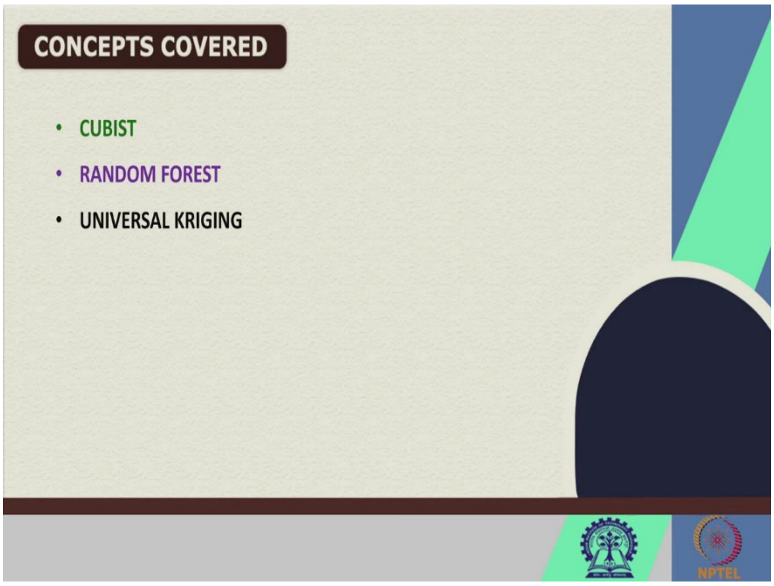
Machine Learning for Soil and Crop Management
Professor Somsubhra Chakraborty
Agricultural and Food Engineering Department
Indian Institute of Technology, Kharagpur
Lecture 55
Digital Soil Mapping with Continuous Variables (Contd.)

Welcome friends to this last 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. We have in our previous lectures we have seen many you know R codes for exploratory data analysis, GIS operation, geo statistical operations, spline fitting, we have seen also, we have seen how to do the model validation using random holdout and then leave one out cross validation.

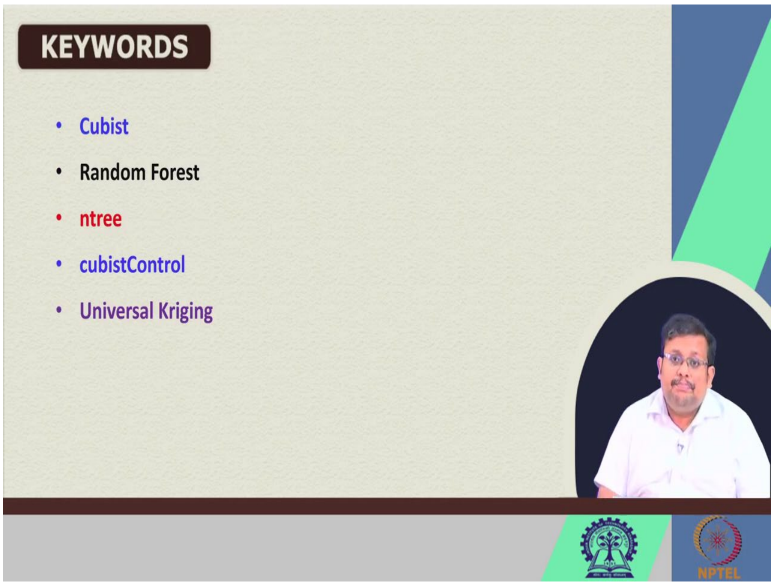
We have seen, we have seen how to produce the model using simple linear regression, we have seen how to produce the model using multiple linear regression and map based on multiple linear regression. Also, we have seen how to develop the classification regression to a decision tree using the continuous data and how to produce the map using that continuous, that decision tree model.

(Refer Slide Time: 01:25)



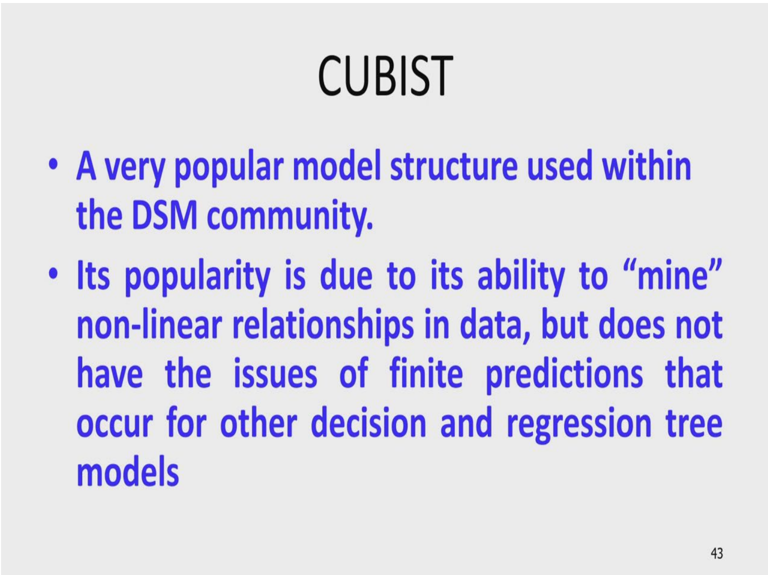
Now, in this lecture we are going to cover these concepts we are going to cover this cubist model, we are going to cover the random forest model and then we are going to discuss a hybrid model that is a hybrid approach that is Universal Kriging.

(Refer Slide Time: 01:38)



So, guys these are the cube these are the keywords for this lecture, cubist, random forest, ntree then cubist control Universal Kriging these we are going to discuss in this lecture.

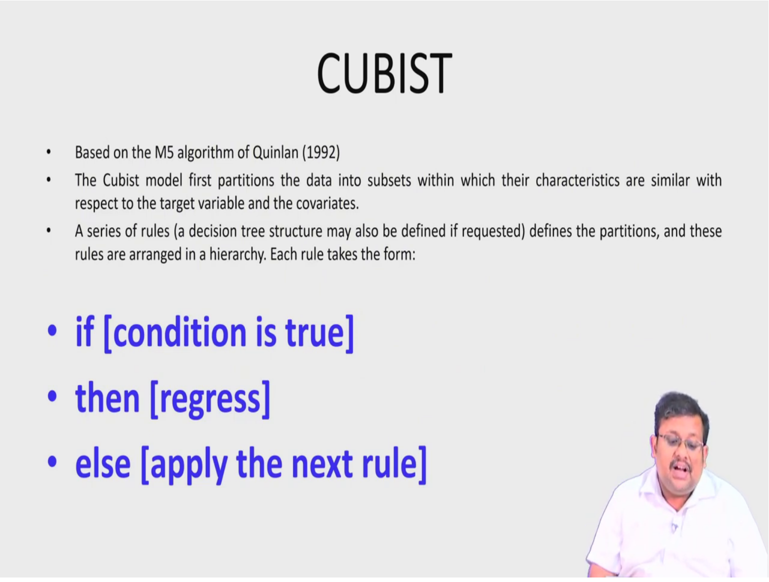
(Refer Slide Time: 01:55)



Now, first we will start discussing about the Cubist, cubist is a very popular model structure used in the within the DSM community and its popularity is due to its ability to mine nonlinear relationship data but does not have the issue of finite prediction that occur for other decision and regression tree models.

Now, in case of decision tree models, the finite predictions is one of the major issue. So, here cubist can address that problem. So, it is ability to mine that non-linear relationship data, but it does not have the issue of this finite prediction. So, this is very popular method nowadays for DSM operations.

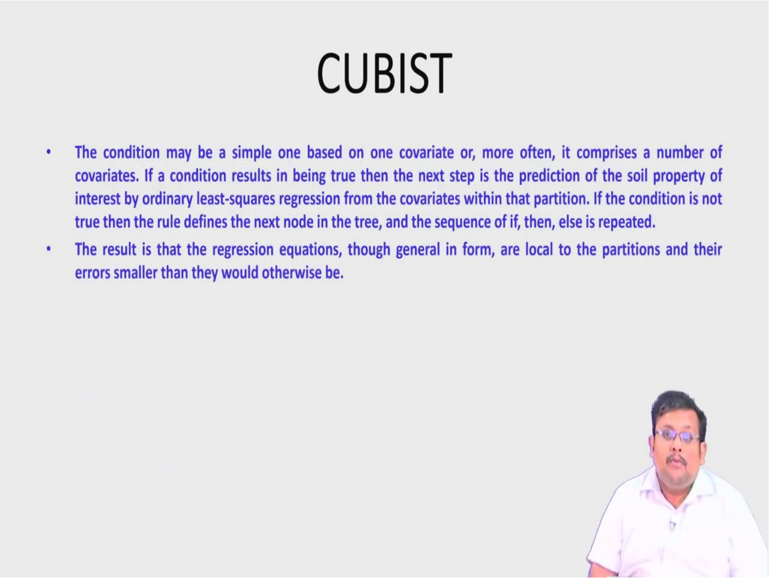
(Refer Slide Time: 02:39)



So, this cubist is based on the M5 algorithm of Quinlan, and also this cubist model… how this cubist model works. So, this cubist model works first by partitioning the data into subsets within which their characteristics are similar with respect to the target variable and the covariates and a series of rules, defines these partitions and these rules are managed in a hierarchy. So, each rule takes this form.

So, you can see here, if a condition is true, then you know you go for regression otherwise or else apply the next rule. So, basically partition the data by these if then else rule and when some observations satisfy some clustered by this partition, then we fit the linear regression in each of these nodes. So, it is basically kind of a hybrid between the nonlinear cart as well as linear model. So, that is why it is able to generalise the nonlinear relationship, but at the same time it can maintain the linearity. So, this is a cubist rule, this is the this is the cubist algorithm.

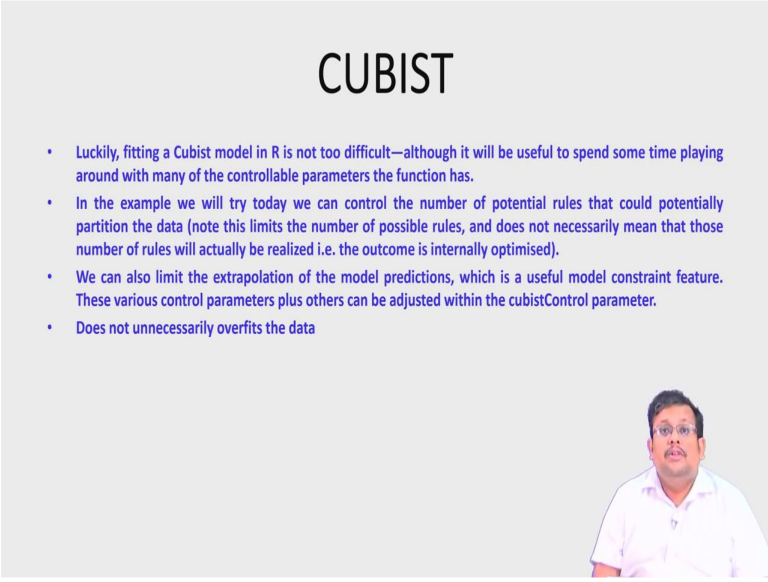
(Refer Slide Time: 03:55)



Now, the conditions may be a simple one based on the one covariate or more often it comprises a number of covariates if a condition results in being true then the next step is the prediction of the soil property of interest by ordinary least squares regression from the covariates within that partition, just like I have seen, I have told you that we partition the data based on the rule if else and you know if then else rule and then we partition the data and based on that partition, we predict the target parameter using our covariate data using lived you know least squares regression linear least squares regression model.

So, if the condition is not true, then the rule defines the next node of the tree and the sequence of it and then else is repeated. So, the results is that regression equation through general inform are very local to the partition and they are error is smaller than they would otherwise be. So, of course, you can see that we are partitioning the data and we are fitting it individual models, linear models in each of this partition. So, though they are general linear, you know, you know this regression equation though their general form they are very much local, because we are fitting the individual models within the individual partitions and as a result of that our error becomes much smaller.

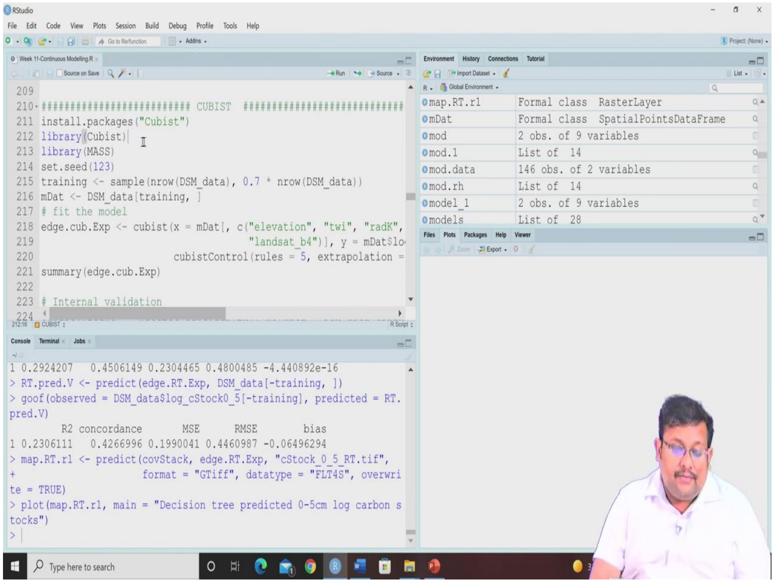
(Refer Slide Time: 05:23)

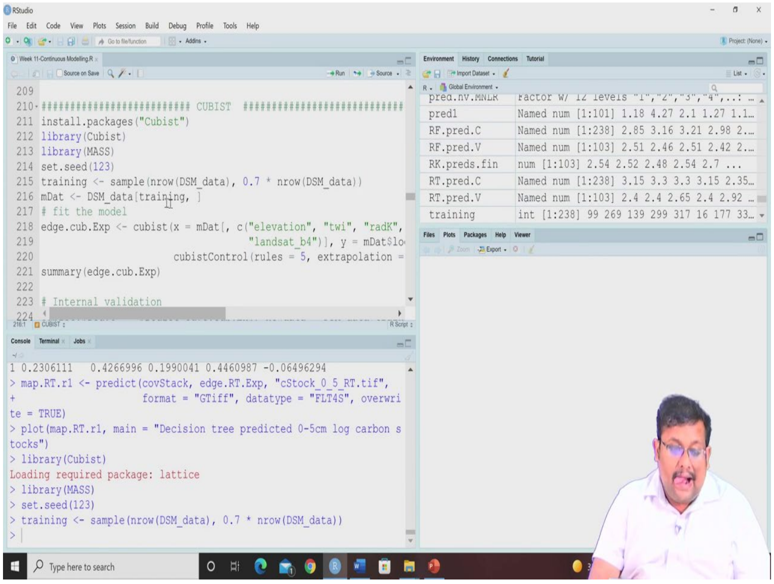


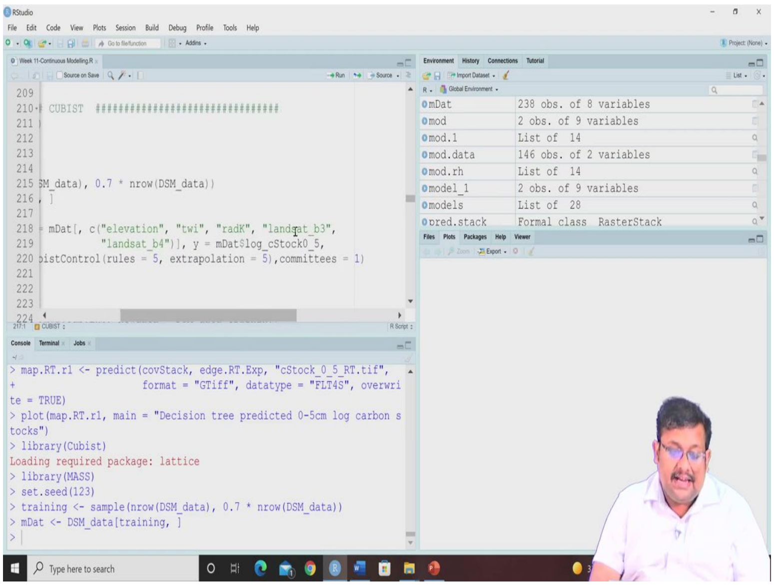
Now, luckily fitting this cubist model in R is not too difficult although it will be useful to spend some time for playing around with many of the controllable parameters the function says, so there is a parameter called cubist control. So, use this cubist control parameter to tune the model. So, if the example will try today, we can control the number of potential rules that could potentially partition the data so, we can control the number of rules this limits the number of possible rules and does not necessarily means that those number of rules will actually be realised, maybe we can give 10 rules, but the algorithm may go with only two rules.

