FUNDAMENTALS OF DATA SCIENCE
ASSIGNMENT - 3

Mrunali Vikas Patil

PROBLEM SET 1

Importing data in R

```
Data <- read.csv("breast_cancer_updated.csv",header = TRUE)
dim(Data)

## [1] 699  11

head(Data)

##   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 the ID Number Column

```
Data$IDNumber <- NULL
```

Excluding rows with NA Values

```
Data <- na.omit(Data)
colMeans(is.na(Data))*100

##      ClumpThickness    UniformCellSize   UniformCellShape
MarginalAdhesion
##                   0                  0                  0
0
## EpithelialCellSize         BareNuclei     BlandChromatin
NormalNucleoli
##                   0                  0                  0
0
##             Mitoses              Class
##                   0                  0
```

Loading the required libraries

```
library(rpart)
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice
```

Question A)

Splitting the data (80% - 20%) into training and test sets

```
set.seed(123)
index <- createDataPartition(Data$Class, p = 0.8, list = FALSE)
train_data <- Data[index, ]
test_data <- Data[-index, ]
```

Decision Tree

```
tree_model <- rpart(Class ~ ., data = train_data, method = "class")
```

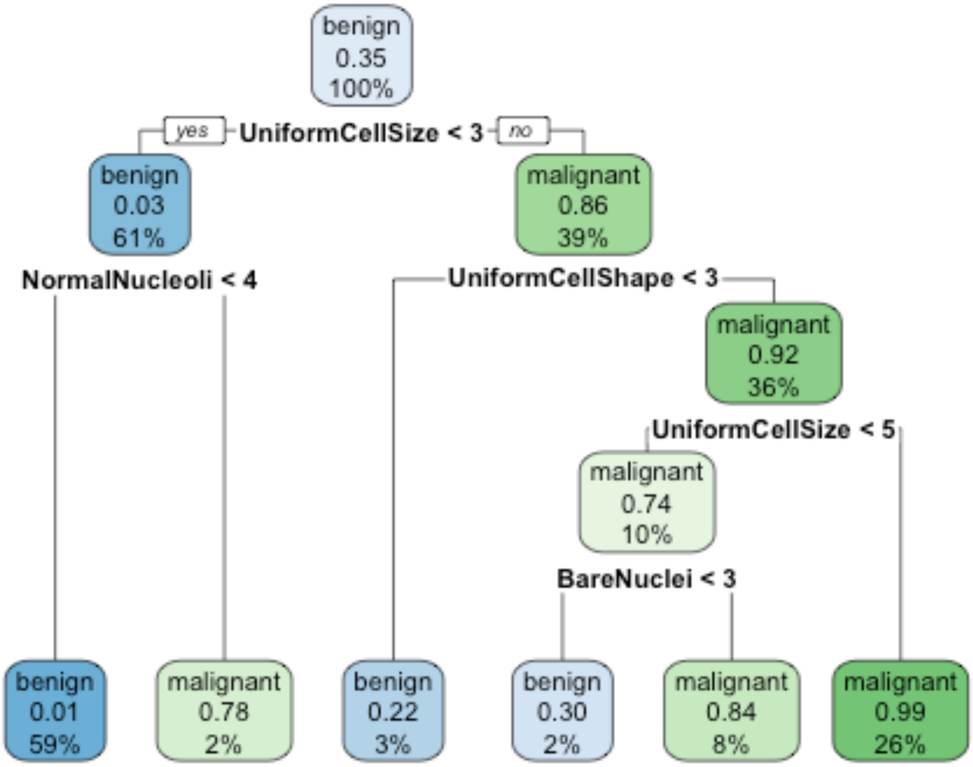
10-Fold cross Validation

```
set.seed(123)
folds <- createFolds(Data$Class, k = 10)
control <- trainControl(method = "cv", number = 10, index = folds)
model <- train(Class ~ ., data = Data, method = "rpart", trControl =
control)
```

Question B)

Visualization of the Decision Tree

```
library(rpart.plot)
rpart.plot(tree_model)
```



Question C)

Generating Full Set of Rules using IF-Then Statements

