1. Run your dataset through Neural Networks/ Deep Learning and Random Forest algorithms either in R or Python AND in h2o (use the tutorials as your technical guide). Then answer the following questions in one single-spaced page:

1. How do these algorithms differ from the algorithms you used last week?

Our Dataset was regarding Student Alcohol Consumption, and last week we worked with tree ,ctree and naïve bayes theorem, using logarithmic based formula to identify what attributes are been identified with best composition, and what best results can be given as output in order to take best business and government decision which can be taken. In order to have best results from dataset, we need to have a good classification, tree would be best used to classify with percentage value shown with the tree, so based on the dataset, we had created a separate class as “ALC” by adjoining WALC and DALC attributes, to identify who drink more and how the student rate of education varies and differs, and what best supportive steps can be taken to reduce alcohol consumption and increase student grade in terms of this. People who drink in weekdays and weekend are considered to have less grade. We had identified this by performing a pre processing steps to the dataset, and classifying with the trees. Trees gives a Dataset to get classified and make a Top and Bottom Pressed to make the Dataset view get shorter, now it is the time to make Ctree , Naïve Bayes theorem to reduce it from Side ways, where when ever a Dataset is taken in consideration, it would always have a shorten path to fetch better results. Now in order to increase the confidence the acquired results are most accurate, it can be identified using the error percentage, and if the error percentage is very less, then we can see that the authoritative that Dataset results resolve with greater trust over the dataset. Random Forest is the algorithm which was been used this week , keeping error percentage in constraint, and then making then scrutinizing and the main functionality is to identify the dataset with deep root from middle of the deep part, and make with error percentage, so with less error percentage, such effective the dataset becomes.  
  
  
###################### Random Forest (for next week) #############################

ind <- sample(2, nrow(student), replace=TRUE, prob=c(0.8, 0.2))

trainDataRF <- student[ind==1,]

testDataRF <- student[ind==2,]

library(randomForest)

rf <- randomForest(Alc ~ ., data=trainDataRF, ntree=100, proximity=TRUE)

rf <- randomForest(Alc ~ ., data=trainDataRF, ntree=1000, proximity=TRUE)

table(predict(rf), trainDataRF$Alc)

print(rf)

attributes(rf)

plot(rf)

importance(rf)

varImpPlot(rf)

studentPred <- predict(rf, newdata=testDataRF)

table(studentPred, testDataRF$Alc)

> table(predict(rf), trainDataRF$Alc)

1 2 3 4 5

1 283 1 1 0 0

2 0 138 16 6 0

3 0 1 42 15 5

4 0 0 0 2 1

5 0 0 0 0 8

> print(rf)

Call:

randomForest(formula = Alc ~ ., data = trainDataRF, ntree = 1000, proximity = TRUE)

Type of random forest: classification

Number of trees: 1000

No. of variables tried at each split: 5

OOB estimate of error rate: 8.86%

Confusion matrix:

1 2 3 4 5 class.error

1 283 0 0 0 0 0.00000000

2 1 138 1 0 0 0.01428571

3 1 16 42 0 0 0.28813559

4 0 6 15 2 0 0.91304348

5 0 0 5 1 8 0.42857143

> importance(rf)