So, when you know after the optimization and also, we can limit the excerpt of extrapolation of the model prediction which is an useful model constraint feature and these variates various control parameters can be adjusted within this cube control parameters. And the one of the benefit of using this cube is easily does not overfit and unnecessarily overfit the data.

(Refer Slide Time: 06:44)



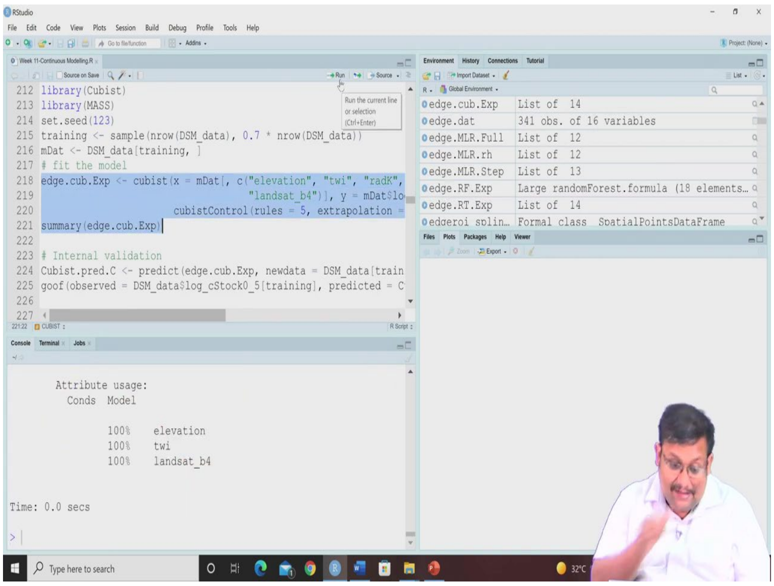


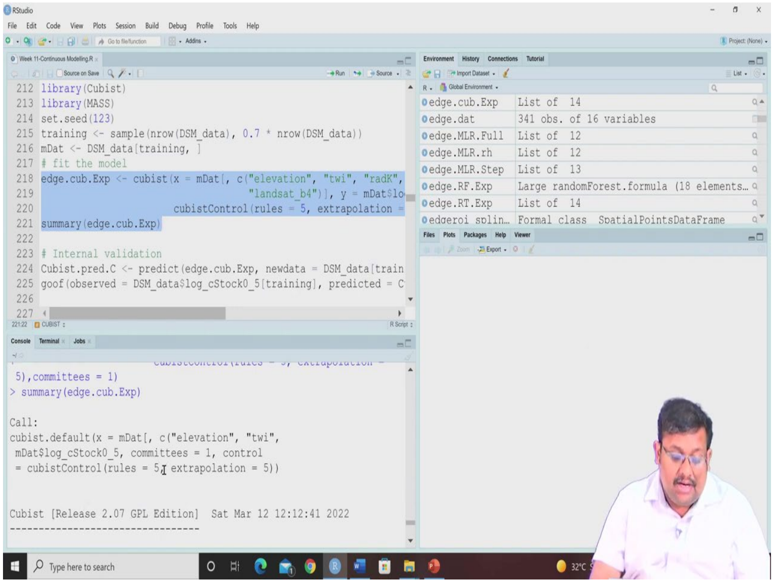


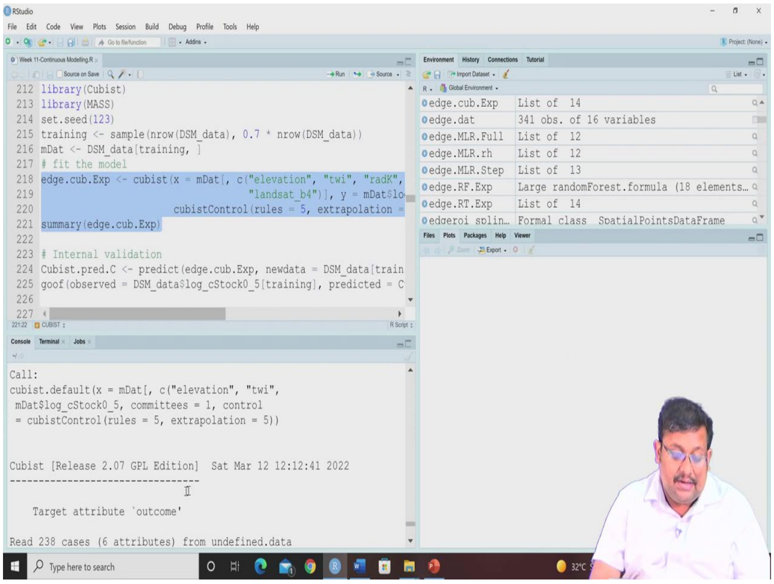
So, let us go ahead and see how we can run this model. So, we have done this card and sorry decision tree and based prediction in our previous lecture. So, here we are going to start with the cubist. So, for cubist we are going to install this cubist package. So, once we have installed the cubist package, let us run this let me just you know, we can call this library cubist, and then we can call this library mass and then we are setting the seed as 123 and then again we are selecting the 70 percent of the samples in the training data set and let us call these training data as model data or m dat. So, these training data calibration data let us call it as m dat or model data.

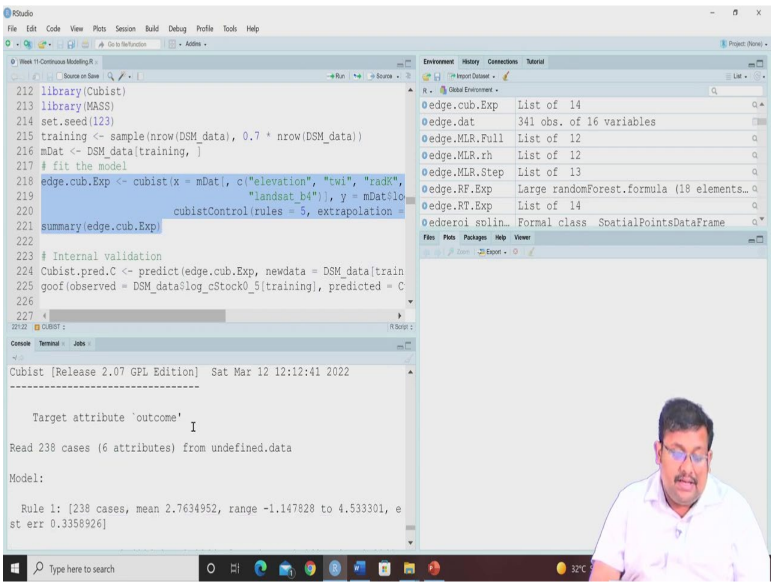
Now, let us fit the cubist model. So, for fitting the cubist model here we are having our x is our model data and our predictors are elevation twi, radk, landset b3, landset b4 and our y is the target parameter that is the carbon stock of 0 to 5 centimetre within this model data our cubist control parameters you can see it is cubist control parameter, we are specifying the 5 number of rows and we are specifying the five extrapolation and number of committees will be 1. So, and so, this is very simple and let us run this summary of this cubist model.

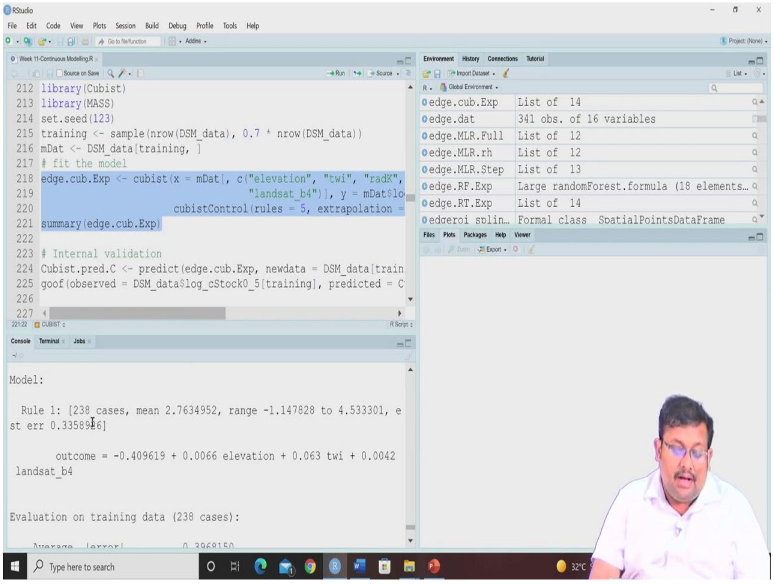
(Refer Slide Time: 08:30)

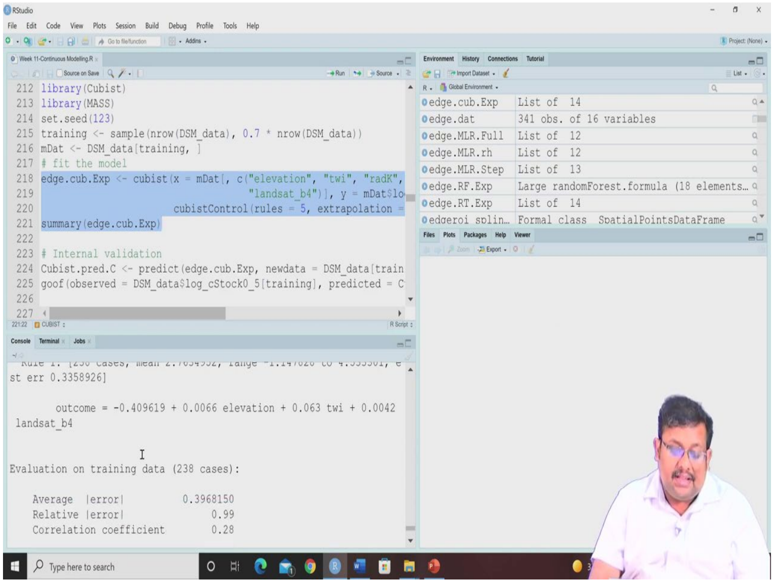


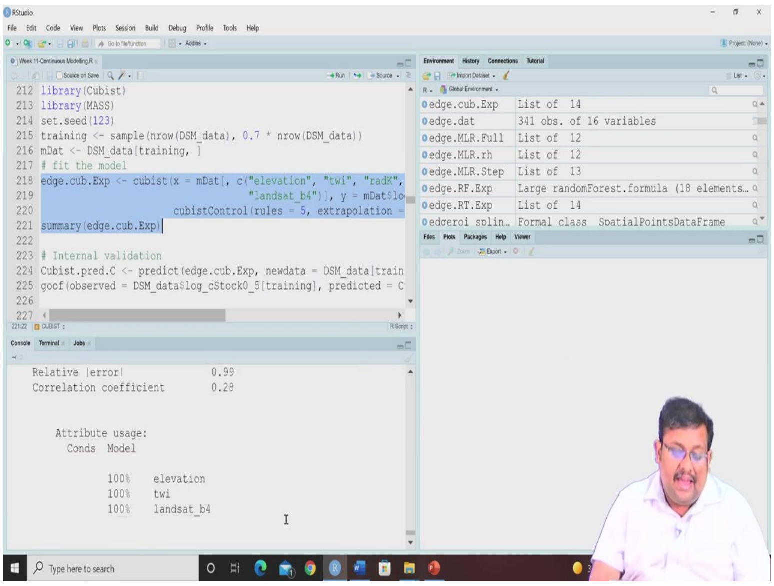

library(Cubist)
library(MASS)
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
mDat <- DSM_data[training,]
# fit the model
edge.cub.Exp <- cubist(x = mDat[, c("elevation", "twi", "radK",
"landsat_b4")], y = mDat$lo
cubistControl(rules = 5, extrapolation =
summary(edge.cub.Exp)
# Internal validation
Cubist.pred.C <- predict(edge.cub.Exp, newdata = DSM_data[train
goof(observed = DSM_data$log_cStock0_5[training], predicted = C
5),committees = 1)
> summary(edge.cub.Exp)
Call:
cubist.default(x = mDat[, c("elevation", "twi",
mDat$log_cStock0_5, committees = 1, control
= cubistControl(rules = 5, extrapolation = 5))
Cubist [Release 2.07 GPL Edition] Sat Mar 12 12:12:41 2022
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
edge.RF.Exp Large randomForest.formula (18 elements…
edge.RT.Exp List of 14
edgeroi_splin… Formal class SpatialPointsDataFrame


Call:
cubist.default(x = mDat[, c("elevation", "twi",
mDat$log_cStock0_5, committees = 1, control
= cubistControl(rules = 5, extrapolation = 5))
Cubist [Release 2.07 GPL Edition] Sat Mar 12 12:12:41 2022
Target attribute `outcome'
Read 238 cases (6 attributes) from undefined.data


Cubist [Release 2.07 GPL Edition] Sat Mar 12 12:12:41 2022
Target attribute `outcome'
Read 238 cases (6 attributes) from undefined.data
Model:
Rule 1: [238 cases, mean 2.7634952, range -1.147828 to 4.533301, e
st err 0.3358926]


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
library(Cubist)
library(MASS)
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
mDat <- DSM_data[training,]
# fit the model
edge.cub.Exp <- cubist(x = mDat[, c("elevation", "twi", "radK",
"landsat_b4")], y = mDat$lo
cubistControl(rules = 5, extrapolation =
summary(edge.cub.Exp)
# Internal validation
Cubist.pred.C <- predict(edge.cub.Exp, newdata = DSM_data[train
goof(observed = DSM_data$log_cStock0_5[training], predicted = C
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
edge.RF.Exp Large randomForest.formula (18 elements…
edge.RT.Exp List of 14
edgeroi_splin… Formal class SpatialPointsDataFrame
Model:
Rule 1: [238 cases, mean 2.7634952, range -1.147828 to 4.533301, e
st err 0.3358926]
outcome = -0.409619 + 0.0066 elevation + 0.063 twi + 0.0042
landsat_b4
Evaluation on training data (238 cases):
Type here to search

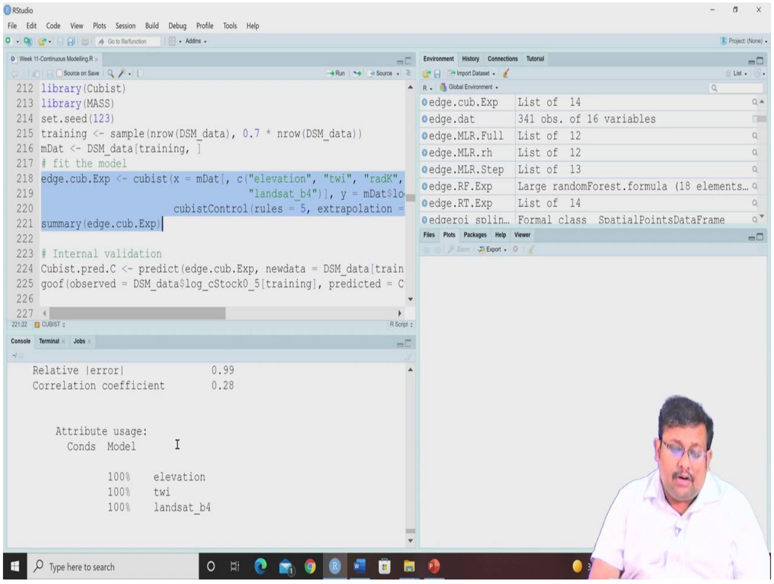

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
library(Cubist)
library(MASS)
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
mDat <- DSM_data[training,]
# fit the model
edge.cub.Exp <- cubist(x = mDat[, c("elevation", "twi", "radK",
"landsat_b4")], y = mDat$lo
cubistControl(rules = 5, extrapolation =
summary(edge.cub.Exp)
# Internal validation
Cubist.pred.C <- predict(edge.cub.Exp, newdata = DSM_data[train
goof(observed = DSM_data$log_cStock0_5[training], predicted = C
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
edge.RF.Exp Large randomForest.formula (18 elements…
edge.RT.Exp List of 14
edgeroi_splin… Formal class SpatialPointsDataFrame
st err 0.3358926]
outcome = -0.409619 + 0.0066 elevation + 0.063 twi + 0.0042
landsat_b4
Evaluation on training data (238 cases):
Average |error| 0.3968150
Relative |error| 0.99
Correlation coefficient 0.28
Type here to search

