Final_Decisiontree

MUNERAH

12/3/2021

```
## Load the file


library(readxl)
df <- read.csv("C:/Users/mnooo/Desktop/MLdrafft/general_data__hr.csv")
```

```
# loading library
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(ggplot2)
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.1.2
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.2
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(ROSE)
```

```
## Warning: package 'ROSE' was built under R version 4.1.2
```

```
## Loaded ROSE 0.0-4
```

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 4.1.2
```

```
## Loading required package: tibble
```

```
## Warning: package 'tibble' was built under R version 4.1.2
```

```
## Loading required package: bitops
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
##
## Attaching package: 'rattle'
```

```
## The following object is masked from 'package:randomForest':
##
##     importance
```

```
library(caret)
```

```
## Loading required package: lattice
```

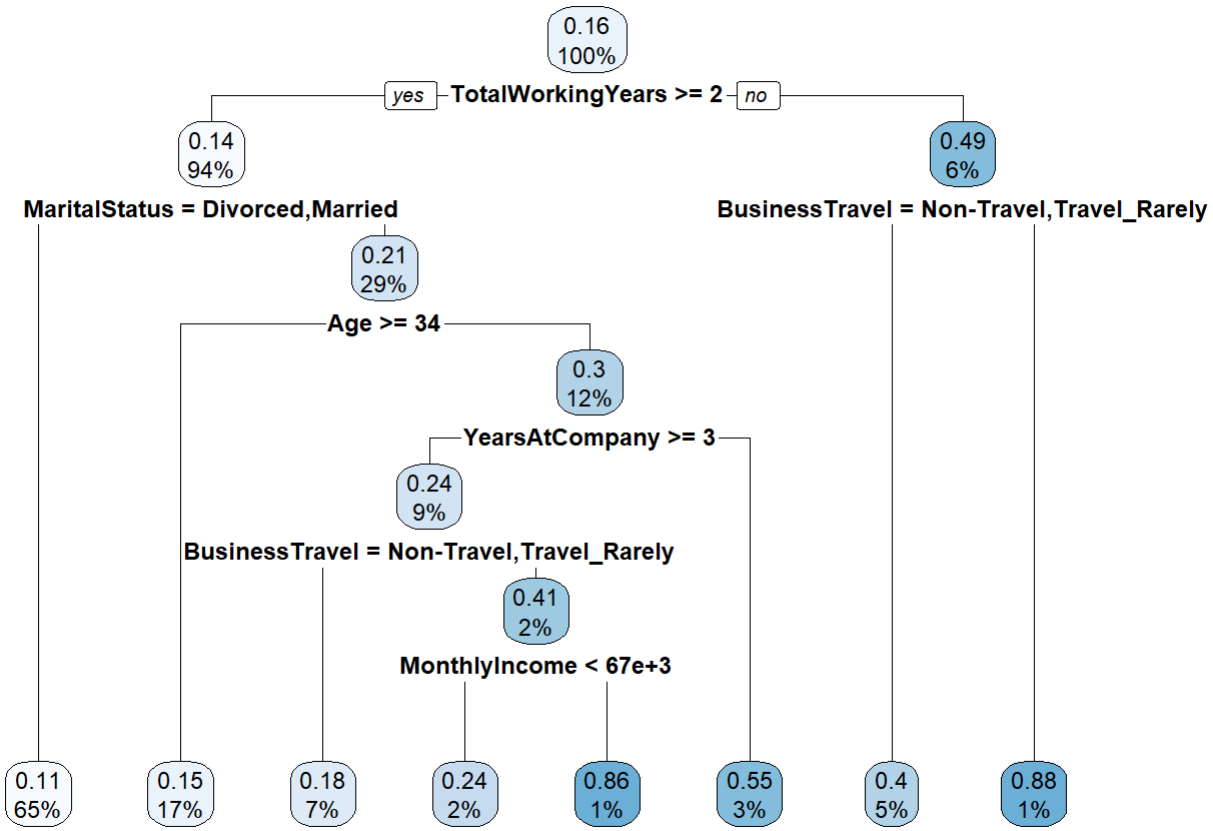
###Preprocessing the dataset

```
#converting chr to factor
df <- df %>% mutate_if(is.character, as.factor)
#converting int to num
df <- df %>% mutate_if(is.integer, as.numeric)

