Assignment_3

Sanket Praveen Patil

2023-10-22

PROBLEM 1

Loading required libraries

```
library(rpart)
library(rpart.plot)
library(tree)
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice
```

Importing data in R

```
df = read.csv("breast_cancer_updated.csv", header = T)
dim(df)

## [1] 699  11

head(df)

##   IDNumber ClumpThickness UniformCellSize UniformCellShape MarginalAdhesion
## 1  1000025              5               1                1                1
## 2  1002945              5               4                4                5
## 3  1015425              3               1                1                1
## 4  1016277              6               8                8                1
## 5  1017023              4               1                1                3
## 6  1017122              8              10               10                8
##   EpithelialCellSize BareNuclei BlandChromatin NormalNucleoli Mitoses     Class
## 1                  2          1              3              1       1    benign
## 2                  7         10              3              2       1    benign
## 3                  2          2              3              1       1    benign
## 4                  3          4              3              7       1    benign
## 5                  2          1              3              1       1    benign
## 6                  7         10              9              7       1 malignant
```

Removing IDNumber column from data

```
df <- df[, !names(df) %in% "IDNumber"]
head(df)
```

```
##   ClumpThickness UniformCellSize UniformCellShape MarginalAdhesion
## 1              5               1                1                1
## 2              5               4                4                5
## 3              3               1                1                1
## 4              6               8                8                1
## 5              4               1                1                3
## 6              8              10               10                8
##   EpithelialCellSize BareNuclei BlandChromatin NormalNucleoli Mitoses     Class
## 1                  2          1              3              1       1    benign
## 2                  7         10              3              2       1    benign
## 3                  2          2              3              1       1    benign
## 4                  3          4              3              7       1    benign
## 5                  2          1              3              1       1    benign
## 6                  7         10              9              7       1 malignant
```

Question 1:

Removing NA values from data

```
colMeans(is.na(df)) * 100

##     ClumpThickness    UniformCellSize   UniformCellShape   MarginalAdhesion 
##           0.000000           0.000000           0.000000           0.000000 
## EpithelialCellSize         BareNuclei     BlandChromatin     NormalNucleoli 
##           0.000000           2.288984           0.000000           0.000000 
##            Mitoses              Class 
##           0.000000           0.000000

df <- na.omit(df)
```

Fit decision tree model

```
cancer_model <- rpart(Class ~ ., data = df)
```

Applying decision tree learning using 10-fold cross-validation

```
set.seed(123)
ctrl <- trainControl(method = "cv", number = 10, savePredictions = TRUE)
model <- train(Class ~ ., data = df, method = "rpart", trControl = ctrl)
```

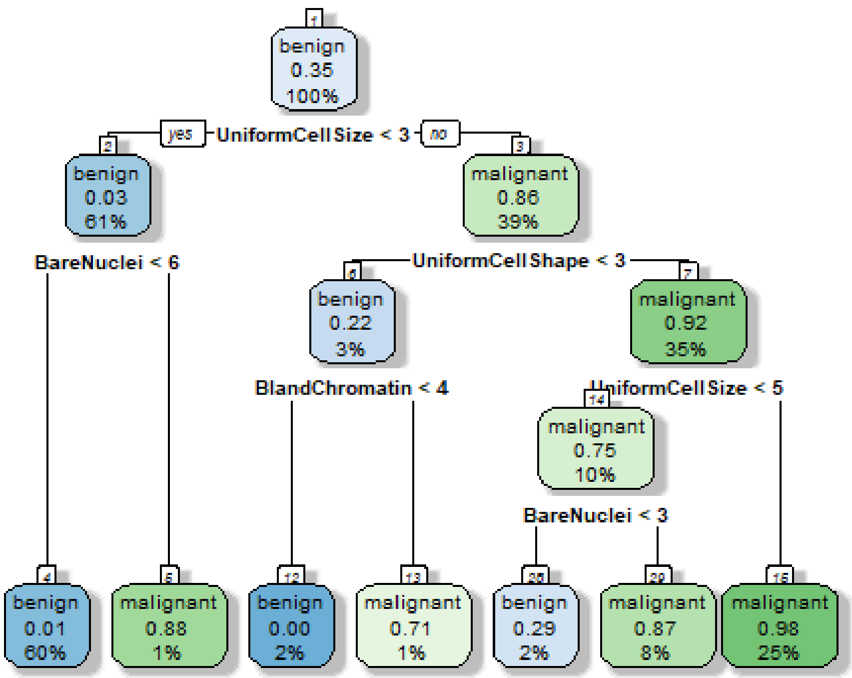
Report accuracy

```
model$results$Accuracy

## [1] 0.9415388 0.9283461 0.8567136
```

Question 2 :

Generating a visualization of the decision tree

```r
rpart.plot(cancer_model, shadow.col = "gray", nn = TRUE)
```



