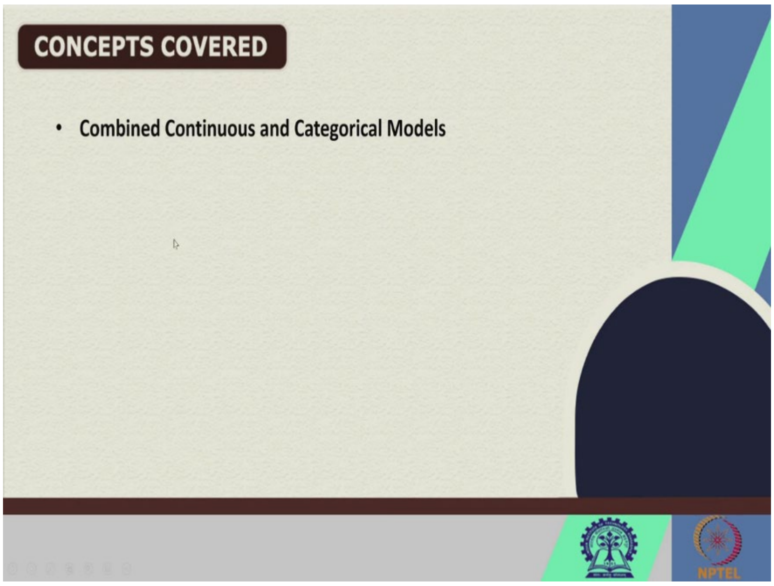
Machine Learning for Soil and Crop Management
Professor Somsubhra Chakraborty
Agricultural and Food Engineering Department
Indian Institute of Technology Kharagpur
Lecture 59
Digital Soil Mapping with Categorical Variables (Contd.)

Welcome friends to this 59th lecture of NPTEL online certification course of Machine Learning for Soil and Crop Management. In this week we are talking about Digital Soil Mapping with the Categorical Variables. In our previous lectures, we have already discussed about different types of categorical models like multinomial logistic model, then C5 decision tree and we have seen what are the important steps of developing the random forest.

We have seen how these bootstrap samples are developed for creating a tree in the random forest and what is M try and what are the features of this random forest model and then how you know out of bag sample is being used for validating the model performance we have already seen.

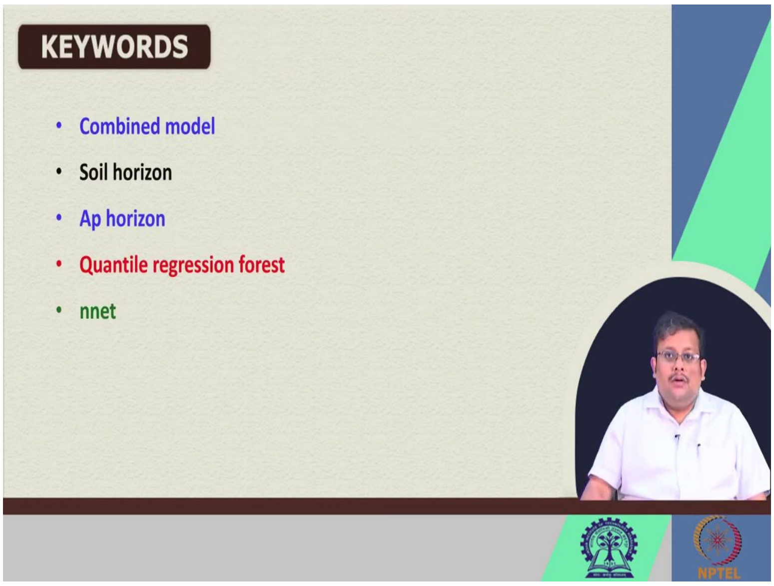
Now, in this lecture first we are going to run the script, R script of random forest and then we are going to discuss another very important topic that is combining the regression as well as classification model. So, that is called combined model. Now, let us first see how to execute the random forest codes and then we will go for the combined model discussion.

(Refer Slide Time: 01:57)



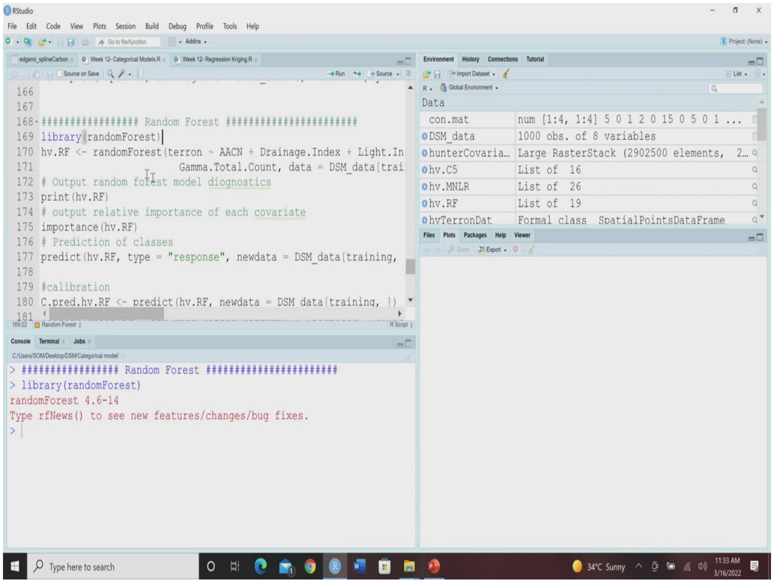
So, these are the concepts which we are going to cover in this lecture apart from running the script of the random forest regression model categorical model, we are also going to discuss the combined continuous and categorical model.

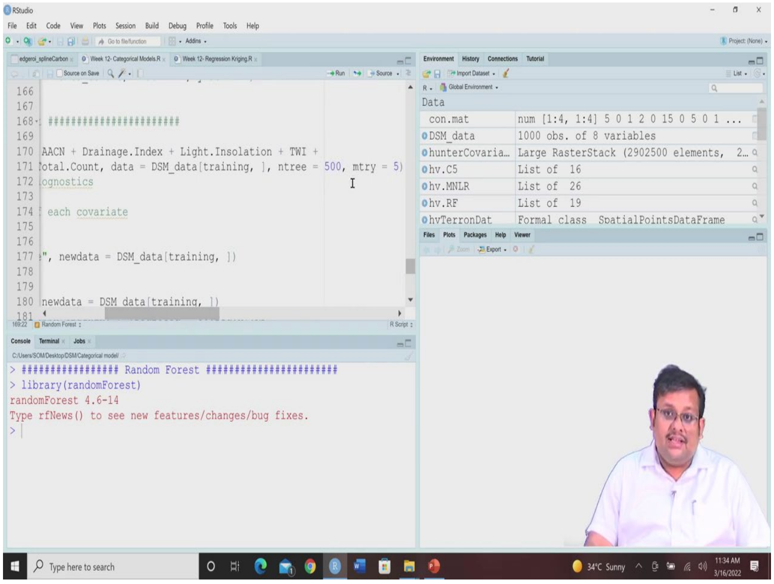
(Refer Slide Time: 02:12)



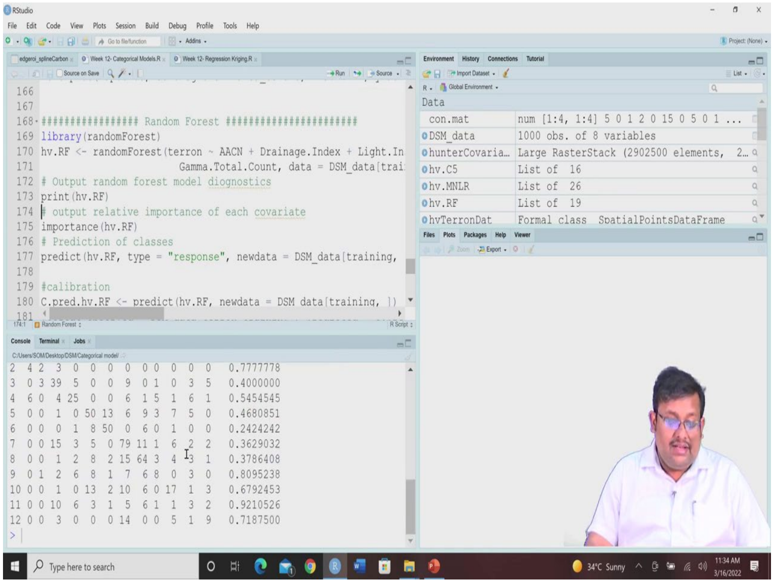
So, in this lecture these are the keywords like Combined model, Soil Horizon, AP horizon, Quantile regression forest, nnet these are the 5 keywords which we are going to discuss in this lecture.

(Refer Slide Time: 02:25)

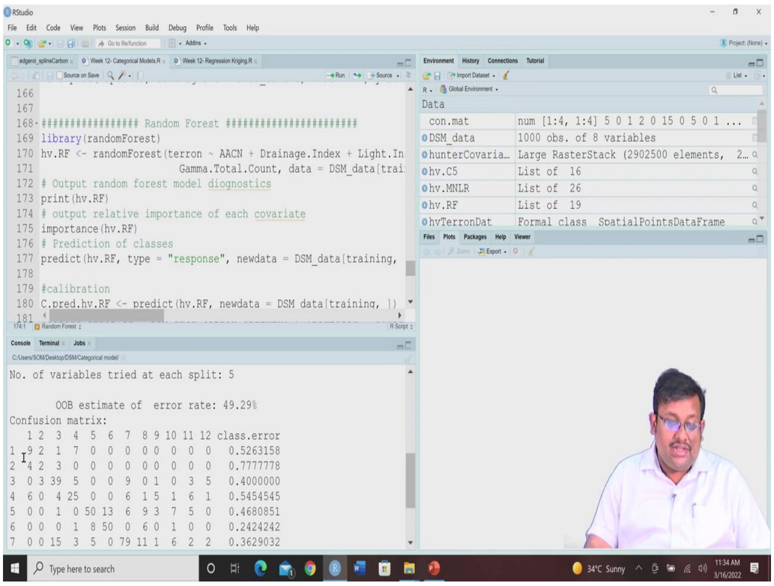




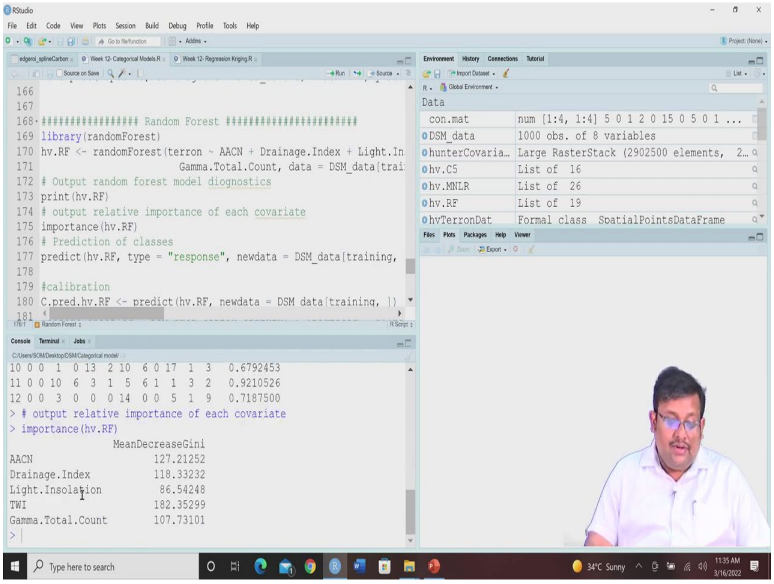
AACN + Drainage.Index + Light.Insolation + TWI +
Total.Count, data = DSM_data[training,], ntree = 500, mtry = 5)
ognostics
each covariate
", newdata = DSM_data[training,])
newdata = DSM_data[training,])
> ################ Random Forest ######################
> library(randomForest)
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.
con.mat num [1:4, 1:4] 5 0 1 2 0 15 0 5 0 1 ...
DSM_data 1000 obs. of 8 variables
hunterCovaria… Large RasterStack (2902500 elements, 2…
hv.C5 List of 16
hv.MNLR List of 26
hv.RF List of 19
hvTerronDat Formal class SpatialPointsDataFrame



################ Random Forest ######################
library(randomForest)
hv.RF <- randomForest(terron ~ AACN + Drainage.Index + Light.In
Gamma.Total.Count, data = DSM_data[trai
# Output random forest model diognostics
print(hv.RF)
# output relative importance of each covariate
importance(hv.RF)
# Prediction of classes
predict(hv.RF, type = "response", newdata = DSM_data[training,
#calibration
C.pred.hv.RF <- predict(hv.RF, newdata = DSM_data[training,])
2 4 2 3 0 0 0 0 0 0 0 0 0 0.7777778
3 0 3 39 5 0 0 9 0 1 0 3 5 0.4000000
4 6 0 4 25 0 0 6 1 5 1 6 1 0.5454545
5 0 0 1 0 50 13 6 9 3 7 5 0 0.4680851
6 0 0 0 1 8 50 0 6 0 1 0 0 0.2424242
7 0 0 15 3 5 0 79 11 1 6 2 2 0.3629032
8 0 0 1 2 8 2 15 64 3 4 3 1 0.3786408
9 0 1 2 6 8 1 7 6 8 0 3 0 0.8095238
10 0 0 1 0 13 2 10 6 0 17 1 3 0.6792453
11 0 0 10 6 3 1 5 6 1 1 3 2 0.9210526
12 0 0 3 0 0 0 14 0 0 5 1 9 0.7187500



################ Random Forest ######################
library(randomForest)
hv.RF <- randomForest(terron ~ AACN + Drainage.Index + Light.In
Gamma.Total.Count, data = DSM_data[trai
# Output random forest model diognostics
print(hv.RF)
# output relative importance of each covariate
importance(hv.RF)
# Prediction of classes
predict(hv.RF, type = "response", newdata = DSM_data[training,
#calibration
C.pred.hv.RF <- predict(hv.RF, newdata = DSM_data[training,])
No. of variables tried at each split: 5
OOB estimate of error rate: 49.29%
Confusion matrix:
1 2 3 4 5 6 7 8 9 10 11 12 class.error
1 9 2 1 7 0 0 0 0 0 0 0 0 0.5263158
2 4 2 3 0 0 0 0 0 0 0 0 0 0.7777778
3 0 3 39 5 0 0 9 0 1 0 3 5 0.4000000
4 6 0 4 25 0 0 6 1 5 1 6 1 0.5454545
5 0 0 1 0 50 13 6 9 3 7 5 0 0.4680851
6 0 0 0 1 8 50 0 6 0 1 0 0 0.2424242
7 0 0 15 3 5 0 79 11 1 6 2 2 0.3629032

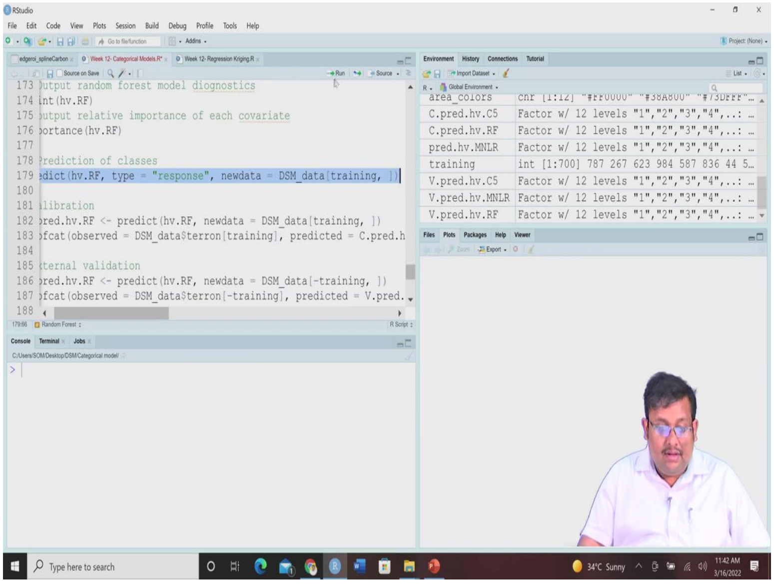


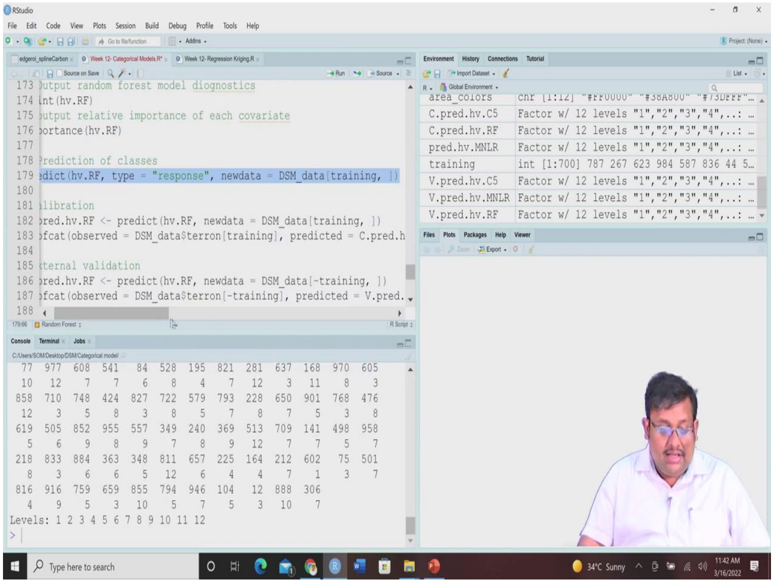
For executing the random forest classification model. Again, you have to call the library random forest, we have already installed this random forest previously, so I am just going to, call this library random forest and as you know our target is terron. And then we are using these covariates like Altitude Above Channel Network, Drainage Index, Light Insulation, TWI, Gamma Total Count all these for predicting the terron class and our data is of course, the training data. And here you can see we are specifying the ntree that is we are going to build 500 number of trees with a number of mtry equal to 5.