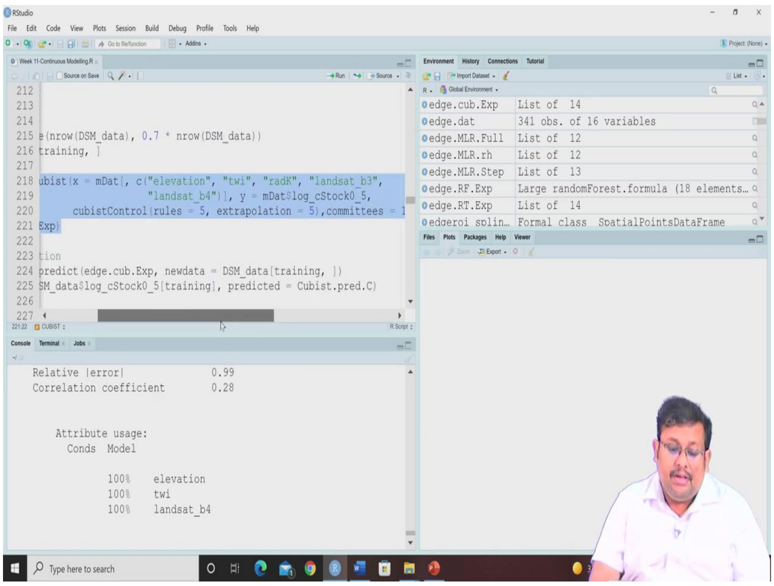

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
library(Cubist)
library(MASS)
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
mDat <- DSM_data[training,]
# fit the model
edge.cub.Exp <- cubist(x = mDat[, c("elevation", "twi", "radK",
"landsat_b4")], y = mDat$lo
cubistControl(rules = 5, extrapolation =
summary(edge.cub.Exp)
# Internal validation
Cubist.pred.C <- predict(edge.cub.Exp, newdata = DSM_data[train
goof(observed = DSM_data$log_cStock0_5[training], predicted = C
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
edge.RF.Exp Large randomForest.formula (18 elements…
edge.RT.Exp List of 14
edgeroi_splin… Formal class SpatialPointsDataFrame
Relative |error| 0.99
Correlation coefficient 0.28
Attribute usage:
Conds Model
100% elevation
100% twi
100% landsat_b4
Type here to search

So, if you run this summary of the cubist model, this is the model results. So, you can see this is the model initial model input and then you can see that target attribute in this case is outcome and we have 238 cases with 6 attributes and then rule 1 is basically using all the 238 cases the mean is 2.76 the range of values is given and estimated error is also given outcome is this model.

So, this model which they have fit this linear regression model which they have fit for this first rule is you know outcome is this is the intercept plus elevation with the slope then twi with the slope and landset b4 with the slope and then you can see that what are the average error what are the linearity error and that is the correlation coefficient.

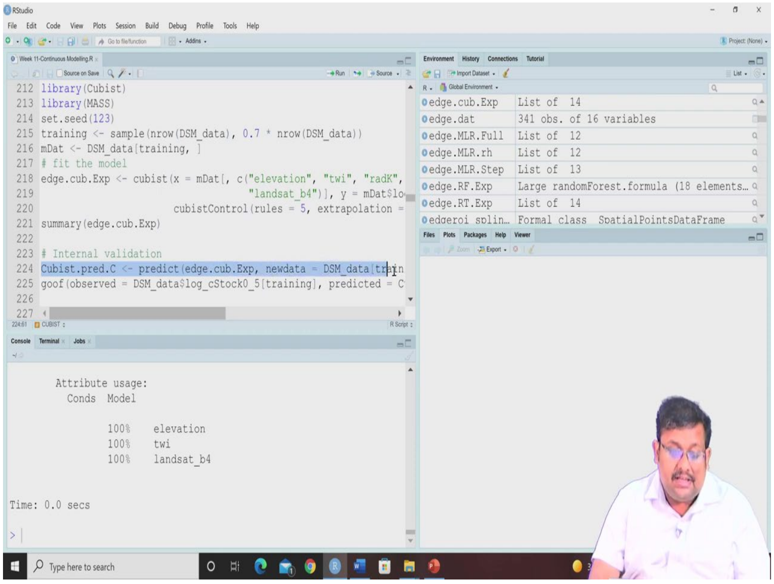
(Refer Slide Time: 09:34)

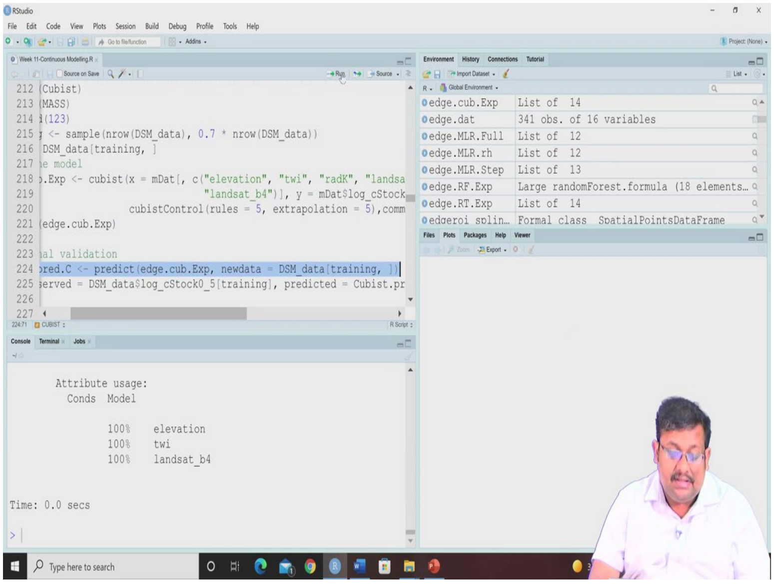


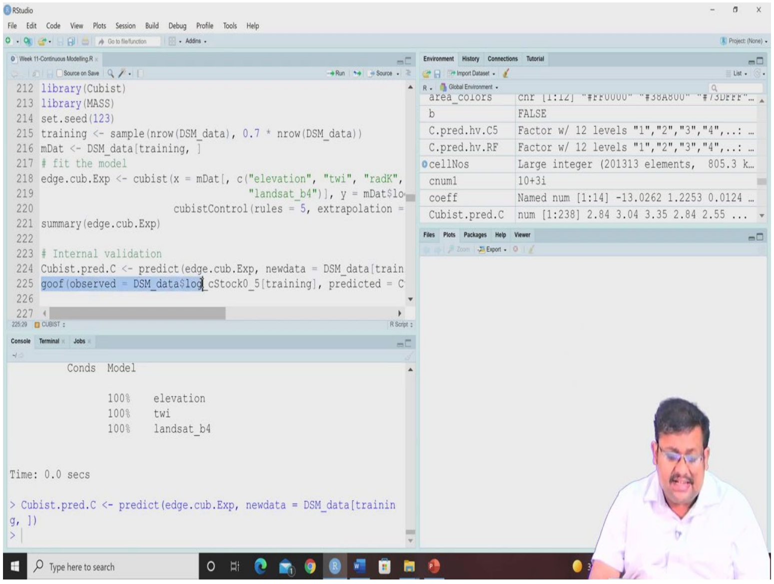


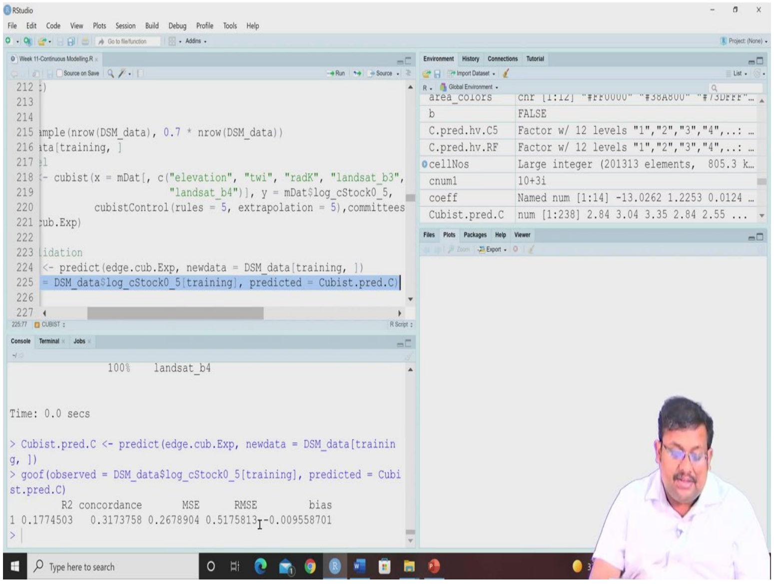
And then you can see that you know, how much these variables have been utilised in the model. So, from there you can see one important thing that although we have fixed you know, five rules, it has used only one role after optimization. So, this is one of the important feature of cubist.

(Refer Slide Time: 10:00)



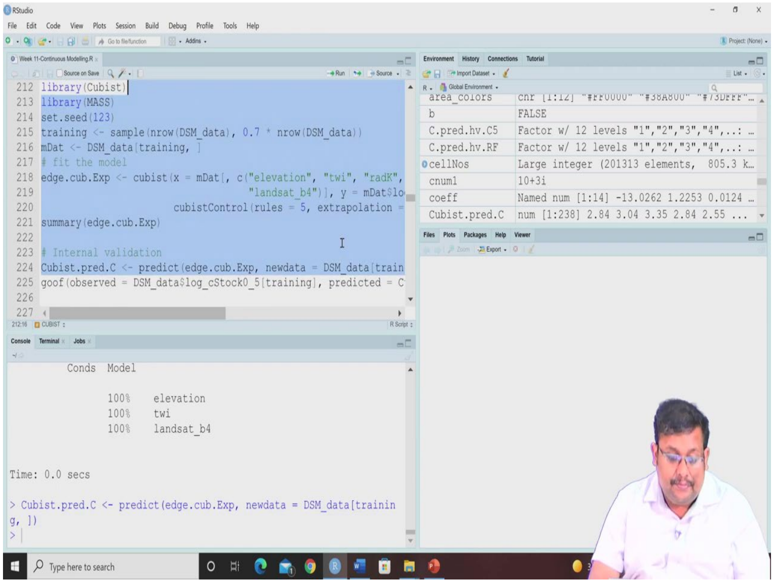


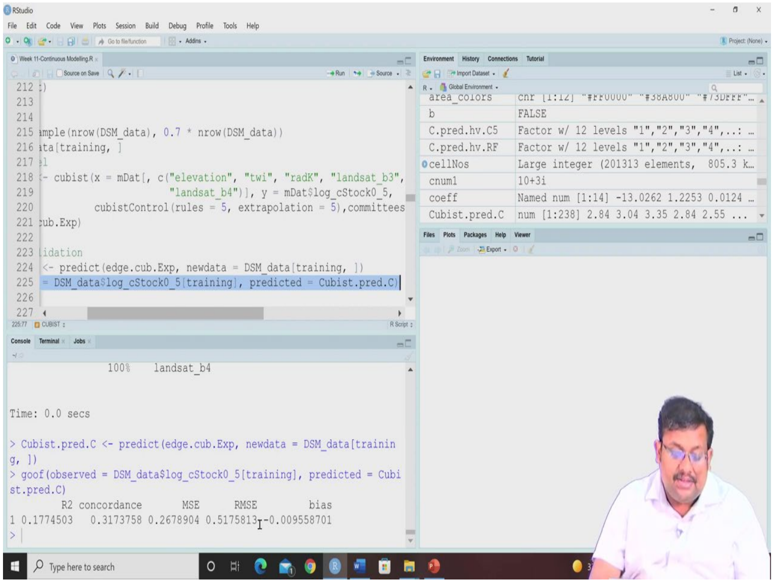


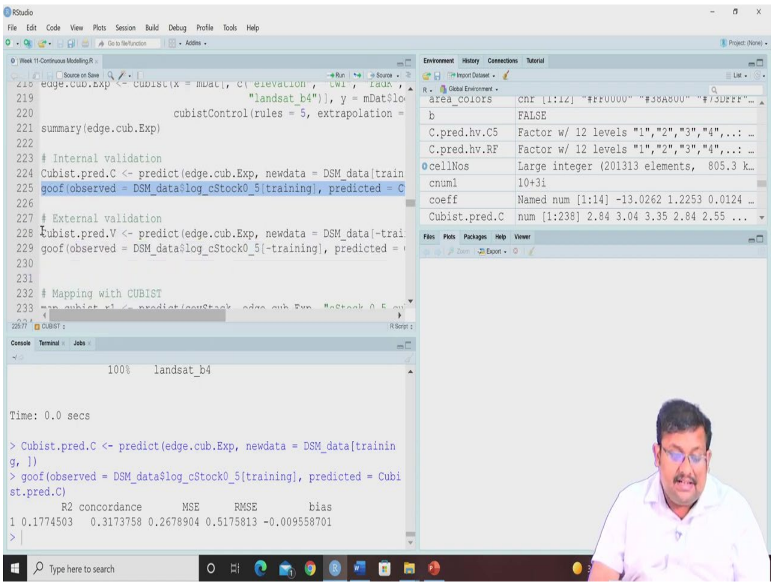


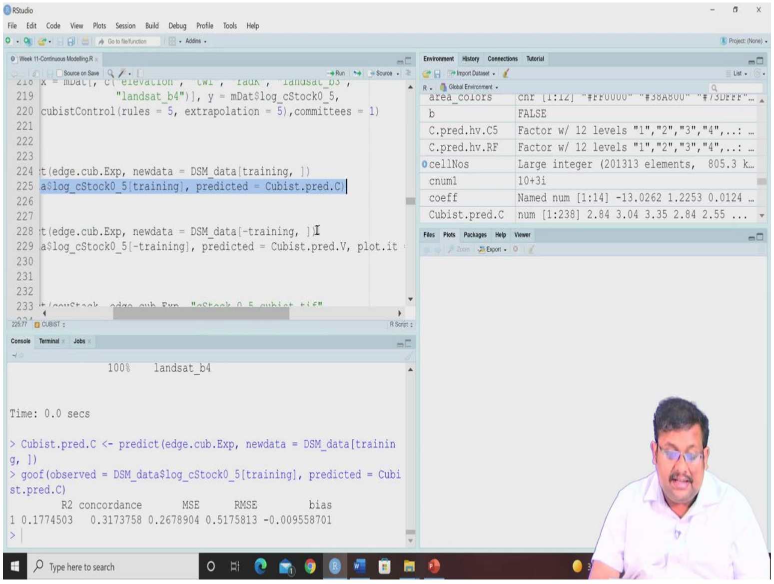
Now, let us do some internal validation. So, for this internal validation We are going to use again the you know first we are going to predict using the predict our dataset based on these training samples. And then internal validation means again the calibration so again goodness of fit statistics observed is our training data set 0 to 5 centimetre carbon data predicted which we have just predicted here cubist dot predict dot C here.