Question3 :

```r
rules <- rpart.rules(cancer_model)
print(rules)
```

- ```
  ##  Class
  ##   0.00 when UniformCellSize >=      3 & UniformCellShape <  3                  & BlandChromatin <  4
  ##   0.01 when UniformCellSize <  3                              & BareNuclei <  6
  ##   0.29 when UniformCellSize is 3 to 5 & UniformCellShape >= 3 & BareNuclei <  3
  ##   0.71 when UniformCellSize >=      3 & UniformCellShape <  3                  & BlandChromatin >= 4
  ##   0.87 when UniformCellSize is 3 to 5 & UniformCellShape >= 3 & BareNuclei >= 3
  ##   0.88 when UniformCellSize <  3                              & BareNuclei >= 6
  ##   0.98 when UniformCellSize >=      5 & UniformCellShape >= 3
  ```
  ```

- Rule 1 - if UniformCellSize >= 3 and UniformCellShape < 3 and BlandChromatin < 4 then Class = 0.00
- Rule 2 -if UniformCellSize < 3 and BareNuclei < 6 then Class = 0.01
- Rule 3 - if UniformCellSize is between 3 and 5 and UniformCellShape >= 3 and BareNuclei < 3 then Class = 0.29
- Rule 4 - if UniformCellSize >= 3 and UniformCellShape < 3 and BlandChromatin >= 4 then Class = 0.71
- Rule 5 - if UniformCellSize is between 3 and 5 and UniformCellShape >= 3 and BareNuclei >= 3 then Class = 0.87
- Rule 6 - if UniformCellSize < 3 and BareNuclei >= 6 then Class = 0.88
- Rule 7 - if UniformCellSize >= 5 and UniformCellShape >= 3 then Class = 0.98

PROBLEM 2 :

Load libraries

```
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(rpart)
library(caret)
```

Load the storms data

```
data(storms, package = "dplyr")
```

View the data

```
str(storms)

## tibble [19,066 × 13] (S3: tbl_df/tbl/data.frame)
##  $ name                        : chr [1:19066] "Amy" "Amy" "Amy" "Amy" ...
##  $ year                        : num [1:19066] 1975 1975 1975 1975 1975 ...
##  $ month                       : num [1:19066] 6 6 6 6 6 6 6 6 6 6 ...
##  $ day                         : int [1:19066] 27 27 27 27 28 28 28 28 29 29 ...
##  $ hour                        : num [1:19066] 0 6 12 18 0 6 12 18 0 6 ...
##  $ lat                         : num [1:19066] 27.5 28.5 29.5 30.5 31.5 32.4 33.3 34 34.4 34 ...
##  $ long                        : num [1:19066] -79 -79 -79 -79 -78.8 -78.7 -78 -77 -75.8 -74.8 ...
```

```
##  $ status                      : Factor w/ 9 levels "disturbance",..: 7 7 7 7 7 7 7 7 8 8 ...
##  $ category                    : num [1:19066] NA NA NA NA NA NA NA NA NA NA ...
##  $ wind                        : int [1:19066] 25 25 25 25 25 25 25 30 35 40 ...
##  $ pressure                    : int [1:19066] 1013 1013 1013 1013 1012 1012 1011 1006 1004 1002 ...
##  $ tropicalstorm_force_diameter: int [1:19066] NA NA NA NA NA NA NA NA NA NA ...
##  $ hurricane_force_diameter    : int [1:19066] NA NA NA NA NA NA NA NA NA NA ...
```

```
dim(storms)
```

```
## [1] 19066    13
```

Convert the target variable (category) to a factor

```
storms$category <- as.factor(storms$category)
```

Removing NA values from data

```
colMeans(is.na(storms)) * 100
```

```
##                         name                         year
##                      0.00000                      0.00000
##                        month                          day
##                      0.00000                      0.00000
##                         hour                          lat
##                      0.00000                      0.00000
##                         long                       status
##                      0.00000                      0.00000
##                     category                         wind
##                     75.43271                      0.00000
##                     pressure tropicalstorm_force_diameter
##                      0.00000                     49.88986
##     hurricane_force_diameter
##                     49.88986
```

```
storms <- na.omit(storms)
```

Checking the unique values and class

```
sapply(storms, function(x) length(unique(x)))
```

```
##                         name                         year
##                          105                           18
##                        month                          day
##                            8                           31
##                         hour                          lat
##                           24                          338
##                         long                       status
##                          647                            1
##                     category                         wind
##                            5                           20
##                     pressure tropicalstorm_force_diameter
```

```
##                        98                         110
##     hurricane_force_diameter
##                        38
```

```
sapply(storms, function(x) class(x))
```

```
##                       name                       year
##                "character"                  "numeric"
##                      month                        day
##                  "numeric"                  "integer"
##                       hour                        lat
##                  "numeric"                  "numeric"
##                       long                     status
##                  "numeric"                   "factor"
##                   category                       wind
##                   "factor"                  "integer"
##                   pressure tropicalstorm_force_diameter
##                  "integer"                  "integer"
##     hurricane_force_diameter
##                  "integer"
```

Removing name variable from data as it will take too much time to train decision tree