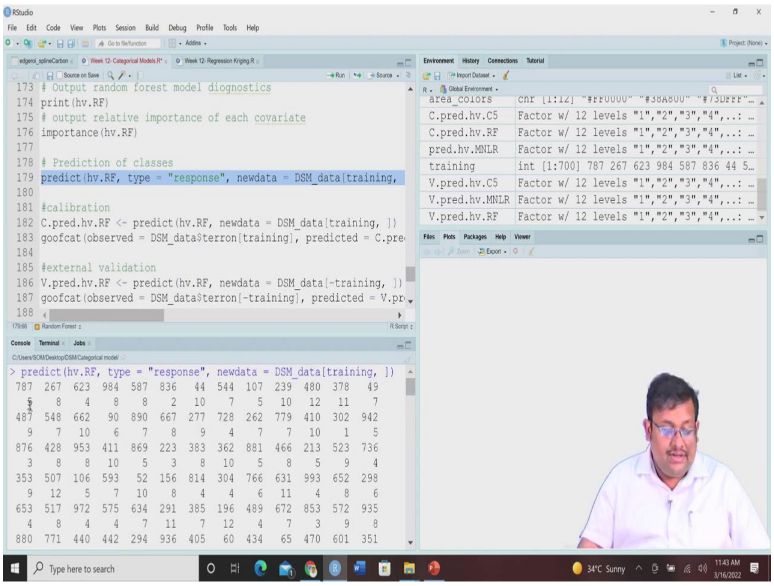
So, in each tree random tree, we are going to use 5 randomly selected features for splitting criteria for identifying the splitting criteria. So, here you can see the mtry is 5 and ntree is 500. So, let us just run it and once we run it, let us see the output of the random forest model diagnostics. So, we are using this print function to see the model output and you can see here the, this is the confusion matrix for all the 12 classes. And of course, the class header is given here.

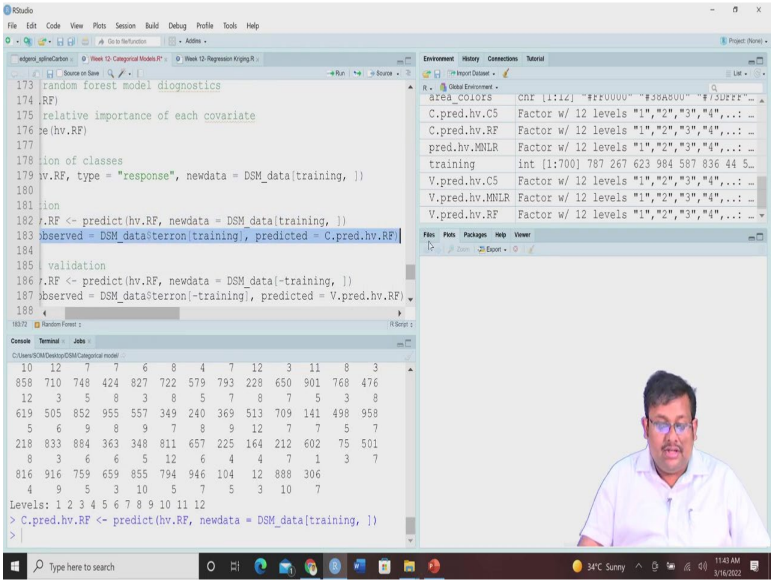
Now, the output if we want to check the output of the relative importance of each of these 5 covariates, which we are using like AACN, Drainage Index, Light Insulation and then TWI and a Gamma Total Count these 5 we want to see their importance in the classification. Again, we are going to use this important function and then we are going to see that according to the mean decrease in Gini impurity criteria, you can see here that TWI is having the highest impact on this model, random forest categorical model followed by Altitude Above Channel Network and then Drainage Index and Gamma Total Count.

(Refer Slide Time: 04:43)

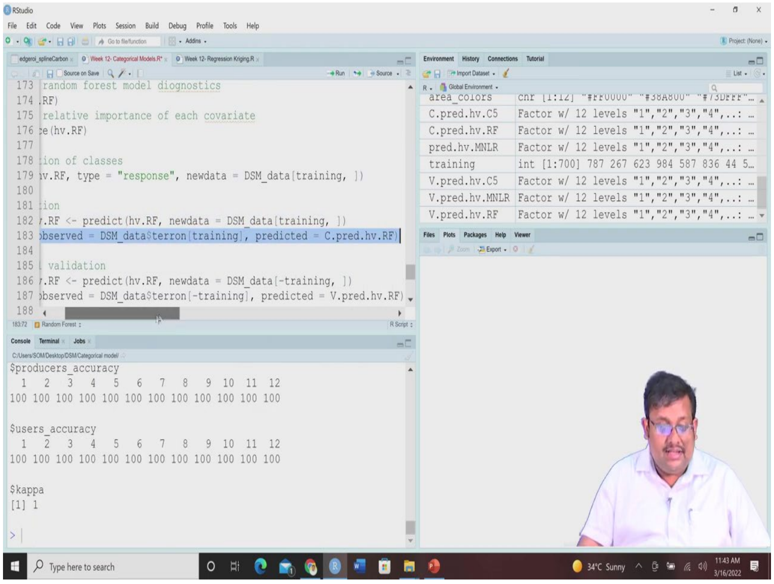




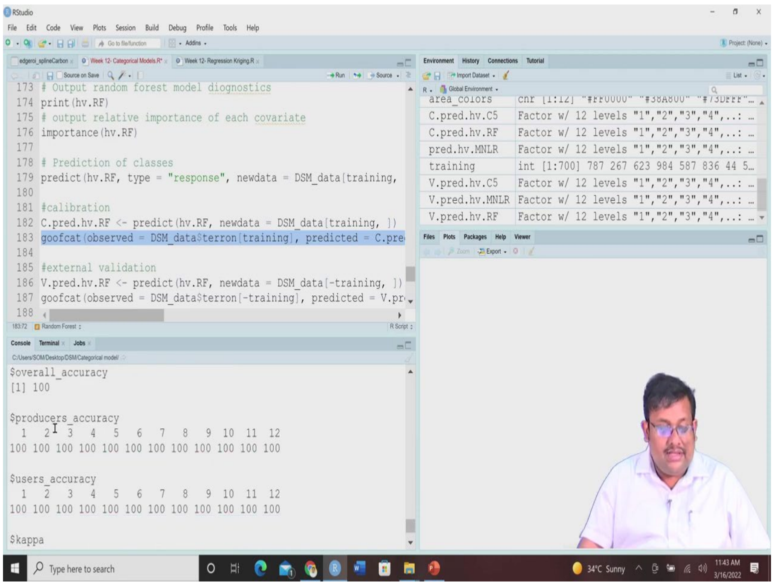




173 random forest model diagnostics
174 .RF)
175 relative importance of each covariate
176 e(hv.RF)
177
178 ion of classes
179 v.RF, type = "response", newdata = DSM_data[training,])
180
181 ion
182 .RF <- predict(hv.RF, newdata = DSM_data[training,])
183 bserved = DSM_data$terron[training], predicted = C.pred.hv.RF)
184
185 validation
186 .RF <- predict(hv.RF, newdata = DSM_data[-training,])
187 bserved = DSM_data$terron[-training], predicted = V.pred.hv.RF)
188
area_colors chr [1:12] "#FF0000" "#38A800" "#73DFFF"...
C.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",..:
C.pred.hv.RF Factor w/ 12 levels "1","2","3","4",..:
pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",..:
training int [1:700] 787 267 623 984 587 836 44 5...
V.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",..:
V.pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",..:
V.pred.hv.RF Factor w/ 12 levels "1","2","3","4",..:
10 12 7 7 6 8 4 7 12 3 11 8 3
858 710 748 424 827 722 579 793 228 650 901 768 476
12 3 5 8 3 8 5 7 8 7 5 3 8
619 505 852 955 557 349 240 369 513 709 141 498 958
5 6 9 8 9 7 8 9 12 7 7 5 7
218 833 884 363 348 811 657 225 164 212 602 75 501
8 3 6 6 5 12 6 4 4 7 1 3 7
816 916 759 659 855 794 946 104 12 888 306
4 9 5 3 10 5 7 5 3 10 7
Levels: 1 2 3 4 5 6 7 8 9 10 11 12
> C.pred.hv.RF <- predict(hv.RF, newdata = DSM_data[training,])



173 random forest model diagnostics
174 .RF)
175 relative importance of each covariate
176 e(hv.RF)
177
178 ion of classes
179 v.RF, type = "response", newdata = DSM_data[training,])
180
181 ion
182 .RF <- predict(hv.RF, newdata = DSM_data[training,])
183 bserved = DSM_data$terron[training], predicted = C.pred.hv.RF)
184
185 validation
186 .RF <- predict(hv.RF, newdata = DSM_data[-training,])
187 bserved = DSM_data$terron[-training], predicted = V.pred.hv.RF)
188
$producers_accuracy
1 2 3 4 5 6 7 8 9 10 11 12
100 100 100 100 100 100 100 100 100 100 100 100
$users_accuracy
1 2 3 4 5 6 7 8 9 10 11 12
100 100 100 100 100 100 100 100 100 100 100 100
$kappa
[1] 1



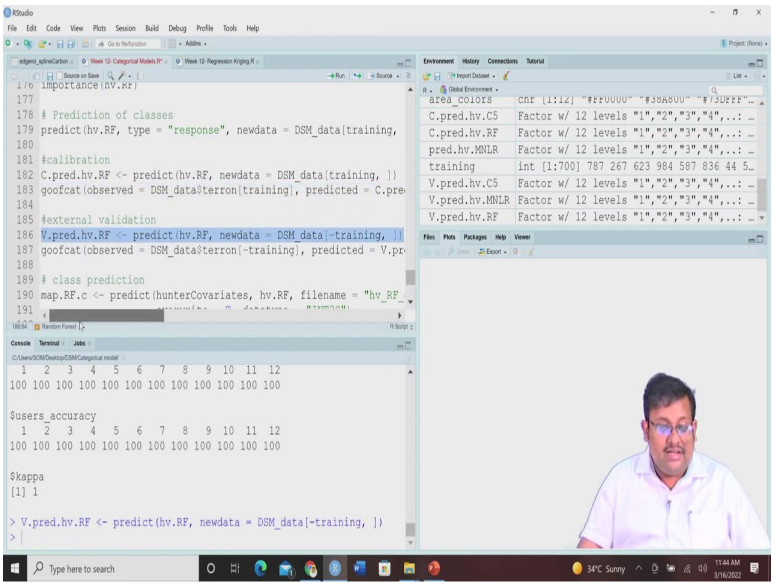
173 # Output random forest model diagnostics
174 print(hv.RF)
175 # output relative importance of each covariate
176 importance(hv.RF)
177
178 # Prediction of classes
179 predict(hv.RF, type = "response", newdata = DSM_data[training,
180
181 #calibration
182 C.pred.hv.RF <- predict(hv.RF, newdata = DSM_data[training,])
183 goofcat(observed = DSM_data$terron[training], predicted = C.pre
184
185 #external validation
186 V.pred.hv.RF <- predict(hv.RF, newdata = DSM_data[-training,])
187 goofcat(observed = DSM_data$terron[-training], predicted = V.pr
188
$overall_accuracy
[1] 100
$producers_accuracy
1 2 3 4 5 6 7 8 9 10 11 12
100 100 100 100 100 100 100 100 100 100 100 100
$users_accuracy
1 2 3 4 5 6 7 8 9 10 11 12
100 100 100 100 100 100 100 100 100 100 100 100
$kappa

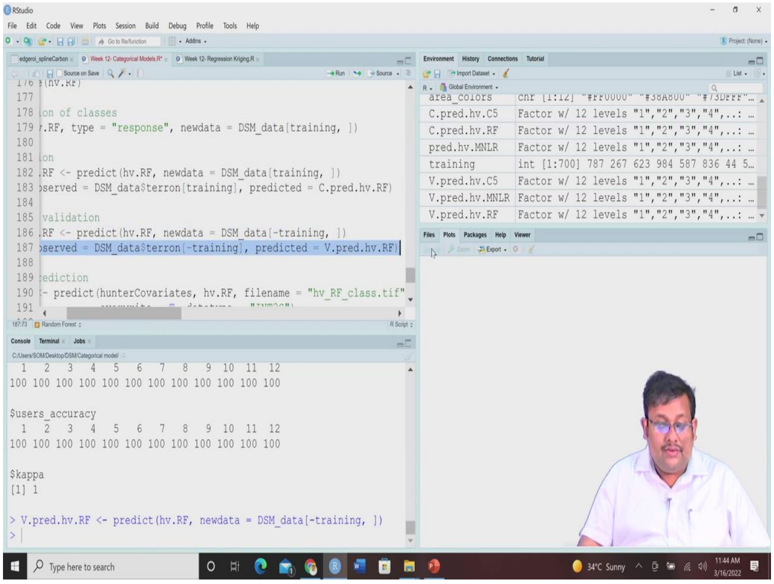
So, now we have predicted the importance we have seen the importance of all the variables. Now, the next step is to use the prediction of the classes. So, for the prediction of the class, we are going to use this type equal to response and then we are going to use the same model to see the prediction of all the classes.

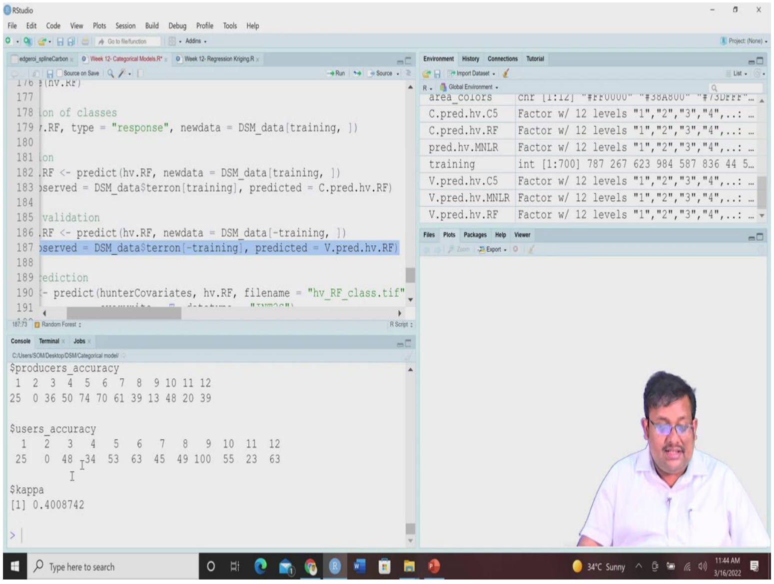
So, you can see that the 787 observation will have the fifth terron and then 267th will have the eighth terron and then 623 will be having the fourth terron and so on. And then we are going to see the goodness of fit for the calibration samples. So, you can see here we are predicting based on the training samples, the predicted values, and at the same time, we are going to run this goofcat function to see the goodness of fit statistics.

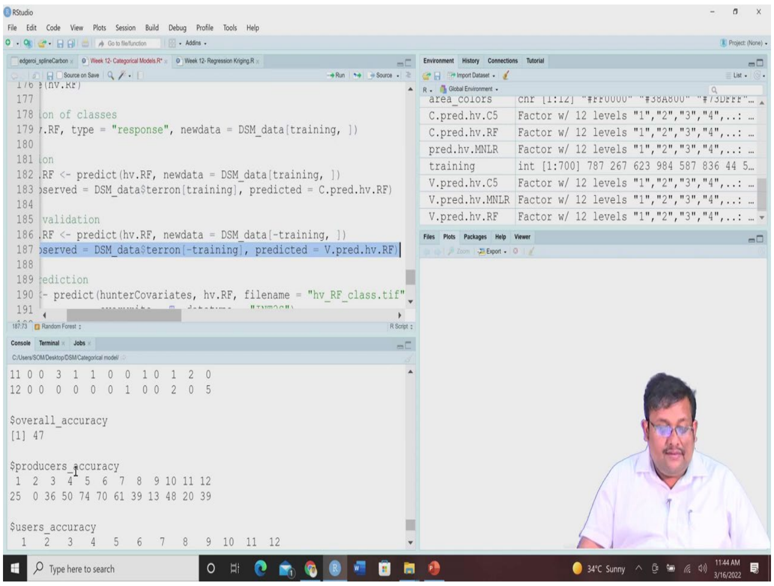
So, here you can see that this is the confusion matrix. And of course, the overall accuracy then producer's accuracy and users accuracy and also the Kappa coefficient. So, it looks like from the calibration model the model is almost perfect. Now, to check whether this calibration model is okay or not, we are going to use the categorical validation or goodness of fit of the validation data. So, we are going to use these validation data that is minus training samples.

