Title: Use Random Forest to prepare a model on fraud data

treating those who have taxable\_income <= 30000 as "Risky" and others are "Good"

Ans::

library(caret)

> library(psych)

> frauddata <- read.csv(file.choose())

> View(frauddata)

> summary(frauddata)

Undergrad Marital.Status Taxable.Income City.Population Work.Experience Urban

Length:600 Length:600 Min. :10003 Min. : 25779 Min. : 0.00 Length:600

Class :character Class :character 1st Qu.:32872 1st Qu.: 66967 1st Qu.: 8.00 Class :character

Mode :character Mode :character Median :55075 Median :106494 Median :15.00 Mode :character

Mean :55208 Mean :108747 Mean :15.56

3rd Qu.:78612 3rd Qu.:150114 3rd Qu.:24.00

Max. :99619 Max. :199778 Max. :30.00

> describe(frauddata)

vars n mean sd median trimmed mad min max range skew kurtosis se

Undergrad\* 1 600 1.52 0.50 2.0 1.52 0.00 1 2 1 -0.08 -2.00 0.02

Marital.Status\* 2 600 2.05 0.82 2.0 2.06 1.48 1 3 2 -0.09 -1.52 0.03

Taxable.Income 3 600 55208.38 26204.83 55074.5 55188.24 33506.02 10003 99619 89616 0.03 -1.21 1069.81

City.Population 4 600 108747.37 49850.08 106493.5 107896.48 61612.41 25779 199778 173999 0.12 -1.13 2035.12

Work.Experience 5 600 15.56 8.84 15.0 15.61 11.86 0 30 30 0.02 -1.17 0.36

Urban\* 6 600 1.50 0.50 2.0 1.50 0.00 1 2 1 -0.01 -2.00 0.02

> pairs(frauddata)

Error in pairs.default(frauddata) : non-numeric argument to 'pairs'

> str(frauddata)

'data.frame': 600 obs. of 6 variables:

$ Undergrad : chr "NO" "YES" "NO" "YES" ...

$ Marital.Status : chr "Single" "Divorced" "Married" "Single" ...

$ Taxable.Income : int 68833 33700 36925 50190 81002 33329 83357 62774 83519 98152 ...

$ City.Population: int 50047 134075 160205 193264 27533 116382 80890 131253 102481 155482 ...

$ Work.Experience: int 10 18 30 15 28 0 8 3 12 4 ...

$ Urban : chr "YES" "YES" "YES" "YES" ...

> attach(frauddata)

The following objects are masked from frauddata (pos = 3):

City.Population, Marital.Status, Taxable.Income, Undergrad, Urban, Work.Experience

> ############### converting taxable to categorical type ###########

> tax\_cat <- ifelse(Taxable.Income<=30000,"risky","good")

> frauddata <- data.frame(tax\_cat,frauddata[,-3])

> table(frauddata$tax\_cat)

good risky

476 124

>

> ############ splitting of data to train and test ########################

> set.seed(100)

> cut <- createDataPartition(tax\_cat,p=0.7,list = F)

> train\_f <- frauddata[cut,]

> test\_f <- frauddata[-cut,]

>

> ############# model building #############################

> #model based on train data

> forest <- randomForest(as.factor(tax\_cat)~.,data = train\_f,importance=TRUE,mtry=2)

> forest

Call:

randomForest(formula = as.factor(tax\_cat) ~ ., data = train\_f, importance = TRUE, mtry = 2)

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 2

OOB estimate of error rate: 22.09%

Confusion matrix:

good risky class.error

good 328 6 0.01796407

risky 87 0 1.00000000

>

> # prediction and accuracy based on train data

> predict\_train <- predict(forest,train\_f)

> mean(predict\_train==train\_f$tax\_cat) # acc = 92.87%

[1] 0.9287411

> confusionMatrix(table(predict\_train,train\_f$tax\_cat))

Confusion Matrix and Statistics

predict\_train good risky

good 334 30

risky 0 57

Accuracy : 0.9287

95% CI : (0.8998, 0.9514)

No Information Rate : 0.7933

P-Value [Acc > NIR] : 1.385e-14

Kappa : 0.7509

Mcnemar's Test P-Value : 1.192e-07

Sensitivity : 1.0000

Specificity : 0.6552

Pos Pred Value : 0.9176

Neg Pred Value : 1.0000

Prevalence : 0.7933

Detection Rate : 0.7933

Detection Prevalence : 0.8646

Balanced Accuracy : 0.8276

'Positive' Class : good

>

>

> #prediction and accuracy based on test data

> predict\_test <- predict(forest,test\_f)

> mean(predict\_test==test\_f$tax\_cat) #acc = 78.77%

[1] 0.7877095

> confusionMatrix(table(predict\_test,test\_f$tax\_cat))

Confusion Matrix and Statistics

predict\_test good risky

good 140 36

risky 2 1

Accuracy : 0.7877

95% CI : (0.7205, 0.8452)