# Treating NA values
df<-na.omit(df)

# dummy coding attrition

df$Attrition<- ifelse(df$Attrition=="Yes",1,0)
```

```
# training and testing split
set.seed(100)
Index <- sample(1:nrow(df),3300)
training <- df[Index,] #75%
testing <- df[-Index,] #25%
```

Building a decision tree model

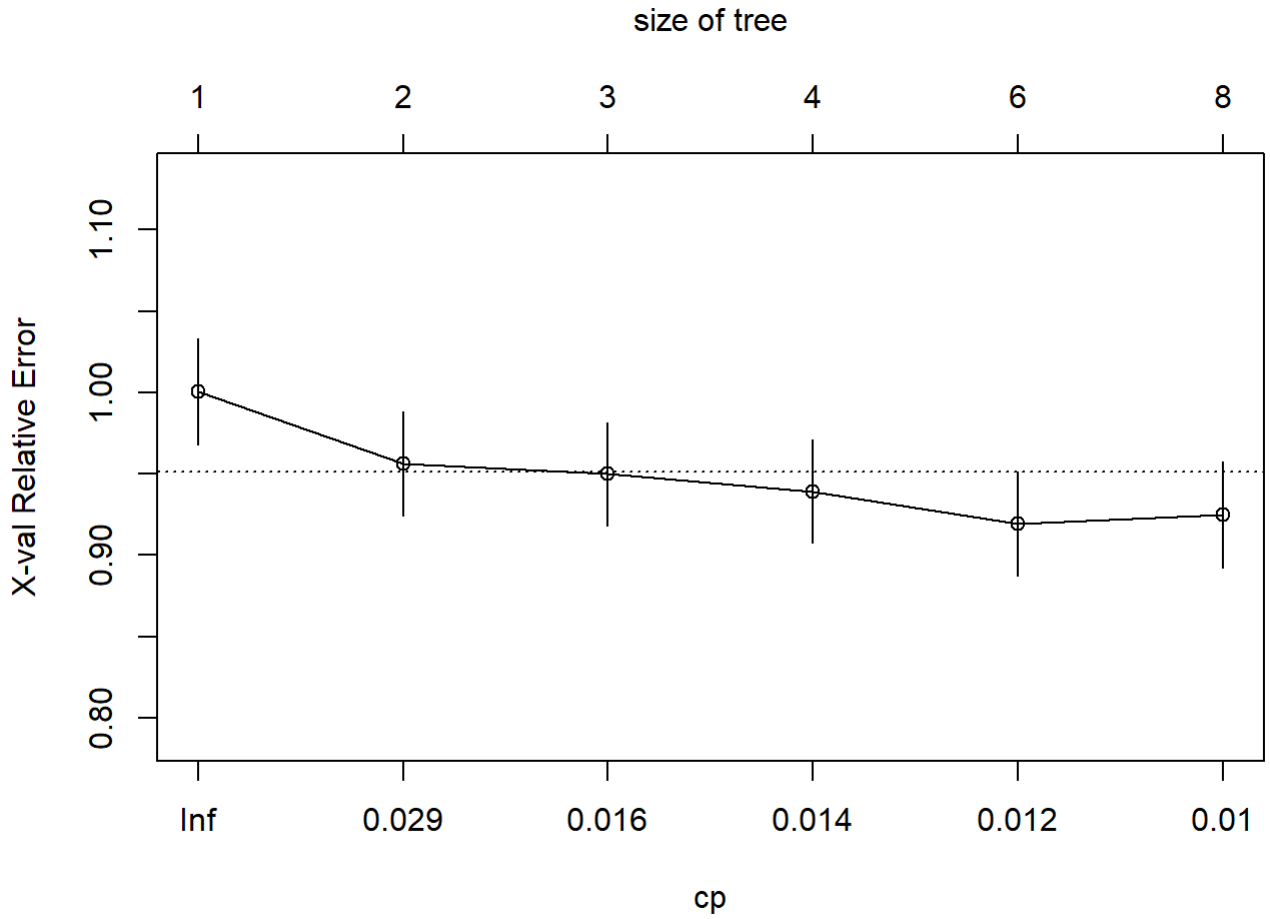
```
model_rpart <- rpart(Attrition~.,data = training)