MeanDecreaseGini

school 1.8144690

sex 5.2295885

age 7.3249242

address 1.6036700

famsize 1.9947052

Pstatus 0.7633893

Medu 5.3528386

Fedu 5.1488638

Mjob 5.9173867

Fjob 4.1911906

reason 5.1019169

guardian 2.6671634

traveltime 3.7952430

studytime 4.3079436

failures 2.3386606

schoolsup 0.8795662

famsup 2.1338523

paid 0.8877970

activities 1.7223965

nursery 1.4111037

higher 1.2412624

internet 1.1455495

romantic 1.9854718

famrel 5.4298800

freetime 6.5820345

goout 9.6761388

Dalc 64.6829969

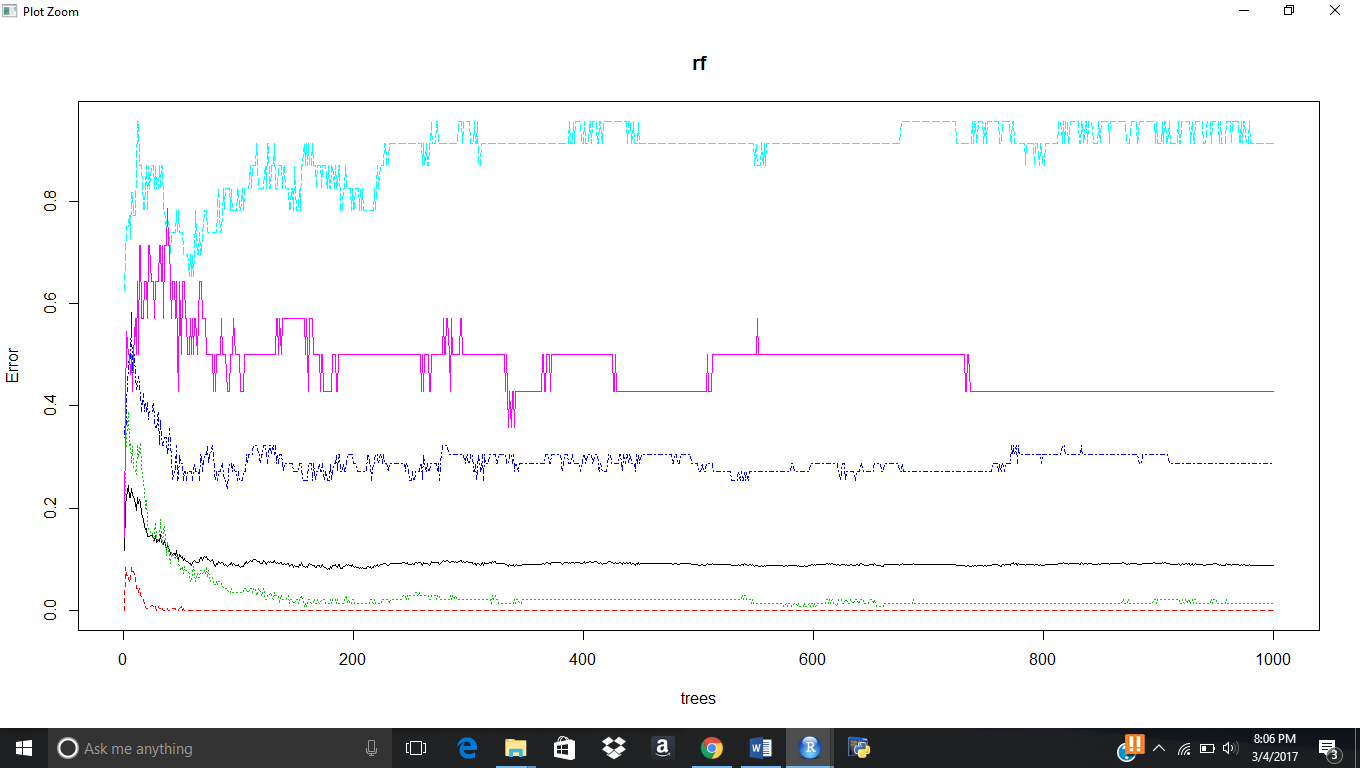
Walc 98.4946683

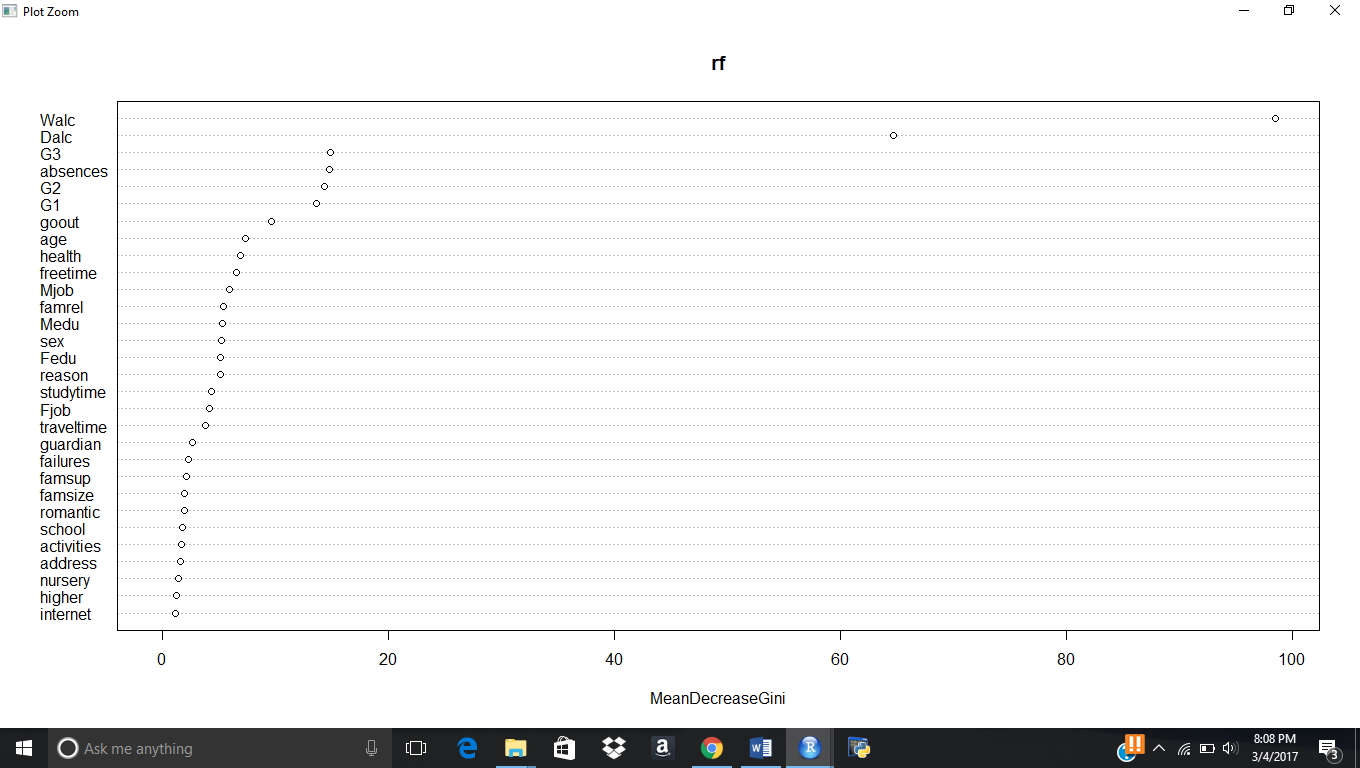
health 6.8987130

absences 14.8198964

G1 13.6395996

G2 14.3276895

G3 14.9041831



> table(studentPred, testDataRF$Alc)

studentPred 1 2 3 4 5

1 71 0 0 0 0

2 0 38 4 0 1

3 0 0 10 6 0

4 0 0 0 0 0

5 0 0 0 0 0

**R Code for Last week for tree:**

Alc=(Dalc\*+/Walc\*2)/7

student<-student\_por1

ind <- sample(2, nrow(student), replace=TRUE, prob=c(0.7, 0.3))

trainDataTree <- student[ind==1,]

testDataTree <- student[ind==2,]

myFormula <- Alc~ school+sex+age+address+famsize+Pstatus+Medu+Fedu+Mjob+Fjob+reason+guardian+traveltime+studytime+failures+schoolsup+famsup+paid+activities+nursery+higher+internet+romantic+famrel+freetime+goout+health+absences

student\_tree <- tree(myFormula, data=trainDataTree)

summary(student\_tree)

print(student\_tree)

plot(student\_tree)

text(student\_tree)

testPred <- predict(student\_tree, newdata = testDataTree)

table(testPred, testDataTree$Alc)

show(testPred)

library(MASS)

studenttree <- tree(myFormula,data=student)

plot(studenttree)

plot(studenttree,col=8)

text(studenttree,digits=2)

summary(studenttree)