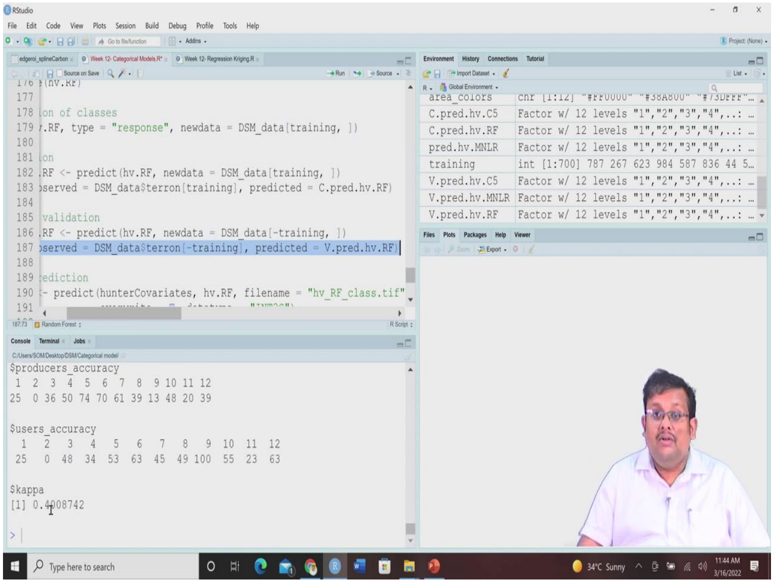
(Refer Slide Time: 06:25)




177
178 ion of classes
179).RF, type = "response", newdata = DSM_data[training,])
181 ion
182 .RF <- predict(hv.RF, newdata = DSM_data[training,])
183 bserved = DSM_data$terron[training], predicted = C.pred.hv.RF)
185 validation
186 .RF <- predict(hv.RF, newdata = DSM_data[-training,])
187 bserved = DSM_data$terron[-training], predicted = V.pred.hv.RF)
189 ediction
190 |- predict(hunterCovariates, hv.RF, filename = "hv_RF_class.tif"
1 2 3 4 5 6 7 8 9 10 11 12
100 100 100 100 100 100 100 100 100 100 100 100
$users_accuracy
1 2 3 4 5 6 7 8 9 10 11 12
100 100 100 100 100 100 100 100 100 100 100 100
$kappa
[1] 1
> V.pred.hv.RF <- predict(hv.RF, newdata = DSM_data[-training,])
C.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",..:
C.pred.hv.RF Factor w/ 12 levels "1","2","3","4",..:
pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",..:
training int [1:700] 787 267 623 984 587 836 44 5...
V.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",..:
V.pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",..:
V.pred.hv.RF Factor w/ 12 levels "1","2","3","4",..:

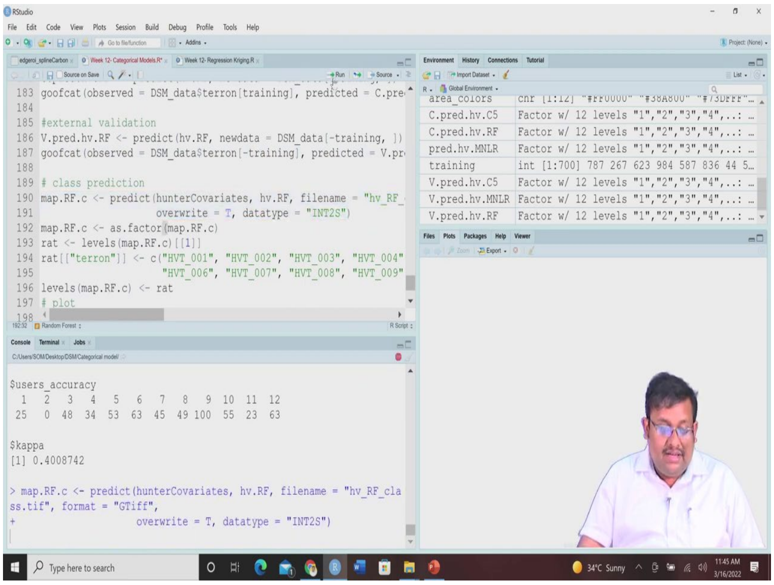

$producers_accuracy
1 2 3 4 5 6 7 8 9 10 11 12
25 0 36 50 74 70 61 39 13 48 20 39
$users_accuracy
1 2 3 4 5 6 7 8 9 10 11 12
25 0 48 34 53 63 45 49 100 55 23 63
$kappa
[1] 0.4008742

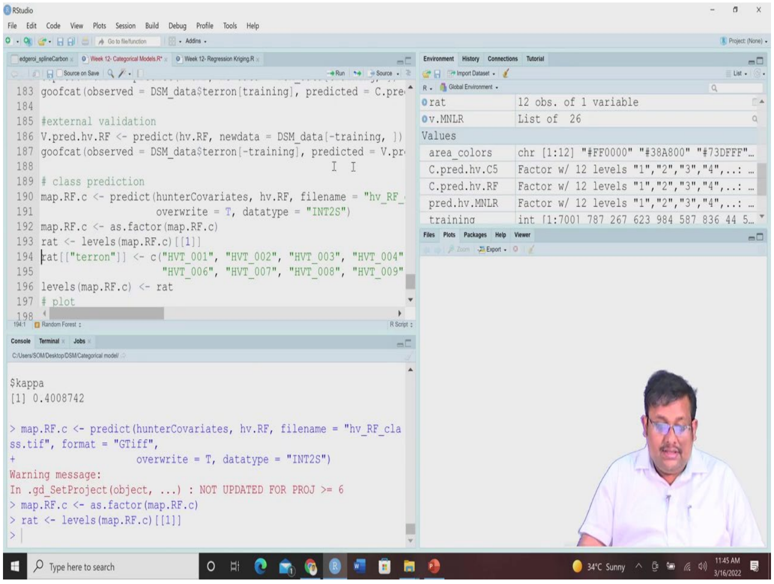

11 0 0 3 1 1 0 0 1 0 1 2 0
12 0 0 0 0 0 0 1 0 0 2 0 5
$overall_accuracy
[1] 47
$producers_accuracy
1 2 3 4 5 6 7 8 9 10 11 12
25 0 36 50 74 70 61 39 13 48 20 39
$users_accuracy
1 2 3 4 5 6 7 8 9 10 11 12

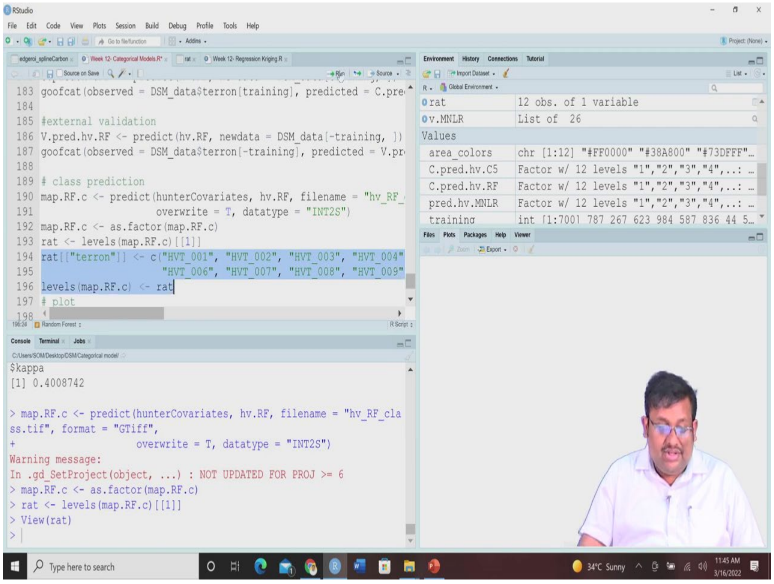


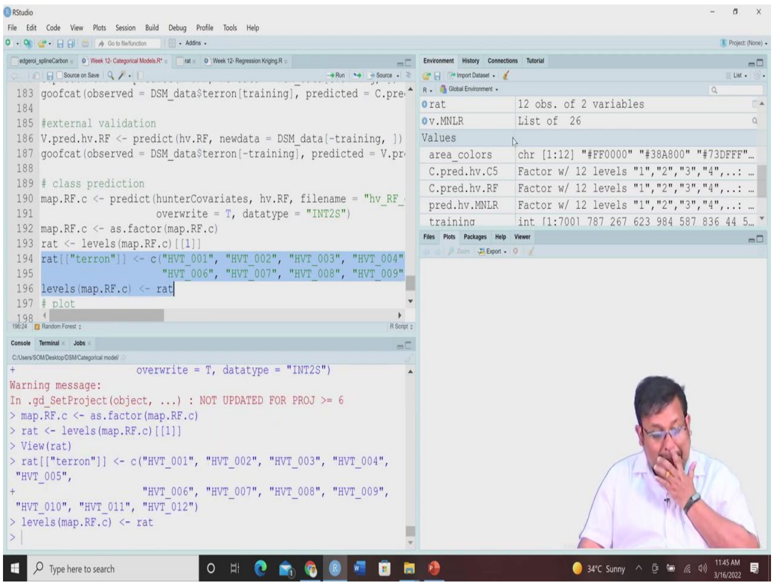
And let us see how it looks like. So, first we are going to predict the values and then we are going to see the goodness of fit statistics. So, will see that the overall accuracy is 47 and the Kappa coefficient is now 0.40. So, there is some difference between the calibration statistics as well as the validation statistics.

(Refer Slide Time: 07:09)



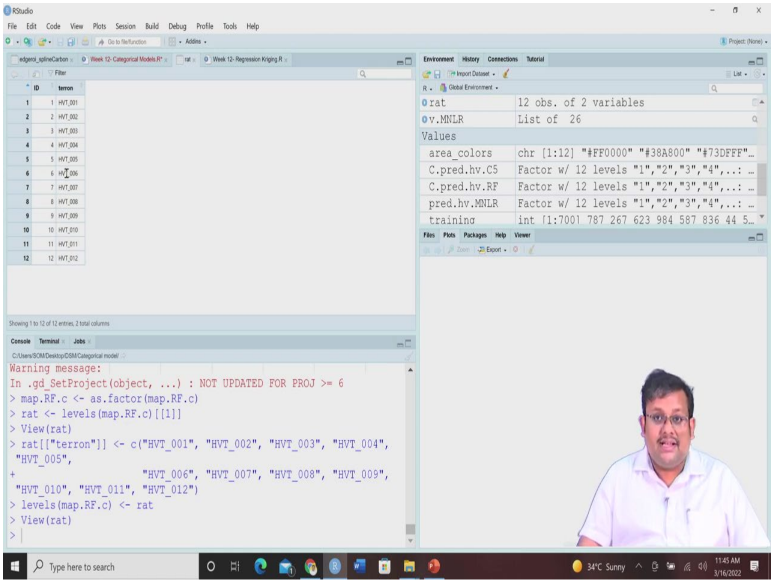

183 goofcat(observed = DSM_data$terron[training], predicted = C.pre
184
185 #external validation
186 V.pred.hv.RF <- predict(hv.RF, newdata = DSM_data[-training,])
187 goofcat(observed = DSM_data$terron[-training], predicted = V.pr
188
189 # class prediction
190 map.RF.c <- predict(hunterCovariates, hv.RF, filename = "hv_RF_
191 overwrite = T, datatype = "INT2S")
192 map.RF.c <- as.factor(map.RF.c)
193 rat <- levels(map.RF.c)[[1]]
194 rat[["terron"]] <- c("HVT_001", "HVT_002", "HVT_003", "HVT_004"
195 "HVT_006", "HVT_007", "HVT_008", "HVT_009"
196 levels(map.RF.c) <- rat
197 # plot
$kappa
[1] 0.4008742
> map.RF.c <- predict(hunterCovariates, hv.RF, filename = "hv_RF_class.tif", format = "GTiff",
+ overwrite = T, datatype = "INT2S")
Warning message:
In .gd_SetProject(object, ...) : NOT UPDATED FOR PROJ >= 6
> map.RF.c <- as.factor(map.RF.c)
> rat <- levels(map.RF.c)[[1]]
rat 12 obs. of 1 variable
v.MNLR List of 26
Values
area_colors chr [1:12] "#FF0000" "#38A800" "#73DFFF"…
C.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",..:
C.pred.hv.RF Factor w/ 12 levels "1","2","3","4",..:
pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",..:
training int [1:700] 787 267 623 984 587 836 44 5…

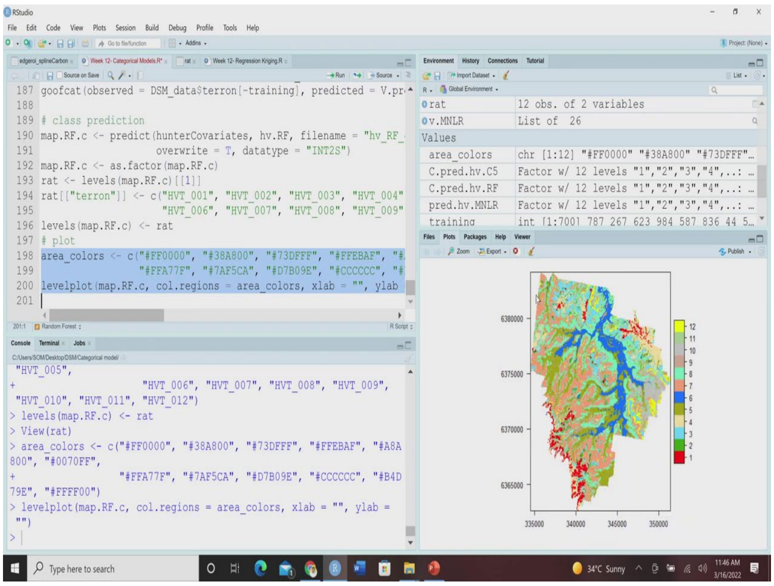

183 goofcat(observed = DSM_data$terron[training], predicted = C.pre
185 #external validation
186 V.pred.hv.RF <- predict(hv.RF, newdata = DSM_data[-training,])
187 goofcat(observed = DSM_data$terron[-training], predicted = V.pr
189 # class prediction
190 map.RF.c <- predict(hunterCovariates, hv.RF, filename = "hv_RF_
191 overwrite = T, datatype = "INT2S")
192 map.RF.c <- as.factor(map.RF.c)
193 rat <- levels(map.RF.c)[[1]]
194 rat[["terron"]] <- c("HVT_001", "HVT_002", "HVT_003", "HVT_004"
195 "HVT_006", "HVT_007", "HVT_008", "HVT_009"
196 levels(map.RF.c) <- rat
197 # plot
$kappa
[1] 0.4008742
> map.RF.c <- predict(hunterCovariates, hv.RF, filename = "hv_RF_class.tif", format = "GTiff",
+ overwrite = T, datatype = "INT2S")
Warning message:
In .gd_SetProject(object, ...) : NOT UPDATED FOR PROJ >= 6
> map.RF.c <- as.factor(map.RF.c)
> rat <- levels(map.RF.c)[[1]]
> View(rat)
rat 12 obs. of 1 variable
v.MNLR List of 26


194 rat[["terron"]] <- c("HVT_001", "HVT_002", "HVT_003", "HVT_004"
195 "HVT_006", "HVT_007", "HVT_008", "HVT_009"
196 levels(map.RF.c) <- rat
+ overwrite = T, datatype = "INT2S")
Warning message:
In .gd_SetProject(object, ...) : NOT UPDATED FOR PROJ >= 6
> map.RF.c <- as.factor(map.RF.c)
> rat <- levels(map.RF.c)[[1]]
> View(rat)
> rat[["terron"]] <- c("HVT_001", "HVT_002", "HVT_003", "HVT_004", "HVT_005",
+ "HVT_006", "HVT_007", "HVT_008", "HVT_009", "HVT_010", "HVT_011", "HVT_012")
> levels(map.RF.c) <- rat
rat 12 obs. of 2 variables
v.MNLR List of 26

Now, class prediction for predict the class again we are going to run those scripts which using the, which uses the hex colour codes as we have seen previously, so we are going to use this predict function for the class prediction and then we are going to map it, now we are going to add a new column called raster in the raster attribute table. So, you will see this rat you can see 12 observation of one variable. So, we are going to add these terrons with these indicators. So, if we just click on this, you will see that.

(Refer Slide Time: 07:52)





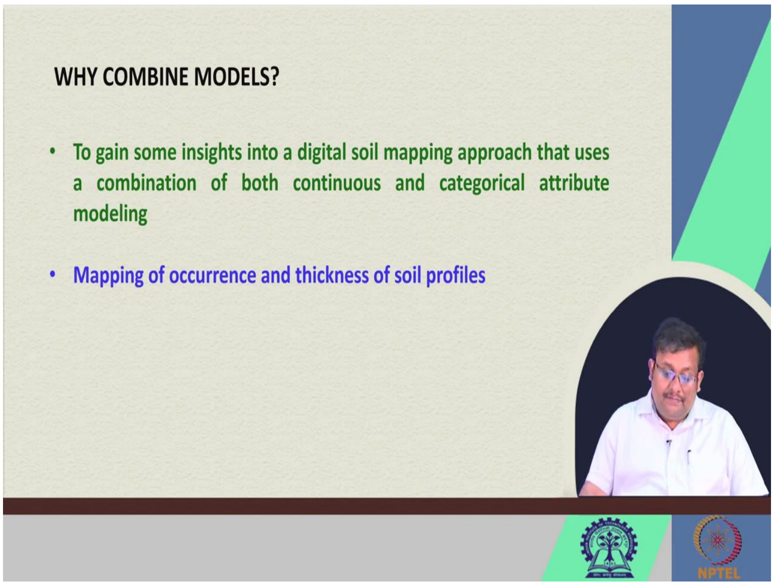
Now, rat will be developed with two variables one is the ID and the second one is the terron. So, from this ID and terron we can see that these are the indicators. Now, let us go back and then we have assigned these plot colours using these hex colour codes. And then we are going to use this just like previously we are going to use this for predicting and mapping the properties of the terrons in the area of interest.

So, guys, what we have done again just like previously, we have first call the library of random forest and we then fitted the model random forest model specifying the number of trees as well as the number of features and once we have done that the model will produce the results we have seen they are variable importance.

Apart from that, we have produced the predicted based on our calibration data set, we have seen the categorical goodness of fit and then we have seen the validation goodness of fit by predicting their values and then using these goofcat function after that, using this random forests module, we have prepared the map and also we have included a new raster attribute table where we have indicated the serial numbers of the terron classes starting from Hunter Valley Terron 001 to Hunter Valley Terron 012.

And then we assign them each of these terron class we have assigned them a colour using the hex colour codes and then we have plotted them just like here. So, this shows that how we can develop this soil classification map using the random forest classification, random forests classification algorithm.