```
storms <- storms[, !names(storms) %in% "name"]
head(storms)
```

```
## # A tibble: 6 × 12
##    year month   day  hour   lat  long status    category  wind pressure
##   <dbl> <dbl> <int> <dbl> <dbl> <dbl> <fct>     <fct>    <int>    <int>
## 1  2004     8     3     6  33   -77.4 hurricane 1           70      983
## 2  2004     8     3    12  34.2 -76.4 hurricane 2           85      974
## 3  2004     8     3    18  35.3 -75.2 hurricane 2           85      972
## 4  2004     8     4     0  36   -73.7 hurricane 1           80      974
## 5  2004     8     4     6  36.8 -72.1 hurricane 1           80      973
## 6  2004     8     4    12  37.3 -70.2 hurricane 2           85      973
## # ℹ 2 more variables: tropicalstorm_force_diameter <int>,
## #   hurricane_force_diameter <int>
```

Question 1:

Training the decision tree model with specified hyperparameters

```
set.seed(123)
cv_model <- train(category ~ ., data = storms,
                  method = "rpart",
                  trControl = trainControl(method = "cv", number = 5),
                  control = rpart.control(maxdepth = 2, minsplit = 5, minbucket = 3))
```

Printing the Accuracy of the model

```
print(cv_model$results$Accuracy)

## [1] 0.8337404 0.7503258 0.5339054
```

Question 2 :

Creating a train/test partition

```
set.seed(789)
splitIndex <- createDataPartition(storms$category, p = 0.8, list = FALSE)
train_storms <- storms[splitIndex, ]
test_storms <- storms[-splitIndex, ]
```

Generating decision tree

```
tree_model_train <- rpart(category ~ ., data = train_storms, method = "class", minsplit = 5, maxdepth =
2, minbucket = 3)
```

Predicting the category of both train and test data

```
predictions_storms_train <- predict(tree_model_train, newdata = train_storms,type = "class")
predictions_storms_test <- predict(tree_model_train, newdata = test_storms, type = "class")
```

Confusion matrix to evaluate accuracy

```
conf_matrix_storms_train <- confusionMatrix(predictions_storms_train, train_storms$category)
conf_matrix_storms_test <- confusionMatrix(predictions_storms_test, test_storms$category)
```

Confusion matrix table

```
conf_matrix_train<-table(predictions_storms_train, train_storms$category)
conf_matrix_test<-table(predictions_storms_test, test_storms$category)
print(conf_matrix_train)

##
## predictions_storms_train   1   2   3   4   5
##                        1 811   0   0   0   0
##                        2   0 332   0   0   0
##                        3   0   0   0   0   0
##                        4   0   0 222 227  52
##                        5   0   0   0   0   0

print(conf_matrix_test)

##
## predictions_storms_test   1   2   3   4   5
##                       1 202   0   0   0   0
```

```
##                    2   0  82   0   0   0
##                    3   0   0   0   0   0
##                    4   0   0  55  56  12
##                    5   0   0   0   0   0

accuracy_storms_train <- conf_matrix_storms_train$overall["Accuracy"]
accuracy_storms_test <- conf_matrix_storms_test$overall["Accuracy"]
```

Print the accuracy

```
print(paste("Accuracy_train:", round(accuracy_storms_train,4)))

## [1] "Accuracy_train: 0.8333"

print(paste("Accuracy_test:", round(accuracy_storms_test,4)))

## [1] "Accuracy_test: 0.8354"
```

- The model performs well to predict class 1 and 2 in both training and testing data which can be proved by iths high diagonal count.
- Whereas, to classify class 3, the model fails. It might tells us that there is lack of representative samples for class 3.
- In both training and testing data, classes 4 and 5 show some misclassifications.
- Also model is working similar on both the data sets i.e. train data and test data.
- As model is performing similar on test and train data, it means that the model has generalized well to new, unseen data.
- Model is maintaining similar performance on training and testing data.
- In conclusion, model is not overfitting the training data.

PROBLEM 3 :

```
library(rpart)
library(ggplot2)
```

Splitting data into 80% 20% split

```
set.seed(678)
splitIndex_3 <- createDataPartition(storms$category, p = 0.8, list = FALSE)
train_data_3 <- storms[splitIndex, ]
test_data_3 <- storms[-splitIndex, ]
```

Tree 1

```
storms_tree_1 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 5, maxdepth = 2,
minbucket = 3)

predictions_train_tree_1 <- predict(storms_tree_1, newdata = train_data_3, type = "class")
predictions_test_tree_1 <- predict(storms_tree_1, newdata = test_data_3, type = "class")
```

```
conf_matrix_train_tree_1 <- confusionMatrix(predictions_train_tree_1, train_data_3$category)
conf_matrix_test_tree_1 <- confusionMatrix(predictions_test_tree_1, test_data_3$category)

accuracy_train_tree_1 <- conf_matrix_train_tree_1$overall["Accuracy"]
accuracy_test_tree_1 <- conf_matrix_test_tree_1$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_1<-sum(storms_tree_1$frame$var == "<leaf>")
```

Creating a dataframe to store the model parameters and accuracy of the tree