```
rules <- rpart.rules(tree_model)
print(rules)
```

```
##  Class
##   0.01 when UniformCellSize <  3
& NormalNucleoli <  4
##   0.22 when UniformCellSize >=      3 & UniformCellShape <  3
##   0.30 when UniformCellSize is 3 to 5 & UniformCellShape >= 3 &
BareNuclei <  3
##   0.78 when UniformCellSize <  3
& NormalNucleoli >= 4
##   0.84 when UniformCellSize is 3 to 5 & UniformCellShape >= 3 &
BareNuclei >= 3
##   0.99 when UniformCellSize >=      5 & UniformCellShape >= 3
```

PROBLEM SET 2

Load required 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 data

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

class(storms$category)

## [1] "numeric"
```

Ensure the ‘category’ variable is a factor

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

Removing name column from the data

```
storms <- subset(storms, select = -name)
```

Removing na values from the data

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

Question A)

Build decision tree using cross-validation

Setting up the train control with cross validation

```
control <- trainControl(method="cv", number=10)
```

Train the model

```
model <- train(category ~ ., data=storms, method="rpart",
               tuneGrid=data.frame(cp=0.01),
               trControl=control,
               control=rpart.control(maxdepth=2, minsplit=5,
minbucket=3))
```

Report the accuracy from cross-validation

```
print(model$results)

##     cp  Accuracy     Kappa  AccuracySD     KappaSD
## 1 0.01 0.8337431 0.7530836 0.002403905 0.003004565
```

Question B)

Create a train/test split

```
set.seed(123) # for reproducibility
index <- createDataPartition(storms$category, p=0.75, list=FALSE)
train_data <- storms[index, ]
test_data <- storms[-index, ]

storms_1 <- rpart(category ~ .,  data = storms, method = "class",
maxdepth=2, minsplit=5, minbucket=3)
```

Predictions on train and test sets

```
predictions_train <- predict(storms_1, train_data, type='class')
predictions_test <- predict(storms_1, test_data, type='class')
```

Generate confusion matrices

```
cm_train <- confusionMatrix(predictions_train, train_data$category)
cm_test <- confusionMatrix(predictions_test, test_data$category)

accuracy_train <- cm_train$overall["Accuracy"]
accuracy_test<- cm_train$overall["Accuracy"]

cat("Accuracy of train: ", accuracy_train, "\n")

## Accuracy of train:  0.8337662

cat("Accuracy of test: ", accuracy_test, "\n")

## Accuracy of test:  0.8337662

table(predictions_train, train_data$category)

##
## predictions_train   1   2   3   4   5
##                 1 760   0   0   0   0
##                 2   0 311   0   0   0
##                 3   0   0   0   0   0
##                 4   0   0 208 213  48
##                 5   0   0   0   0   0
```

```r
table(predictions_test, test_data$category)

##
## predictions_test   1   2   3   4   5
##                1 253   0   0   0   0
##                2   0 103   0   0   0
##                3   0   0   0   0   0
##                4   0   0  69  70  16
##                5   0   0   0   0   0
```

The model's performance is consistent between the training and test datasets. Since the mistakes made by the model in the training set are similar to those made in the test set, it suggests that the model is not overfitting. Instead, the model might have inherent biases or issues with distinguishing between certain classes (especially Class 3, 4, and 5).

The model has difficulty classifying Class 3, 4, and 5.

Specifically:
Class 3 is consistently misclassified as Class 4.
Some Class 4 examples are misclassified as Class 5.
All Class 5 instances are predicted as Class 4.

The consistency of these mistakes between training and test datasets suggests that the model is not overfitting but has inherent classification challenges with these classes.

PROBLEM SET 3

Load required libraries

```r
library(dplyr)
library(rpart)
```

Question A)

```r
set.seed(123)
splitIndex <- createDataPartition(storms$category, p = 0.8, list =
FALSE)
train_data <- storms[splitIndex, ]
test_data <- storms[-splitIndex, ]
```

Question B)

