ams580\_hw3

Mustafa Yigit Isik

2023-02-20

#**Load Packages**

if (!requireNamespace("caTools")) install.packages('caTools')

## Loading required namespace: caTools

if (!requireNamespace("tidyverse")) install.packages('tidyverse')

## Loading required namespace: tidyverse

if (!requireNamespace("caret")) install.packages('caret')

## Loading required namespace: caret

if (!requireNamespace("rpart")) install.packages('rpart')  
if (!requireNamespace("rattle")) install.packages('rattle')

## Loading required namespace: rattle

library(caTools)  
library(tidyverse)

## ── Attaching packages  
## ───────────────────────────────────────  
## tidyverse 1.3.2 ──

## ✔ ggplot2 3.4.0 ✔ purrr 1.0.1   
## ✔ tibble 3.1.8 ✔ dplyr 1.0.10  
## ✔ tidyr 1.3.0 ✔ stringr 1.5.0   
## ✔ readr 2.1.4 ✔ forcats 1.0.0   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(caret)

## Loading required package: lattice  
##   
## Attaching package: 'caret'  
##   
## The following object is masked from 'package:purrr':  
##   
## lift

library(rpart)  
library(rattle)

## Loading required package: bitops  
## Rattle: A free graphical interface for data science with R.  
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

#**Read the data**

data <- read.csv("/Users/mustafayigitisik/Desktop/stuff/semesters/spring 2023/ams 580/hw3\_CART/banknote.csv")  
  
#removes potential rows where at least one cell is empty   
data <- na.omit(data)  
cat('There are', nrow(data), 'rows left.')

## There are 1372 rows left.

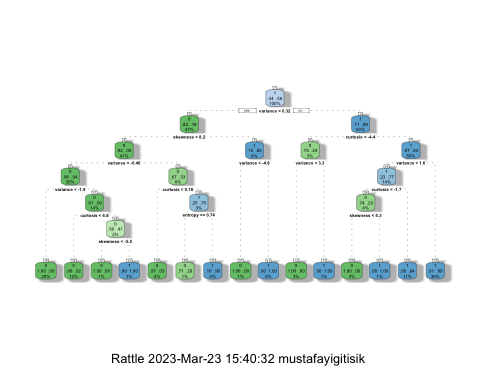
data$class <- as.factor(data$class)

#**Split Data**

set.seed(123)  
training.samples <- data$class %>%   
 createDataPartition(p = 0.8, list = FALSE)  
training <- data[training.samples, ]  
testing <- data[-training.samples, ]

#**Full Tree and its Confusion Matrix ~ Problem 2**

model <- rpart(class ~., data = training, control = rpart.control(cp=0))  
par(xpd = NA)  
fancyRpartPlot(model)



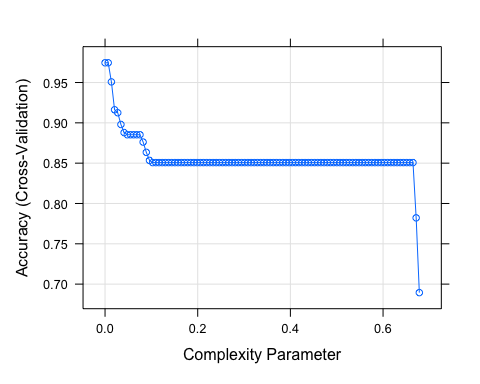
pred <- predict(model,newdata = testing, type ='class')  
pred <- ifelse(pred == 1, 'real', 'forged')  
table(pred,testing$class)

##   
## pred 0 1  
## forged 115 5  
## real 7 147

#**Prune the Data ~ Problem 3**

To make tree more robust, weprune the full tree using the training data with 10-fold cross-validation. Please (1) show the complexity plot, (2) report the best CP value, and (3) draw the pruned tree using rattle.

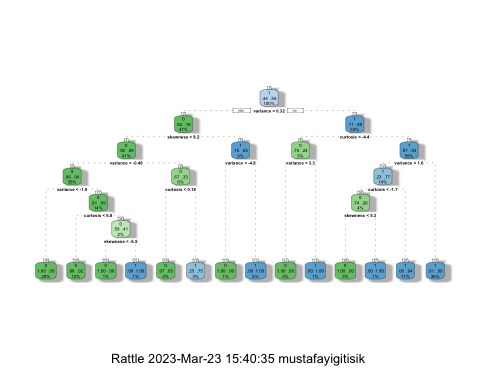
set.seed(123)  
model2 <- train(  
 class ~., data = training, method = "rpart",  
 trControl = trainControl("cv", number = 10),  
 tuneLength = 100)  
plot(model2)



model2$bestTune

## cp  
## 2 0.0068513

fancyRpartPlot(model2$finalModel)



#**Check testing data via pruned version ~ Problem 4**

Using optimal pruned tree to predict whether each check is forged or not in the testing data.