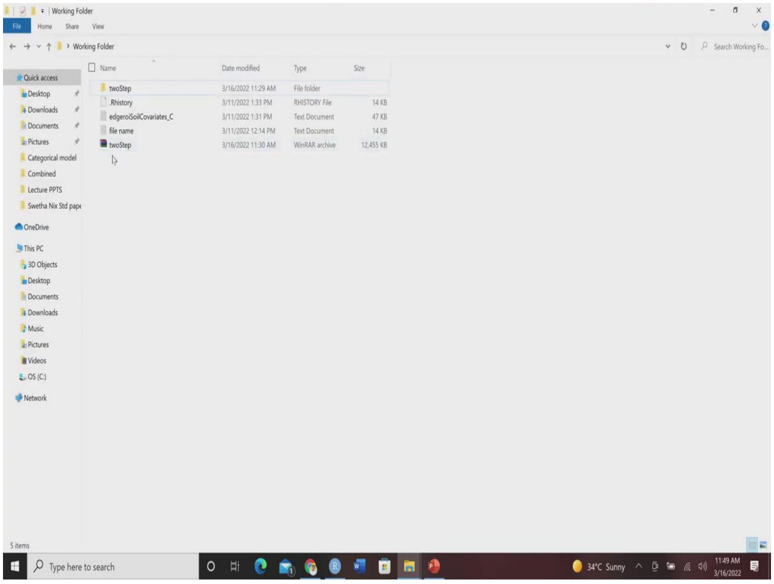
(Refer Slide Time: 10:10)

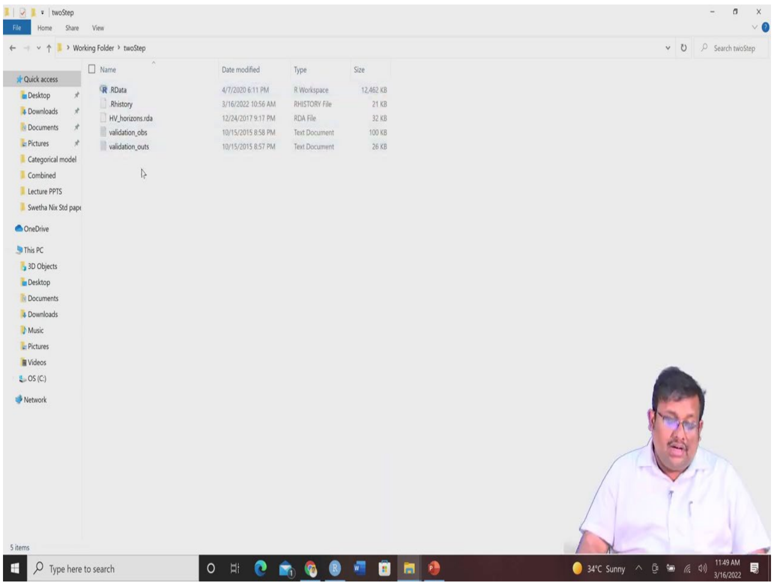


Now, let us go back to our Combine Model, now, what is the motivation between this Combine Model, why we go for the combining this model here, I will show you how we can combine this categorical model and then we can also, how we can combine this categorical model and continuous model together in digital soil mapping using different types of machine learning algorithms.

So, our objective is to gain insight into digital soil mapping approach that uses a combination of both continuous and categorical attribute modelling first of all, we are going to see the occurrence that means presence or absence of particular soil horizon in a soil profile and then if they are present then we are going to measure we are going to predict their depth using a regression model. So, first we are going to use the categorical model, then we are going to use the continuous model.

(Refer Slide Time: 11:20)

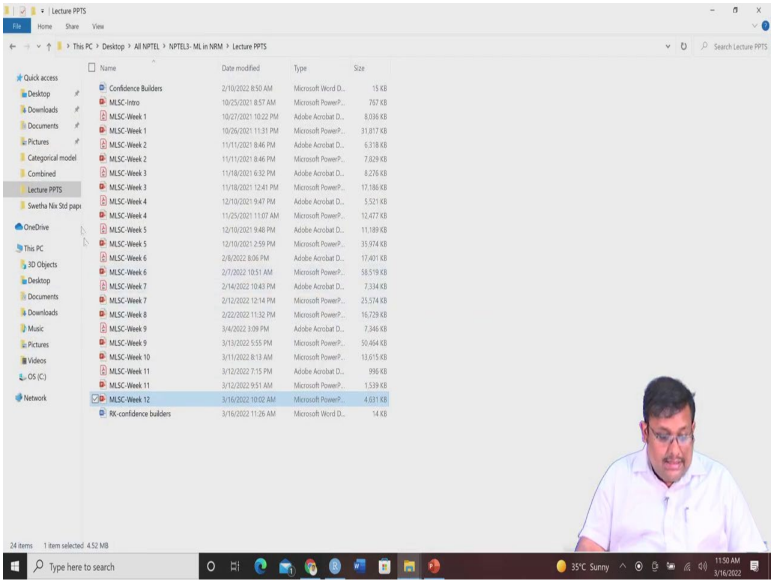


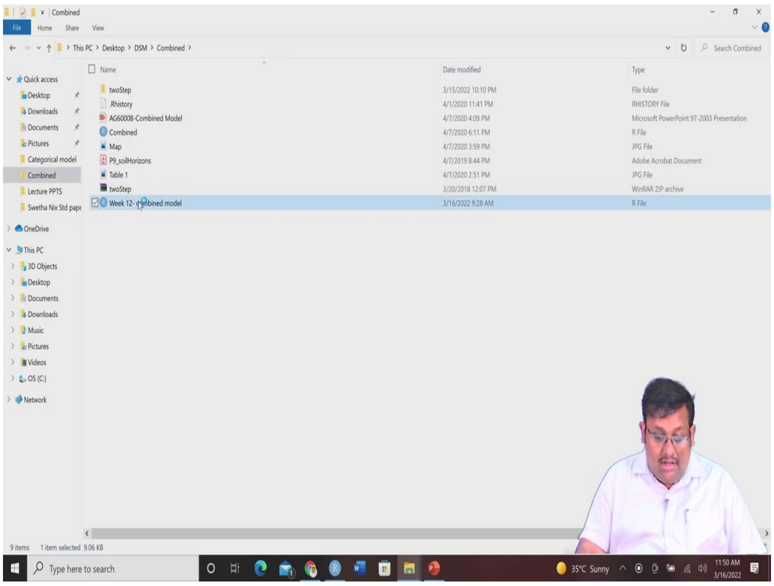


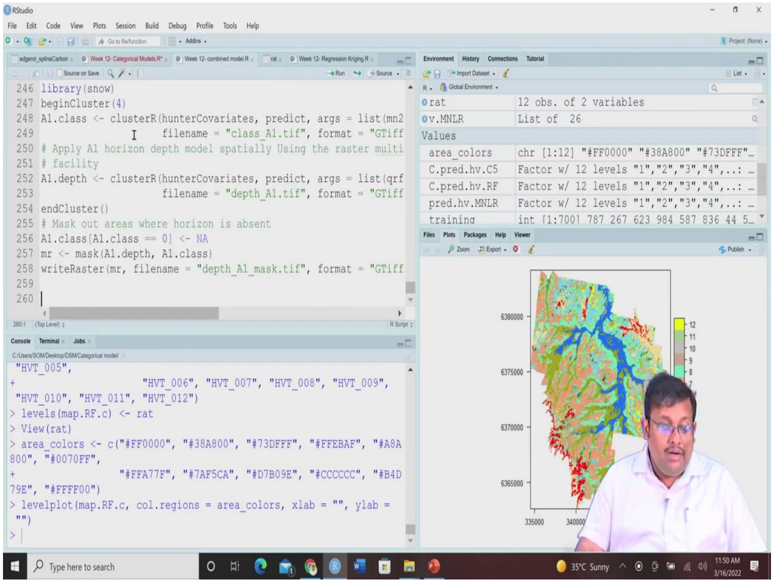
Now, for this, remember guys, we have to use some files. So, this file I will attach in our class forum, please download it. So, if I go back to our working folder, in the working folder, you will see there is a file called twoStep. So, this twoStep model, I will attach, I will post it in our class forum, and then you can extract this into your working folder. So, if you open this twoStep folder, you will see there is a data file and also there will be the observation, validation observation as well as the validation outputs.

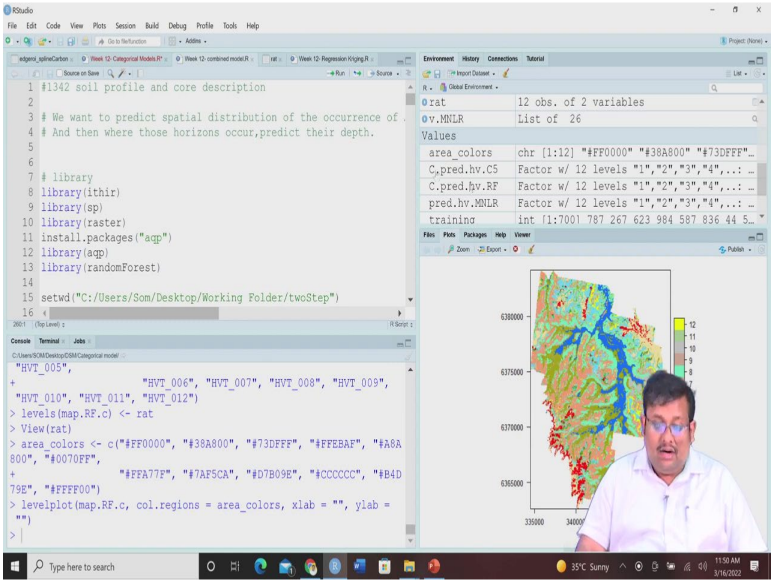
So, these will be required for different operation in this combined model. So, let me just go through and these codes and walk you through these codes and let you know, how we use these things. So, once you download this code of this twoStep, you will just download this folder and extract it in your working folder, so that you can work with it.

(Refer Slide Time: 12:25)



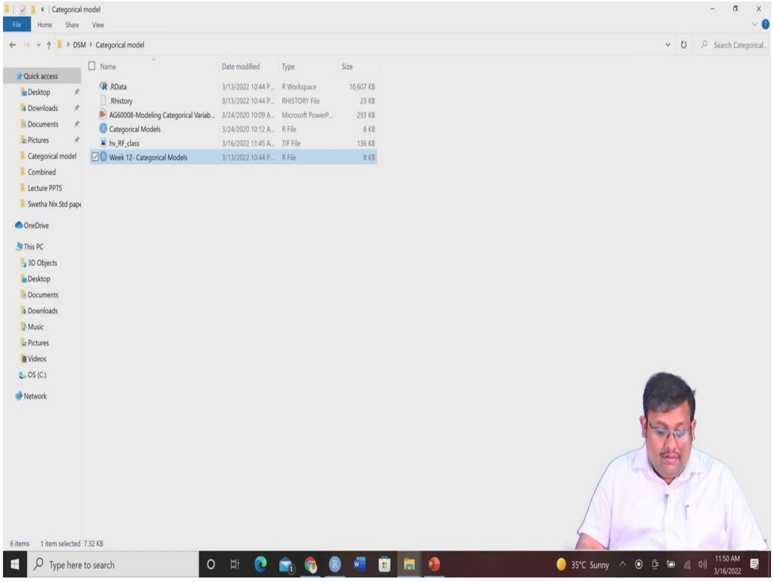


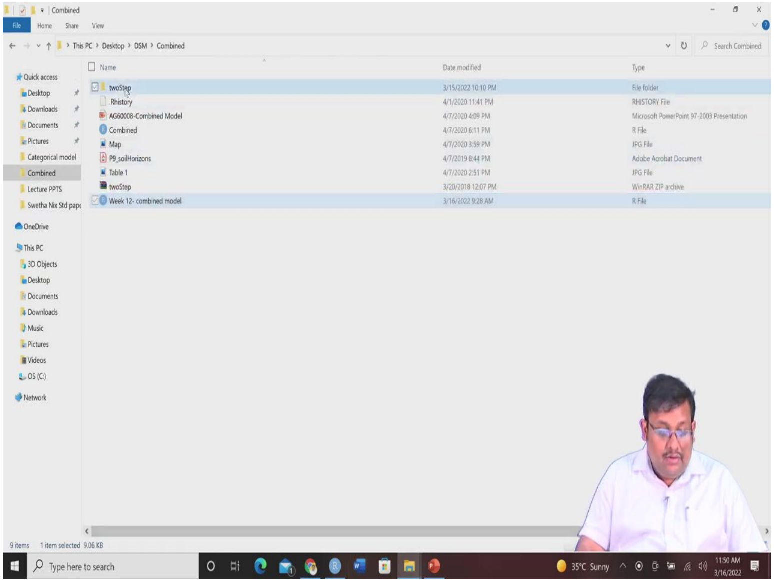


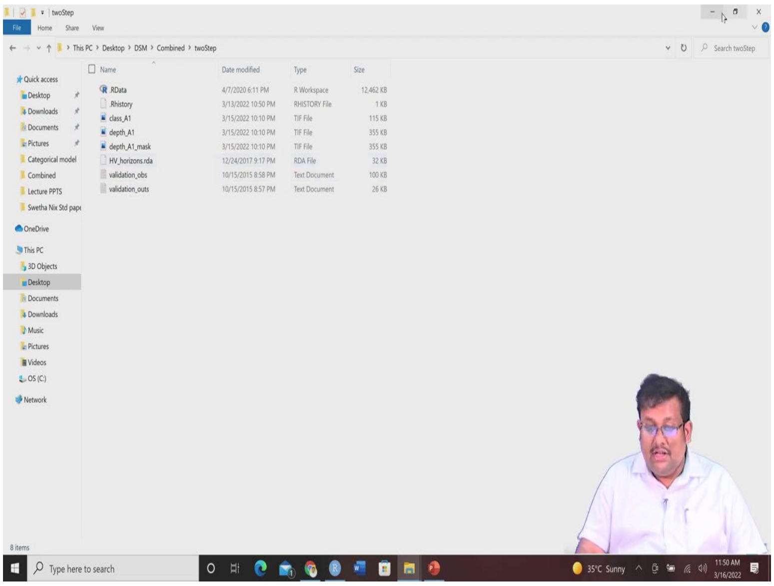


So, let me just go ahead and use the combined model. So, these are the scripts for the combined model.

(Refer Slide Time: 12:42)

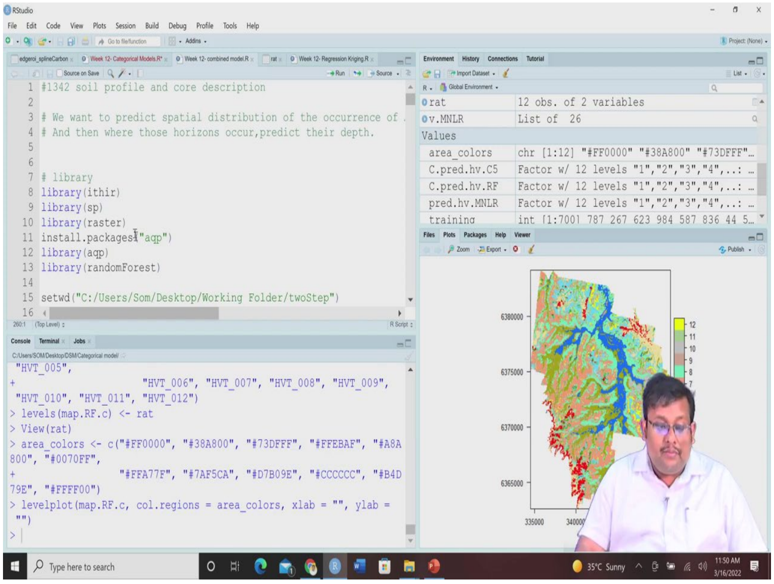


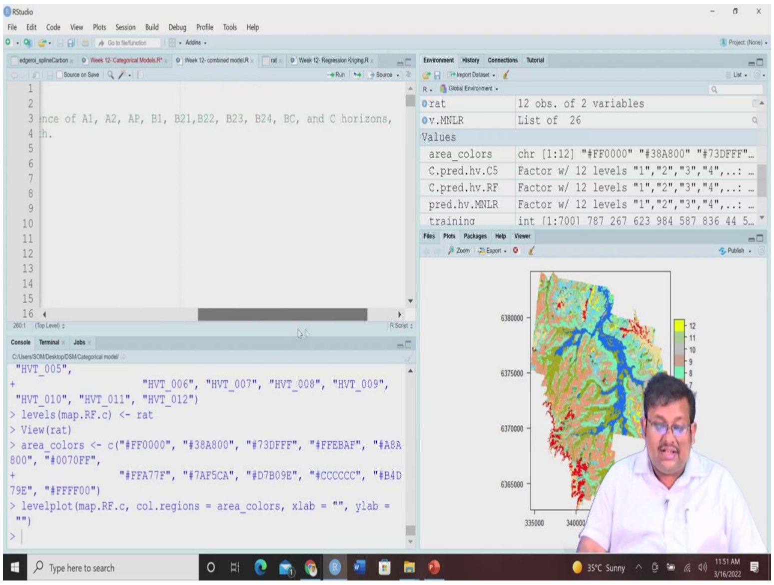


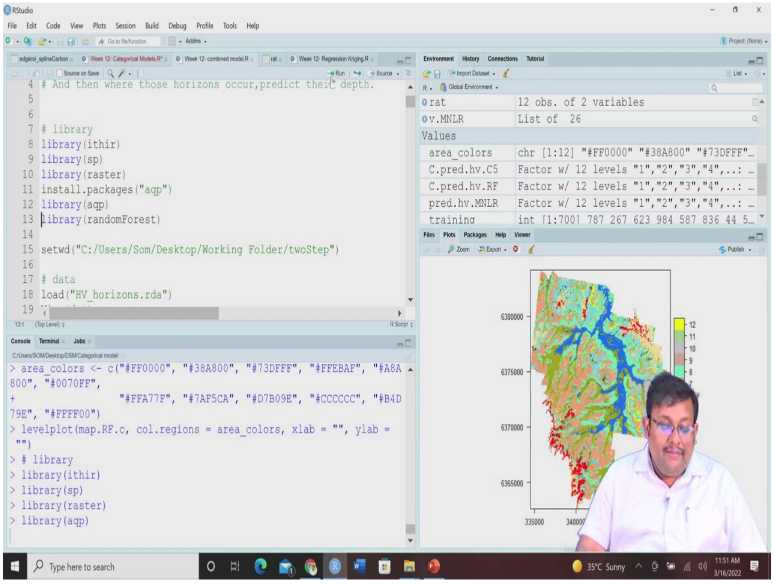


So, in this data set of this twoStep model, if you go to the twoStep model, you will see that there will be HV horizons data.