Tree 1

```
tree_1 <- rpart(category ~ ., data = train_data, method = "class",
minsplit = 5, maxdepth = 2, minbucket = 3)

predictions_tree_1_train <- predict(tree_1, newdata = train_data, type
= "class")
predictions_tree_1_test <- predict(tree_1, newdata = test_data, type =
"class")

confusion_matrix_tree_1_train <-
confusionMatrix(predictions_tree_1_train, train_data$category)
confusion_matrix_tree_1_test <-
confusionMatrix(predictions_tree_1_test, test_data$category)

accuracy_tree_1_train <-
confusion_matrix_tree_1_train$overall["Accuracy"]
accuracy_tree_1_test <-
confusion_matrix_tree_1_test$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_tree_1<-sum(tree_1$frame$var == "<leaf>")

parameters_table <- data.frame("Nodes" = nodes_tree_1, "TrainAccuracy"
= accuracy_tree_1_train, "TestAccuracy" = accuracy_tree_1_test,
                       "Minsplit" = 5, "Maxdepth" = 2, "Minbucket" =
3)
```

Tree 2

```
tree_2<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 10, maxdepth = 2, minbucket = 6)

predictions_tree_2_train <- predict(tree_2, newdata = train_data, type
= "class")
predictions_tree_2_test <- predict(tree_2, newdata = test_data, type =
"class")
```

```r
confusion_matrix_tree_2_train <-
confusionMatrix(predictions_tree_2_train, train_data$category)
confusion_matrix_tree_2_test <-
confusionMatrix(predictions_tree_2_test, test_data$category)

accuracy_tree_2_train <-
confusion_matrix_tree_2_train$overall["Accuracy"]
accuracy_tree_2_test <-
confusion_matrix_tree_2_test$overall["Accuracy"]
```

Checking the nodes of the tree

```r
nodes_tree_2<-sum(tree_2$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_2,
accuracy_tree_2_train, accuracy_tree_2_test, 10, 2, 6))
```

Tree 3

```r
tree_3<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 15, maxdepth = 2, minbucket = 9)

predictions_tree_3_train <- predict(tree_3, newdata = train_data, type
= "class")
predictions_tree_3_test <- predict(tree_3, newdata = test_data, type =
"class")

confusion_matrix_tree_3_train <-
confusionMatrix(predictions_tree_3_train, train_data$category)
confusion_matrix_tree_3_test <-
confusionMatrix(predictions_tree_3_test, test_data$category)

accuracy_tree_3_train <-
confusion_matrix_tree_3_train$overall["Accuracy"]
accuracy_tree_3_test <-
confusion_matrix_tree_3_test$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_tree_3<-sum(tree_3$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_3,
accuracy_tree_3_train, accuracy_tree_3_test, 15, 2, 9))
```

Tree 4

```
tree_4<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 20, maxdepth = 3, minbucket = 12)

predictions_tree_4_train <- predict(tree_4, newdata = train_data, type
= "class")
predictions_tree_4_test <- predict(tree_4, newdata = test_data, type =
"class")

confusion_matrix_tree_4_train <-
confusionMatrix(predictions_tree_4_train, train_data$category)
confusion_matrix_tree_4_test <-
confusionMatrix(predictions_tree_4_test, test_data$category)

accuracy_tree_4_train <-
confusion_matrix_tree_4_train$overall["Accuracy"]
accuracy_tree_4_test <-
confusion_matrix_tree_4_test$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_tree_4<-sum(tree_4$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_4,
accuracy_tree_4_train, accuracy_tree_4_test, 20, 3, 12))
```

Tree 5

```
tree_5<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 25, maxdepth = 3, minbucket = 15)

predictions_tree_5_train <- predict(tree_5, newdata = train_data, type
= "class")
```

```
predictions_tree_5_test <- predict(tree_5, newdata = test_data, type =
"class")

confusion_matrix_tree_5_train <-
confusionMatrix(predictions_tree_5_train, train_data$category)
confusion_matrix_tree_5_test <-
confusionMatrix(predictions_tree_5_test, test_data$category)

accuracy_tree_5_train <-
confusion_matrix_tree_5_train$overall["Accuracy"]
accuracy_tree_5_test <-
confusion_matrix_tree_5_test$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_tree_5<-sum(tree_5$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_5,
accuracy_tree_5_train, accuracy_tree_5_test, 25, 3, 15))
```