(Refer Slide Time: 10:30)



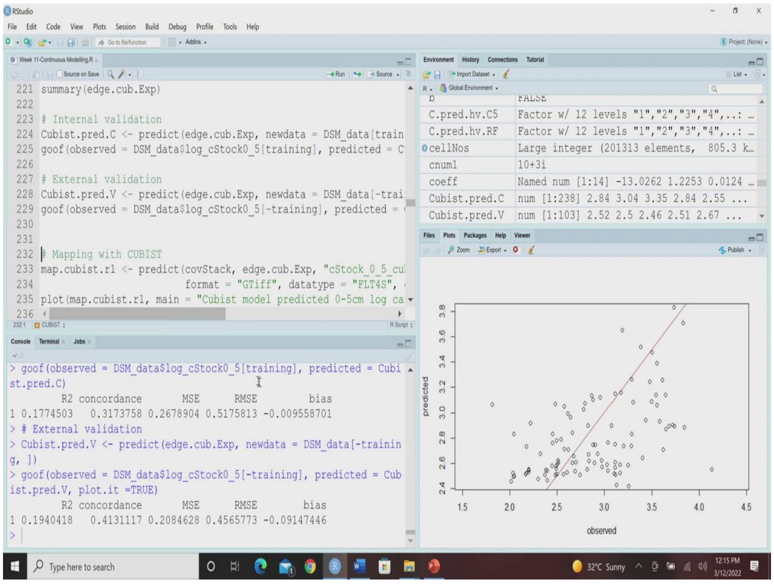


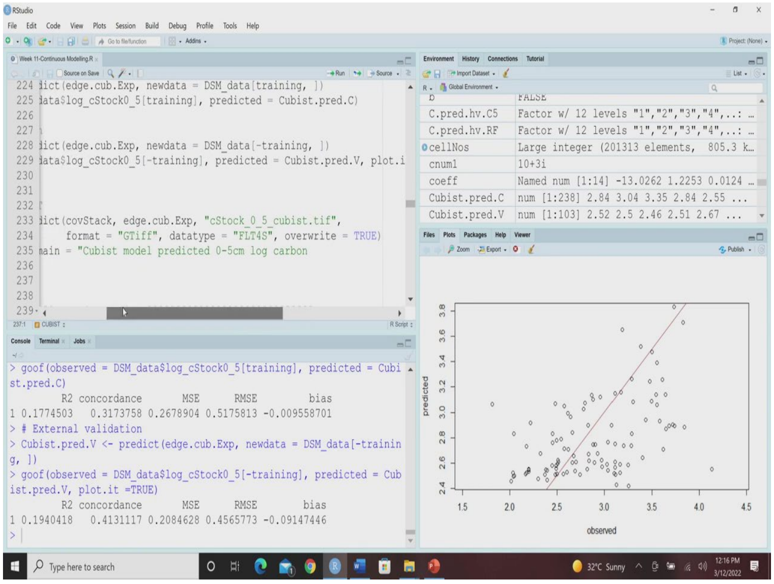




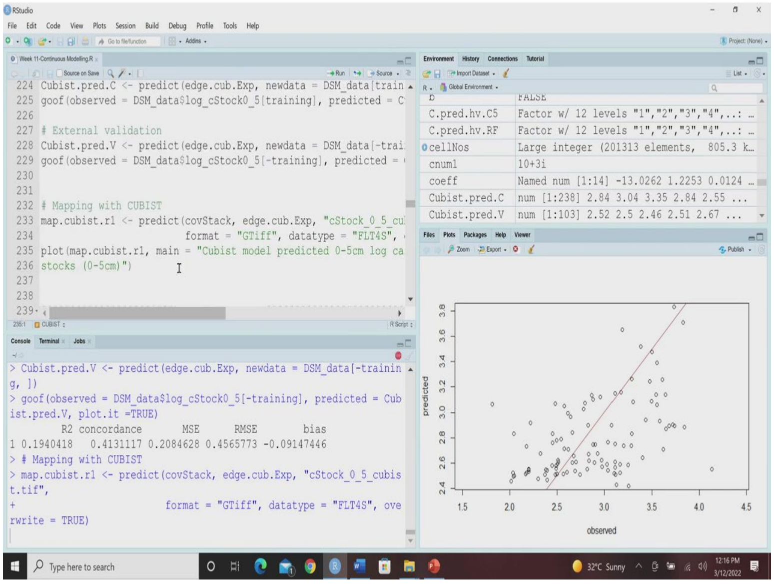
So, if we run this thing, we will see we will see the model results So, R square 0.17 concordance 0.31, MSE 0.26, RMSE 0.51, bias is 0.00 external validation again for validation data set minus training you can see we are first predicting the values and then from the observed values and the predicted values again we are going to use this goof function and we want to plot it simultaneously.

(Refer Slide Time: 11:05)

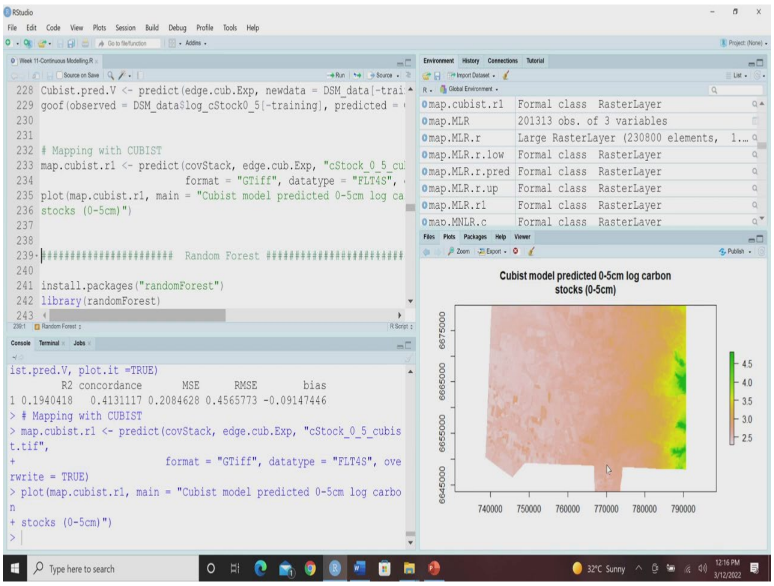




iict(edge.cub.Exp, newdata = DSM_data[training,])
iata$log_cStock0_5[training], predicted = Cubist.pred.C)
iict(edge.cub.Exp, newdata = DSM_data[-training,])
iata$log_cStock0_5[-training], predicted = Cubist.pred.V, plot.i
iict(covStack, edge.cub.Exp, "cStock_0_5_cubist.tif",
format = "GTiff", datatype = "FLT4S", overwrite = TRUE)
nain = "Cubist model predicted 0-5cm log carbon
> goof(observed = DSM_data$log_cStock0_5[training], predicted = Cubist.pred.C)
R2 concordance MSE RMSE bias
1 0.1774503 0.3173758 0.2678904 0.5175813 -0.009558701
> # External validation
> Cubist.pred.V <- predict(edge.cub.Exp, newdata = DSM_data[-training,])
> goof(observed = DSM_data$log_cStock0_5[-training], predicted = Cubist.pred.V, plot.it =TRUE)
R2 concordance MSE RMSE bias
1 0.1940418 0.4131117 0.2084628 0.4565773 -0.09147446
C.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",..: …
C.pred.hv.RF Factor w/ 12 levels "1","2","3","4",..: …
cellNos Large integer (201313 elements, 805.3 k…
cnum1 10+3i
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] 2.52 2.5 2.46 2.51 2.67 ...
predicted
observed



224 Cubist.pred.C <- predict(edge.cub.Exp, newdata = DSM_data[train
225 goof(observed = DSM_data$log_cStock0_5[training], predicted = C
227 # External validation
228 Cubist.pred.V <- predict(edge.cub.Exp, newdata = DSM_data[-trai
229 goof(observed = DSM_data$log_cStock0_5[-training], predicted =
232 # Mapping with CUBIST
233 map.cubist.r1 <- predict(covStack, edge.cub.Exp, "cStock_0_5_cu
234 format = "GTiff", datatype = "FLT4S",
235 plot(map.cubist.r1, main = "Cubist model predicted 0-5cm log ca
236 stocks (0-5cm)")
> Cubist.pred.V <- predict(edge.cub.Exp, newdata = DSM_data[-training,])
> goof(observed = DSM_data$log_cStock0_5[-training], predicted = Cubist.pred.V, plot.it =TRUE)
R2 concordance MSE RMSE bias
1 0.1940418 0.4131117 0.2084628 0.4565773 -0.09147446
> # Mapping with CUBIST
> map.cubist.r1 <- predict(covStack, edge.cub.Exp, "cStock_0_5_cubist.tif",
+ format = "GTiff", datatype = "FLT4S", overwrite = TRUE)
predicted
observed



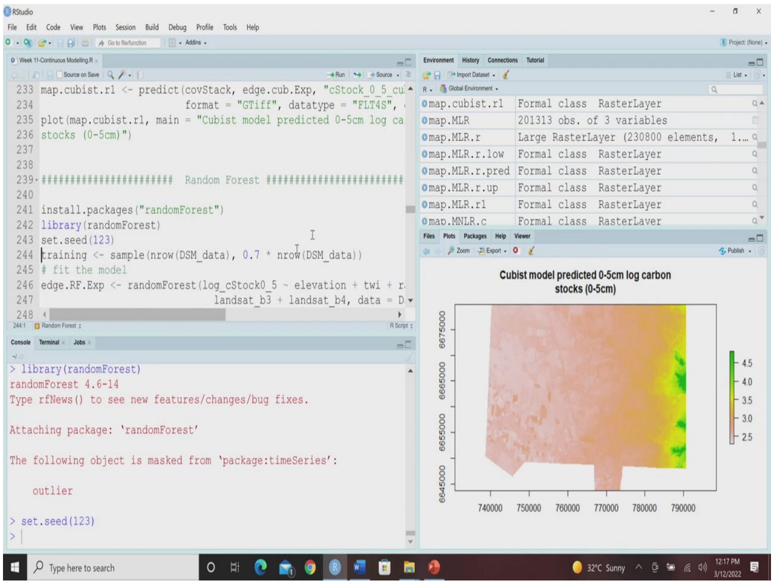
239 ##################### Random Forest #####################
241 install.packages("randomForest")
242 library(randomForest)
map.cubist.r1 Formal class RasterLayer
map.MLR 201313 obs. of 3 variables
map.MLR.r Large RasterLayer (230800 elements, 1…
map.MLR.r.low Formal class RasterLayer
map.MLR.r.pred Formal class RasterLayer
map.MLR.r.up Formal class RasterLayer
map.MLR.r1 Formal class RasterLayer
map.MNLR.c Formal class RasterLayer
> # Mapping with CUBIST
> map.cubist.r1 <- predict(covStack, edge.cub.Exp, "cStock_0_5_cubist.tif",
+ format = "GTiff", datatype = "FLT4S", overwrite = TRUE)
> plot(map.cubist.r1, main = "Cubist model predicted 0-5cm log carbon
+ stocks (0-5cm)")
Cubist model predicted 0-5cm log carbon stocks (0-5cm)

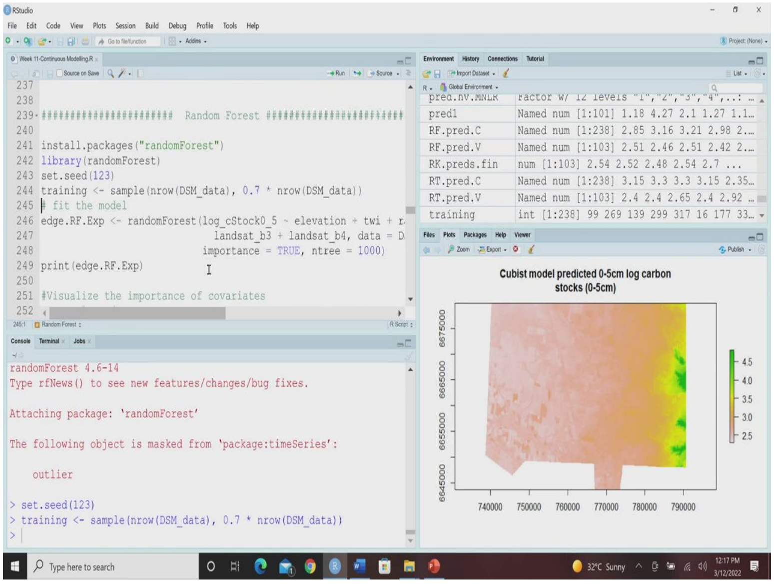
So, you can see this is the result 0.19 concordance I may see RMSE bias, when you do not one thing when using the goof function, when you do not specify any argument of type, then by default it will give you the result for digital soil mapping or DSM if you specify the argument at either DSM or spec, then only they will they will take it into consideration, but if you do not specify the argument, then it will by default it will take the DSM argument.

So, this is a predicted versus measured results and then if you want to map based on this cube is values, so, basically you have to use this predict function and then you know based on this predict function our model is edge dot cube dot exp our covStack remember whatever modelling, whatever mapping you are doing, you have to based on these covariate data.

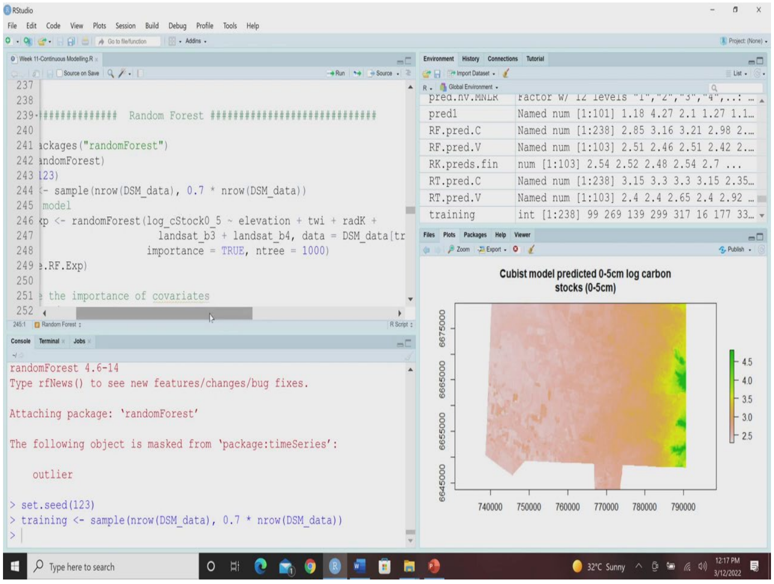
So, covariate data stack covariate is taking into consideration so, we are running it and then if you want to plot this, you just have to use this plot function and see how it will appear. So, this is the cubist model predicted 0 to 5 centimetre no log carbonStock. So, using the cubist model, so, this is how you can guys you can predict the view based on the cubist model and you can produce the map of soil properties using the cubist predicted values. So, we have completed the cubist model.