```
comp_tbl <- data.frame("Nodes" = nodes_1, "TrainAccuracy" = accuracy_train_tree_1, "TestAccuracy"
= accuracy_test_tree_1,"Minsplit" = 5, "Maxdepth" = 2, "Minbucket" = 3)
```

Tree 2

```
storms_tree_2 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 10, maxdepth = 2,
minbucket = 6)

predictions_train_tree_2 <- predict(storms_tree_2, newdata = train_data_3, type = "class")
predictions_test_tree_2 <- predict(storms_tree_2, newdata = test_data_3, type = "class")

conf_matrix_train_tree_2 <- confusionMatrix(predictions_train_tree_2, train_data_3$category)
conf_matrix_test_tree_2 <- confusionMatrix(predictions_test_tree_2, test_data_3$category)

accuracy_train_tree_2 <- conf_matrix_train_tree_2$overall["Accuracy"]
accuracy_test_tree_2 <- conf_matrix_test_tree_2$overall["Accuracy"]

nodes_2<-sum(storms_tree_2$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_2, accuracy_train_tree_2, accuracy_test_tree_2, 10, 2, 6))
```

Tree 3

```
storms_tree_3 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 15, maxdepth = 2,
minbucket = 9)

predictions_train_tree_3 <- predict(storms_tree_3, newdata = train_data_3, type = "class")
predictions_test_tree_3 <- predict(storms_tree_3, newdata = test_data_3, type = "class")

conf_matrix_train_tree_3 <- confusionMatrix(predictions_train_tree_3, train_data_3$category)
conf_matrix_test_tree_3 <- confusionMatrix(predictions_test_tree_3, test_data_3$category)

accuracy_train_tree_3 <- conf_matrix_train_tree_3$overall["Accuracy"]
accuracy_test_tree_3 <- conf_matrix_test_tree_3$overall["Accuracy"]
```

```
nodes_3<-sum(storms_tree_3$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_3, accuracy_train_tree_3, accuracy_test_tree_3, 15, 2, 9))
```

Tree 4

```
storms_tree_4 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 5, maxdepth = 3,
minbucket = 3)

predictions_train_tree_4 <- predict(storms_tree_4, newdata = train_data_3, type = "class")
predictions_test_tree_4 <- predict(storms_tree_4, newdata = test_data_3, type = "class")

conf_matrix_train_tree_4 <- confusionMatrix(predictions_train_tree_4, train_data_3$category)
conf_matrix_test_tree_4 <- confusionMatrix(predictions_test_tree_4, test_data_3$category)

accuracy_train_tree_4 <- conf_matrix_train_tree_4$overall["Accuracy"]
accuracy_test_tree_4 <- conf_matrix_test_tree_4$overall["Accuracy"]

nodes_4<-sum(storms_tree_4$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_4, accuracy_train_tree_4, accuracy_test_tree_4, 5, 3, 3))
```

Tree 5

```
storms_tree_5 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 10, maxdepth = 3,
minbucket = 6)

predictions_train_tree_5 <- predict(storms_tree_5, newdata = train_data_3, type = "class")
predictions_test_tree_5 <- predict(storms_tree_5, newdata = test_data_3, type = "class")

conf_matrix_train_tree_5 <- confusionMatrix(predictions_train_tree_5, train_data_3$category)
conf_matrix_test_tree_5 <- confusionMatrix(predictions_test_tree_5, test_data_3$category)

accuracy_train_tree_5 <- conf_matrix_train_tree_5$overall["Accuracy"]
accuracy_test_tree_5 <- conf_matrix_test_tree_5$overall["Accuracy"]

nodes_5<-sum(storms_tree_5$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_5, accuracy_train_tree_5, accuracy_test_tree_5, 10, 3, 6))
```

Tree 6

```
storms_tree_6 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 15, maxdepth = 3,
minbucket = 9)

predictions_train_tree_6 <- predict(storms_tree_6, newdata = train_data_3, type = "class")
predictions_test_tree_6 <- predict(storms_tree_6, newdata = test_data_3, type = "class")
```

```
conf_matrix_train_tree_6 <- confusionMatrix(predictions_train_tree_6, train_data_3$category)
conf_matrix_test_tree_6 <- confusionMatrix(predictions_test_tree_6, test_data_3$category)

accuracy_train_tree_6 <- conf_matrix_train_tree_6$overall["Accuracy"]
accuracy_test_tree_6 <- conf_matrix_test_tree_6$overall["Accuracy"]

nodes_6<-sum(storms_tree_6$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_6, accuracy_train_tree_6, accuracy_test_tree_6, 15, 3, 9))
```

Tree 7