Tree 6

```
tree_6<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 40, maxdepth = 8, minbucket = 30)

predictions_tree_6_train <- predict(tree_6, newdata = train_data, type
= "class")
predictions_tree_6_test <- predict(tree_6, newdata = test_data, type =
"class")

confusion_matrix_tree_6_train <-
confusionMatrix(predictions_tree_6_train, train_data$category)
confusion_matrix_tree_6_test <-
confusionMatrix(predictions_tree_6_test, test_data$category)

accuracy_tree_6_train <-
confusion_matrix_tree_6_train$overall["Accuracy"]
accuracy_tree_6_test <-
confusion_matrix_tree_6_test$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_tree_6<-sum(tree_6$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_6,
accuracy_tree_6_train, accuracy_tree_6_test, 40, 8, 30))
```

Tree 7

```
tree_7<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 70, maxdepth = 8, minbucket = 30)

predictions_tree_7_train <- predict(tree_7, newdata = train_data, type
= "class")
predictions_tree_7_test <- predict(tree_7, newdata = test_data, type =
"class")

confusion_matrix_tree_7_train <-
confusionMatrix(predictions_tree_7_train, train_data$category)
confusion_matrix_tree_7_test <-
confusionMatrix(predictions_tree_7_test, test_data$category)

accuracy_tree_7_train <-
confusion_matrix_tree_7_train$overall["Accuracy"]
accuracy_tree_7_test <-
confusion_matrix_tree_7_test$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_tree_7<-sum(tree_7$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_7,
accuracy_tree_7_train, accuracy_tree_7_test, 70, 8, 30))
```

Tree 8

```
tree_8<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 120, maxdepth = 12, minbucket = 80)

predictions_tree_8_train <- predict(tree_8, newdata = train_data, type
= "class")
```

```r
predictions_tree_8_test <- predict(tree_8, newdata = test_data, type =
"class")

confusion_matrix_tree_8_train <-
confusionMatrix(predictions_tree_8_train, train_data$category)
confusion_matrix_tree_8_test <-
confusionMatrix(predictions_tree_8_test, test_data$category)

accuracy_tree_8_train <-
confusion_matrix_tree_8_train$overall["Accuracy"]
accuracy_tree_8_test <-
confusion_matrix_tree_8_test$overall["Accuracy"]
```

Checking the nodes of the tree

```r
nodes_tree_8<-sum(tree_8$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_8,
accuracy_tree_8_train, accuracy_tree_8_test, 120, 12, 80))
```

Tree 9

```r
tree_9<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 180, maxdepth = 15, minbucket = 120)

predictions_tree_9_train <- predict(tree_9, newdata = train_data, type
= "class")
predictions_tree_9_test <- predict(tree_9, newdata = test_data, type =
"class")

confusion_matrix_tree_9_train <-
confusionMatrix(predictions_tree_9_train, train_data$category)
confusion_matrix_tree_9_test <-
confusionMatrix(predictions_tree_9_test, test_data$category)

accuracy_tree_9_train <-
confusion_matrix_tree_9_train$overall["Accuracy"]
accuracy_tree_9_test <-
confusion_matrix_tree_9_test$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_tree_9<-sum(tree_9$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_9,
accuracy_tree_9_train, accuracy_tree_9_test, 180, 15, 120))
```

Tree 10

```
tree_10<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 200, maxdepth = 20, minbucket = 150)

predictions_tree_10_train <- predict(tree_10, newdata = train_data,
type = "class")
predictions_tree_10_test <- predict(tree_10, newdata = test_data, type
= "class")

confusion_matrix_tree_10_train <-
confusionMatrix(predictions_tree_10_train, train_data$category)
confusion_matrix_tree_10_test <-
confusionMatrix(predictions_tree_10_test, test_data$category)

accuracy_tree_10_train <-
confusion_matrix_tree_10_train$overall["Accuracy"]
accuracy_tree_10_test <-
confusion_matrix_tree_10_test$overall["Accuracy"]
```

Checking the nodes of the tree

```
nodes_tree_10<-sum(tree_10$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_10,
accuracy_tree_10_train, accuracy_tree_10_test, 200, 20, 150))
```

