ams580\_quiz3

Mustafa Yigit Isik

2023-02-22

#**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

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

## Loading required namespace: xlsx

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.

library(xlsx)

#**Read the data**

data <- read.csv("/Users/mustafayigitisik/Desktop/stuff/semesters/spring 2023/ams 580/quizzes/quiz 3/GreatUnknown.csv")  
  
#removes potential rows where at least one cell is empty   
data <- subset(data, !is.na(w1) | !is.na(w2) | !is.na(w3) | !is.na(w4) | !is.na(w5) | !is.na(w6) | !is.na(w7) | !is.na(w8) | !is.na(w9) | !is.na(w10)  
| !is.na(w11)| !is.na(w12)| !is.na(y))  
cat('There are', nrow(data), 'rows left.')

## There are 4601 rows left.

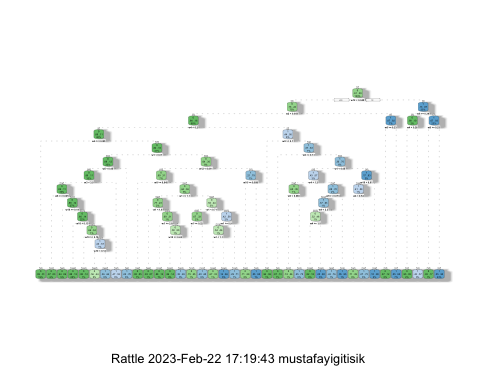
data$y <- as.factor(data$y)

#**Split Data**

set.seed(456)  
training.samples <- data$y %>%   
 createDataPartition(p = 0.75, list = FALSE)  
training <- data[training.samples, ]  
testing <- data[-training.samples, ]

#**Full Tree, its Confusion Matrix, and Sens/Spec/Accuracy ~ Problem 2**

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



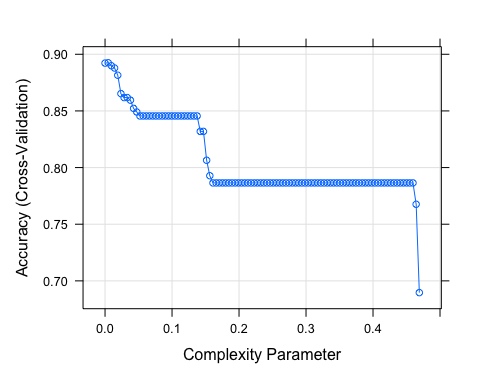
pred1 <- predict(model,newdata = testing, type ='class')  
pred1 <- ifelse(pred1 == 1, '1', '0')  
confusionMatrix(factor(pred1), factor(testing$y), positive = '1')

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 653 63  
## 1 44 390  
##   
## Accuracy : 0.907   
## 95% CI : (0.8887, 0.9231)  
## No Information Rate : 0.6061   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.8037   
##   
## Mcnemar's Test P-Value : 0.08184   
##   
## Sensitivity : 0.8609   
## Specificity : 0.9369   
## Pos Pred Value : 0.8986   
## Neg Pred Value : 0.9120   
## Prevalence : 0.3939   
## Detection Rate : 0.3391   
## Detection Prevalence : 0.3774   
## Balanced Accuracy : 0.8989   
##   
## 'Positive' Class : 1   
##

#**Prune the Data ~ Problem 3**

To make tree more robust, we prune 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.

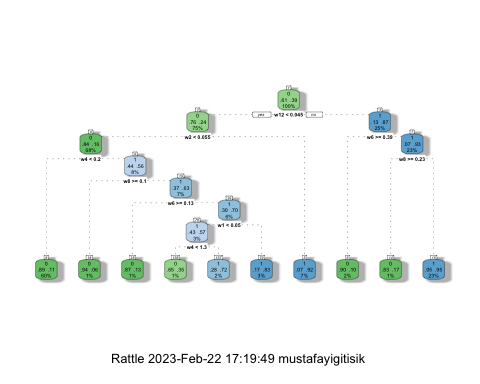
# set.seed(456)  
model2 <- train(  
 y ~., data = training, method = "rpart",  
 trControl = trainControl("cv", number = 10),  
 tuneLength = 100)  
plot(model2)



model2$bestTune

## cp  
## 2 0.004738562

fancyRpartPlot(model2$finalModel)



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

Using optimal pruned tree to predict whether each check is 0 or 1 in the testing data.