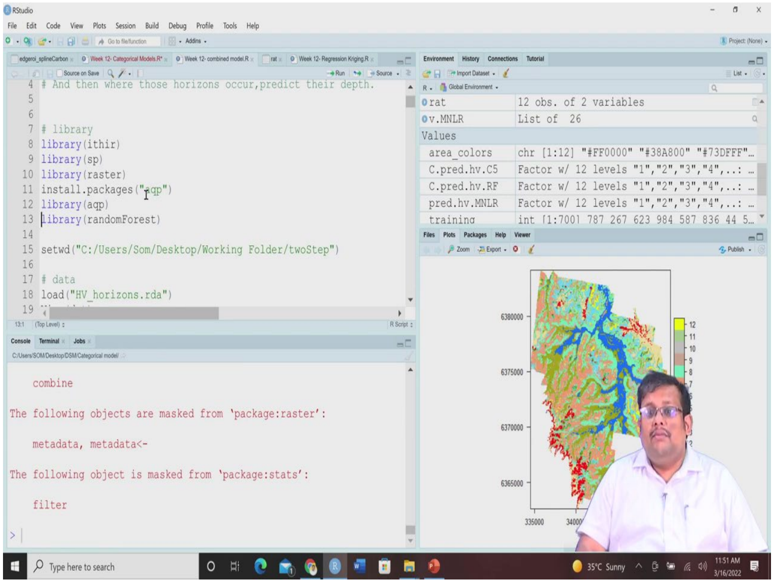
(Refer Slide Time: 12:51)



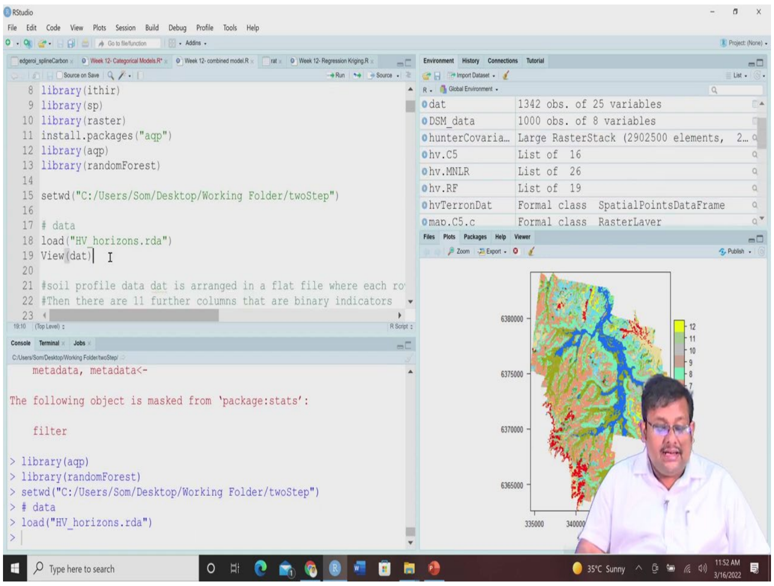




# And then where those horizons occur,predict their depth.
# library
library(ithir)
library(sp)
library(raster)
install.packages("aqp")
library(aqp)
library(randomForest)
setwd("C:/Users/Som/Desktop/Working Folder/twoStep")
# data
load("HV_horizons.rda")
rat 12 obs. of 2 variables
v.MNLR List of 26
Values
area_colors chr [1:12] "#FF0000" "#38A800" "#73DFFF"…
C.pred.hv.C5 Factor w/ 12 levels "1","2","3","4",..:
C.pred.hv.RF Factor w/ 12 levels "1","2","3","4",..:
pred.hv.MNLR Factor w/ 12 levels "1","2","3","4",..:
training int [1:700] 787 267 623 984 587 836 44 5…
> area_colors <- c("#FF0000", "#38A800", "#73DFFF", "#FFEBAF", "#A8A800", "#0070FF",
+ "#FFA77F", "#7AF5CA", "#D7B09E", "#CCCCCC", "#B4D79E", "#FFFF00")
> levelplot(map.RF.c, col.regions = area_colors, xlab = "", ylab = "")
> # library
> library(ithir)
> library(sp)
> library(raster)
> library(aqp)



# And then where those horizons occur,predict their depth.
# library
library(ithir)
library(sp)
library(raster)
install.packages("aqp")
library(aqp)
library(randomForest)
setwd("C:/Users/Som/Desktop/Working Folder/twoStep")
# data
load("HV_horizons.rda")
combine
The following objects are masked from 'package:raster':
metadata, metadata<-
The following object is masked from 'package:stats':
filter



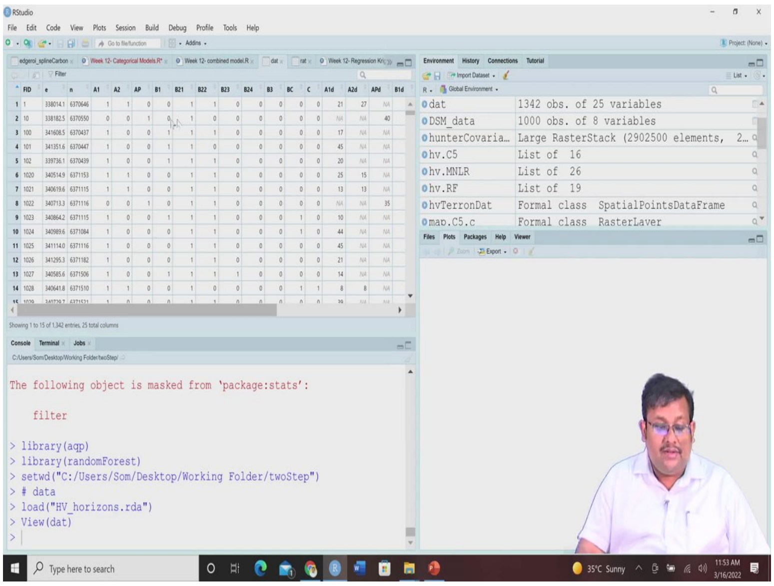
library(ithir)
library(sp)
library(raster)
install.packages("aqp")
library(aqp)
library(randomForest)
setwd("C:/Users/Som/Desktop/Working Folder/twoStep")
# data
load("HV_horizons.rda")
View(dat)
#soil profile data dat is arranged in a flat file where each ro
#Then there are 11 further columns that are binary indicators
dat 1342 obs. of 25 variables
DSM_data 1000 obs. of 8 variables
hunterCovaria… Large RasterStack (2902500 elements, 2…
hv.C5 List of 16
hv.MNLR List of 26
hv.RF List of 19
hvTerronDat Formal class SpatialPointsDataFrame
map.C5.c Formal class RasterLayer
metadata, metadata<-
The following object is masked from 'package:stats':
filter
> library(aqp)
> library(randomForest)
> setwd("C:/Users/Som/Desktop/Working Folder/twoStep")
> # data
> load("HV_horizons.rda")

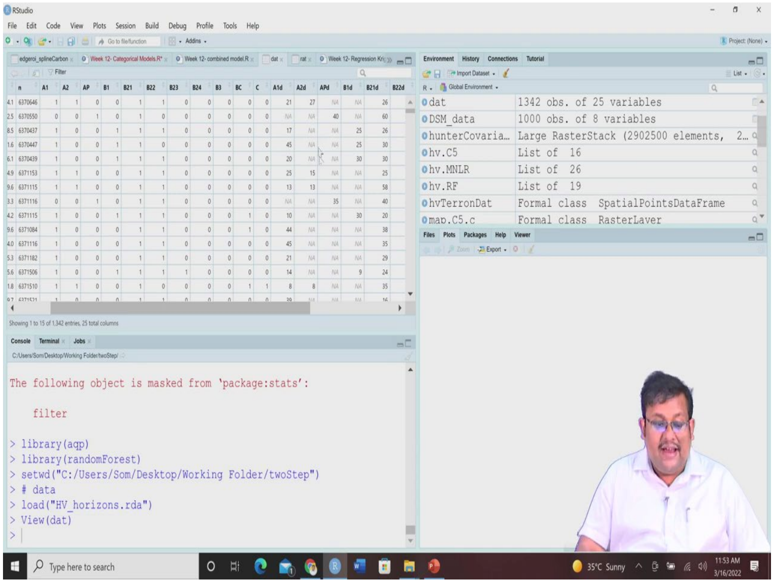
So, these HV horizon data is basically having 1342 soil profile and code description. So, we are going to use the, this data set and we are going to we want to predict the spatial distribution of their occurrence specifically on, for A1, A2, AP and then B1, B21, B22, B23, B24, BC and C horizons and where they are presents, when they are present we are going to predict their depth. So, let us call all the required libraries.

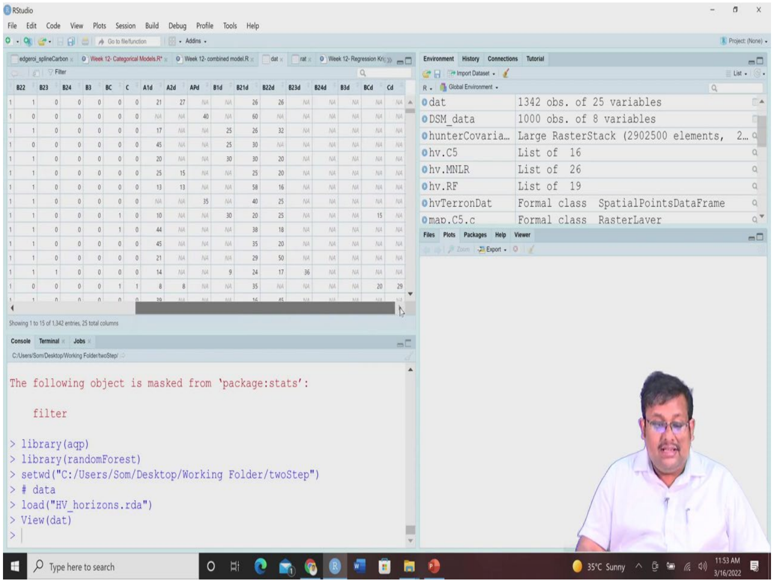
So, the required library already you know that is ithir library is required, the library sp will be required, library raster will be required. And then you have to install this package called a aqp once you install then call this library aqp, aqp is basically having the is useful for doing some soil DSMs calculations. So, this is very important. Now, once we have, call this library a aqp the next is calling the library random forest because we are going to use this random forest model.

So, let us call it after this we have to set our working directory, we can either manually go and see set the twoStep folder or we can give directly the path of this twoStep folder here I have used the path of this twoStep folder it is contained within this working folder and for this I have used the set working directory function. So, let me just go and select this working directory twoStep because all the maps and all these outputs will be produced in this twoStep folder now. Now, in the twoStep folder there is HV underscore horizon dot rda file. So, let us call these things and then let us view the file.

(Refer Slide Time: 14:52)





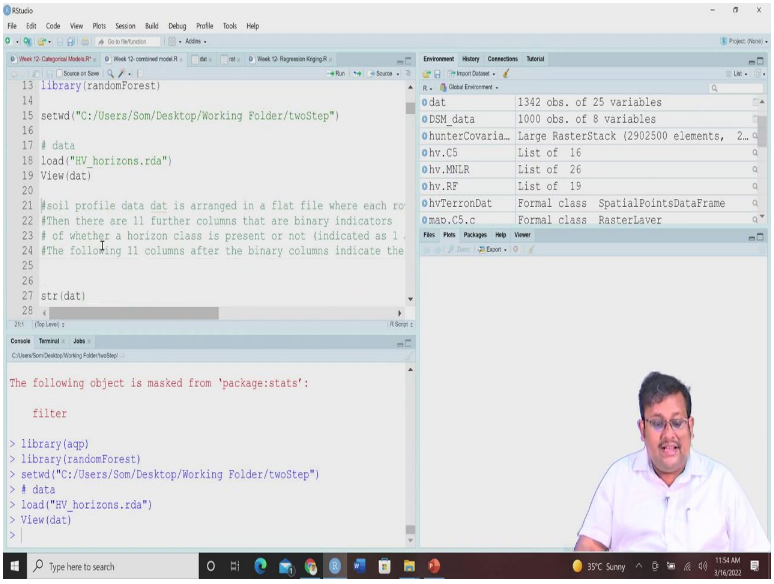


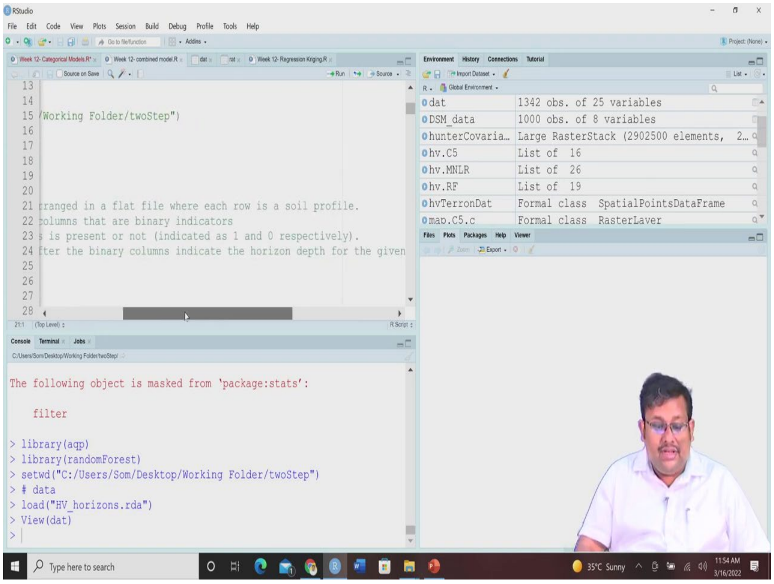
So, if you see this file, this data set of 1340 observation around so you will see there will be easting northing and then A1, A2, AP, B1, B2 and B22, B23, B24, B3, BC, C and all these horizons. So, this 1 and 0 basically shows the presence and absence. So, presence is denoted by 1 and absence is denoted by 0.

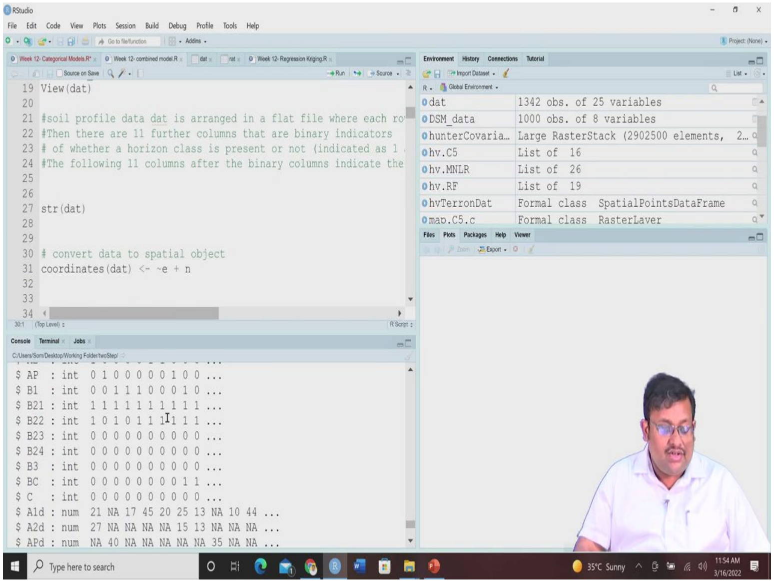
So, here you can see these are the samples the number of samples and you can see these A1, A2 are basically this is showing each row is showing a soil profile. So, this is for soil profile these another soil profile and the soil profile it shows the arrangement of the horizon. So, the first soil profile gives you the arrangement of the depth of these horizons and second profile gives you the arrangement of these horizons and so, on.

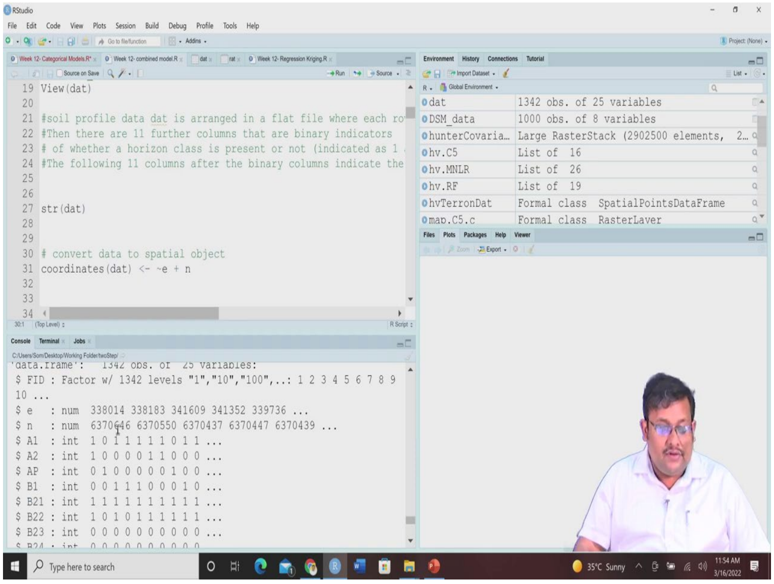
And after this C horizon, you will see these A1d, A2d and all these things, so the d stands for the depth. So, when there are present so, for example, A1 is present, so, the A1 depth is 21 centimetre, then A2 is present, so A2 depth is 27 centimetre and when there are absent there having the any values. So, for this you can see, this is the dataset. So, once we have this data set, let us go back and see.