```
storms_tree_7 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 30, maxdepth = 3,
minbucket = 20)

predictions_train_tree_7 <- predict(storms_tree_7, newdata = train_data_3, type = "class")
predictions_test_tree_7 <- predict(storms_tree_7, newdata = test_data_3, type = "class")

conf_matrix_train_tree_7 <- confusionMatrix(predictions_train_tree_7, train_data_3$category)
conf_matrix_test_tree_7 <- confusionMatrix(predictions_test_tree_7, test_data_3$category)

accuracy_train_tree_7 <- conf_matrix_train_tree_7$overall["Accuracy"]
accuracy_test_tree_7 <- conf_matrix_test_tree_7$overall["Accuracy"]

nodes_7<-sum(storms_tree_7$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_7, accuracy_train_tree_7, accuracy_test_tree_7, 30, 3,
20))
```

Tree 8

```
storms_tree_8 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 40, maxdepth = 10,
minbucket = 30)

predictions_train_tree_8 <- predict(storms_tree_8, newdata = train_data_3, type = "class")
predictions_test_tree_8 <- predict(storms_tree_8, newdata = test_data_3, type = "class")

conf_matrix_train_tree_8 <- confusionMatrix(predictions_train_tree_8, train_data_3$category)
conf_matrix_test_tree_8 <- confusionMatrix(predictions_test_tree_8, test_data_3$category)

accuracy_train_tree_8 <- conf_matrix_train_tree_8$overall["Accuracy"]
accuracy_test_tree_8 <- conf_matrix_test_tree_8$overall["Accuracy"]

nodes_8<-sum(storms_tree_8$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_8, accuracy_train_tree_8, accuracy_test_tree_8, 40, 10,
30))
```

Tree 9

```
storms_tree_9 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 60, maxdepth = 20,
minbucket = 40)

predictions_train_tree_9 <- predict(storms_tree_9, newdata = train_data_3, type = "class")
predictions_test_tree_9 <- predict(storms_tree_9, newdata = test_data_3, type = "class")

conf_matrix_train_tree_9 <- confusionMatrix(predictions_train_tree_9, train_data_3$category)
conf_matrix_test_tree_9 <- confusionMatrix(predictions_test_tree_9, test_data_3$category)

accuracy_train_tree_9 <- conf_matrix_train_tree_9$overall["Accuracy"]
accuracy_test_tree_9 <- conf_matrix_test_tree_9$overall["Accuracy"]

nodes_9<-sum(storms_tree_9$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_9, accuracy_train_tree_9, accuracy_test_tree_9, 60, 20,
40))
```

Tree 10

```
storms_tree_10 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 200, maxdepth =
25, minbucket = 100)

predictions_train_tree_10 <- predict(storms_tree_10, newdata = train_data_3, type = "class")
predictions_test_tree_10 <- predict(storms_tree_10, newdata = test_data_3, type = "class")

conf_matrix_train_tree_10 <- confusionMatrix(predictions_train_tree_10, train_data_3$category)
conf_matrix_test_tree_10 <- confusionMatrix(predictions_test_tree_10, test_data_3$category)

accuracy_train_tree_10 <- conf_matrix_train_tree_10$overall["Accuracy"]
accuracy_test_tree_10 <- conf_matrix_test_tree_10$overall["Accuracy"]

nodes_10<-sum(storms_tree_10$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_10, accuracy_train_tree_10, accuracy_test_tree_10, 200,
25, 100))
```

Tree 11

```
storms_tree_11 <- rpart(category ~ ., data = train_data_3, method = "class", minsplit = 300, maxdepth =
25, minbucket = 200)

predictions_train_tree_11 <- predict(storms_tree_11, newdata = train_data_3, type = "class")
predictions_test_tree_11 <- predict(storms_tree_11, newdata = test_data_3, type = "class")

conf_matrix_train_tree_11 <- confusionMatrix(predictions_train_tree_11, train_data_3$category)
conf_matrix_test_tree_11 <- confusionMatrix(predictions_test_tree_11, test_data_3$category)
```

```r
accuracy_train_tree_11 <- conf_matrix_train_tree_11$overall["Accuracy"]
accuracy_test_tree_11 <- conf_matrix_test_tree_11$overall["Accuracy"]

nodes_11<-sum(storms_tree_11$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_11, accuracy_train_tree_11, accuracy_test_tree_11, 300,
25, 200))
```

Tree 12

```r
storms_tree_12 <- rpart(category ~ ., data = train_data_3, method = "class",
                        minsplit = 500, maxdepth = 25, minbucket = 500)

predictions_train_tree_12 <- predict(storms_tree_12, newdata = train_data_3, type = "class")
predictions_test_tree_12 <- predict(storms_tree_12, newdata = test_data_3, type = "class")

conf_matrix_train_tree_12 <- confusionMatrix(predictions_train_tree_12, train_data_3$category)
conf_matrix_test_tree_12 <- confusionMatrix(predictions_test_tree_12, test_data_3$category)

accuracy_train_tree_12 <- conf_matrix_train_tree_12$overall["Accuracy"]
accuracy_test_tree_12 <- conf_matrix_test_tree_12$overall["Accuracy"]

nodes_12<-sum(storms_tree_12$frame$var == "<leaf>")