(Refer Slide Time: 12:58)

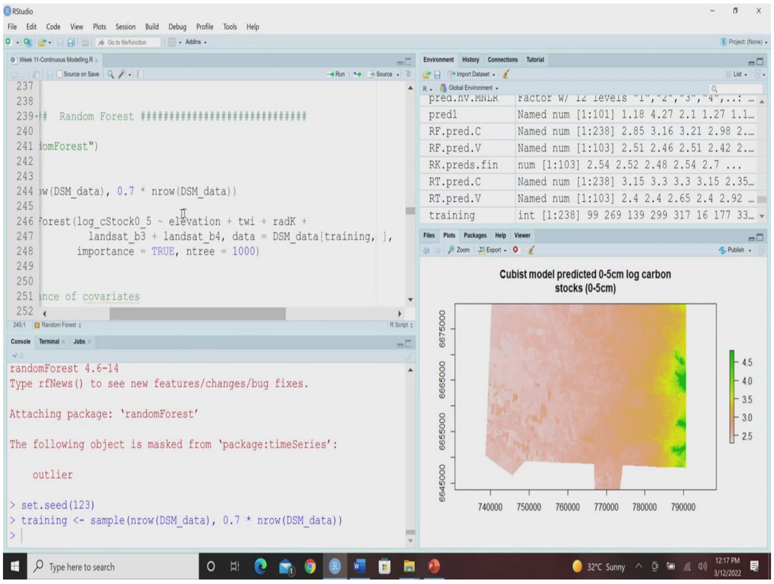




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Week 11-Continuous Modeling.R
############################ Random Forest ############################
install.packages("randomForest")
library(randomForest)
set.seed(123)
training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
# fit the model
edge.RF.Exp <- randomForest(log_cStock0_5 ~ elevation + twi + r
landsat_b3 + landsat_b4, data = D
importance = TRUE, ntree = 1000)
print(edge.RF.Exp)
#Visualize the importance of covariates
Console Terminal Jobs
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.
Attaching package: 'randomForest'
The following object is masked from 'package:timeSeries':
outlier
> set.seed(123)
> training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
Environment History Connections Tutorial
pred1 Named num [1:101] 1.18 4.27 2.1 1.27 1.1...
RF.pred.C Named num [1:238] 2.85 3.16 3.21 2.98 2...
RF.pred.V Named num [1:103] 2.51 2.46 2.51 2.42 2...
RK.preds.fin num [1:103] 2.54 2.52 2.48 2.54 2.7 ...
RT.pred.C Named num [1:238] 3.15 3.3 3.3 3.15 2.35...
RT.pred.V Named num [1:103] 2.4 2.4 2.65 2.4 2.92 ...
training int [1:238] 99 269 139 299 317 16 177 33...
Files Plots Packages Help Viewer
Cubist model predicted 0-5cm log carbon stocks (0-5cm)
6675000 6665000 6655000 6645000
740000 750000 760000 770000 780000 790000
4.5 4.0 3.5 3.0 2.5
Type here to search
32°C Sunny 12:17 PM 3/12/2022



RStudio
############# Random Forest ############################
ckages("randomForest")
ndomForest)
123)
- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
model
xp <- randomForest(log_cStock0_5 ~ elevation + twi + radK +
landsat_b3 + landsat_b4, data = DSM_data[tr
importance = TRUE, ntree = 1000)
.RF.Exp)
the importance of covariates
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.
Attaching package: 'randomForest'
The following object is masked from 'package:timeSeries':
outlier
> set.seed(123)
> training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
Cubist model predicted 0-5cm log carbon stocks (0-5cm)
Type here to search
32°C Sunny 12:17 PM 3/12/2022

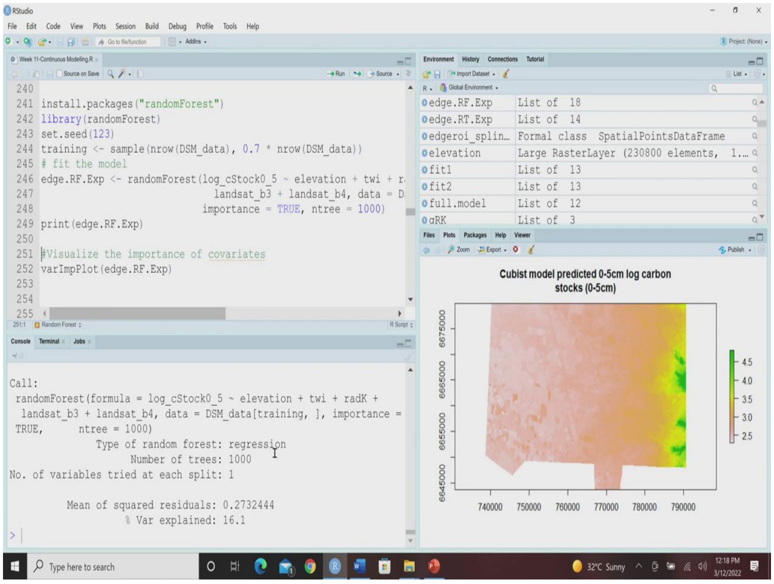


RStudio
Random Forest
omForest")
w(DSM_data), 0.7 * nrow(DSM_data))
orest(log_cStock0_5 ~ elevation + twi + radK +
landsat_b3 + landsat_b4, data = DSM_data[training,],
importance = TRUE, ntree = 1000)
nce of covariates
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.
Attaching package: 'randomForest'
The following object is masked from 'package:timeSeries':
outlier
> set.seed(123)
> training <- sample(nrow(DSM_data), 0.7 * nrow(DSM_data))
Cubist model predicted 0-5cm log carbon stocks (0-5cm)
Type here to search
32°C Sunny 12:17 PM 3/12/2022

Now, the next important model, which has been widely used in DSM domain is the random forest for Random Forests the in the you have to install this package, random forest. So, please go ahead and install, I have already installed so, I am not going to further install I am just going to call this library random forest.

Again, I am setting the seed 123 for selecting randomly selecting the data, the calibration set. So, here we are selecting the 70 percent of the data in the calibration set and then we are fitting the model so, for fitting the model the function is random forest function here we are targeting these log carbon stocks 0 to 5 centimetre our parameters or variables or elevation twi, radk, landset b3, landset b4 data is the training data set of the you know of the of these DSM underscore data and importance true and the number of tree we want to grow is 1000 number of trees. So, the model is now built.

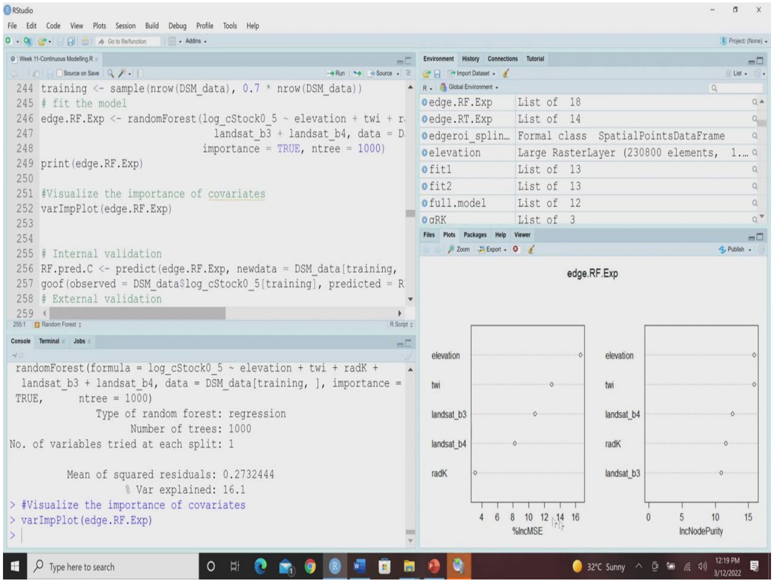
(Refer Slide Time: 14:00)



Now, let us see how this model will. So, here you can see this is the output of the model. So, type of random forest is a regression because our target is a continuous variable number of trees we have grown 1000, number of variables tried at each split is 1, mean of squared residuals 0.27, so MSE is basically 0.27.

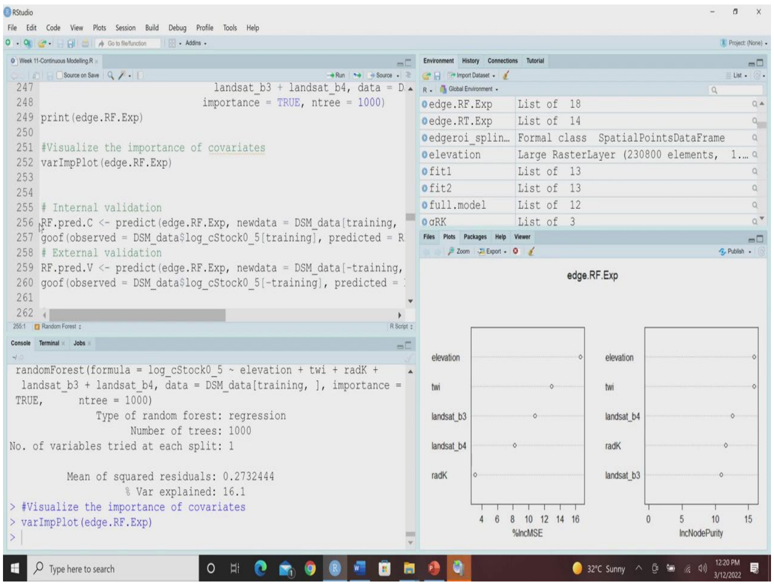
So, if you take the square root of it, it will give you the RMSE and number of way and the percentage of variance explained is 16.1 percentage. So, you can see here this is the result from the random forest. Now, another very important thing is using the random forest, you will be able to get the variable importance that means you can map or you can produce the plot showing which variable is more important than other variables. So, for that we are going to use this varImpPlot function and this is the our full model.

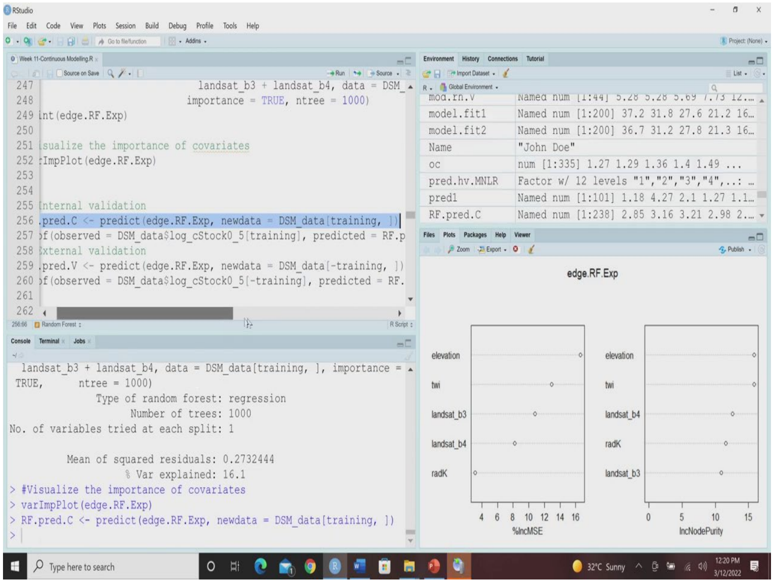
(Refer Slide Time: 15:00)

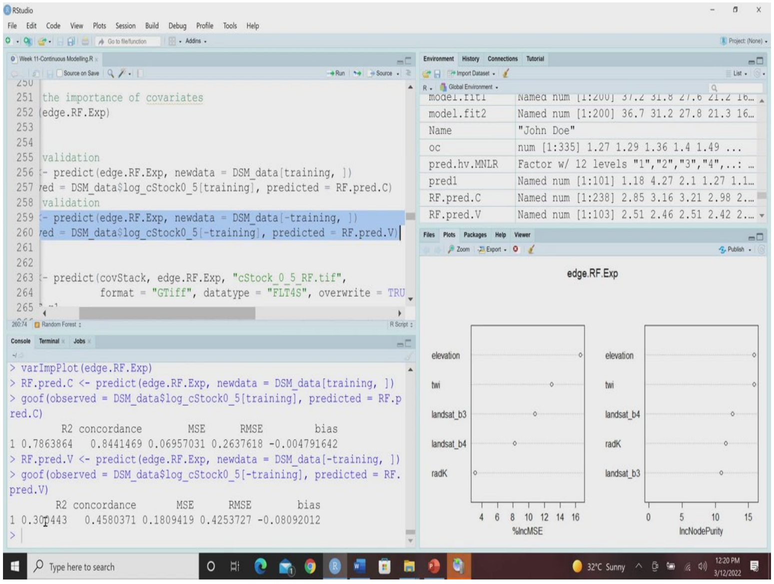


So, let us run it and you can see here these two expressions or these two plots will be generated based on percentage increase in MSE and increase in Node purity. So, you see that percent increase in MSE basically that means a based on that elevation has the maximum importance followed by twi followed by landset b3, landset b4 and radiometric potassium similar type of results we can get an increase in node purity, INC node purity means increasing node purity. So, you can also get that elevation is the highest important variable followed by twi, landset b4 radk, landset b3. So, based on this, we can say that elevation is the highest important variable followed by twi, landset b3 and twi and then landset b4 and so on, so forth.

(Refer Slide Time: 16:05)



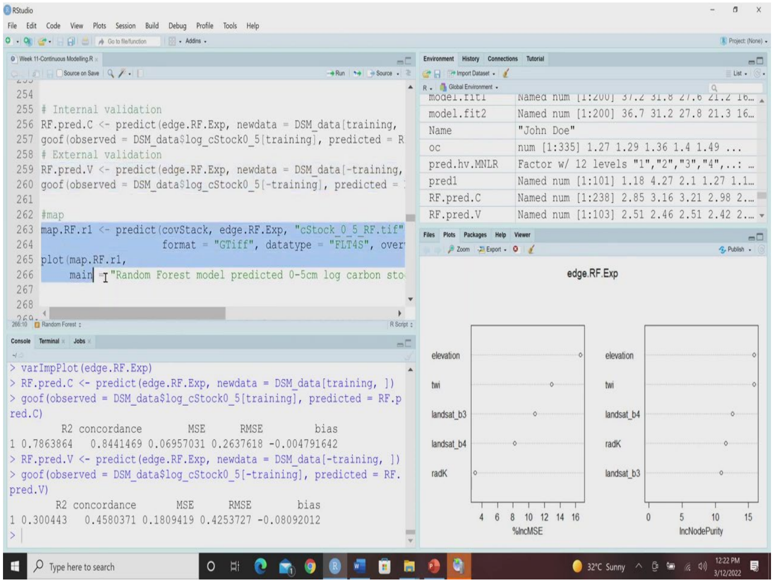


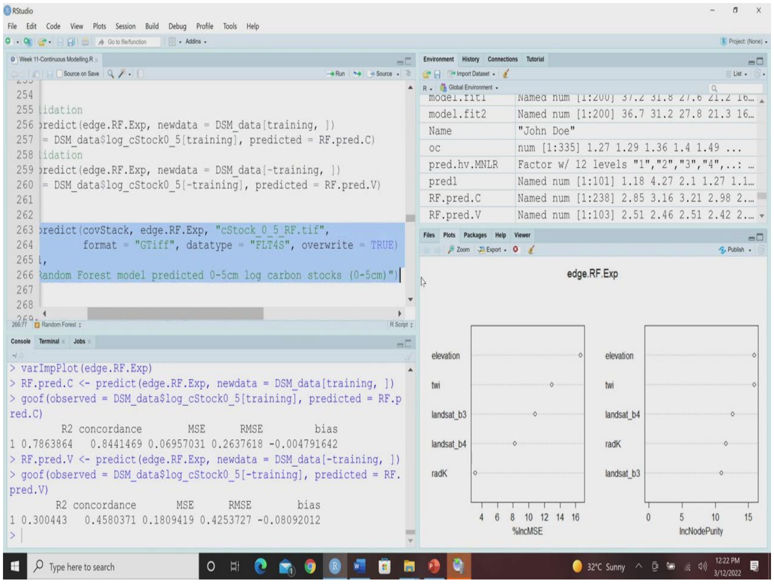


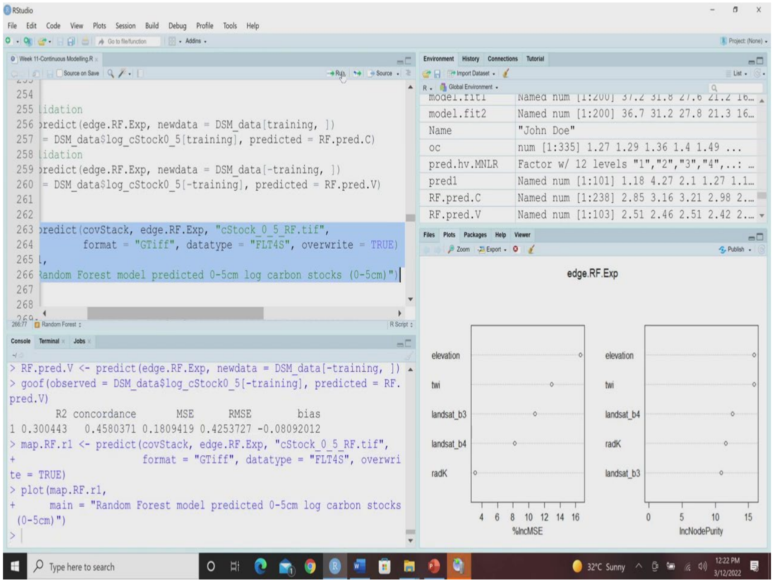
So, once we have done that, next is doing the internal validation the same just like previous, we are going to do this with the calibration set we are going to first predict based on the calibration set, and then we are going to use this cubist, we are going to use this goof function to predict the results and then we are going to see that R square is 0.78, concordance is 0.8, for MSE 0.06, RMSE 0.26, bias minus 0.00 and then external validation that means original validation with the holdout validation sample.