Compute the confusion matrix and report the sensitivity, specificity and the overall accuracy for the testing data.

pred <- predict(model2, newdata = testing)  
pred <- ifelse(pred == 1, 'real' , 'forged')  
matrix <- table(pred,testing$class)  
matrix

##   
## pred 0 1  
## forged 112 5  
## real 10 147

Sensitivity

matrix[2,2]/(matrix[1,2] + matrix[2,2])

## [1] 0.9671053

Specificity

matrix[1,1]/(matrix[1,1] + matrix[2,1])

## [1] 0.9180328

Accuracy

(matrix[1,1]+matrix[2,2])/(sum(matrix))

## [1] 0.9452555

#**Regression Model ~ Problem 5**

data$class <- as.factor(data$class)  
model3 <- glm(class ~ . , data = training, family = binomial)

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

summary(model3)

##   
## Call:  
## glm(formula = class ~ ., family = binomial, data = training)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.49913 -0.00005 0.00000 0.00000 1.44236   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -8.5895 2.1863 -3.929 8.54e-05 \*\*\*  
## variance 9.3611 2.4703 3.789 0.000151 \*\*\*  
## skewness 4.6968 1.2673 3.706 0.000210 \*\*\*  
## curtosis 6.1372 1.6414 3.739 0.000185 \*\*\*  
## entropy 0.5193 0.4208 1.234 0.217156   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1508.568 on 1097 degrees of freedom  
## Residual deviance: 33.991 on 1093 degrees of freedom  
## AIC: 43.991  
##   
## Number of Fisher Scoring iterations: 13

#Confusion Matrix based on Reg

predict\_training<-predict(model3, training, type= 'response' )  
predict\_testing<-predict(model3, testing, type= 'response' )  
confusion\_matrix\_training <- table(actual = training$class, prediction = predict\_training > 0.8)  
confusion\_matrix\_testing <- table(actual = testing$class, prediction = predict\_testing > 0.8)  
confusion\_matrix\_training

## prediction  
## actual FALSE TRUE  
## 0 487 1  
## 1 10 600

#Accuracy

accuracy\_training<-sum(diag(confusion\_matrix\_training))/sum(confusion\_matrix\_training)  
accuracy\_training

## [1] 0.9899818

accuracy\_testing<-sum(diag(confusion\_matrix\_testing))/sum(confusion\_matrix\_testing)  
accuracy\_testing

## [1] 0.9854015

#Sensitivity

sensitivity\_training <-confusion\_matrix\_training[2,2]/(confusion\_matrix\_training[2,2]+confusion\_matrix\_training[2,1])  
sensitivity\_training

## [1] 0.9836066

sensitivity\_testing <-confusion\_matrix\_testing[2,2]/(confusion\_matrix\_testing[2,2]+confusion\_matrix\_testing[2,1])  
sensitivity\_testing

## [1] 0.9802632

#Specificity

specificity\_training = confusion\_matrix\_training[1,1]/(confusion\_matrix\_training[1,1]+confusion\_matrix\_training[1,2])  
specificity\_training

## [1] 0.9979508

specificity\_testing = confusion\_matrix\_testing[1,1]/(confusion\_matrix\_testing[1,1]+confusion\_matrix\_testing[1,2])  
specificity\_testing

## [1] 0.9918033

#Problem 6

**From majority voting classifier, we import our findings [115, 5, 10, 147] here to test for sensitivity, accuracy, and specificity**

majority <- c(115, 5, 10, 147)  
majority\_matrix <- matrix(majority, nrow=2)  
majority\_matrix

## [,1] [,2]  
## [1,] 115 10  
## [2,] 5 147

Sensitivity

majority\_matrix[2,2]/(majority\_matrix[1,2] + majority\_matrix[2,2])

## [1] 0.9363057

Specificity

majority\_matrix[1,1]/(majority\_matrix[1,1] + majority\_matrix[2,1])

## [1] 0.9583333

Accuracy

(majority\_matrix[1,1]+majority\_matrix[2,2])/sum(majority\_matrix)

## [1] 0.9458484