comp_tbl <- comp_tbl %>% rbind(list(nodes_12, accuracy_train_tree_12, accuracy_test_tree_12, 500,
25, 500))
```

Final table

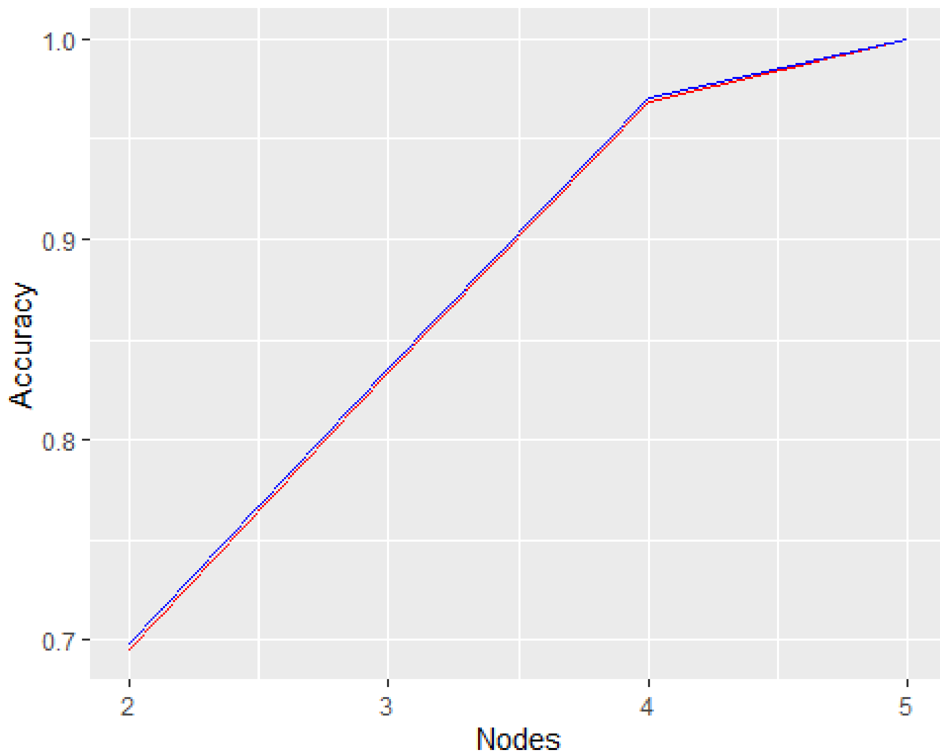
```r
print(comp_tbl)
```

```
##          Nodes TrainAccuracy TestAccuracy Minsplit Maxdepth Minbucket
## Accuracy     3     0.8333333    0.8353808        5        2         3
## 1            3     0.8333333    0.8353808       10        2         6
## 11           3     0.8333333    0.8353808       15        2         9
## 12           4     0.9683698    0.9705160        5        3         3
## 13           4     0.9683698    0.9705160       10        3         6
## 14           4     0.9683698    0.9705160       15        3         9
## 15           4     0.9683698    0.9705160       30        3        20
## 16           5     1.0000000    1.0000000       40       10        30
## 17           5     1.0000000    1.0000000       60       20        40
## 18           4     0.9683698    0.9705160      200       25       100
## 19           4     0.9683698    0.9705160      300       25       200
## 110          2     0.6952555    0.6977887      500       25       500
```

Observing the accuracies with line graph

```r
ggplot(comp_tbl, aes(x=Nodes)) +
  geom_line(aes(y = TrainAccuracy), color = "red") +
```

```
geom_line(aes(y = TestAccuracy), color="blue") +
ylab("Accuracy")
```



- **By Looking at the graph and table, we can conclude that at node = 5, tree performed best and gave highest accuracy. If we compare the line graph for train and test, there is not a single point where test accuracy decreased as compared to train accuracy. Hence we can say that there is no inflection point in generated models.**

Question 3 :

- We will choose the tree number 8 as our final tree.
- Parameters - Nodes = 5, Minsplit = 40, Maxdepth = 10 and Minbucket = 30

Confusion matrix

```
conf_matrix_train_tree_8_table <- table(predictions_train_tree_8, train_data_3$category)
conf_matrix_test_tree_8_table <- table(predictions_test_tree_8, test_data_3$category)
```

Printing the final results of matrix

```
print(conf_matrix_train_tree_8_table)
```

```
##
## predictions_train_tree_8   1   2   3   4   5
##                        1 811   0   0   0   0
##                        2   0 332   0   0   0
##                        3   0   0 222   0   0
##                        4   0   0   0 227   0
##                        5   0   0   0   0  52
```

```
print(conf_matrix_test_tree_8_table)
```

```
##
## predictions_test_tree_8   1   2   3   4   5
##                       1 202   0   0   0   0
##                       2   0  82   0   0   0
##                       3   0   0  55   0   0
##                       4   0   0   0  56   0
##                       5   0   0   0   0  12
```

- **With the help of above results, we can clearly conclude that our model is performing well enough to predict all the classes with 0 miss classifications.**

Using same parameters to train a model with 10 fold cross validation.