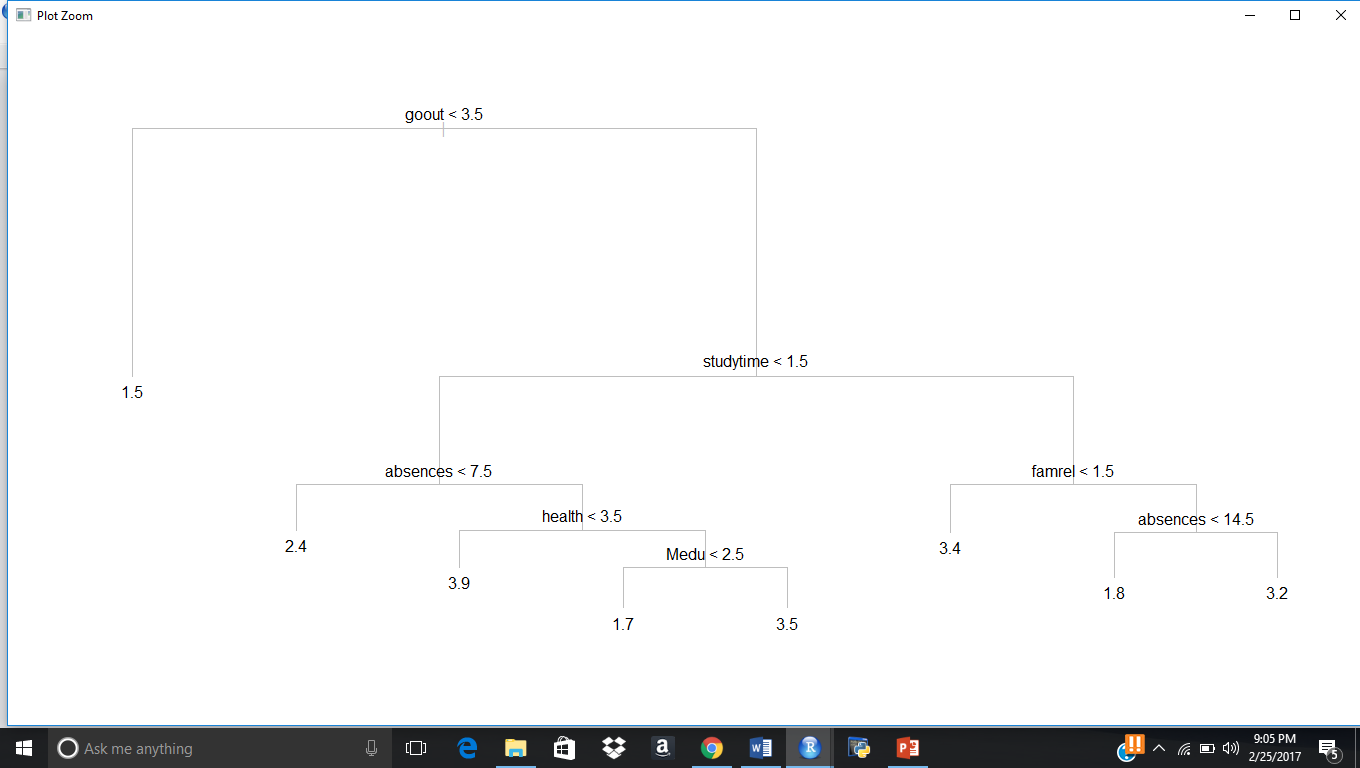
studentsnip=snip.tree(studenttree,nodes=c(7,12))

irissnip

plot(studentsnip)

text(studentsnip)

summary(studentsnip)



> table(testPred, testDataTree$Alc)

testPred 1 2 3 4 5

1.33333333333333 2 0 0 0 0

1.41463414634146 9 2 3 0 0

1.44827586206897 63 31 4 1 0

1.55 3 1 1 0 0

1.71428571428571 1 0 3 0 0

1.91304347826087 7 4 1 1 0

2.1 2 0 0 0 0

2.28571428571429 1 5 1 0 0

2.31578947368421 3 3 4 1 1

2.66666666666667 1 1 0 1 1

3 4 2 0 2 0

3.33333333333333 4 3 1 1 1

3.42857142857143 1 2 2 0 0

4.16666666666667 1 0 1 0 1

################# C TREE (conditional inference) ###################

set.seed(1234)

ind <- sample(2, nrow(student), replace=TRUE, prob=c(0.7, 0.3))

trainData <- student[ind==1,]

testData <- student[ind==2,]

library(party)

student\_ctree <- ctree(myFormula, data=trainData)

table(predict(iris\_ctree), trainData$Species)

print(student\_ctree)

plot(student\_ctree)

testPred <- predict(student\_ctree, newdata = testData)

table(testPred, testData$Alc)

> table(predict(student\_ctree), trainData$Alc)

1 2 3 4 5

1 227 90 20 6 1

2 10 19 7 1 1

3 14 23 27 11 10

4 0 0 0 0 0

5 0 0 0 0 0

> table(testPred, testData$Alc)