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

pred2 <- predict(model2, newdata = testing)  
pred2 <- ifelse(pred2 == 1, '1' , '0')  
confusionMatrix(factor(pred2), factor(testing$y), positive = '1')

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 662 72  
## 1 35 381  
##   
## Accuracy : 0.907   
## 95% CI : (0.8887, 0.9231)  
## No Information Rate : 0.6061   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8023   
##   
## Mcnemar's Test P-Value : 0.0005009   
##   
## Sensitivity : 0.8411   
## Specificity : 0.9498   
## Pos Pred Value : 0.9159   
## Neg Pred Value : 0.9019   
## Prevalence : 0.3939   
## Detection Rate : 0.3313   
## Detection Prevalence : 0.3617   
## Balanced Accuracy : 0.8954   
##   
## 'Positive' Class : 1   
##

#**Regression Model ~ Problem 5**

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

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

summary(model3)

##   
## Call:  
## glm(formula = y ~ ., family = binomial, data = training)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -5.3903 -0.7292 -0.0559 0.3173 4.6410   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.12745 0.07279 -15.490 < 2e-16 \*\*\*  
## w1 0.46264 0.07545 6.131 8.71e-10 \*\*\*  
## w2 3.85286 0.41750 9.228 < 2e-16 \*\*\*  
## w3 0.20408 0.07341 2.780 0.00544 \*\*   
## w4 0.78275 0.10200 7.674 1.66e-14 \*\*\*  
## w5 2.73875 0.56608 4.838 1.31e-06 \*\*\*  
## w6 -2.69331 0.29528 -9.121 < 2e-16 \*\*\*  
## w7 0.08907 0.23452 0.380 0.70409   
## w8 -2.24841 0.37651 -5.972 2.35e-09 \*\*\*  
## w9 -0.56056 0.33297 -1.683 0.09228 .   
## w10 -0.48270 0.26824 -1.800 0.07194 .   
## w11 -1.47321 0.94990 -1.551 0.12092   
## w12 10.77729 0.73890 14.586 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 4628.1 on 3450 degrees of freedom  
## Residual deviance: 2374.4 on 3438 degrees of freedom  
## AIC: 2400.4  
##   
## Number of Fisher Scoring iterations: 8

#Confusion Matrix based on Reg

probabilities <- model3 %>% predict(testing, type = "response")  
pred3 <- ifelse(probabilities > 0.5, 1, 0)  
confusionMatrix(factor(pred3), factor(testing$y), positive = '1')

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 666 101  
## 1 31 352  
##   
## Accuracy : 0.8852   
## 95% CI : (0.8654, 0.9031)  
## No Information Rate : 0.6061   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7529   
##   
## Mcnemar's Test P-Value : 1.905e-09   
##   
## Sensitivity : 0.7770   
## Specificity : 0.9555   
## Pos Pred Value : 0.9191   
## Neg Pred Value : 0.8683   
## Prevalence : 0.3939   
## Detection Rate : 0.3061   
## Detection Prevalence : 0.3330   
## Balanced Accuracy : 0.8663   
##   
## 'Positive' Class : 1   
##

#**Problem 6**

update.test = cbind(testing, pred1, pred2, pred3)  
write.xlsx(update.test, 'testingquiz3.xlsx', sheetName = 'testingquiz3')  
  
pred = cbind(pred1, pred2, pred3)  
pred.m = apply(pred,1,function(x) names(which.max(table(x))))   
pred.m = as.numeric(pred.m)  
confusionMatrix(factor(pred.m), factor(testing$y), positive = '1')

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 666 73  
## 1 31 380  
##   
## Accuracy : 0.9096   
## 95% CI : (0.8915, 0.9255)  
## No Information Rate : 0.6061   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8075   
##   
## Mcnemar's Test P-Value : 5.81e-05   
##   
## Sensitivity : 0.8389   
## Specificity : 0.9555   
## Pos Pred Value : 0.9246   
## Neg Pred Value : 0.9012   
## Prevalence : 0.3939   
## Detection Rate : 0.3304   
## Detection Prevalence : 0.3574   
## Balanced Accuracy : 0.8972   
##   
## 'Positive' Class : 1   
##