```
train_control = trainControl(method = "cv", number = 10)

hypers = rpart.control(minsplit =  40, maxdepth = 10, minbucket = 30)
tree8_cv <- train(category ~ ., data = train_data_3, control = hypers, trControl = train_control, method =
"rpart1SE")
```

Report accuracy

```
tree8_cv$results$Accuracy

## [1] 1
```

With cross validation also, we can see our model is providing accuracy = 1.

PROBLEM 4 :

Load necessary libraries

```
library(rpart)
```

Loading data

```
BankData <- read.csv("Bank_Modified.csv")
```

Removing the ID column

```
BankData <- BankData[, !names(BankData) %in% "X"]
head(BankData)
```

```
##   cont1 cont2 cont3 bool1 bool2 cont4 bool3 cont5 cont6 approval credit.score
## 1 30.83 0.000  1.25     t     t     1     f   202     0        +       664.60
## 2 58.67 4.460  3.04     t     t     6     f    43   560        +       693.88
## 3 24.50 0.500  1.50     t     f     0     f   280   824        +       621.82
## 4 27.83 1.540  3.75     t     t     5     t   100     3        +       653.97
## 5 20.17 5.625  1.71     t     f     0     f   120     0        +       670.26
## 6 32.08 4.000  2.50     t     f     0     t   360     0        +       672.16
##   ages
## 1   58
## 2   54
## 3   62
## 4   51
## 5   58
## 6   37
```

Converting the target variable ‘approval’ to a factor

```
BankData$approval <- as.factor(BankData$approval)
```

Question 1 :

Building the initial decision tree model with the help of hyperparameters

```
set.seed(123)
split_index <- createDataPartition(BankData$approval, p = 0.8, list = FALSE)
train_data_1 <- BankData[split_index, ]
test_data_1 <- BankData[-split_index, ]

BankData_model <- rpart(approval ~ ., data = train_data_1, method = "class", minsplit = 10, maxdepth =
20)

predictions_1 <- predict(BankData_model, newdata = test_data_1, type = "class")
```

Confusion matrix to evaluate accuracy

```
conf_matrix <- confusionMatrix(predictions_1, test_data_1$approval)
accuracy_1 <- conf_matrix$overall["Accuracy"]
```

Print the accuracy

```
print(paste("Accuracy:", round(accuracy_1,4)))

## [1] "Accuracy: 0.8832"
```

Number of leaf nodes

```
sum(BankData_model$frame$var == "<leaf>")

## [1] 7
```

Question 2 :

```
library(caret)
```

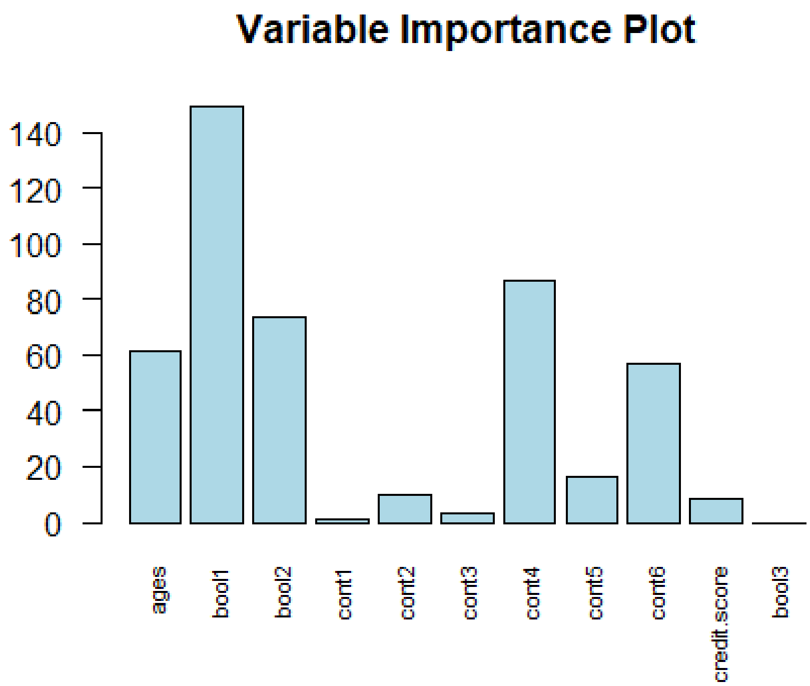
Running the variable importance analysis on the model

```
var_importance <- varImp(BankData_model)
print(var_importance)

##               Overall
## ages        61.475110
## bool1      149.164983
## bool2       73.772412
## cont1        1.046187
## cont2        9.777572
## cont3        2.962987
## cont4       86.936452
## cont5       16.387493
## cont6       57.328878
## credit.score  8.333000
## bool3        0.000000
```

Question 3 :

Plotting variable importance

```
barplot(var_importance$Overall, main = "Variable Importance Plot",
        col = "lightblue", cex.names = 0.7, las = 2, names.arg = rownames(var_importance))
```



Question 4 :

- By looking at the graph, we can see top 6 variables with high importance are bool1, cont4, bool2, ages, cont3 and cont6.

Rebuild the model with top six variables