testPred 1 2 3 4 5

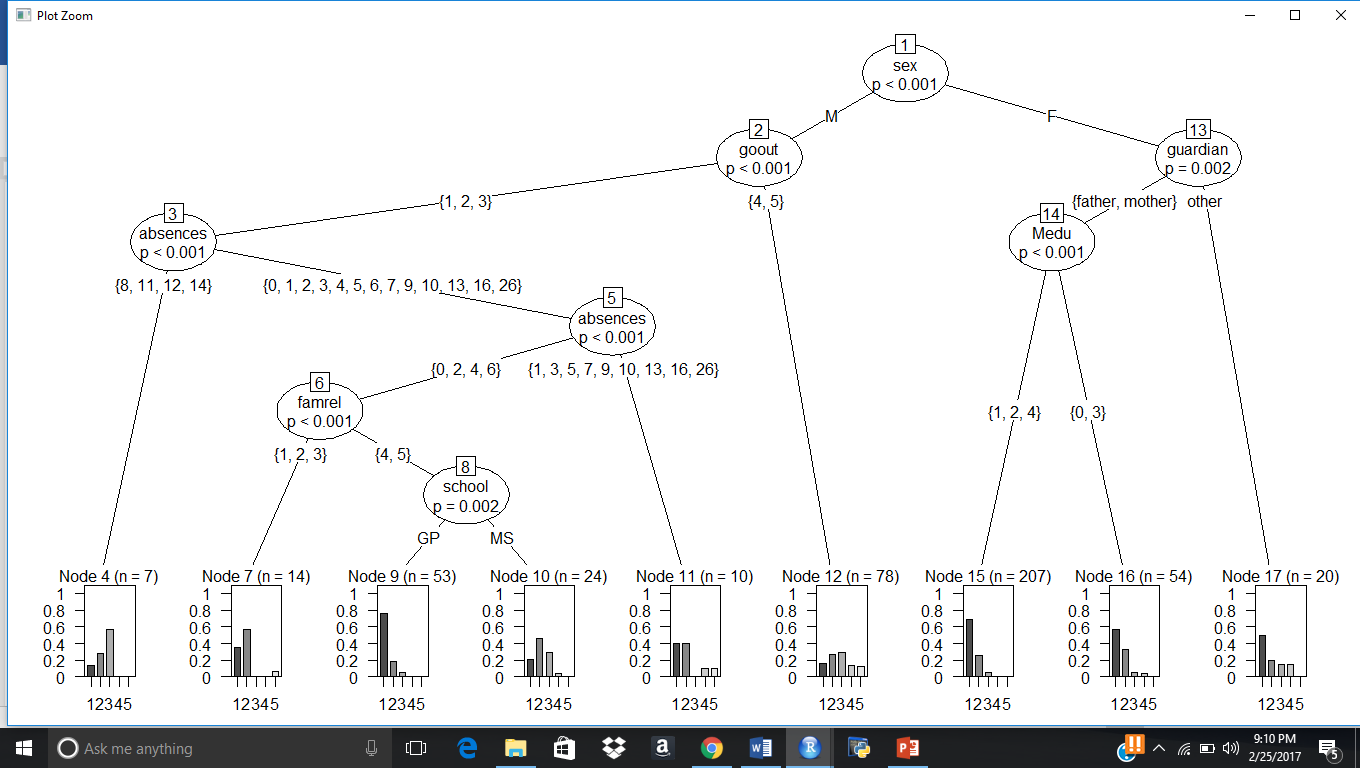
1 85 31 5 5 2

2 10 8 2 0 0

3 8 7 12 6 1

4 0 0 0 0 0

5 0 0 0 0 0



2. Are their results more useful or less useful than the results you computed last week? Why?

R Code :   
  
  
  
  
###################### Random Forest (for next week) #############################

ind <- sample(2, nrow(student), replace=TRUE, prob=c(0.8, 0.2))

trainDataRF <- student[ind==1,]

testDataRF <- student[ind==2,]

library(randomForest)

rf <- randomForest(Alc ~ ., data=trainDataRF, ntree=100, proximity=TRUE)

rf <- randomForest(Alc ~ ., data=trainDataRF, ntree=1000, proximity=TRUE)

table(predict(rf), trainDataRF$Alc)

print(rf)

attributes(rf)

plot(rf)

importance(rf)

varImpPlot(rf)

studentPred <- predict(rf, newdata=testDataRF)

table(studentPred, testDataRF$Alc)

> table(predict(rf), trainDataRF$Alc)

1 2 3 4 5

1 283 1 1 0 0

2 0 138 16 6 0

3 0 1 42 15 5

4 0 0 0 2 1

5 0 0 0 0 8

> print(rf)

Call:

randomForest(formula = Alc ~ ., data = trainDataRF, ntree = 1000, proximity = TRUE)

Type of random forest: classification

Number of trees: 1000

No. of variables tried at each split: 5

OOB estimate of error rate: 8.86%

Confusion matrix:

1 2 3 4 5 class.error

1 283 0 0 0 0 0.00000000

2 1 138 1 0 0 0.01428571

3 1 16 42 0 0 0.28813559

4 0 6 15 2 0 0.91304348

5 0 0 5 1 8 0.42857143

> importance(rf)