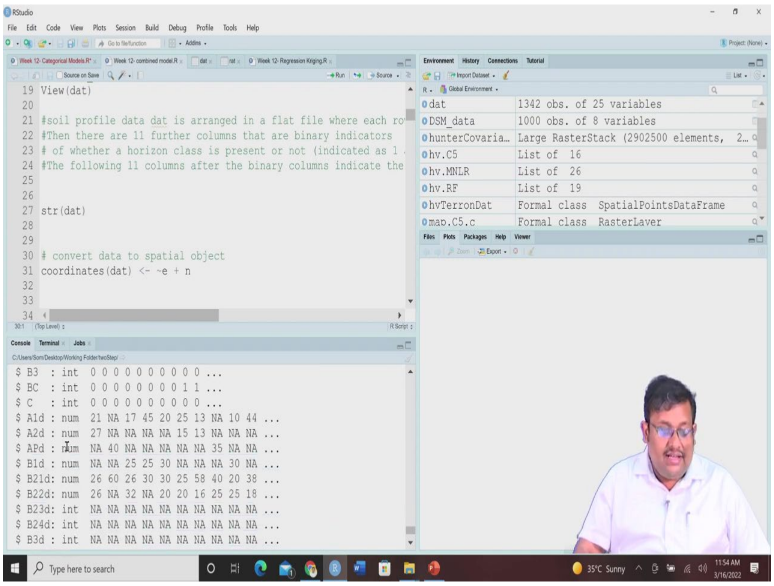
(Refer Slide Time: 16:17)






View(dat)
#soil profile data dat is arranged in a flat file where each ro
#Then there are 11 further columns that are binary indicators
# of whether a horizon class is present or not (indicated as 1
#The following 11 columns after the binary columns indicate the
str(dat)
# convert data to spatial object
coordinates(dat) <- ~e + n
$ AP : int 0 1 0 0 0 0 0 1 0 0 ...
$ B1 : int 0 0 1 1 1 0 0 0 1 0 ...
$ B21 : int 1 1 1 1 1 1 1 1 1 1 ...
$ B22 : int 1 0 1 0 1 1 1 1 1 1 ...
$ B23 : int 0 0 0 0 0 0 0 0 0 0 ...
$ B24 : int 0 0 0 0 0 0 0 0 0 0 ...
$ B3 : int 0 0 0 0 0 0 0 0 0 0 ...
$ BC : int 0 0 0 0 0 0 0 0 1 1 ...
$ C   : int 0 0 0 0 0 0 0 0 0 0 ...
$ A1d : num 21 NA 17 45 20 25 13 NA 10 44 ...
$ A2d : num 27 NA NA NA NA 15 13 NA NA NA ...
$ APd : num NA 40 NA NA NA NA NA 35 NA NA ...
dat 1342 obs. of 25 variables
DSM_data 1000 obs. of 8 variables
hunterCovaria… Large RasterStack (2902500 elements, 2…
hv.C5 List of 16
hv.MNLR List of 26
hv.RF List of 19
hvTerronDat Formal class SpatialPointsDataFrame
map.C5.c Formal class RasterLayer

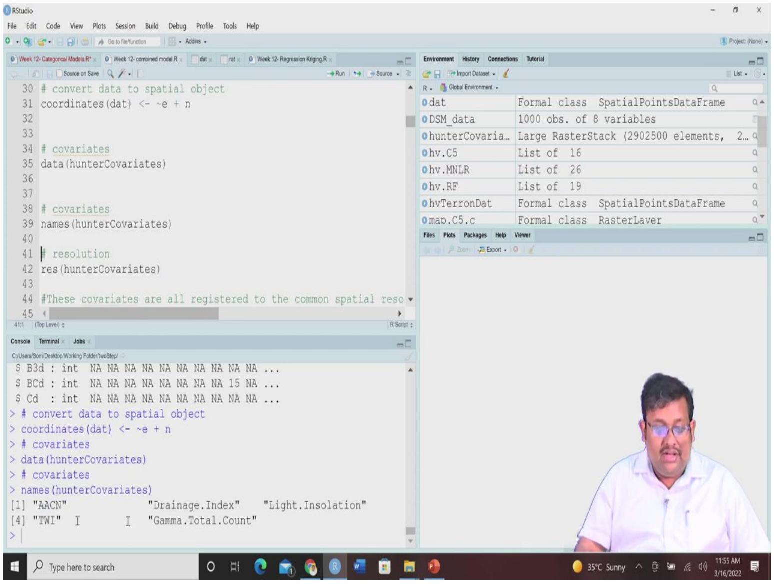

View(dat)
#soil profile data dat is arranged in a flat file where each ro
#Then there are 11 further columns that are binary indicators
# of whether a horizon class is present or not (indicated as 1
#The following 11 columns after the binary columns indicate the
str(dat)
# convert data to spatial object
coordinates(dat) <- ~e + n
'data.frame': 1342 obs. of 25 variables:
$ FID : Factor w/ 1342 levels "1","10","100",..: 1 2 3 4 5 6 7 8 9 10 ...
$ e   : num 338014 338183 341609 341352 339736 ...
$ n   : num 6370646 6370550 6370437 6370447 6370439 ...
$ A1 : int 1 0 1 1 1 1 1 0 1 1 ...
$ A2 : int 1 0 0 0 0 1 1 0 0 0 ...
$ AP : int 0 1 0 0 0 0 0 1 0 0 ...
$ B1 : int 0 0 1 1 1 0 0 0 1 0 ...
$ B21 : int 1 1 1 1 1 1 1 1 1 1 ...
$ B22 : int 1 0 1 0 1 1 1 1 1 1 ...
$ B23 : int 0 0 0 0 0 0 0 0 0 0 ...
dat 1342 obs. of 25 variables
DSM_data 1000 obs. of 8 variables
hunterCovaria… Large RasterStack (2902500 elements, 2…
hv.C5 List of 16
hv.MNLR List of 26
hv.RF List of 19
hvTerronDat Formal class SpatialPointsDataFrame
map.C5.c Formal class RasterLayer

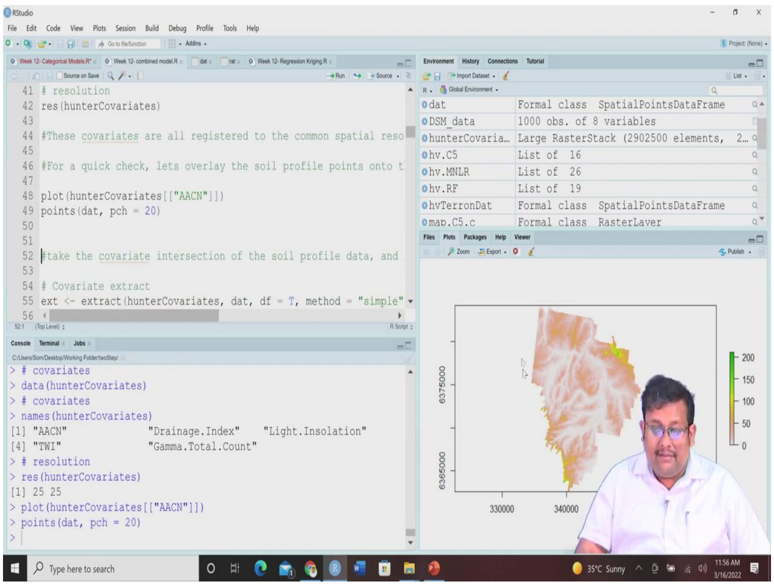

View(dat)
#soil profile data dat is arranged in a flat file where each ro
#Then there are 11 further columns that are binary indicators
# of whether a horizon class is present or not (indicated as 1
#The following 11 columns after the binary columns indicate the
str(dat)
# convert data to spatial object
coordinates(dat) <- ~e + n
$ B3 : int 0 0 0 0 0 0 0 0 0 0 ...
$ BC : int 0 0 0 0 0 0 0 0 1 1 ...
$ C   : int 0 0 0 0 0 0 0 0 0 0 ...
$ A1d : num 21 NA 17 45 20 25 13 NA 10 44 ...
$ A2d : num 27 NA NA NA NA 15 13 NA NA NA ...
$ APd : num NA 40 NA NA NA NA NA 35 NA NA ...
$ B1d : num NA NA 25 25 30 NA NA NA 30 NA ...
$ B21d: num 26 60 26 30 30 25 58 40 20 38 ...
$ B22d: num 26 NA 32 NA 20 20 16 25 25 18 ...
$ B23d: int NA NA NA NA NA NA NA NA NA NA ...
$ B24d: int NA NA NA NA NA NA NA NA NA NA ...
$ B3d : int NA NA NA NA NA NA NA NA NA NA ...
dat 1342 obs. of 25 variables
DSM_data 1000 obs. of 8 variables
hunterCovaria… Large RasterStack (2902500 elements, 2…
hv.C5 List of 16
hv.MNLR List of 26
hv.RF List of 19
hvTerronDat Formal class SpatialPointsDataFrame
map.C5.c Formal class RasterLayer

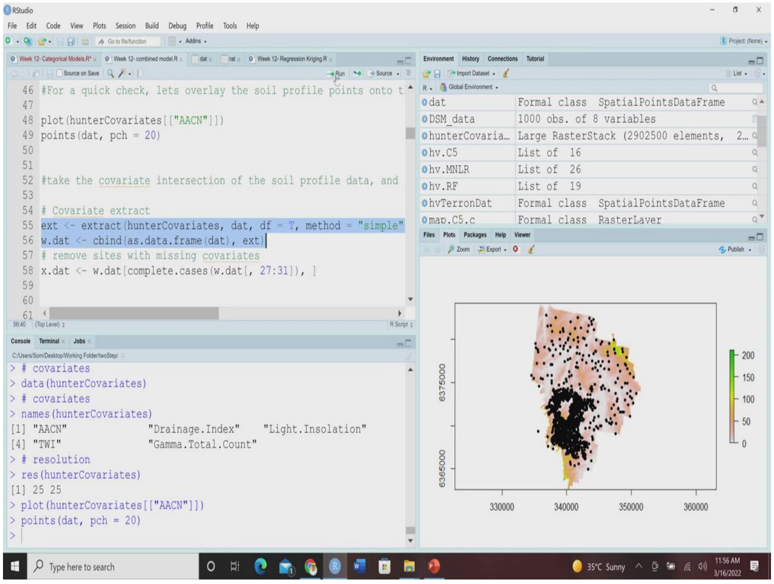
So, you can see here, the soil profile data is arranged in a flat file, where each row is a soil profile, then there is 11 further columns who are, that they are binary indicators 1 or 0 I have already told you whether horizon is class is present or not they indicated by 1 and 0 and the following 11 columns are showing the binary column after this binary columns are showing the horizon depth.

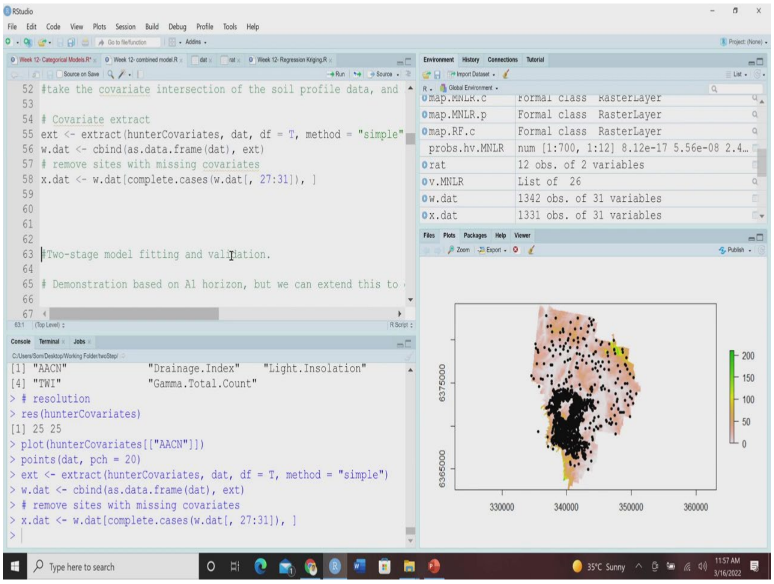
Now, once we have seen these data set let us see the structure of the datasets, so the structure of the data set will be again very clear starting from the factor then e and n of course, these are the numerical variables and then the integer variables are there and finally, their depth which is also numerical variables. Now, the next now it is a tabular format it is a data frame format next we have to convert it to the spatial objects.

(Refer Slide Time: 17:15)









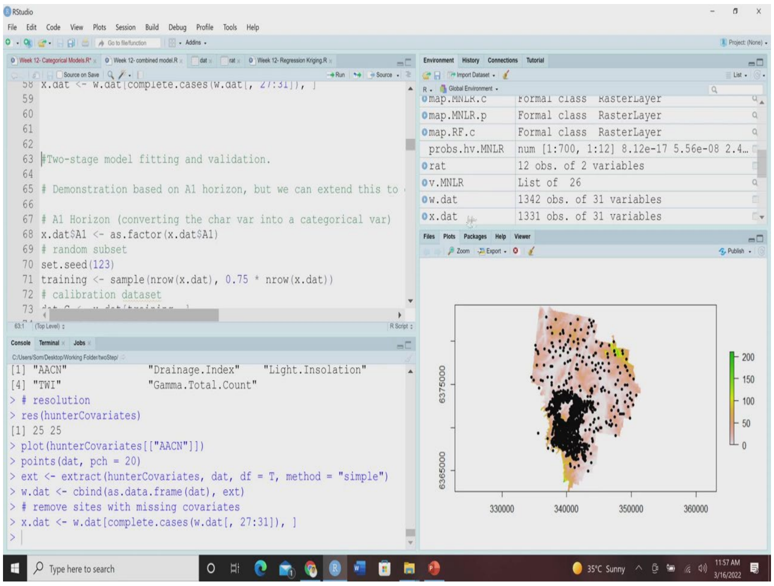
So, we must understand we must instruct R that consider these e and n as the coordinates. So, R now, you will see there is a spatial points data frame. Now, once we have converted this into spatial points data frame, we are downloading these Hunter Covariates data after we download the Hunter Covariates data of course, let us see what are the names of these there will be 5, 1, 2, 3, 4, 5 ACNN, Drainage Index, Light Installation and then TWI and Gamma Total Count and let us see this resolution.

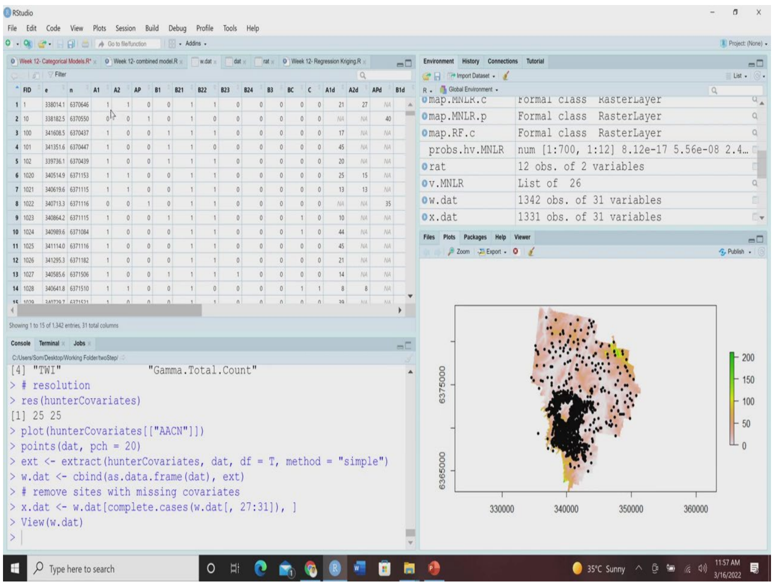
So, it is a raster stack and you will see the resolution is 25 metre by 25. So, it is a large raster stack where we have stacked all these 5 covariate and the resolution is 25 by 25 they are having all the common spatial resolution. So, for a quick display, let us first plot these for example, this ACNN and display the points and overlay the points, sampling points and this will look like this.

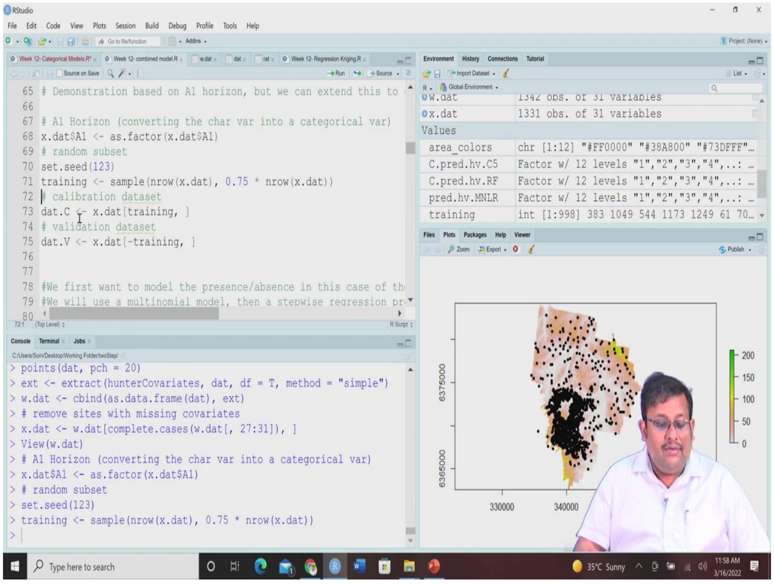
So, we are plotting first the Altitude Above Channel Network raster file and overlaying the samples over there, the point samples over there. So, 1342 samples whatever we may have. So, it is yes 1342 samples, so 1342 I did not been plotted here. Now, once we have these the next step is to extract these covariates.

So, there are 5 covariates just like previously, we are going to extract these covariates and then we are we want to combine these extracted covariates with the original data frame of 1342 observation and we are giving this name w dot dat which will be already created here and you can see here 1342 observation of 31 variables. Now, let us remove the sides with the missing covariates. So, where there are missing covariates we are removing those sides. Now, our data is now prepared for two stage model fitting and validation.