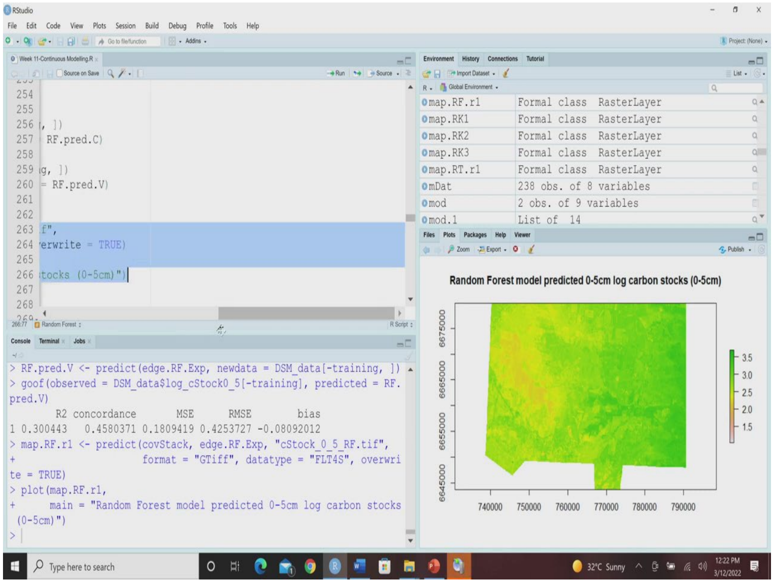
So, if you can see that, it is also 30 percent. Now, one thing you can see for clear here that here the model is showing somewhat you know, over prediction or you know sorry overfitting So, here you can see R squared value is 0.78 however, for the calibration data set for what the validation data set, we are getting an R square value of 0.30, so that showing some amount of overfitting. So, you should be very very careful by checking these… calibration validation statistics. So, you know, so we have seen how to do how to check the variable importance and we have also seen how to you know, how to get the model output from the calibration and the validation.

(Refer Slide Time: 17:37)





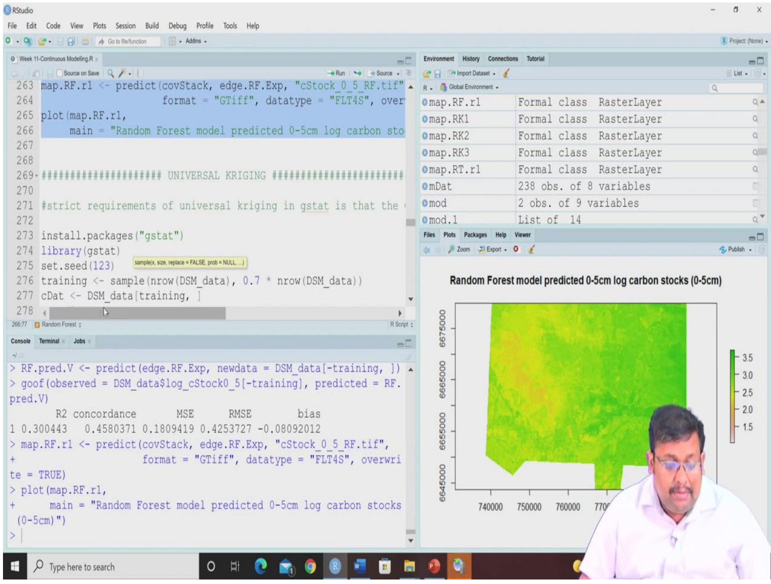


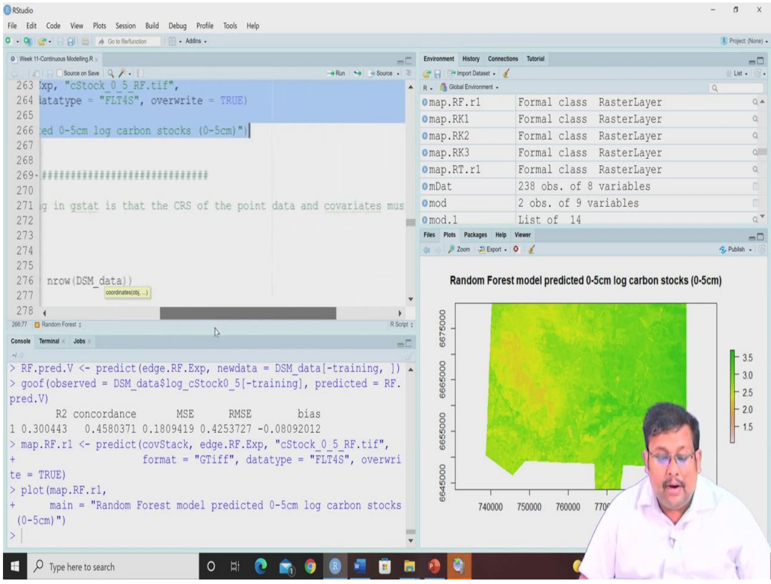


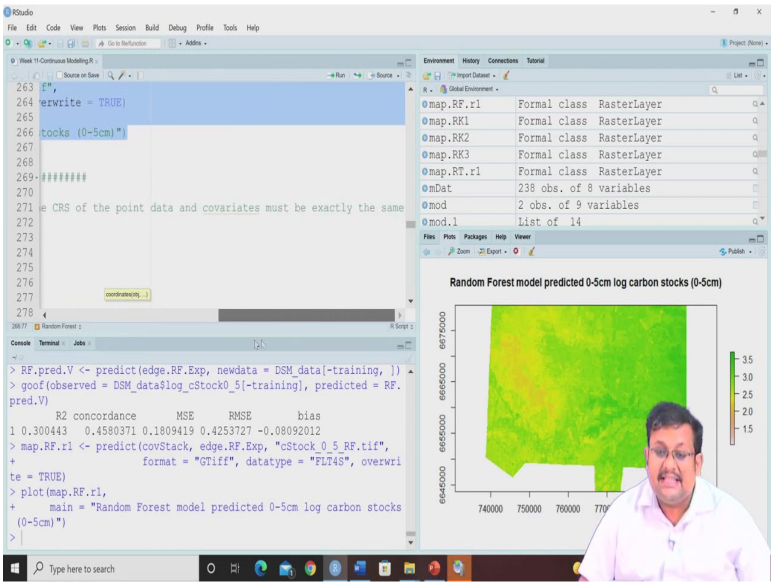
Now, let us see how to produce the map using this using the random forest, just like previously we have also used this call stack function, so call stack function sorry call stack or stack covariate we are going to predict based on the stack covariate using these edge dot rf dot exp model, our format data type all these things are given here. So, let us first predict and then map using and then plot using the plot function.

So, you can see here the plot will be generated momentarily showing the random forest predicted map of 0 to 5-centimetre block carbon stock. So, this is how this map looks like this is the random forest predicted organic carbon map of soil. So, guys this is how you produce the maps using these cubist and Random Forest. Now, next thing is Universal Kriging.

(Refer Slide Time: 18:43)







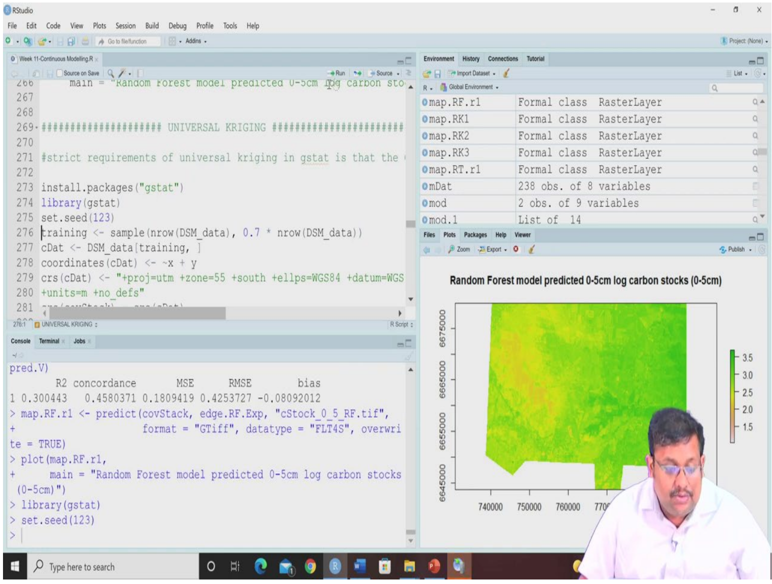
Now, Universal Kriging is a basically that it is a kind of a hybrid approach. Now, what is a hybrid approach it is basically combination of both regression as well as Kriging. So, here we do the regression by using the covariates and whatever residuals are there, we are going to Krige those residuals and then combine the results from the regression as well as from the Kriging interpolation.

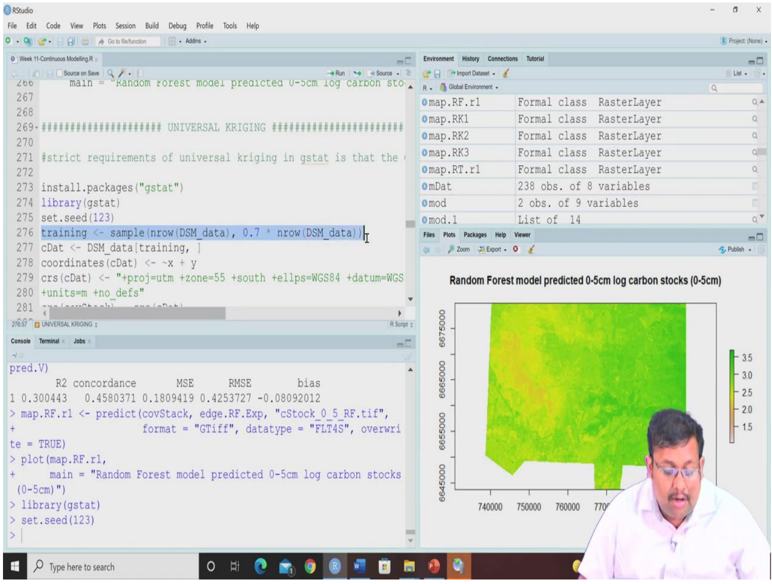
So, the final output will be the combination of the model output plus the Kriging interpolation output. Now, this is the in general this is called regression Kriging we are going to talk about regression Kriging in next week of lectures, but just remember that in this case, if we are very much strict about the model, that is this regression model, which should be linear then it will be called a Universal model problem.

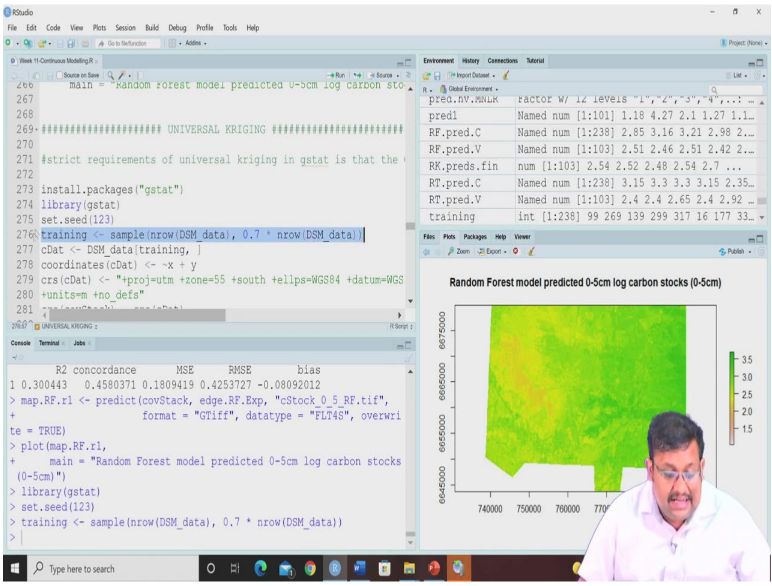
So, Universal Kriging approach, but, if that prediction model is variable, maybe most of the time you will see that nonlinear models are more you know, more appropriate for handling or for predicting the soil properties, then we can tell that, in that case it will be a true regression Kriging but the difference between a regression Kriging and then a Universal Kriging is in case of regression Kriging the prediction model could be any way you know, it could be nonlinear, but in case of Universal Kriging the prediction model will be always linear.

So, I will show you now, how to you know deal with these Universal Kriging in R. So, remember that in case of Universal Kriging, there is a strict requirement of Universal you know Universal Kriging in gstat is that the coordinate reference system of the point data and the covariate must be exactly the same. So, for maintaining, these coordinate, the coordinate reference system for these for the of the point data set as well as the covariate data must be same, so, we need to periodically check this if they are same or not.

(Refer Slide Time: 21:18)

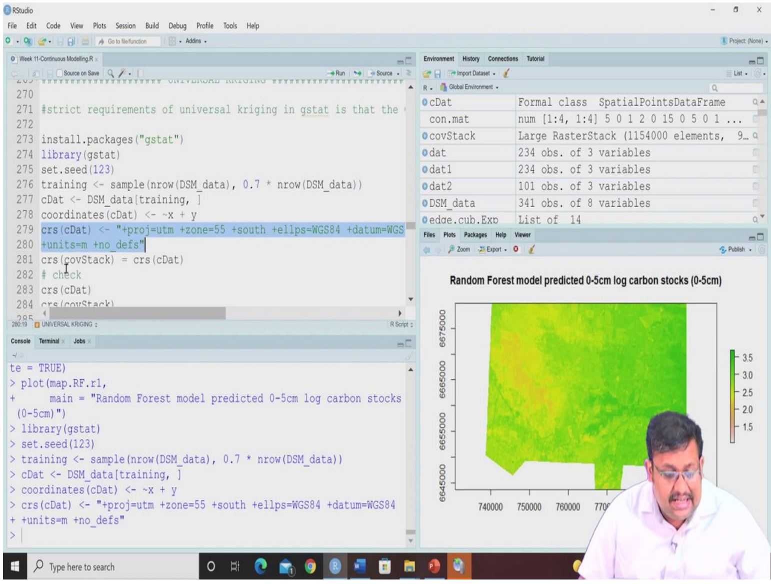


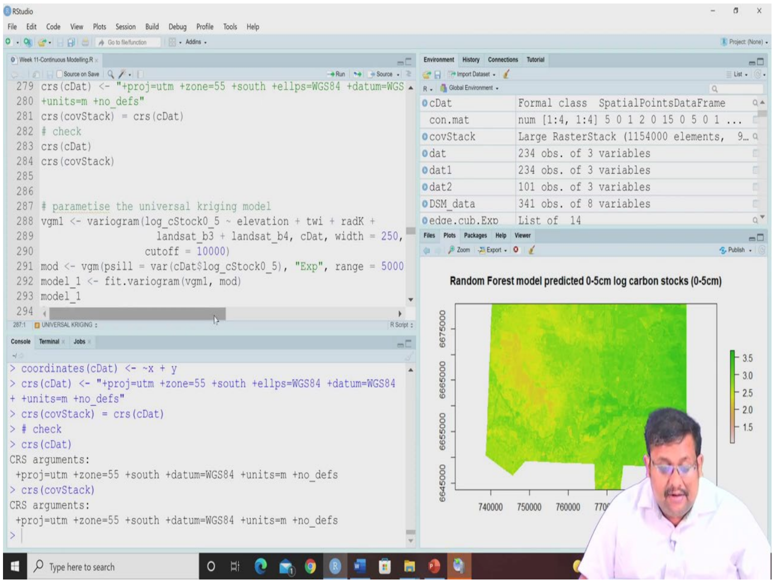


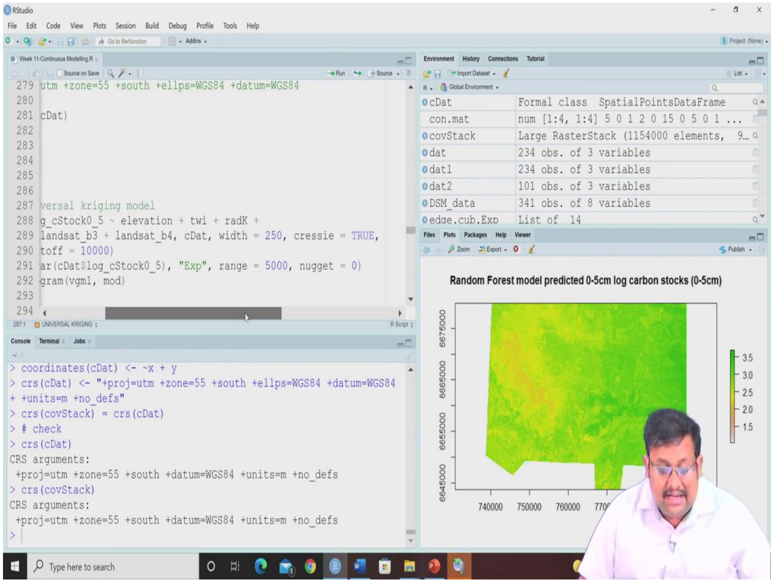


So, for this we have to install this gstat package, and then we have to call this library gstat. And then again, setting the seed 123 again just like previously, we are going to select the model and the training model let us call it as a calibration dataset. See that, and then we are going to see the coordinate of the see that so, of course, it will be we can see that we are going to assign these coordinates that utm zone 55 south WGS 84.