MeanDecreaseGini

school 1.8144690

sex 5.2295885

age 7.3249242

address 1.6036700

famsize 1.9947052

Pstatus 0.7633893

Medu 5.3528386

Fedu 5.1488638

Mjob 5.9173867

Fjob 4.1911906

reason 5.1019169

guardian 2.6671634

traveltime 3.7952430

studytime 4.3079436

failures 2.3386606

schoolsup 0.8795662

famsup 2.1338523

paid 0.8877970

activities 1.7223965

nursery 1.4111037

higher 1.2412624

internet 1.1455495

romantic 1.9854718

famrel 5.4298800

freetime 6.5820345

goout 9.6761388

Dalc 64.6829969

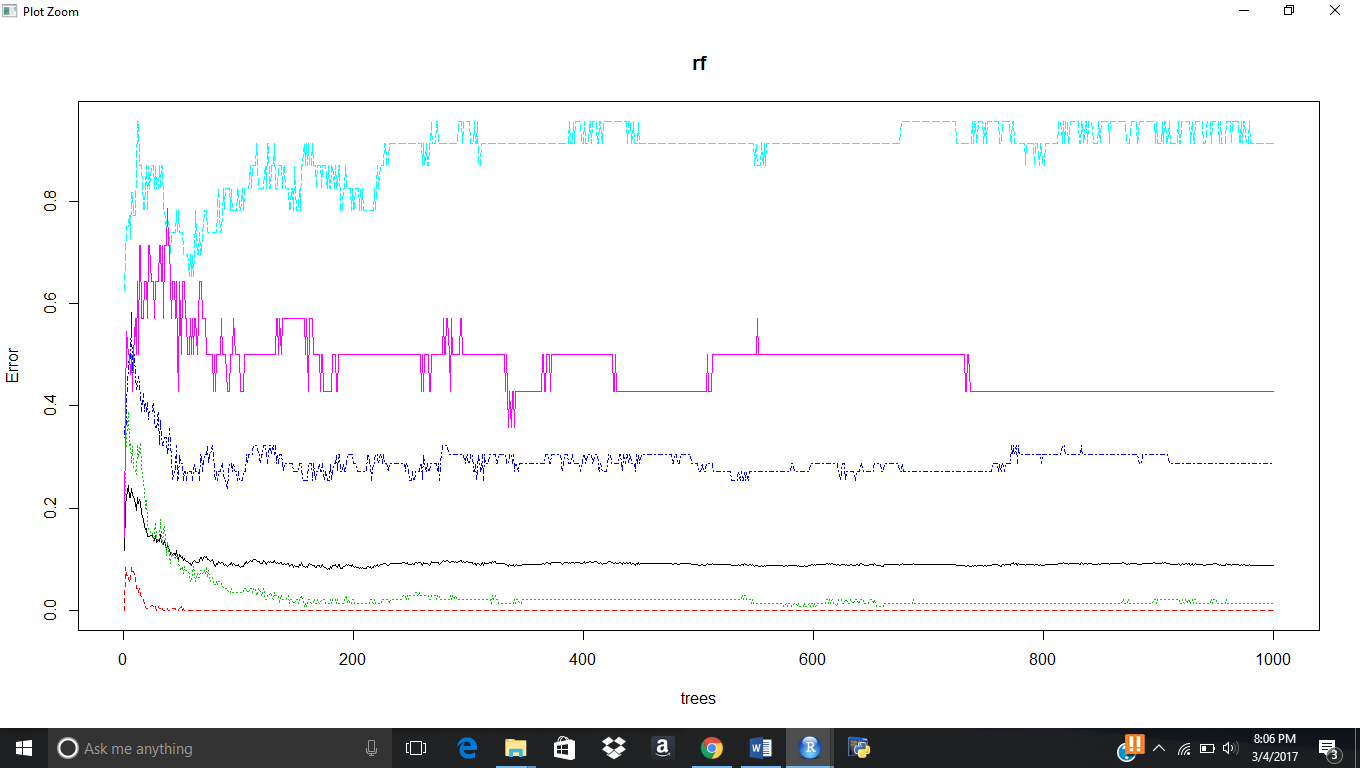
Walc 98.4946683

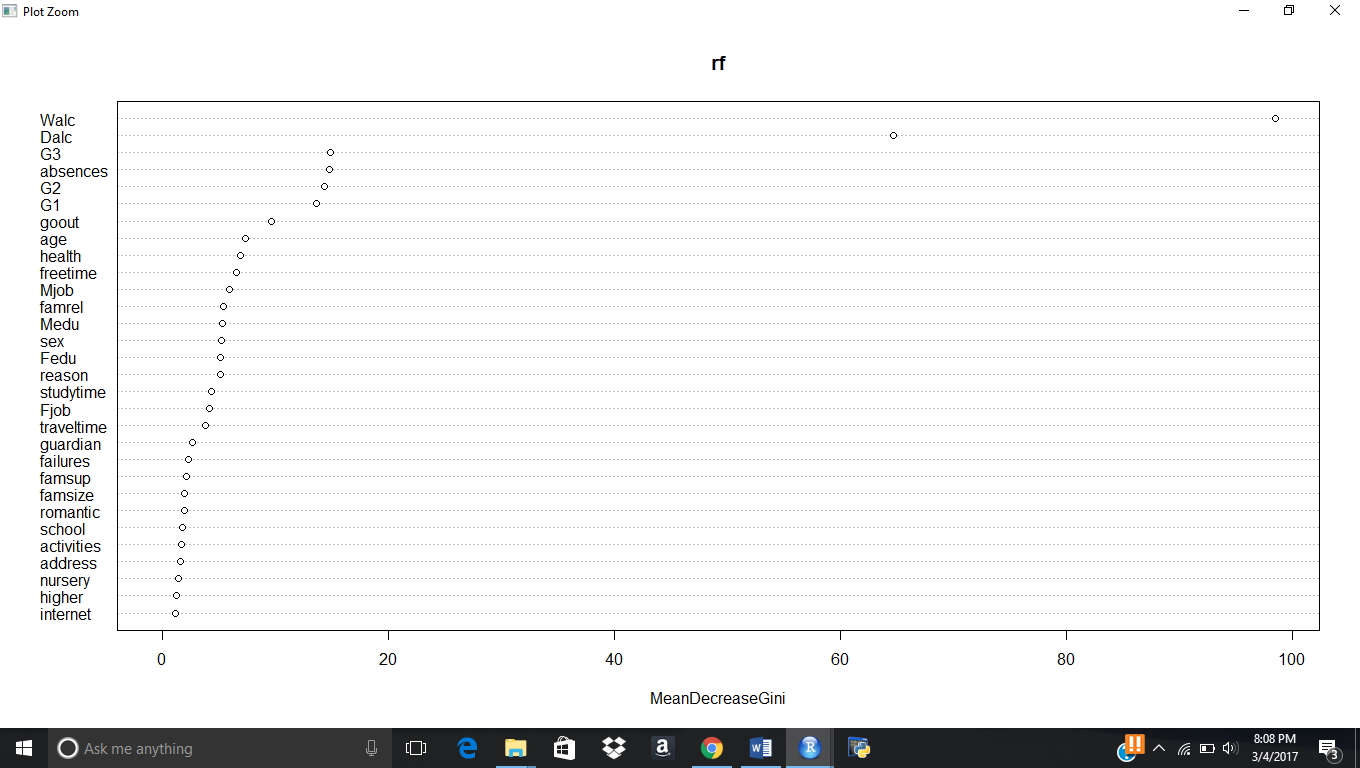
health 6.8987130

absences 14.8198964

G1 13.6395996

G2 14.3276895

G3 14.9041831



> table(studentPred, testDataRF$Alc)

studentPred 1 2 3 4 5

1 71 0 0 0 0

2 0 38 4 0 1

3 0 0 10 6 0

4 0 0 0 0 0

5 0 0 0 0 0

**R Code for Last week for tree:**

Alc=(Dalc\*+/Walc\*2)/7

student<-student\_por1

ind <- sample(2, nrow(student), replace=TRUE, prob=c(0.7, 0.3))

trainDataTree <- student[ind==1,]

testDataTree <- student[ind==2,]

myFormula <- Alc~ school+sex+age+address+famsize+Pstatus+Medu+Fedu+Mjob+Fjob+reason+guardian+traveltime+studytime+failures+schoolsup+famsup+paid+activities+nursery+higher+internet+romantic+famrel+freetime+goout+health+absences

student\_tree <- tree(myFormula, data=trainDataTree)

summary(student\_tree)

print(student\_tree)

plot(student\_tree)

text(student\_tree)

testPred <- predict(student\_tree, newdata = testDataTree)

table(testPred, testDataTree$Alc)

show(testPred)

library(MASS)

studenttree <- tree(myFormula,data=student)

plot(studenttree)

plot(studenttree,col=8)

text(studenttree,digits=2)

summary(studenttree)