(Refer Slide Time: 19:23)





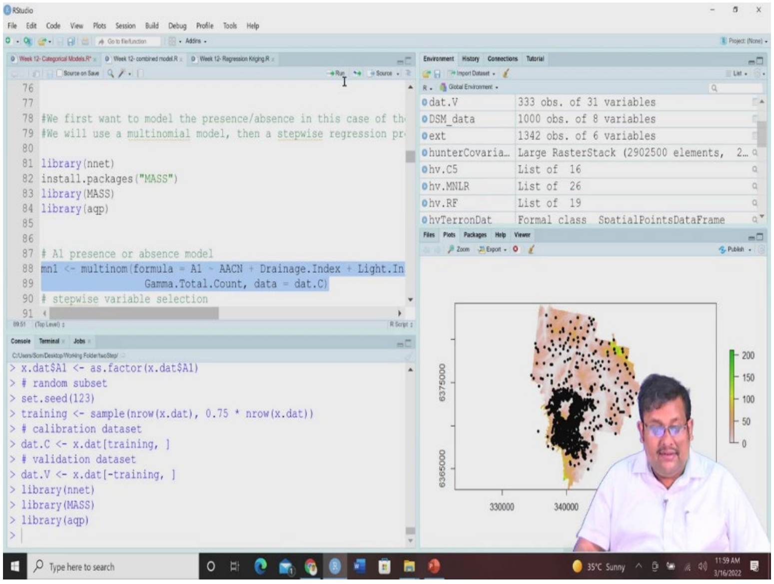


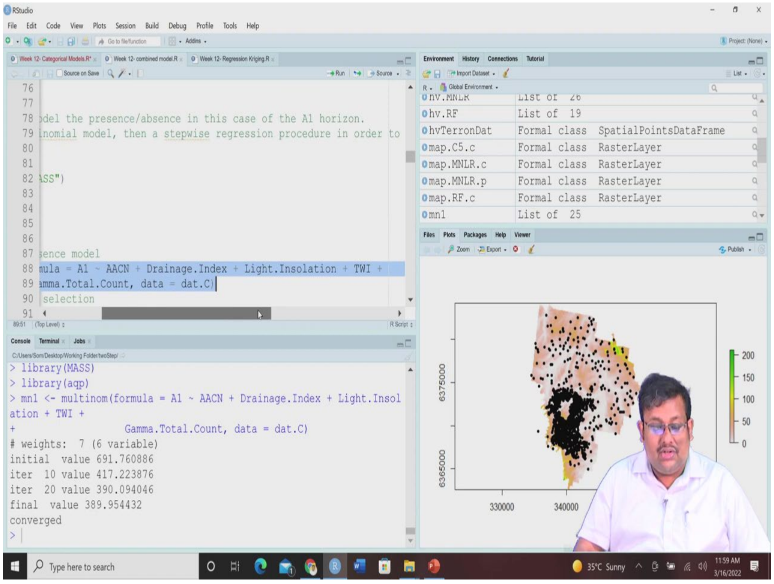
Now, we are going to demonstrate these two-stage model fitting for based on the A1 horizon. But remember that we can extend to any other horizons. So, you can perform this operation with any other horizon. Now, that in the data file 1341 observation so you can see these are considered as a numerical variable by R at this point of time, but we have to indicate R that these are the character variables because one indicates the presence of a proof of a horizon and 0 indicates the absence of that particular soil horizon.

So, we are going to first this converting we have to first convert this character variable into categorical variable. So, for that we are going we are going to use this x dot factor function for these A1. So, now it will understand R will understand that 1 and 0 are the two characters.

Now, again just like previously, we are going to set the random seed 123. We are going to now select 75 percent of the data, you can do it with the 70 percent data also. So, you can use the 75 percent of the data as the training samples and then you can use this data set and you can rename this data set as calibration data and of course, the validation data you can need a name as dat dot V. So, dat dot C and dat dot V are showing the calibration data and validation data. So, once we did, once we have separated this calibration and validation file.

(Refer Slide Time: 21:05)

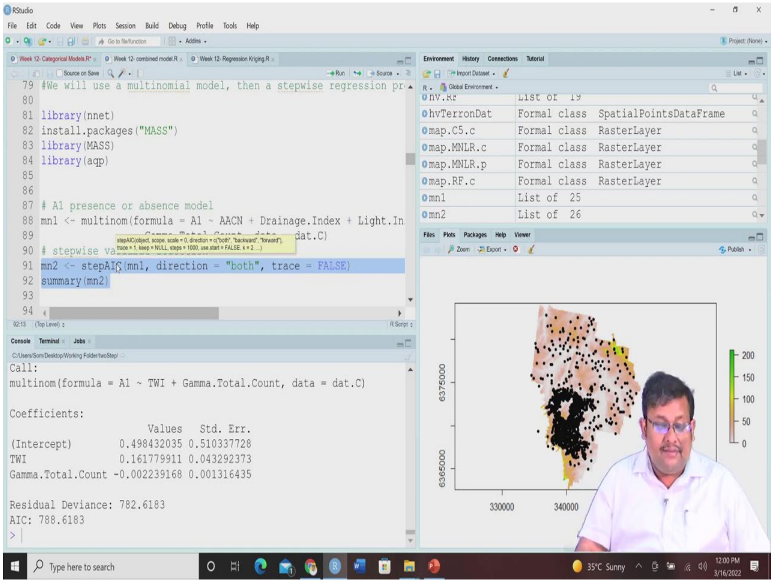




Now, suppose our first objective is to model the presence or absence in this case we are going to use this a A1 horizon. So, for this we are going to use the multinomial logistic regression model. So, we are going to use this nnet function and then we are going to install this mass package and then library aqp.

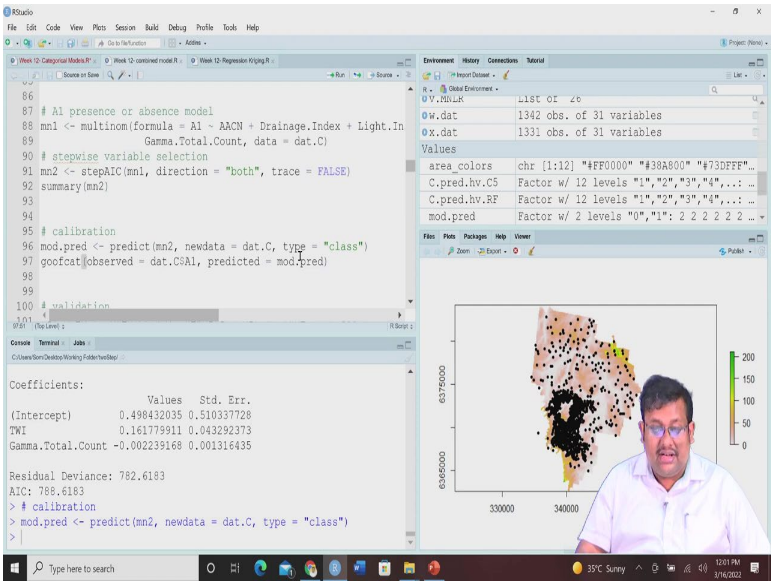
So, for predicting the presence or absence of the predicting the presence or absence of A1 horizon, we are going to use this multinom function and let us do this just like previously here our target is A1 presence of A1 or absence of A1 and here AACN, Drainage Index and then Light Insulation, TWI, Gamma Total Count all these are given here.

(Refer Slide Time: 22:15)



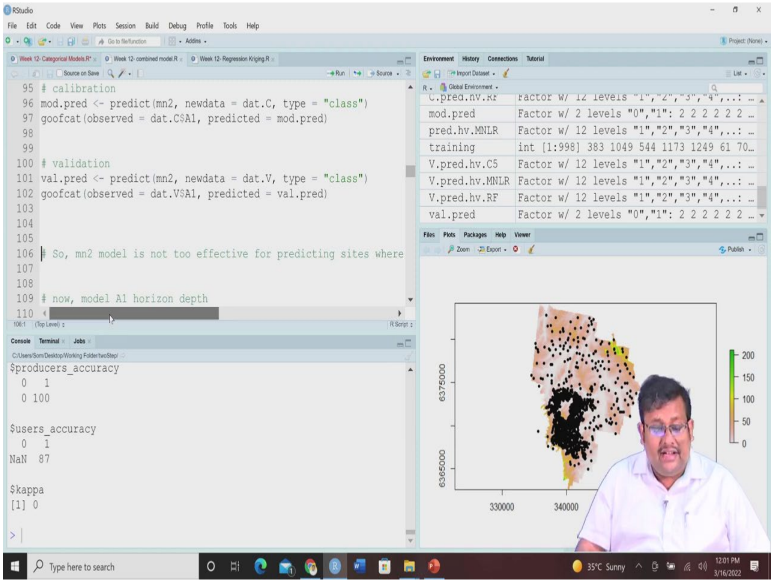
And then for the next step we are going to select the important variables using the stepwise regression. So, stepwise variable selection we are going to use step AIC function and then let us see what are the variables or features this algorithm can select. So, you can see here TWI and Gamma Total Count have been selected as important predictors and rest of them were discarded. So, will go with this mn2 model, which will be more parsimonious model than mn1, for subsequent statistical operations and mapping operations.

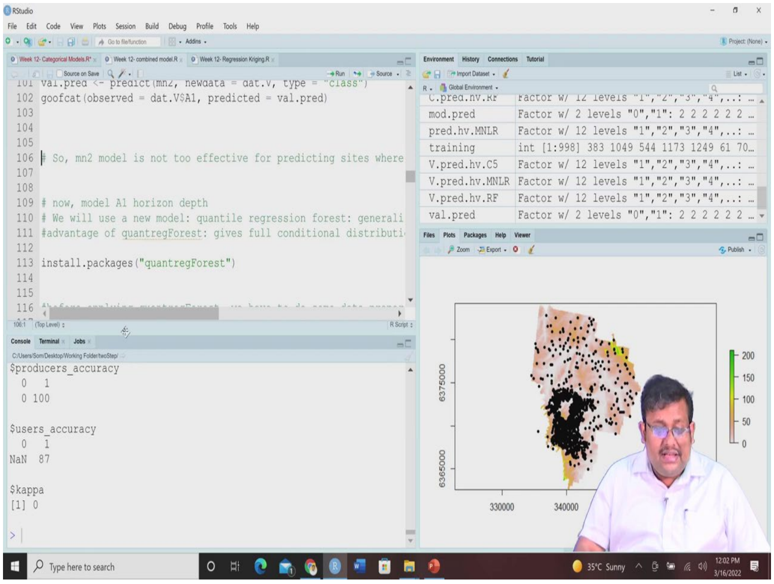
(Refer Slide Time: 22:50)

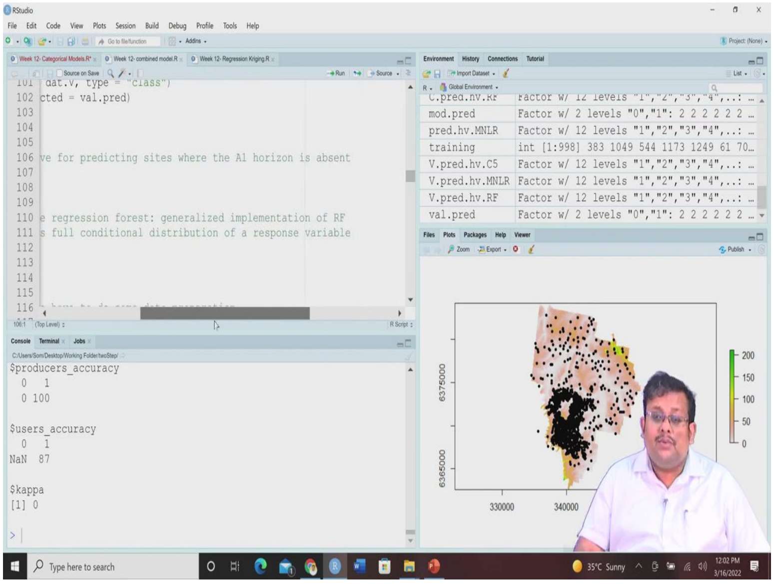


So, here you can see we are keeping these mn2 model and then we are going to predict the values for these training samples and we are going to predict the class using this calibration data using this mn2 model. So, we are doing this and then we are using this goofcat function to produce the classification accuracy and you can see here again the confusion matrix overall accuracy produces accuracy users accuracy and Kappa coefficient are generated and then we are also in the next step we are doing the prediction of the validation samples.

(Refer Slide Time: 23:30)





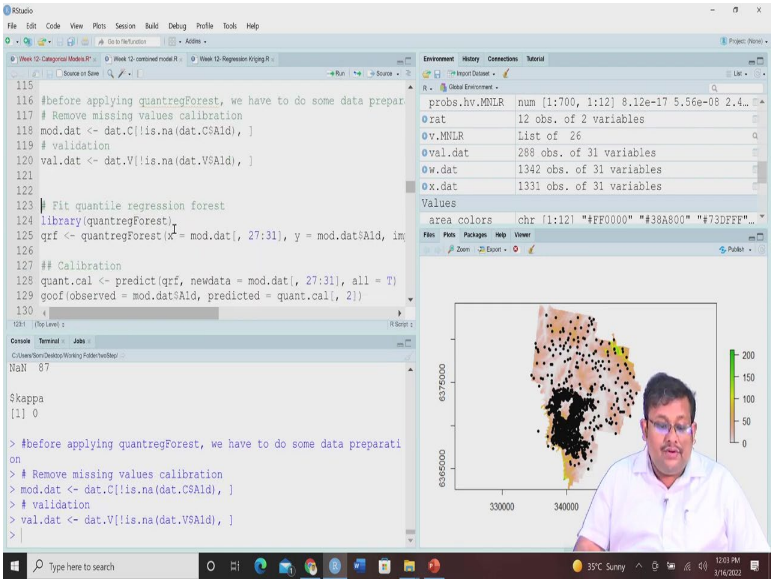


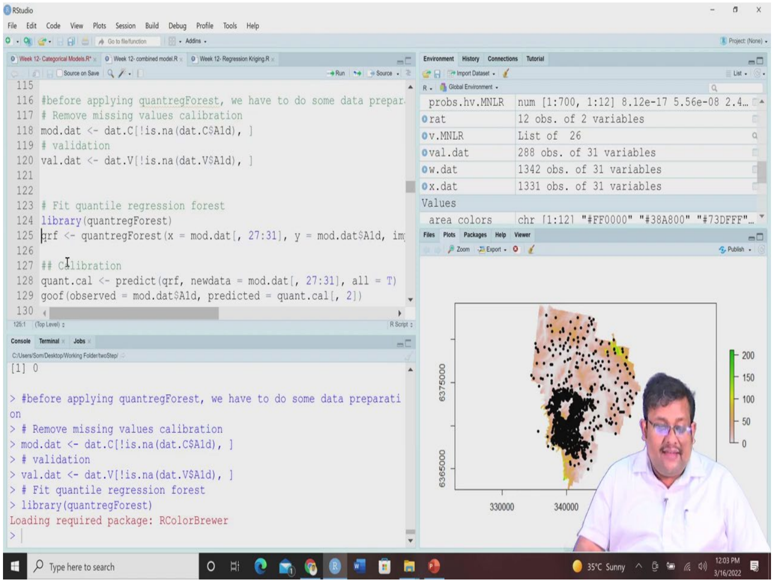
So, we are predicting the validation samples and we are again using this goofcat function to see that is also from this result, this is the confusion matrix overall accuracy produces accuracy, users accuracy and Kappa coefficient. So, we can see clearly from this that mn2 to model is not too effective for predicting the sides where even horizon is absent. So, here you can see when the A1 horizon is absent in both the cases these mn2 model is not very much effective for predicting those absence of A1 horizons.

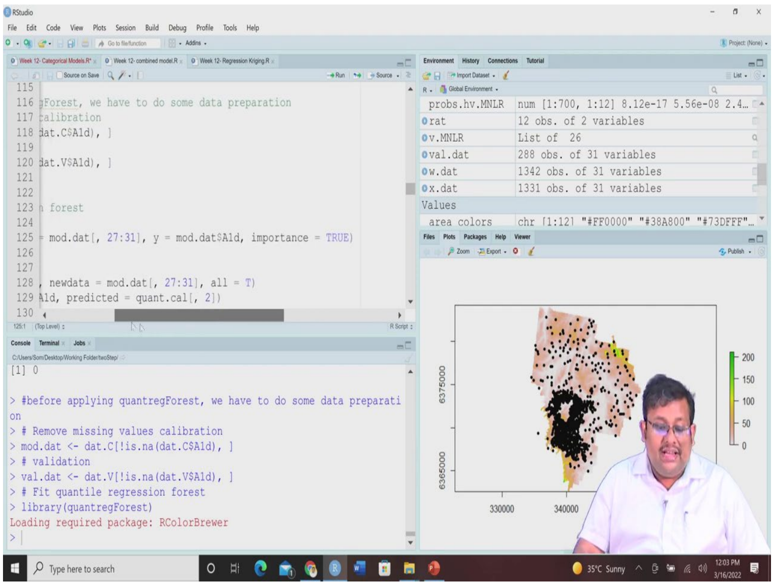
Now, once we have done this categorical model, next step is to predict the A1 horizon depth. So, we will use a new model that is called quantile regression forest. It is generalised implementation of the random forests we know that random forest, how to use these random forests to predict the continuous properties, continuous variables.