Tree 11

```
tree_11<- rpart(category ~ ., data = train_data, method = "class",
minsplit = 300, maxdepth = 25, minbucket = 200)

predictions_tree_11_train <- predict(tree_11, newdata = train_data,
type = "class")
```

```
predictions_tree_11_test <- predict(tree_11, newdata = test_data, type
= "class")

confusion_matrix_tree_11_train <-
confusionMatrix(predictions_tree_11_train, train_data$category)
confusion_matrix_tree_11_test <-
confusionMatrix(predictions_tree_11_test, test_data$category)

accuracy_tree_11_train <-
confusion_matrix_tree_11_train$overall["Accuracy"]
accuracy_tree_11_test <-
confusion_matrix_tree_11_test$overall["Accuracy"]
```

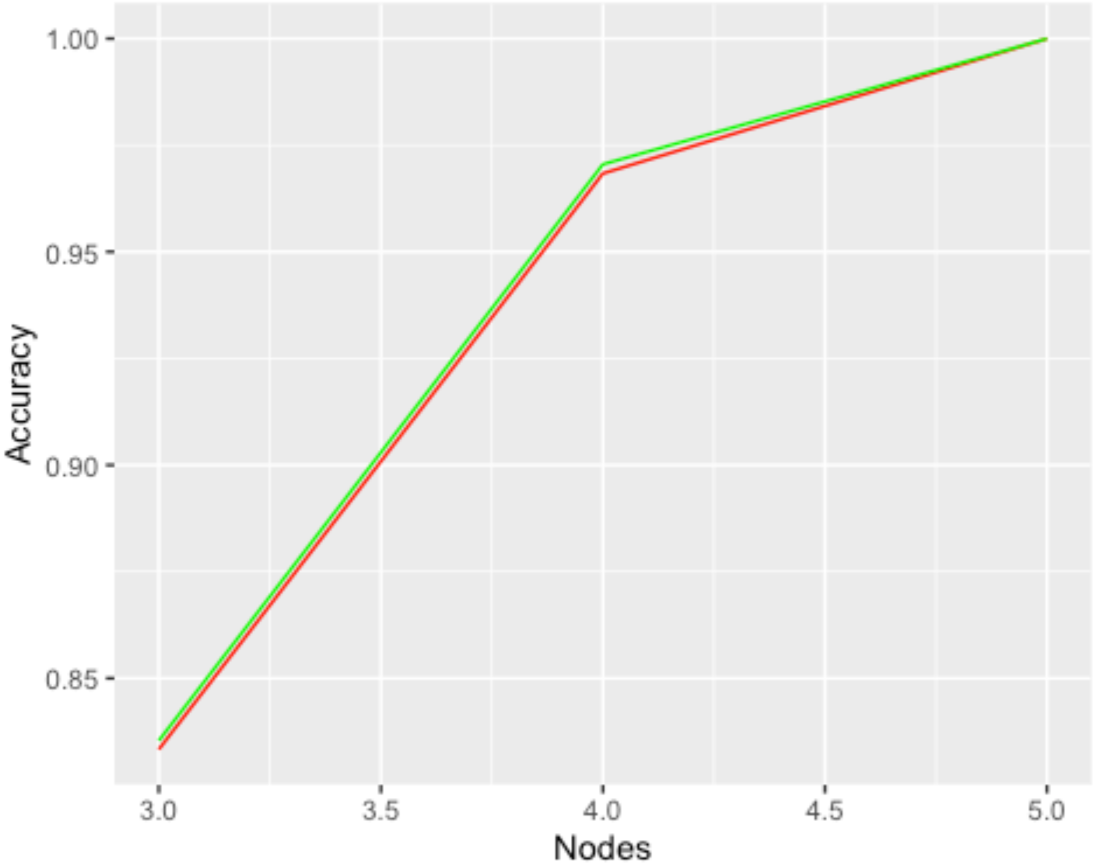
Checking the nodes of the tree

```
nodes_tree_11<-sum(tree_11$frame$var == "<leaf>")

parameters_table <- parameters_table %>% rbind(list(nodes_tree_11,
accuracy_tree_11_train, accuracy_tree_11_test, 300, 25, 200))


print(parameters_table)

##           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       20        3
12
## 13            4     0.9683698    0.9705160       25        3
15
## 14            5     1.0000000    1.0000000       40        8
30
## 15            5     1.0000000    1.0000000       70        8
30
## 16            4     0.9683698    0.9705160      120       12
80
```

```
## 17              4     0.9683698    0.9705160      180        15
120
## 18              4     0.9683698    0.9705160      200        20
150
## 19              4     0.9683698    0.9705160      300        25
200
```

Observing the accuracies with line graph

```
ggplot(parameters_table, aes(x=Nodes)) +
  geom_line(aes(y = TrainAccuracy), color = "red") +
  geom_line(aes(y = TestAccuracy), color="green") +
  ylab("Accuracy")
```



As accuracy of the training and testing data looks same, we can say there is no inflection point.