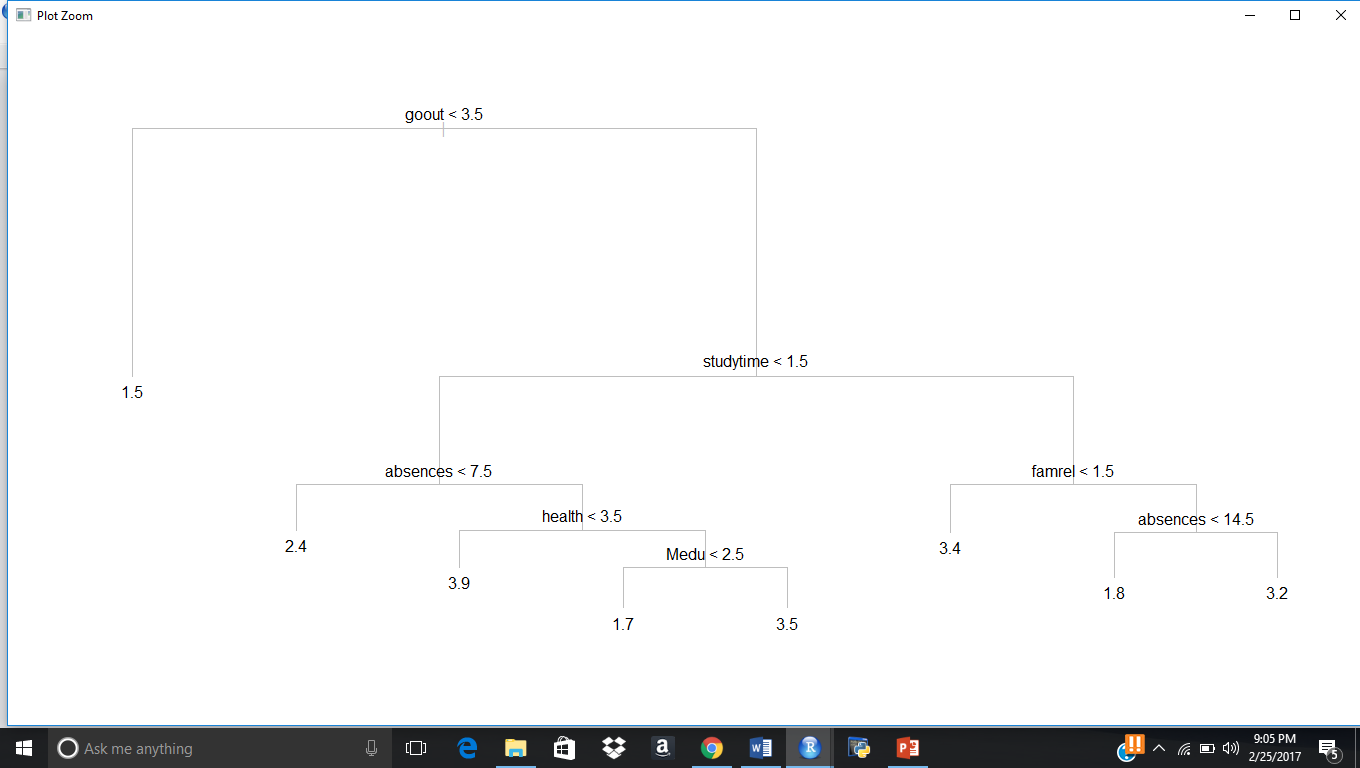
studentsnip=snip.tree(studenttree,nodes=c(7,12))

irissnip

plot(studentsnip)

text(studentsnip)

summary(studentsnip)



> table(testPred, testDataTree$Alc)

testPred 1 2 3 4 5

1.33333333333333 2 0 0 0 0

1.41463414634146 9 2 3 0 0

1.44827586206897 63 31 4 1 0

1.55 3 1 1 0 0

1.71428571428571 1 0 3 0 0

1.91304347826087 7 4 1 1 0

2.1 2 0 0 0 0

2.28571428571429 1 5 1 0 0

2.31578947368421 3 3 4 1 1

2.66666666666667 1 1 0 1 1

3 4 2 0 2 0

3.33333333333333 4 3 1 1 1

3.42857142857143 1 2 2 0 0

4.16666666666667 1 0 1 0 1

################# C TREE (conditional inference) ###################

set.seed(1234)

ind <- sample(2, nrow(student), replace=TRUE, prob=c(0.7, 0.3))

trainData <- student[ind==1,]

testData <- student[ind==2,]

library(party)

student\_ctree <- ctree(myFormula, data=trainData)

table(predict(iris\_ctree), trainData$Species)

print(student\_ctree)

plot(student\_ctree)

testPred <- predict(student\_ctree, newdata = testData)

table(testPred, testData$Alc)

> table(predict(student\_ctree), trainData$Alc)

1 2 3 4 5

1 227 90 20 6 1

2 10 19 7 1 1

3 14 23 27 11 10

4 0 0 0 0 0

5 0 0 0 0 0

> table(testPred, testData$Alc)

