… as the number of splits are increasing that means, the model is becoming more and more complex.

So, you can choose where to stop and where to prune to prevent the overfitting of the model. So, one thumb rule is you select the CP for which you have the you minimise the x error. Another thumb rule is you choose the lowest level for which these rel error plus e standard you know the summation of both of them will be less than this e error.

So, based on these model values, you can select the optimum number of split for which you should run the model and you can prevent the overfitting and also the variable importance you can see here elevation has the maximum variable importance followed by twi, the landset b3, radk and landset b2. So, from there you can have any idea about how much important those variables are for predicting the carbon stock of logarithmic converted carbon stock for 0 to 5 centimetre.

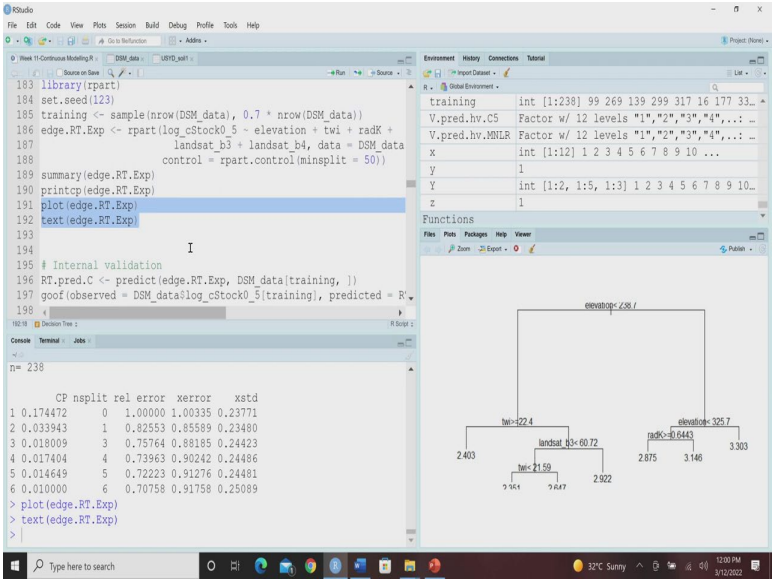
(Refer Slide Time: 34:52)

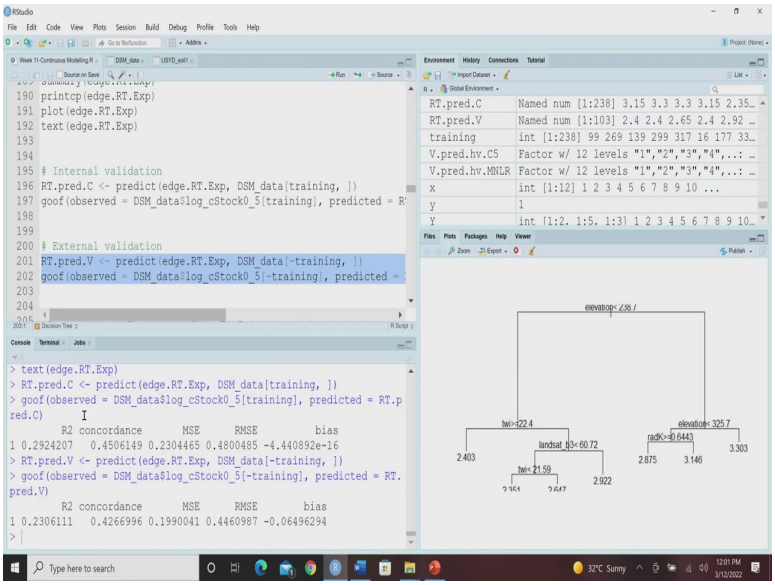


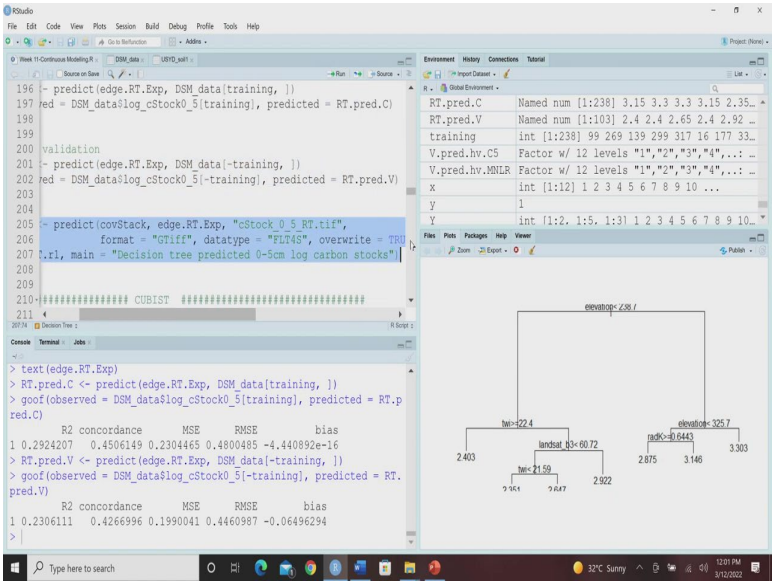


Then for each of these nodes, how many observations were there? What was the complexity parameter, what is the mean of observation, what was the MSE of observation, mean squared error, how many observations were in the one side and other how many observation in the other side and then primary splitting rules and all these things you can get from and for all the nodes, you will have these details. So, this is how you create this decision tree, you can print this decision tree also. And also, you can plot this decision tree by using this plot and text comment.