Question C)

Final Choice of Model and Evaluation

The best model parameter is tree number 6.

Parameters:Nodes = 5, Minsplit = 40, Maxdepth = 8 and Minbucket = 30

Confusion matrix

```
confusion_matrix_tree_6_train <-
confusionMatrix(predictions_tree_6_train, train_data$category)
confusion_matrix_tree_6_test <-
confusionMatrix(predictions_tree_6_test, test_data$category)
```

Final Model Result

```
print(confusion_matrix_tree_6_train)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   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
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9978, 1)
##     No Information Rate : 0.4933
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
```

```
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity            1.0000   1.0000    1.000   1.0000  1.00000
## Specificity            1.0000   1.0000    1.000   1.0000  1.00000
## Pos Pred Value         1.0000   1.0000    1.000   1.0000  1.00000
## Neg Pred Value         1.0000   1.0000    1.000   1.0000  1.00000
## Prevalence             0.4933   0.2019    0.135   0.1381  0.03163
## Detection Rate         0.4933   0.2019    0.135   0.1381  0.03163
## Detection Prevalence   0.4933   0.2019    0.135   0.1381  0.03163
## Balanced Accuracy      1.0000   1.0000    1.000   1.0000  1.00000
```

```
print(confusion_matrix_tree_6_test)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   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
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.991, 1)
##     No Information Rate : 0.4963
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity            1.0000   1.0000   1.0000   1.0000  1.00000
## Specificity            1.0000   1.0000   1.0000   1.0000  1.00000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000  1.00000
## Neg Pred Value          1.0000   1.0000   1.0000   1.0000  1.00000
## Prevalence              0.4963   0.2015   0.1351   0.1376  0.02948
## Detection Rate          0.4963   0.2015   0.1351   0.1376  0.02948
## Detection Prevalence    0.4963   0.2015   0.1351   0.1376  0.02948
## Balanced Accuracy       1.0000   1.0000   1.0000   1.0000  1.00000
```

The above data makes it clear that our model functions well with accuracy of 1 and it has zero miss classifications.

Evaluating the model with 10-folds cross-validation.

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

hypers = rpart.control(minsplit =  40, maxdepth = 8, minbucket = 30)
tree6 <- train(category ~ ., data = train_data, control = hypers,
trControl = traincontrol, method = "rpart1SE")
```

Report accuracy

```
tree6$results$Accuracy

## [1] 1
```

We can state that our model provides accuracy = 1 through cross validation.

PROBLEM SET 4)

Loading the required Libraries

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

Importing data in R

```
Bankdata <- read.csv("Bank_Modified.csv", header = TRUE)
dim(Bankdata)

## [1] 690  13

head(Bankdata)
```

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

Removing the ID Column

```
Bankdata <- subset(Bankdata, select = -X)
```

Convert ‘approval’ to factor

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

Question A)

Decision Tree Model