testPred 1 2 3 4 5

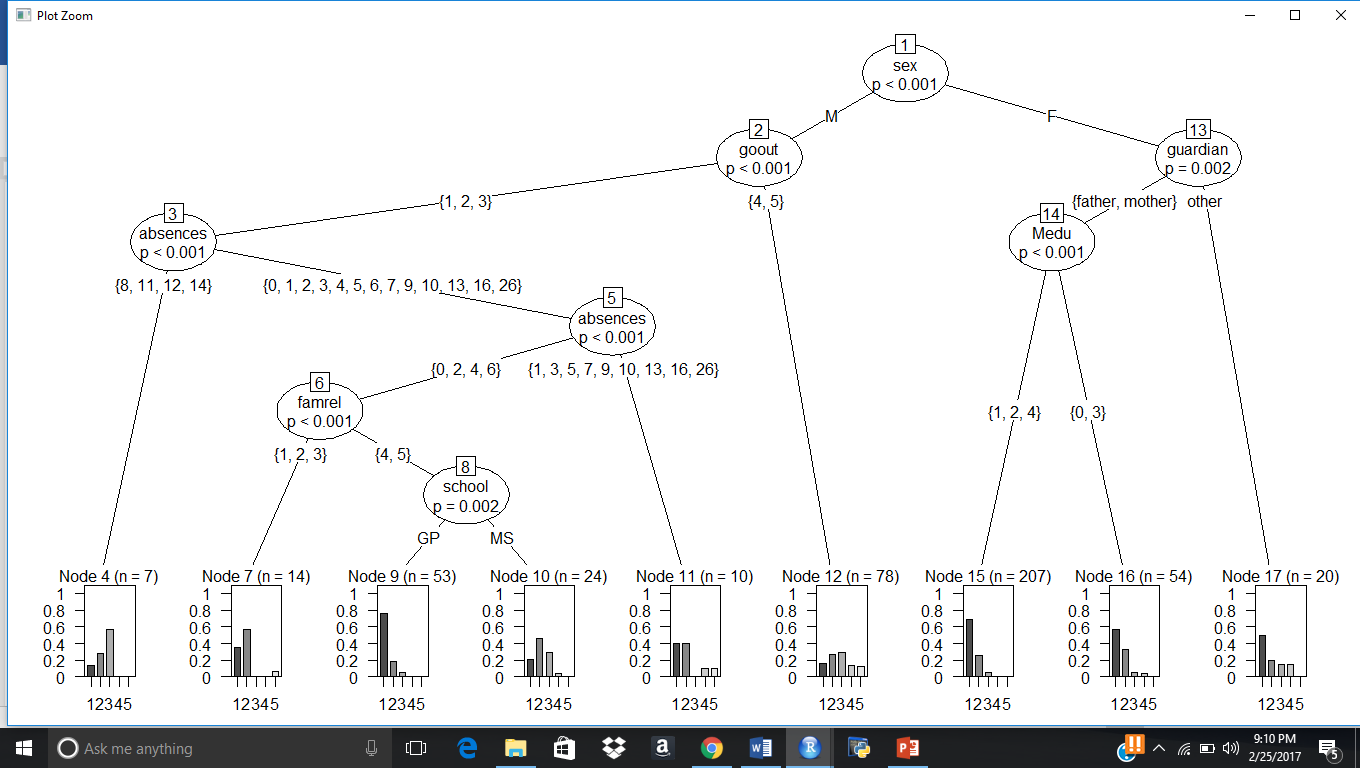
1 85 31 5 5 2

2 10 8 2 0 0

3 8 7 12 6 1

4 0 0 0 0 0

5 0 0 0 0 0



###################### NAIVE BAYES (conditional probability) ###############################

library(mlbench)

plot(student[,34])

title(main="number of alcohol consumption", xlab="level of consumption", ylab="number of student")

student[,"train"] <- ifelse(runif(nrow(student))<0.80,1,0)

trainColNum <- grep('train', names(student))

trainstudent <- student[student$train==1,-trainColNum]

teststudent <- student[student$train==0,-trainColNum]

library(e1071)

nb\_model <- naiveBayes(Alc~.,data = trainstudent)

nb\_model

summary(nb\_model)

str(nb\_model)

nb\_test\_predict <- predict(nb\_model,teststudent[,-1])

table(pred=nb\_test\_predict,true=teststudent$Alc)

mean(nb\_test\_predict==teststudent$Alc)

nb\_multiple\_runs <- function(train\_fraction,n){

fraction\_correct <- rep(NA,n)

for (i in 1:n){

student[,"train"] <- ifelse(runif(nrow(student))<train\_fraction,1,0)

trainColNum <- grep('train',names(student))

trainstudent <- student[student$train==1,-trainColNum]

teststudent <- student[student$train==0,-trainColNum]

nb\_model <- naiveBayes(Alc~.,data = trainstudent)

nb\_test\_predict <- predict(nb\_model,teststudent[,-1])

fraction\_correct[i] <- mean(nb\_test\_predict==teststudent$Alc)

}

return(fraction\_correct)

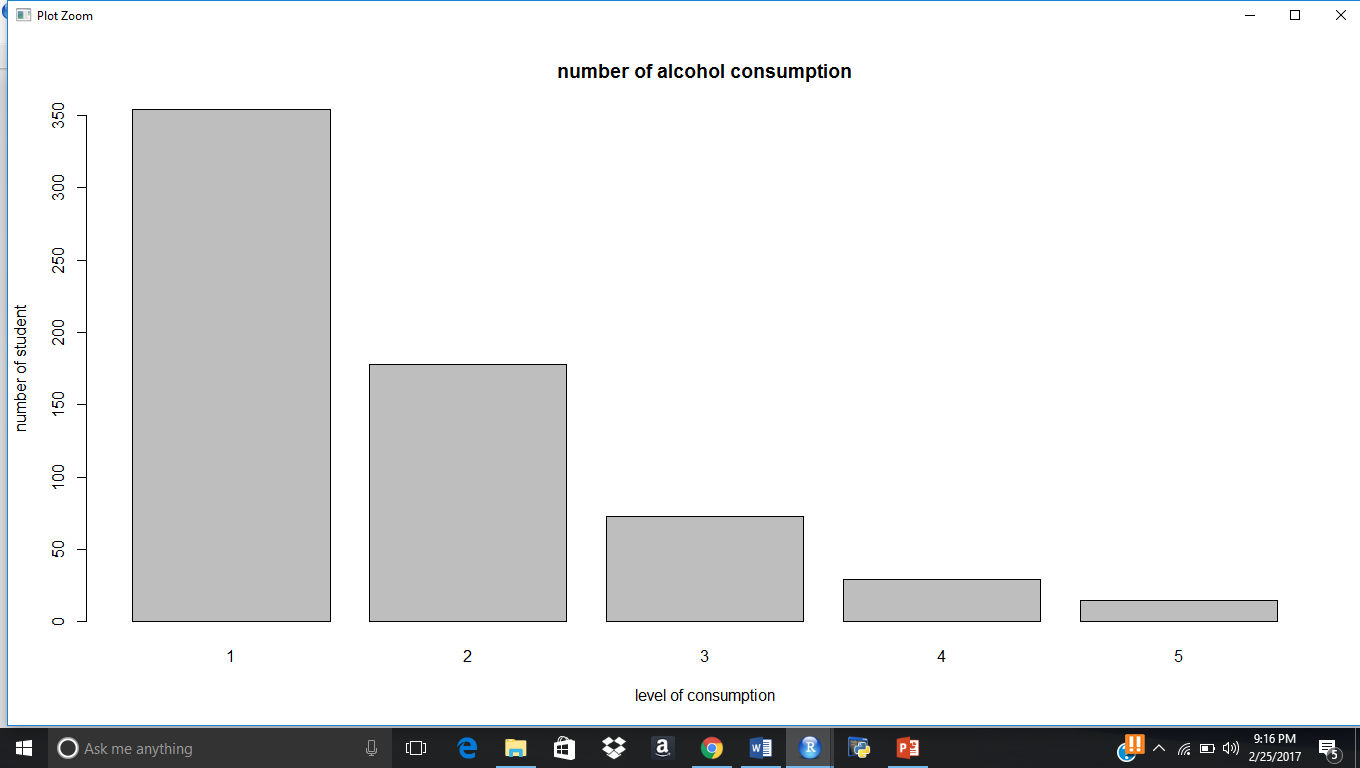
}

fraction\_correct\_predictions <- nb\_multiple\_runs(0.8,20)

fraction\_correct\_predictions

summary(fraction\_correct\_predictions)

sd(fraction\_correct\_predictions)



> summary(nb\_model)

Length Class Mode

apriori 5 table numeric

tables 33 -none- list

levels 5 -none- character

call 4 -none- call

> nb\_test\_predict <- predict(nb\_model,teststudent[,-1])