(Refer Slide Time: 35:32)



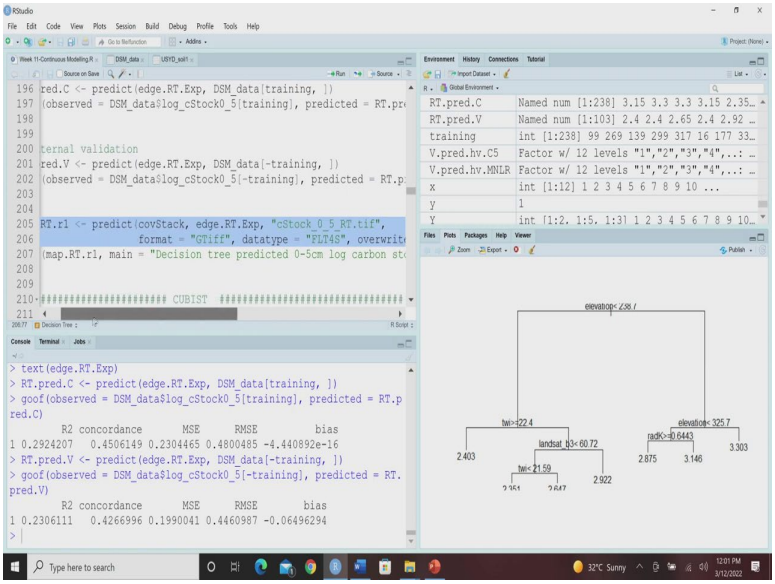


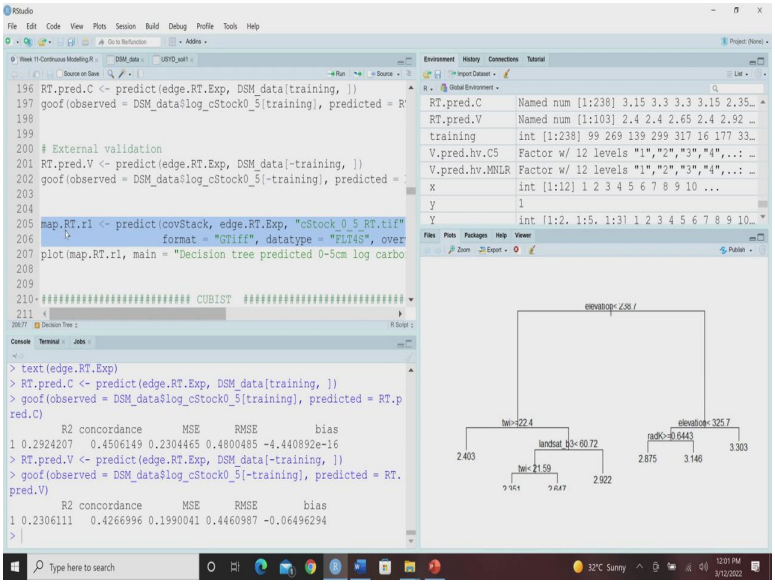


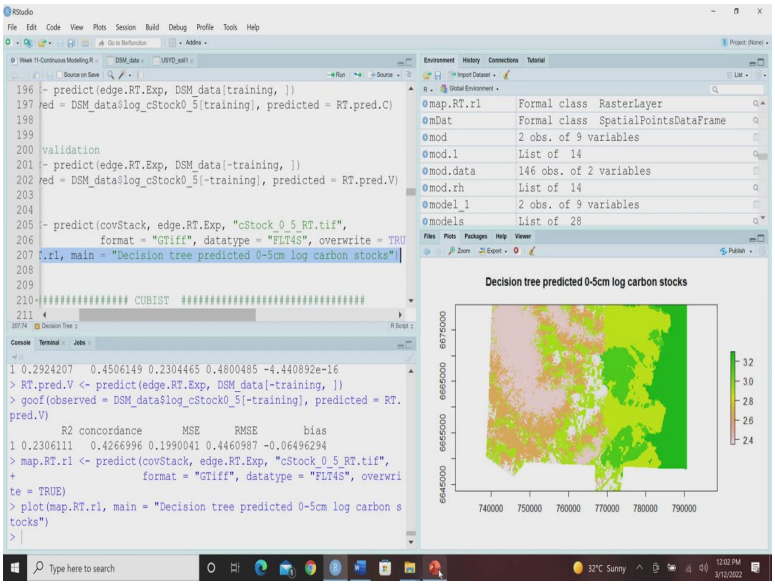
Of course, there are many other packages called part e part k 8 packages for producing more aesthetically beautiful, decision tree representation, but here you can see some ideas basic ideas. So, now, if we do the internal validation, remember guys internal validation is synonymous to the calibration. So, here we are going to use these you know, calibration.

So, basically using the training samples, we are going to do the internal, internal validation. So, we are getting the results here an external validation is the origin of validation using minus training. So, you can see how the results are. So, you can see here, this is the, results from the calibration and validation.

(Refer Slide Time: 36:35)







And also you can map it for you know, you can map the result map based on the cart model, you can predict based on the stack covariates and you can plot the decision tree predicted organic carbon stock so, this is the decision tree predicted 0 to 5 centimetre log carbon stock and which is using the covariates values for fitting the model and based on those module cart model we have predicted the results and this is the map based on the classification regression tree.

(Refer Slide Time: 37:15)

So, let us wrap up this lecture, I think you have learned something new, please run these codes to gain more and more confidence all these codes are having these annotation for your better understanding, this is the reference and let us wrap up this lecture. And thank you for joining. And in the next lecture we will be going from here we will be discussing the Cubist model and random forest using R. And will see how we can use this cubist model and random forests to produce the map of soil properties. Thank you.