(Refer Slide Time: 21:57)

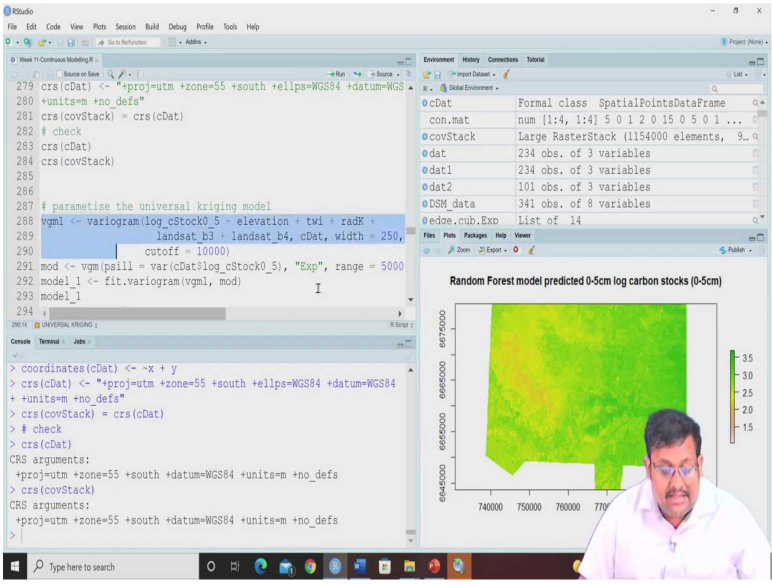


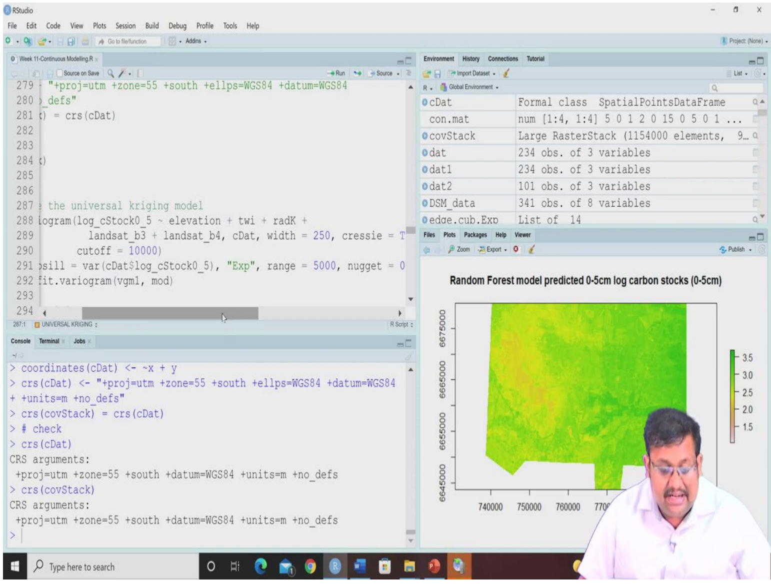


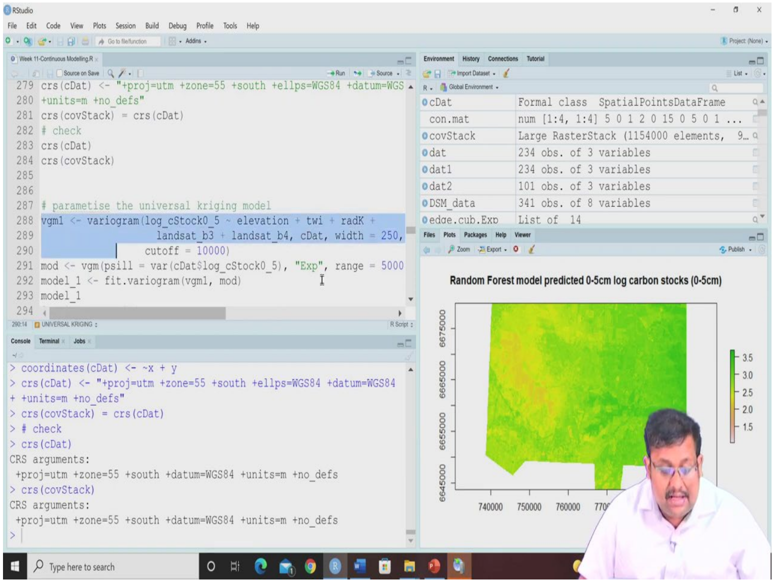


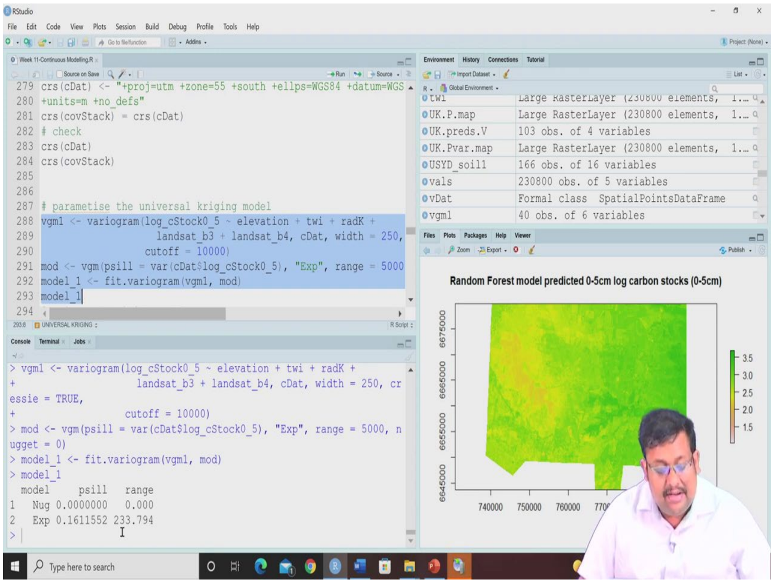
So, let us see whether our coordinate reference system of this calibration data set matches well with the coordinate reference system of the of the stack covariate or not. So, if we give this command we will cDat yes, they are both same. So, if we check further the coordinate reference system of cDat and then coordinate reference system of covStack we will see that in both the condition there will be zone 55 South datum a WGS 84 in both cases.

(Refer Slide Time: 22:30)




"+proj=utm +zone=55 +south +ellps=WGS84 +datum=WGS84
_defs"
) = crs(cDat)
the universal kriging model
ogram(log_cStock0_5 ~ elevation + twi + radK +
landsat_b3 + landsat_b4, cDat, width = 250, cressie = T
cutoff = 10000)
sill = var(cDat$log_cStock0_5), "Exp", range = 5000, nugget = 0
fit.variogram(vgm1, mod)
> coordinates(cDat) <- ~x + y
> crs(cDat) <- "+proj=utm +zone=55 +south +ellps=WGS84 +datum=WGS84
+ +units=m +no_defs"
> crs(covStack) = crs(cDat)
> # check
> crs(cDat)
CRS arguments:
+proj=utm +zone=55 +south +datum=WGS84 +units=m +no_defs
> crs(covStack)
CRS arguments:
+proj=utm +zone=55 +south +datum=WGS84 +units=m +no_defs
cDat Formal class SpatialPointsDataFrame
con.mat num [1:4, 1:4] 5 0 1 2 0 15 0 5 0 1 ...
covStack Large RasterStack (1154000 elements,
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
Random Forest model predicted 0-5cm log carbon stocks (0-5cm)

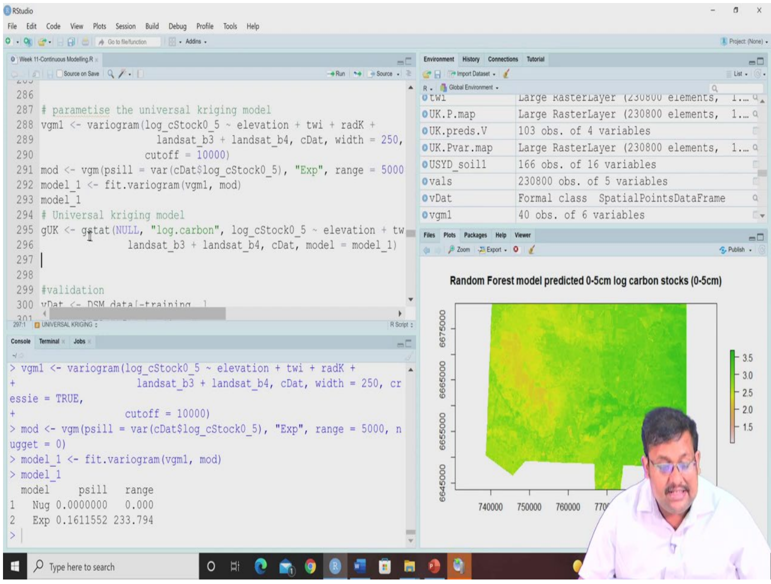

crs(cDat) <- "+proj=utm +zone=55 +south +ellps=WGS84 +datum=WGS
+units=m +no_defs"
crs(covStack) = crs(cDat)
# check
crs(cDat)
crs(covStack)
# parametise the universal kriging model
vgm1 <- variogram(log_cStock0_5 ~ elevation + twi + radK +
landsat_b3 + landsat_b4, cDat, width = 250,
cutoff = 10000)
mod <- vgm(psill = var(cDat$log_cStock0_5), "Exp", range = 5000
model_1 <- fit.variogram(vgm1, mod)
model_1
Random Forest model predicted 0-5cm log carbon stocks (0-5cm)

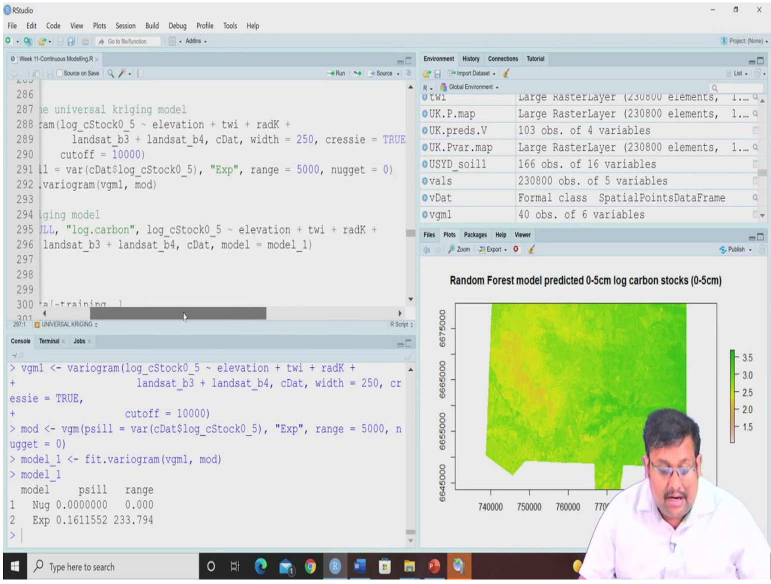

> vgm1 <- variogram(log_cStock0_5 ~ elevation + twi + radK +
+ landsat_b3 + landsat_b4, cDat, width = 250, cr
essie = TRUE,
+ cutoff = 10000)
> mod <- vgm(psill = var(cDat$log_cStock0_5), "Exp", range = 5000, n
ugget = 0)
> model_1 <- fit.variogram(vgm1, mod)
> model_1
model psill range
1 Nug 0.0000000 0.000
2 Exp 0.1611552 233.794
UK.P.map Large RasterLayer (230800 elements,
UK.preds.V 103 obs. of 4 variables
UK.Pvar.map Large RasterLayer (230800 elements,
USYD_soil1 166 obs. of 16 variables
vals 230800 obs. of 5 variables
vDat Formal class SpatialPointsDataFrame
vgm1 40 obs. of 6 variables
Random Forest model predicted 0-5cm log carbon stocks (0-5cm)

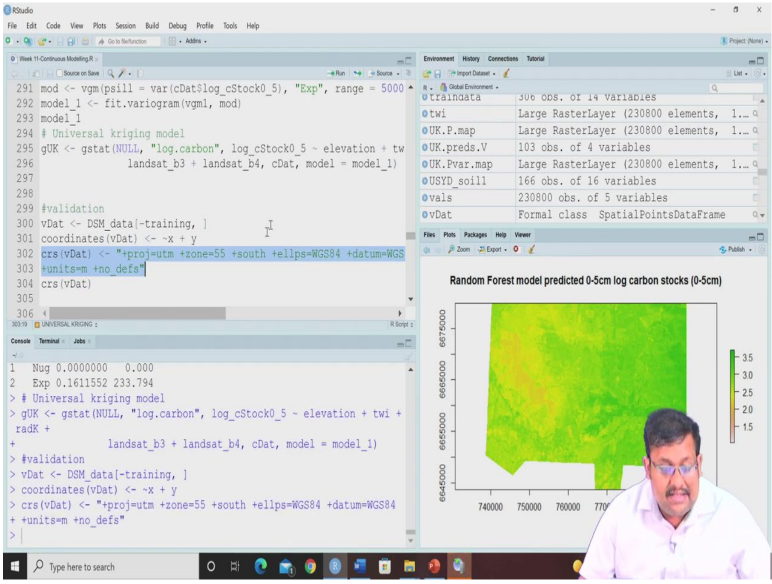
Now, next thing is to parameterize the Universal Kriging model. So, first we have to fit the variogram we have already seen how to fit the variogram so we are going to use these variogram function. Our target is logged in… log converted carbonStocks 0 to 5 centimetre our variables are elevation twi, radk then landsat b3, landsat b4 our model is calibration data, our data set its calibration data, width is again default 250 cressie variogram cutoff values is 10,000 and then here, we are again going to fix, the fit the model using an exponential model, and then let us see how this variogram parameters will come.

So, here you can see negative 0 which is almost… which is ideal, and then here are the for a exponential model, we are gating the partial sill. So, the total sill will also be 0.16 and range parameter is 233. So, from there we can have any idea about the variogram parameters. And based on these variogram parameters, we are going to feed the Universal Kriging model.