> table(pred=nb\_test\_predict,true=teststudent$Alc)

true

pred 1 2 3 4 5

1 105 0 0 1 0

2 4 44 5 2 1

3 0 2 20 3 1

4 0 0 0 2 4

5 0 0 0 0 1

> mean(nb\_test\_predict==teststudent$Alc)

[1] 0.8820513

> fraction\_correct\_predictions

[1] 0.9387755 0.8709677 0.8661972 0.8270677 0.9000000 0.8666667 0.8870968 0.9268293

[9] 0.8769231 0.9126984 0.8947368 0.8333333 0.8802817 0.8560000 0.8939394 0.8936170

[17] 0.8661417 0.8769231 0.8768116 0.8906250

> summary(fraction\_correct\_predictions)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0.8271 0.8665 0.8786 0.8818 0.8941 0.9388

> sd(fraction\_correct\_predictions)

[1] 0.02718371

Bayes algorithm is given best results than the other algorithm which was been used , because of this we are able to identify the percentage of the attribute best inter related, and make it best pre processed forwarded to make a best relation so that it suffice the big question using the class attribute how ever it also give how much trust and relation does ALC is given in compared to other attributes to make it with other attributes, if we can identify the percentage way it gives more information and descriptive format for the client and the owner to make more informative decision to be taken on this.

3. What new insights about the data in the dataset do the results give you?

By using Random Forest we had identified the below results are been fetched using Random Forest, and in order to make it effective, and so for that we had given 1000 trees in consideration and ofr our dataset, and we had given number of split of 5, and then error percentage was 8.86% where it was a resemblance how much interdependence is there between ,

randomForest(formula = Alc ~ ., data = trainDataRF, ntree = 1000, proximity = TRUE)

Type of random forest: classification

Number of trees: 1000

No. of variables tried at each split: 5

OOB estimate of error rate: 8.86%  
  
  
Now I had made with h20 used in R tools, so made with effective results which resolve with better decision maker,   
  
R Code:   
  
  
h2o.init(ip="localhost",port = 54321,max\_mem\_size = "2000m")

x<-setdiff(colnames(student),c("Alc","Walc","Dalc"))

y="Alc"

set.seed(1234)

ind<-sample(2,nrow(student), replace=TRUE,prob=c(0.7, 0.3))

trainDF<-student[ind==1,]

testDF<-student[ind==2,]

model<-h2o.deeplearning(x=x,

y=y,

training\_frame = as.h2o(trainDF),

nfolds = 3,

stopping\_rounds = 7,

epochs=400,

overwrite\_with\_best\_model = TRUE,

activation = "Tanh",

input\_dropout\_ratio = 0.2,

hidden = c(10,10,10),

l1=6e-4,

loss = "Automatic",

distribution = "AUTO",

stopping\_metric = "MSE")

predictions1 =predict(object = model, newdata = as.h2o(trainDF))

predictions = as.data.frame(predictions1)

str(predictions)

head(predictions)

tail(predictions)

performance = h2o.performance(model = model)

print(performance)

predictions = as.data.frame(predict(object = model, newdata = as.h2o(testDF)))

head(predictions)

tail(predictions)

performance = h2o.performance(model = model)

print(performance)

Training dataset result

> head(predictions)

predict

1 1

2 1

3 2

4 1

5 1

6 1

MSE: (Extract with `h2o.mse`) 0.02781112

RMSE: (Extract with `h2o.rmse`) 0.1667667

Logloss: (Extract with `h2o.logloss`) 0.1083186

Mean Per-Class Error: 0.03513097

Confusion Matrix: vertical: actual; across: predicted

1 2 3 4 5 Error Rate

1 243 0 0 0 0 0.0000 = 0 / 243

2 8 108 1 0 1 0.0847 = 10 / 118

3 3 0 50 0 2 0.0909 = 5 / 55

4 0 0 0 21 0 0.0000 = 0 / 21

5 0 0 0 0 14 0.0000 = 0 / 14

Totals 254 108 51 21 17 0.0333 = 15 / 451

Test dataset result

> head(predictions)

predict

1 1

2 2

3 2

4 1

5 3

6 1

Extract training frame with `h2o.getFrame("trainDF")`

MSE: (Extract with `h2o.mse`) 0.02781112

RMSE: (Extract with `h2o.rmse`) 0.1667667

Logloss: (Extract with `h2o.logloss`) 0.1083186

Mean Per-Class Error: 0.03513097

Confusion Matrix: Extract with `h2o.confusionMatrix(<model>,train = TRUE)`)

=========================================================================

Confusion Matrix: vertical: actual; across: predicted

1 2 3 4 5 Error Rate

1 243 0 0 0 0 0.0000 = 0 / 243

2 8 108 1 0 1 0.0847 = 10 / 118

3 3 0 50 0 2 0.0909 = 5 / 55

4 0 0 0 21 0 0.0000 = 0 / 21

5 0 0 0 0 14 0.0000 = 0 / 14

Totals 254 108 51 21 17 0.0333 = 15 / 451

By using this code it how best the prediction helps with it provide with error rate with the best related attributes what best error rate and less error rate proves that it has given good confidence and trust increases on dataset, and in order to prove it, we had error rate resolves with approximately with zero. The training and test Dataset results with h20 are much more effective to provide effective results for dataset.   
  
  
Mean per class error is 0.03, and logloss is 0.10 and how many rate it identified with number of rate to total number of results, and so it takes what best 5 attributes are been taken to refine with best results.

4. What business, political, or medical decisions could management make on the basis of your results?

So based on this kind of results, yes we can identify some decision, which would make government to take good decision on student as considered as student to be considered as next growth of the country.

Some of the decisions said to be as without having time given to student for education, it helps student to drink more, so they can start time in the schooling as study time to make them avigated with some other work for education which make student to decrease intke of alcohol. Family time is something very important, so to build more parks for students so that they can spend more time with family, and events for students so that they can time with family. Health is one factor, to make a regular check up in school to check student health checkup done on monthly basis. Providing more jobs for women would also make student to get inspired from child hood and make them start working.  
  
  
These are some of the important things which we considered as important points which are been found in our results, and also as on we make more reverse interpreted relation to it, it would generate more results for government to take more results.

Trust provided by using this algorithm was good and support given with error percentage was very less in H20 algorithm, and then using Random forest it has given with 8% of error percentage as less percentage as effective with the dataset, and then to identify H20 more effective output and RF is used to identify to treat the attributes with best scrutinize way, and give the dataset reading with middle rounding and which attach and collect with more datasets.