```
tree_model <- rpart(approval ~., data = Bankdata, method = "class",
minsplit = 10, maxdepth = 20)
```

Question B)

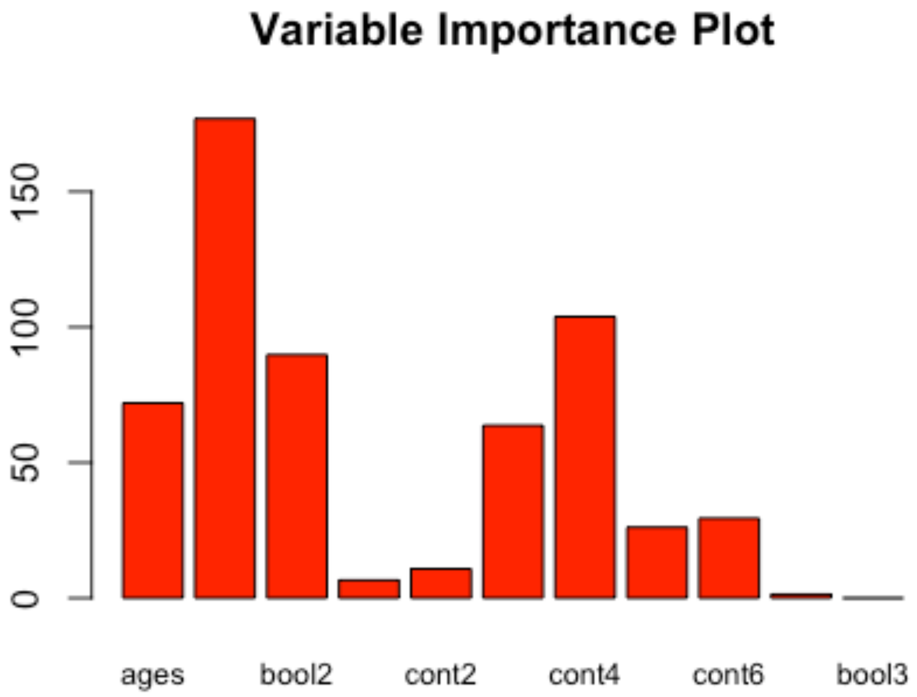
Variable Importance Analysis

```
importance <- varImp(tree_model, scale = FALSE)
```

Question C)

Plot variable importance

```
barplot(importance$Overall, main = "Variable Importance Plot",
        col = "red", cex.names = 0.8,  names.arg =
rownames(importance))
```



Question D)

The top six variables based on the Variable Importance Analysis are bool1 , bool2, ages, cont4, cont3, cont6

```
set.seed(123)
index <- createDataPartition(Bankdata$approval, p = 0.8, list = FALSE)
bankdata_train<- Bankdata[index, ]
bankdata_test <- Bankdata[-index, ]
```

Correctly extracting top 6 variables and constructing the formula

```
Bankdata_1 <- rpart(approval ~ bool1 + bool2 + ages + cont4 + cont3 +
cont6,  data = bankdata_train, method = "class", minsplit = 10,
maxdepth = 20)


predictions_initial <- predict(tree_model, newdata = bankdata_test,
type = "class")
predictions <- predict(Bankdata_1, newdata = bankdata_test, type =
"class")
```

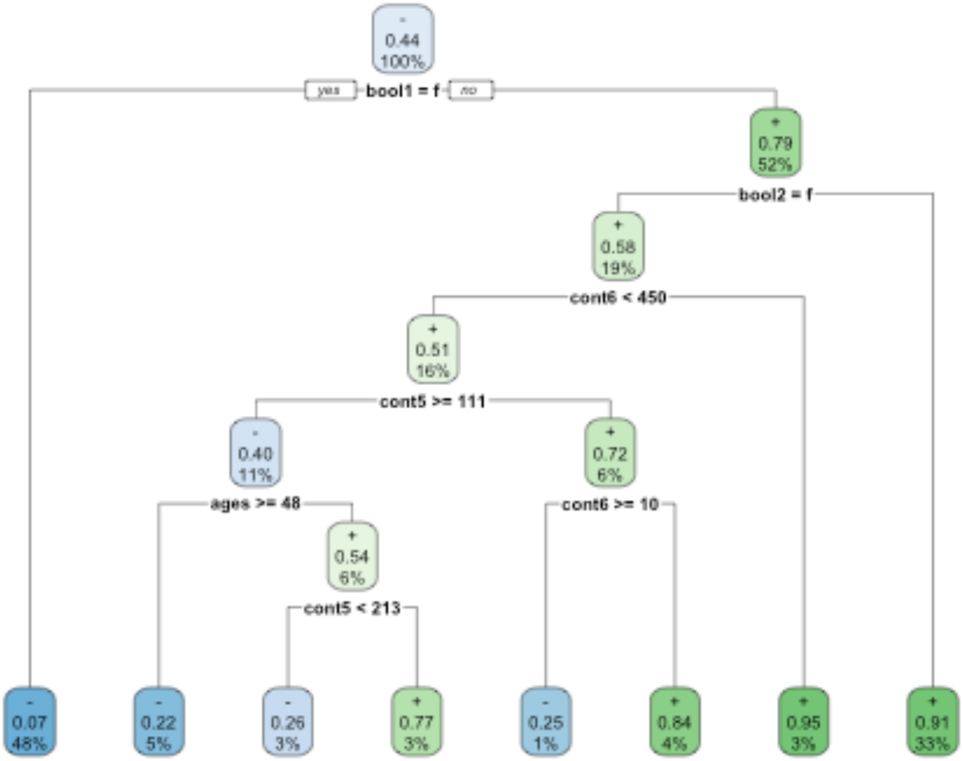
Confusion matrix to evaluate accuracy