(Refer Slide Time: 23:46)



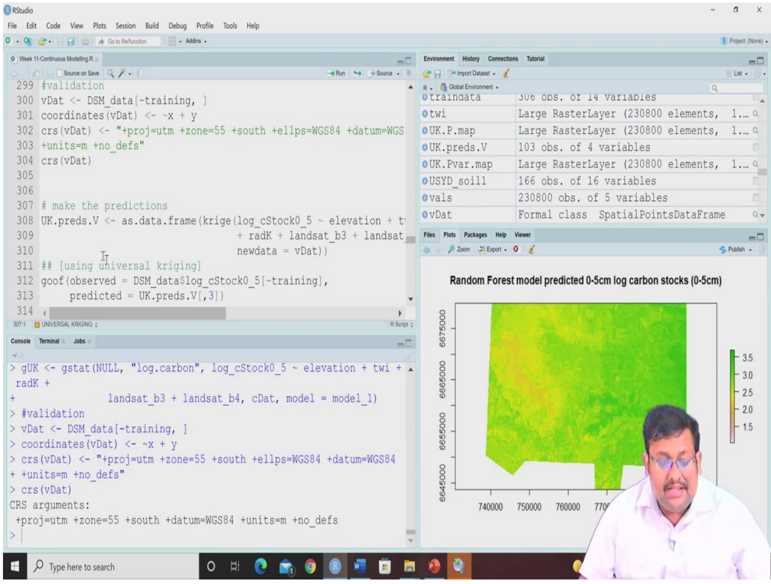


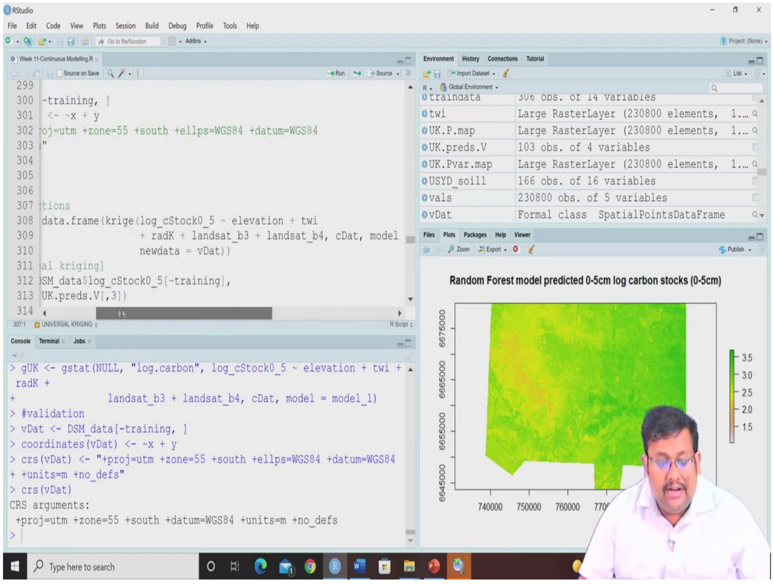


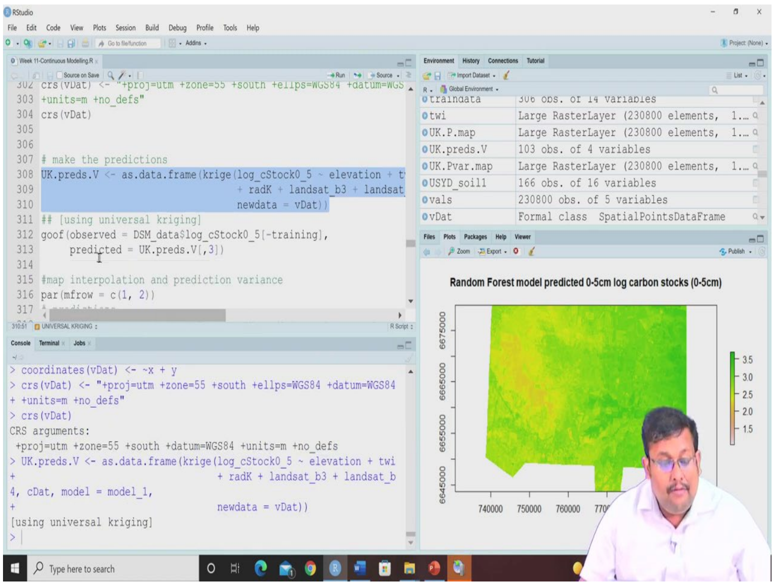
So, for fitting the Universal Kriging model, the function is gstat function we are going to use our target is a natural log converted carbonStock, our predictors are elevation twi, radk, landsat b3, landsat b4 our data set is calibration data set our model is model 1 which we have just fitted using the exponential function.

So, once we have this fitted this model, let us validate so validation data set is vDat again, minus training samples coordinates again we are x we are, instructing R that these x and y you should understand these add the coordinates. So, we are assigning these coordinates same just like previously utm zone beautified south and from there WGS 84 and then we want to see the coordinate reference system, so you will see that utm zone 35 south WGS 84.

(Refer Slide Time: 24:45)

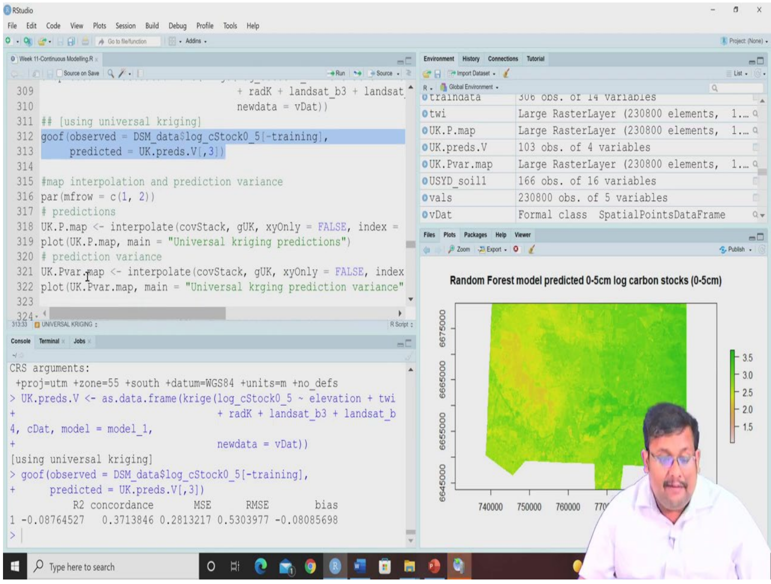


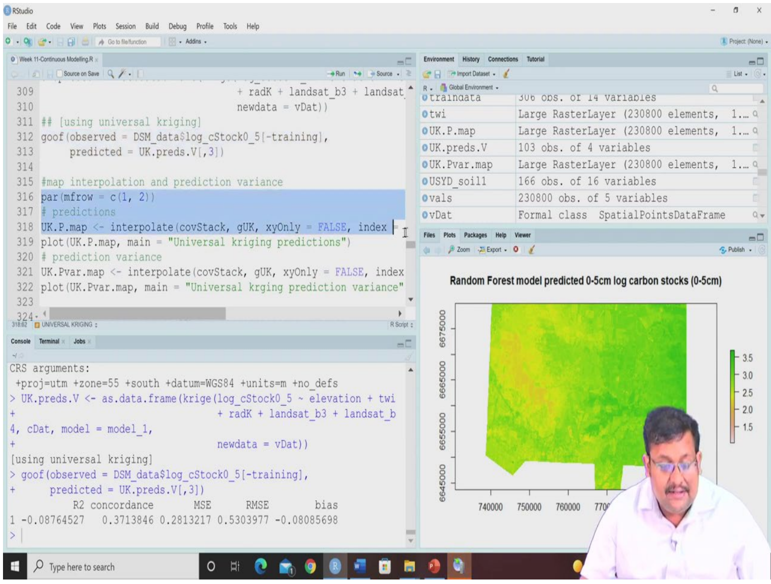


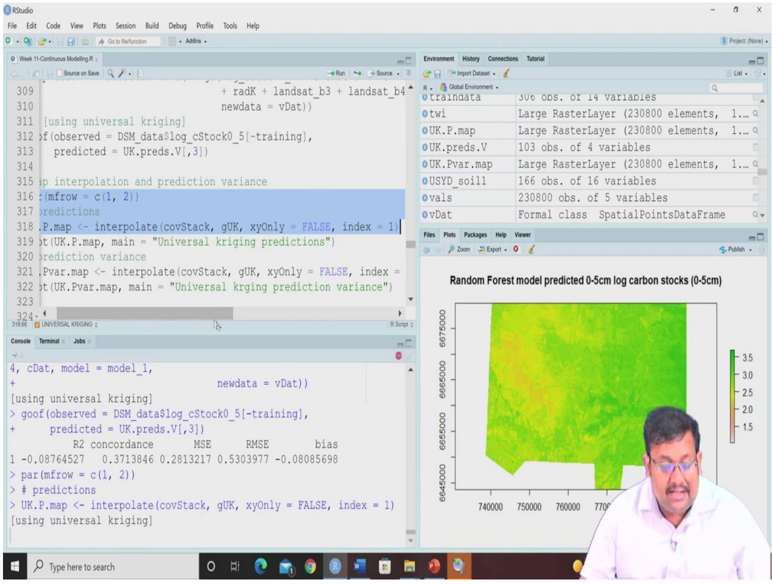


And now we can predict based on the predict this validation data based on our Universal Kriging model which have already fitted, so we are going to you know, predict the validation data set based on the previously fitted model.

(Refer Slide Time: 25:13)

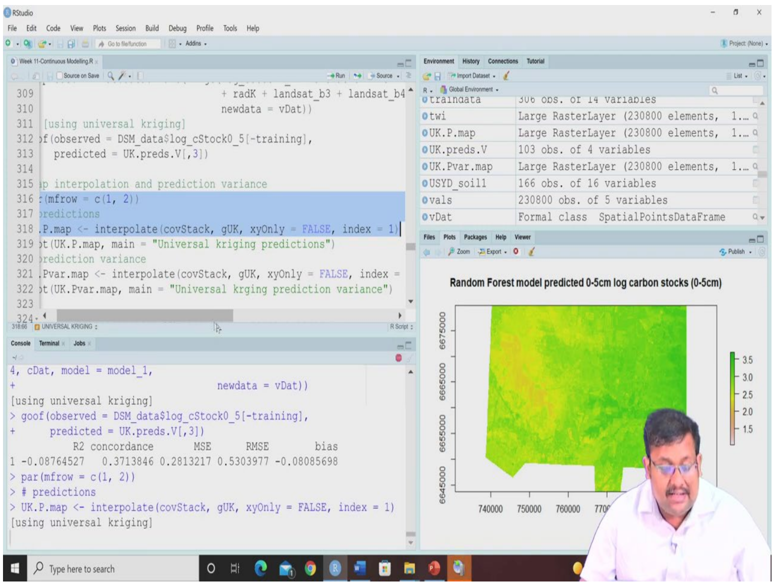


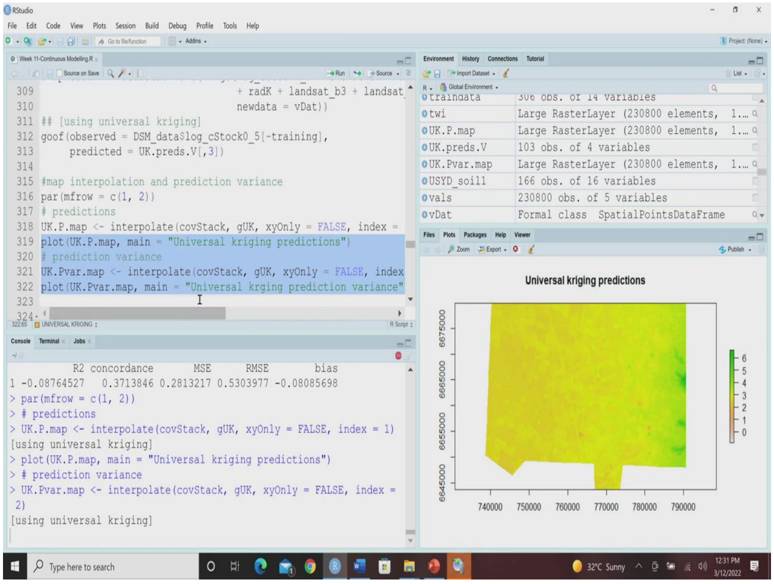


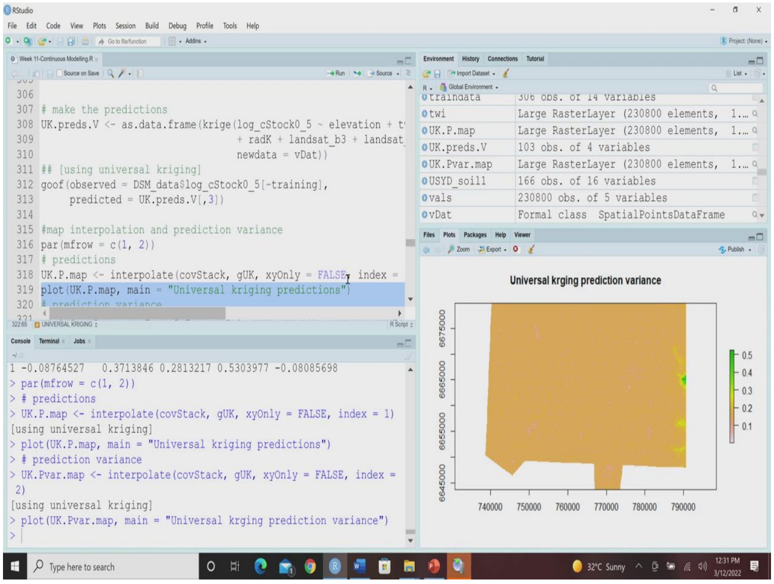


Now, we are going to see the goodness of fit statistics of the observed as well as predicted results. So, you can see that the here the goodness of fit statistics is not that high we are seeing some extrapolation. So, that is why we are getting the negative R square negative R square value, but, you know, you will have it varies from data set to data set in some data set you will may have some good R squared values in this just in this case, we are not getting very good result that shows the prior the linear model may not be the ideal model to handle this dataset. So, once we have these now, we can map the interpolation and prediction variance. So, first we want to predict based on this Universal Kriging model, and once we do that we can produce.

(Refer Slide Time: 26:13)



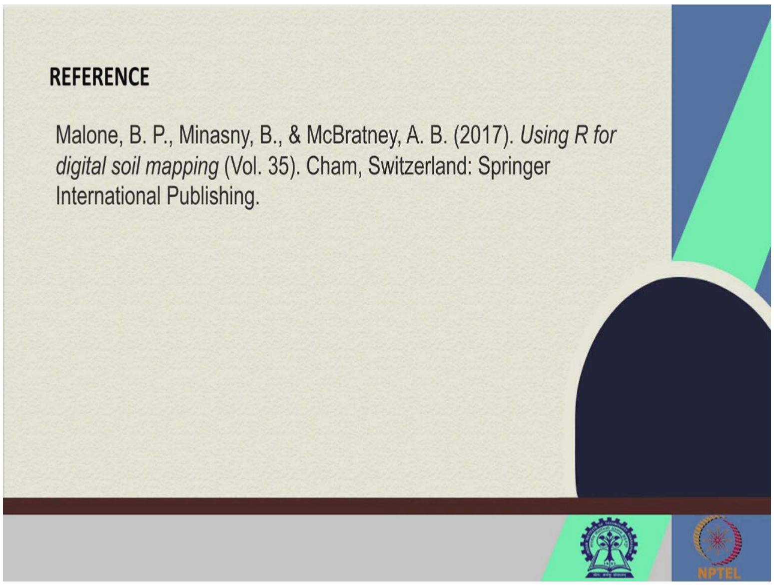




So, let us first remove this plot so, that, so first you have to predict or interpolate based on these covStack and then you are using these gUK or Universal Kriging and so here you are giving the index 1 for producing the predictions, you can also produce the prediction variance by using the same by changing only the index and you can see the both the plots. So, let us first run all these and we will see their plot together.

So, this is the Universal Kriging predictions and it will do some calculation and this is the Universal Kriging prediction variance. So, not only you will get the prediction result, but at the same time you will get the prediction variance also. So, this is how you use the Universal Kriging to produce the hybrid results.

(Refer Slide Time: 27:39)



So, guys, I hope that the things are you have got some good knowledge from this lecture and you will be now able to run these codes and grow your confidence on these type of approaches, where we have already seen the cubist which is an important machine learning approach we have already seen the random forests which is an advanced machine learning approach and we have also seen an hybrid approach called Universal Kriging.

So, let us wrap up our lecture here. And let us wrap up our week 11 and in week 12 we will be discussing the Regression Kriging which is the most advanced, one of the advanced hybrid approach for digital soil mapping. And then we will be also discussing how to deal with the categorical models using R so please stay tuned and let us meet in the lectures of week 12. Thank you.