```
BankData_model_new <- rpart(approval ~ bool1 + cont4 + bool2 + ages + cont5 + cont6, data =
train_data_1, method = "class", minsplit = 10, maxdepth = 20)

predictions <- predict(BankData_model_new, newdata = test_data_1, type = "class")
```

Confusion matrix to evaluate accuracy

```
conf_matrix <- confusionMatrix(predictions, test_data_1$approval)
accuracy <- conf_matrix$overall["Accuracy"]
```

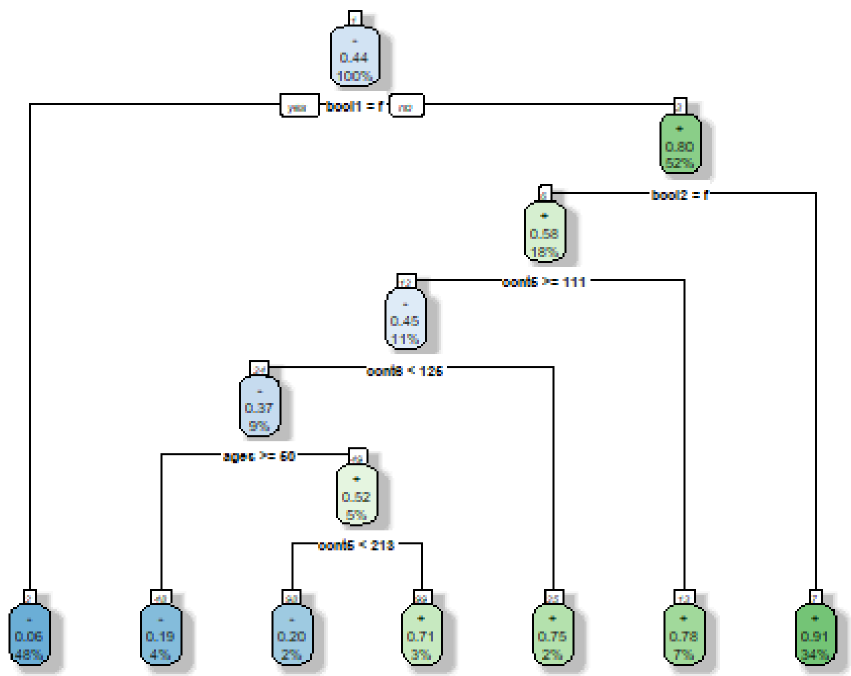
Print the accuracy

```
print(paste("Accuracy:", round(accuracy,4)))

## [1] "Accuracy: 0. 0.8832"

# There is no change in the accuracy of the model after rebuilding the model with 6 most important
variables.
```

Question 5:

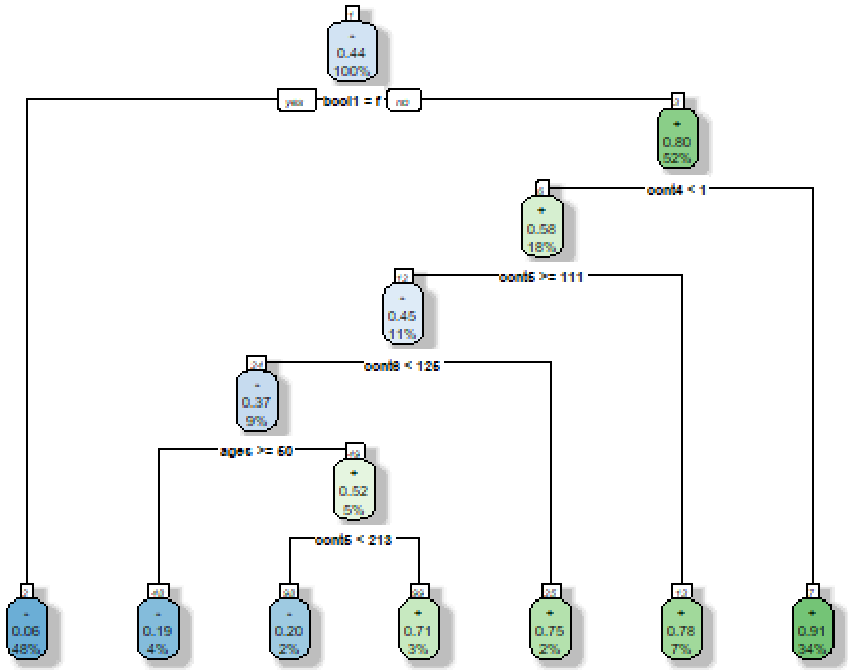
Visualize the initial tree

```r
library(rpart.plot)
rpart.plot(BankData_model, shadow.col = "gray", nn = TRUE)
```



Visualize the tree with top six variables

```r
rpart.plot(BankData_model_new, shadow.col = "gray", nn = TRUE)
```



- We checked the accuracy of the model after choosing 6 important variables. We can see that there is no change in the accuracy and also there is no change in the size of the decision tree as well.