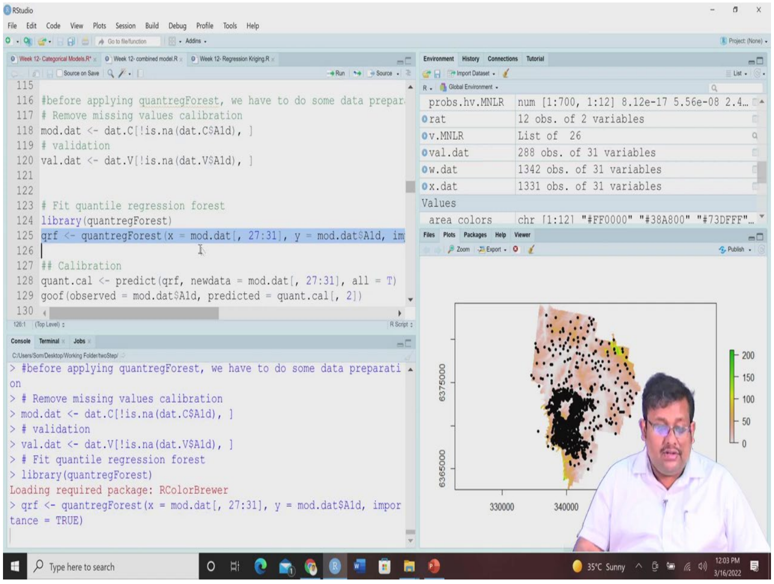
However, we are going to now use the quantile regression for this time, we are going to show you the quantile regression for just to give you an idea about the the variety of different types of machine learning models which we use for digital soil mapping purposes. So, what is the major advantage of using these quantile regression forest. So, for basic advantage of this quantile regression forest is it gives the full conditional distribution of a response variable. So, I have already given these these annotations here so, you can follow these.

(Refer Slide Time: 25:15)



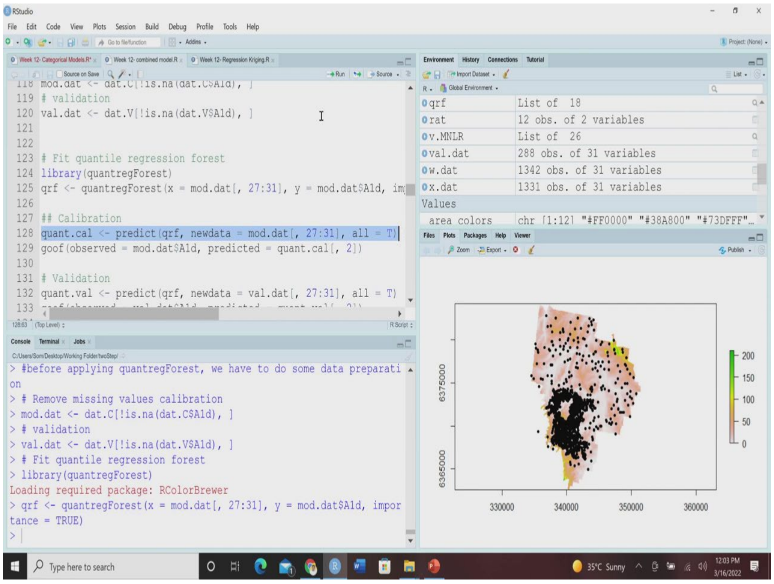


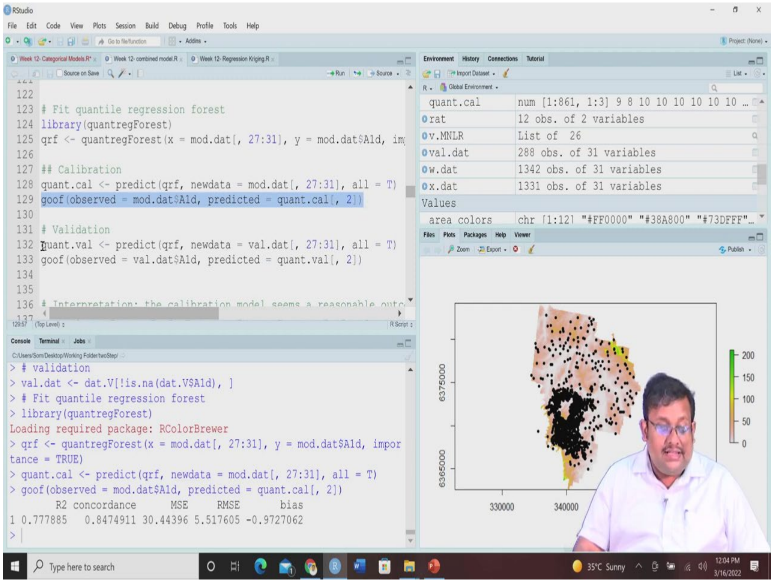




Now, for this we need to install this for this quantile regression forest we need to install this quant regression forest, and then before applying this quant regression forest, we have to do some data preparation. So, we are going to remove the missing data from the model data as well as from the validation data, and then we are going to call the library quant regression forest and then we are going to fit this quant regression forest model and then we are going to predict there, here you can see, our x is model data and our y is also model data with the depth of A1 horizon.

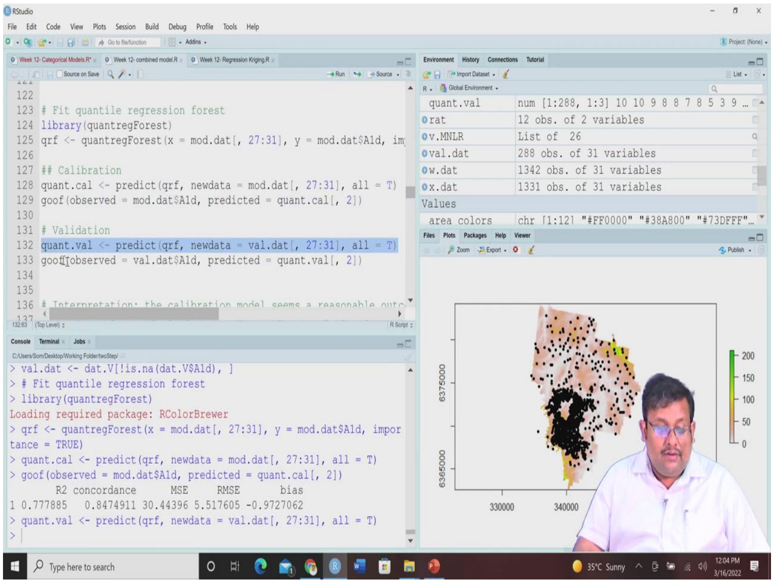
(Refer Slide Time: 25:59)

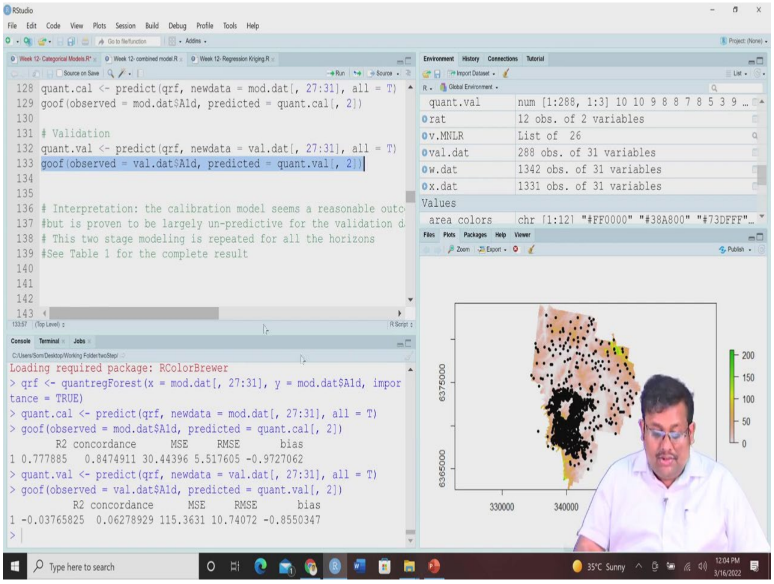




So, we are going to run it and let us see how it looks like. So, the model has been implemented. Now, we are going to calibrate based on our calibration data. So, our calibration data we have, we have predicted and now, we are going to use this goof function since it is a continuous model, we will be using the simple goof function and the results are as you can see here, 0.77 concordance 0.284.

(Refer Slide Time: 26:27)





Now, let us see that, what is the validation output? So, from this validation output, you can again we are going to use these goof function to see the results. So, we can see here from this that our interpretation should be the calibration model seems a reasonable outcome for predicting the depth of the A1 horizon. However, it is largely unpredicted for the validation samples. So, this is this two-stage modelling is then repeated for all the horizons, and will see table 1 for the complete result.

(Refer Slide Time: 27:08)

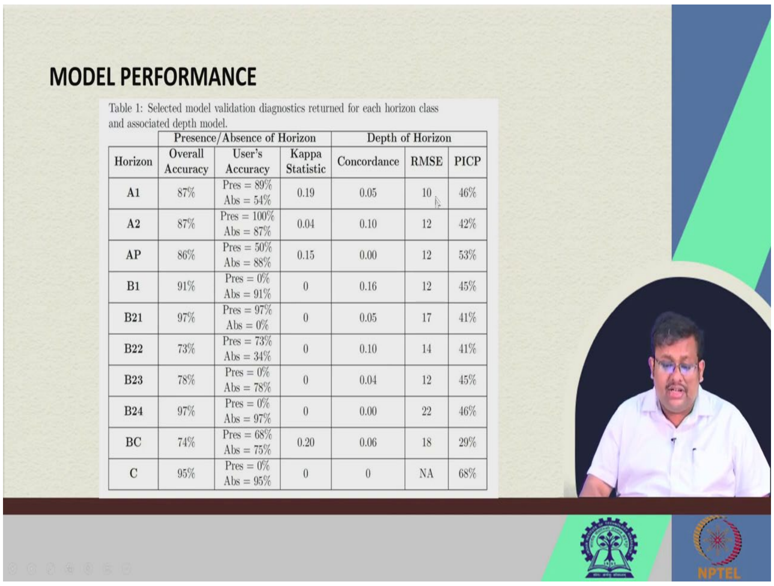


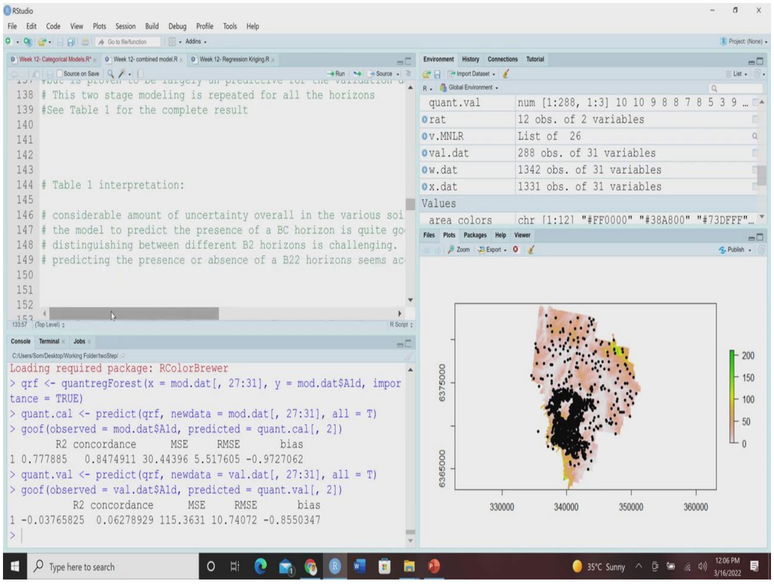
Table 1: Selected model validation diagnostics returned for each horizon class and associated depth model.

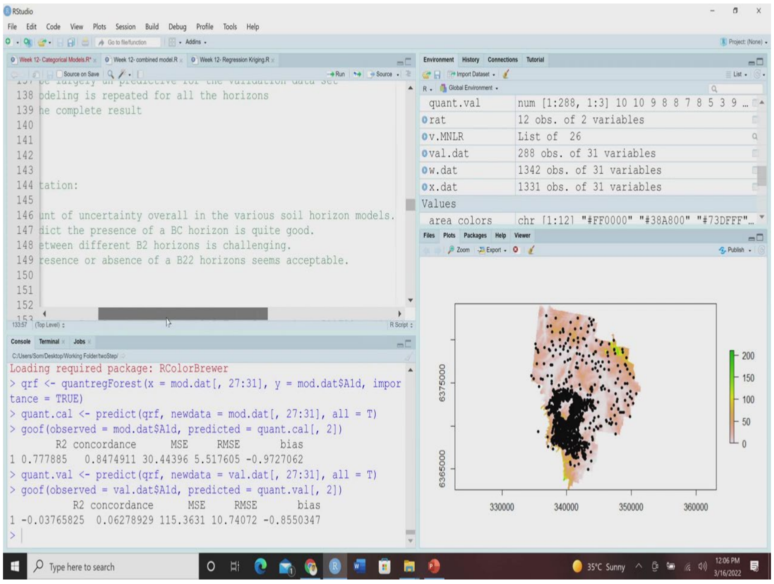
	Presence/Absence of Horizon			Depth of Horizon		
Horizon	Overall Accuracy	User's Accuracy	Kappa Statistic	Concordance	RMSE	PICP
A1	87%	Pres = 89% Abs = 54%	0.19	0.05	10	46%
A2	87%	Pres = 100% Abs = 87%	0.04	0.10	12	42%
AP	86%	Pres = 50% Abs = 88%	0.15	0.00	12	53%
B1	91%	Pres = 0% Abs = 91%	0	0.16	12	45%
B21	97%	Pres = 97% Abs = 0%	0	0.05	17	41%
B22	73%	Pres = 73% Abs = 34%	0	0.10	14	41%
B23	78%	Pres = 0% Abs = 78%	0	0.04	12	45%
B24	97%	Pres = 0% Abs = 97%	0	0.00	22	46%
BC	74%	Pres = 68% Abs = 75%	0.20	0.06	18	29%
C	95%	Pres = 0% Abs = 95%	0	0	NA	68%

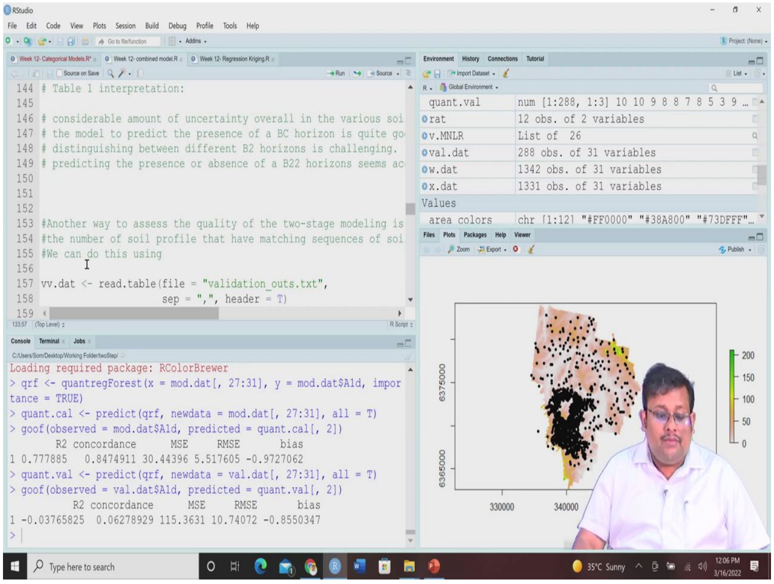
So, if we go back to this table 1 which I have produced here, so, you can see here that this is the table 1 and these are the horizons, they are the presence and absence of the horizon you can see here and then there depth of the horizon is predicted here. So, from this table 1 we can infer we can see that some important observations that here you can see the presence is 0 percent absence is 91 percent, for B21 it is 97 0, B22 73 34, B23 0 to 78 and then B24 0 to 97.

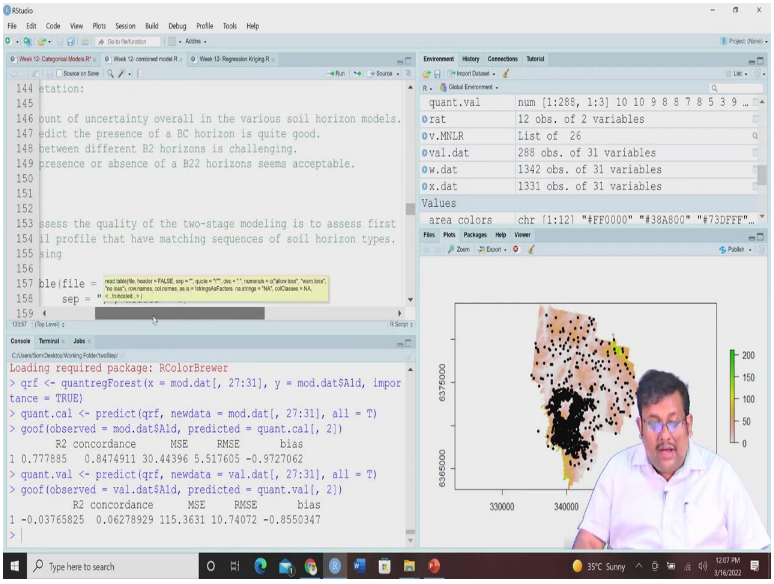
So, you can see clearly that prediction of the presence of an absence of Horizon specifically in these horizons B2 horizons are not very accurate and it is very challenging not accurate, I would not say I would say it is very challenging at the same time the presence and absence of B22 Horizon prediction looks okay. But at the and also, if you see the Kappa statics, from this kappa statistics, we can see that presence and absence of Horizon BC horizon also looks okay. These are the concordance correlation coefficient and RMSE values.

(Refer Slide Time: 28:33)









So, let us go back to our script and see that this table 1 interpretation I have given here that considerable amount of uncertainty overall in the various soil horizon model we have already seen, the model to predict the presence of BC horizon is quite good because of the Kappa statistics distinguishing between the different B2 horizon is challenging we know that and predicting the presence or absence of a B22 horizon seems okay acceptable.

Now, another way to access or assess the quality of the two-stage modelling is basically to assess first the number of soil profile that have matching sequence of soil horizons types and we can do this by using different methods. So, let us wrap up our lecture here, and we will discuss this thing, this part in our upcoming lecture.

And we will see how to use the other methods to see the fidelity between the predicted profile as well as the observed profile whether how to how to get the matches how to get the match between the observed profile as well as the predicted profile, what are the different ways we are going to see in our upcoming lecture. So, guys I hope that you have learned something new in this lecture and we will meet in our next lecture to discuss from here and we will discuss how to use this for producing the maps, thank you very much.