```
confusion_matrix <- confusionMatrix(predictions,
bankdata_test$approval)
accuracy <- confusion_matrix$overall["Accuracy"]
print(accuracy)

##  Accuracy
## 0.8540146

confusion_matrix <- confusionMatrix(predictions_initial,
bankdata_test$approval)
accuracy_initial <- confusion_matrix$overall["Accuracy"]
print(accuracy_initial)

##  Accuracy
## 0.8905109

cat("Accuracy of initial model: ", accuracy_initial, "\n")

## Accuracy of initial model:  0.8905109

cat("Accuracy of top 6 vars model: ", accuracy, "\n")

## Accuracy of top 6 vars model:  0.8540146
```

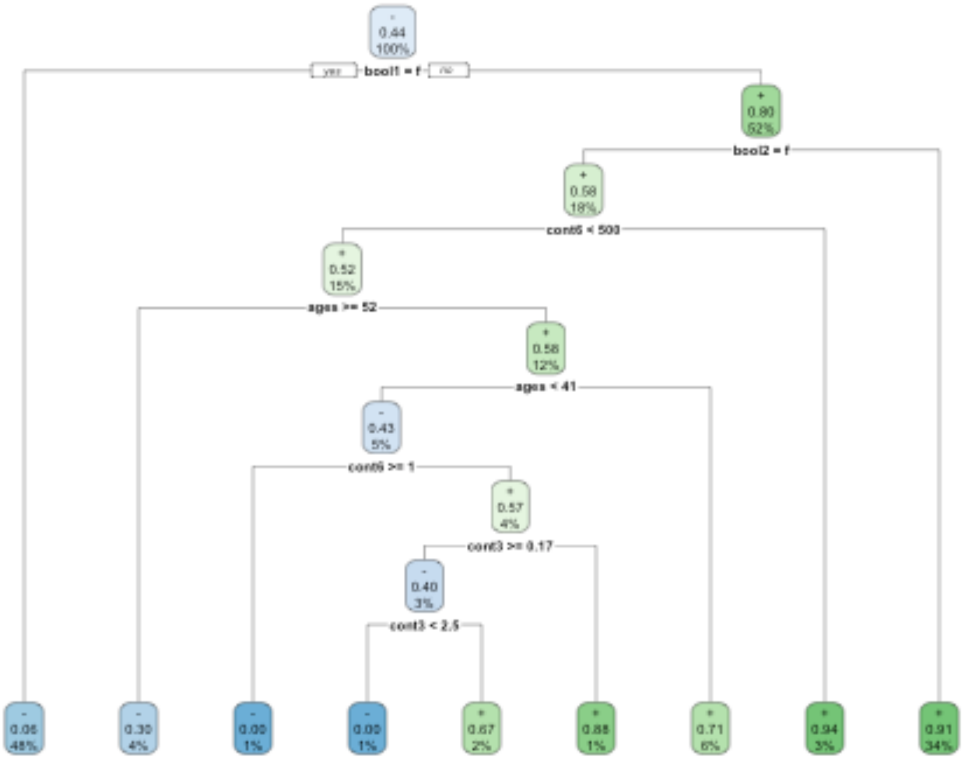
Initially, the accuracy of the model was 0.89 and after rebuilding the model with top 6 variables, accuracy decreased to 0.86.

Question e:

```
library(rpart.plot)
rpart.plot(tree_model)
```



```
rpart.plot(Bankdata_1)
```



By reducing the number of variables, size of the tree decreased.