No Information Rate : 0.7933

P-Value [Acc > NIR] : 0.6154

Kappa : 0.0196

Mcnemar's Test P-Value : 8.636e-08

Sensitivity : 0.98592

Specificity : 0.02703

Pos Pred Value : 0.79545

Neg Pred Value : 0.33333

Prevalence : 0.79330

Detection Rate : 0.78212

Detection Prevalence : 0.98324

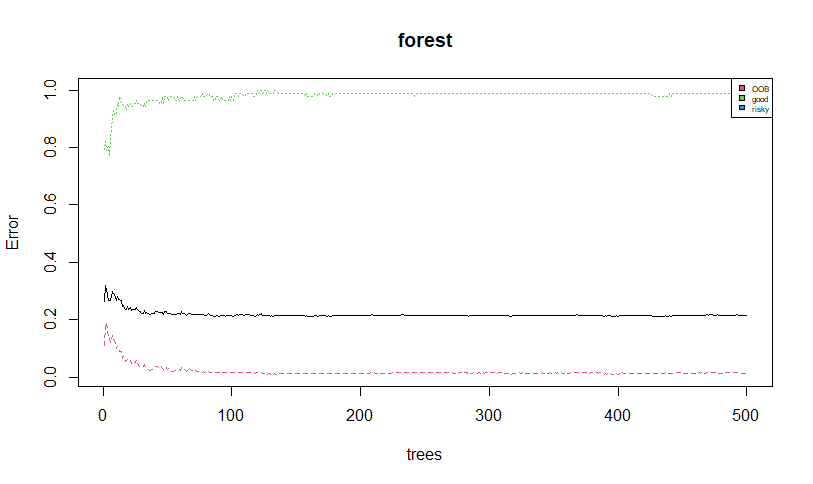
Balanced Accuracy : 0.50647

'Positive' Class : good

>

> plot(forest)

> legend("topright",col = 2:5,colnames(forest$err.rate),fill = 2:5,cex = 0.5)

> 

>

> importance(forest)

good risky MeanDecreaseAccuracy MeanDecreaseGini

Undergrad -1.1014921 -5.2906194 -3.375289 4.959548

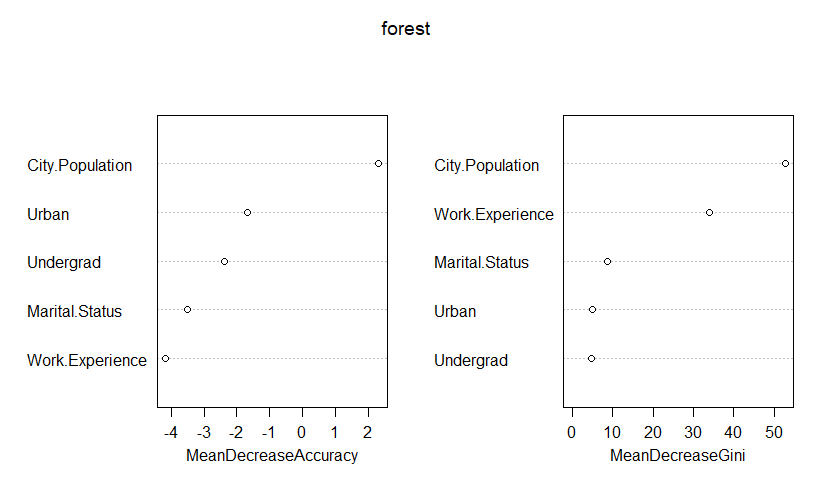
Marital.Status -0.3369408 -4.7546176 -2.349864 8.797790

City.Population 1.7216991 -0.5847524 1.186898 53.572585

Work.Experience -3.2165859 -4.6628961 -4.794466 34.031060

Urban -1.2095217 -0.9991161 -1.481211 5.183193

> varImpPlot(forest)



> #conclusion : most significant variable is city population

>

> # bagging

> acc <- c()

> i=2

> for(i in 2:10){

+ set.seed(100)

+ d <- createDataPartition(tax\_cat,p=0.8,list = F)

+ train\_d <- frauddata[d,]

+ test\_d <- frauddata[-d,]

+ model\_d <- randomForest(as.factor(tax\_cat)~.,data = train\_d,mtry=i)

+ pred\_b <- predict(model\_d,test\_d)

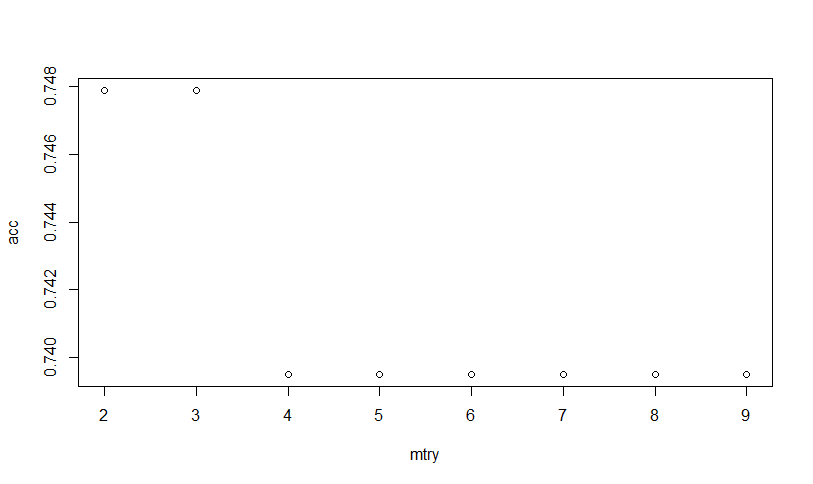
+ acc[i-2]=mean(pred\_b==test\_d$tax\_cat)

+ }

> acc

[1] 0.7478992 0.7478992 0.7394958 0.7394958 0.7394958 0.7394958 0.7394958 0.7394958

> plot(2:9,acc,xlab = "mtry",ylab = "acc")



> #higher accuracy is for mtry=2

>

> #choosen mtry as 2 with reasonable OOB error

>

> finalmodel <- randomForest(as.factor(tax\_cat)~.,data = train\_f,mtry=2)

> finalmodel

Call:

randomForest(formula = as.factor(tax\_cat) ~ ., data = train\_f, mtry = 2)

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 2

OOB estimate of error rate: 22.09%

Confusion matrix:

good risky class.error

good 326 8 0.0239521

risky 85 2 0.9770115

> mean(predict(finalmodel,test\_f)==test\_f$tax\_cat)

[1] 0.7988827