rpart.plot(model_rpart) ## plot the decision tree
```



```
printcp(model_rpart) ## complexity values and other detaied result
```

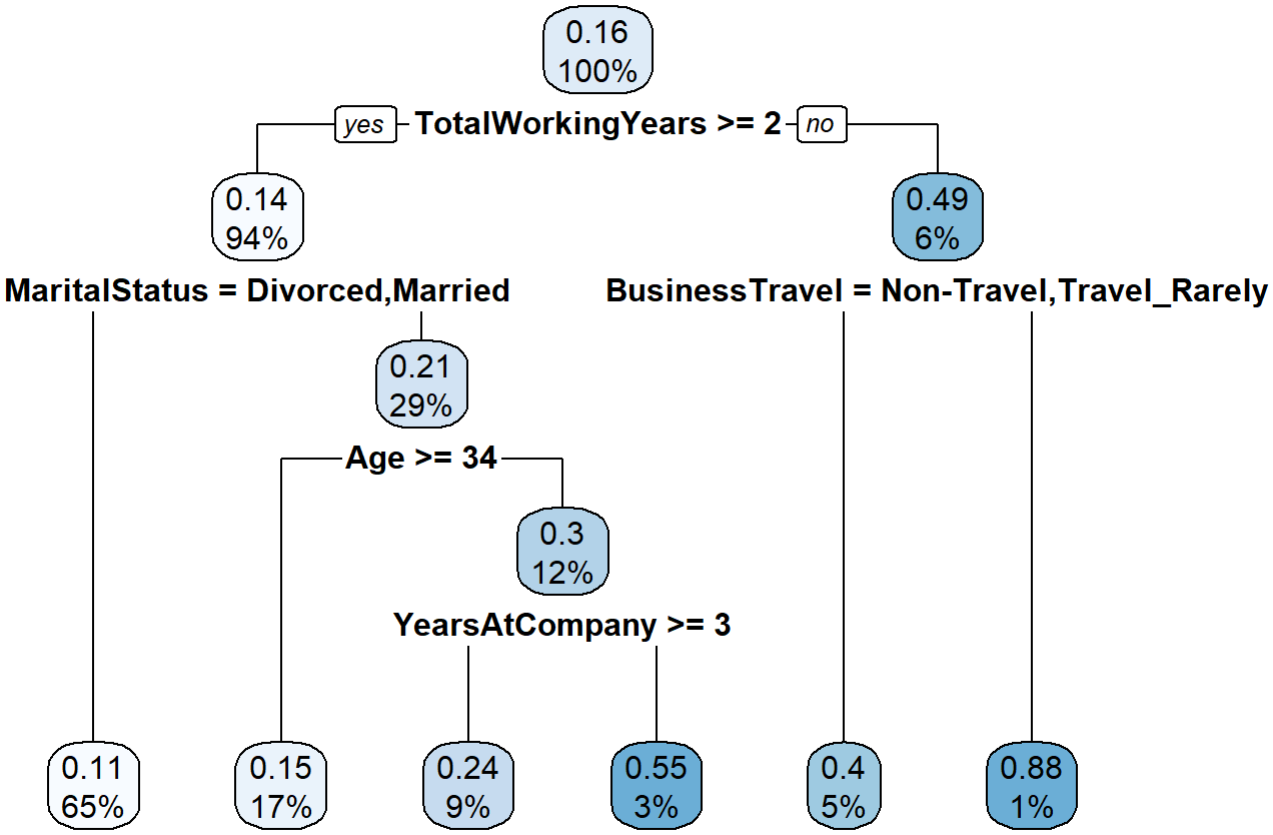
```
##
## Regression tree:
## rpart(formula = Attrition ~ ., data = training)
##
## Variables actually used in tree construction:
## [1] Age               BusinessTravel    MaritalStatus     MonthlyIncome
## [5] TotalWorkingYears YearsAtCompany
##
## Root node error: 441.48/3300 = 0.13378
##
## n= 3300
##
##         CP nsplit rel error  xerror     xstd
## 1 0.049940      0   1.00000 1.00065 0.032471
## 2 0.016499      1   0.95006 0.95637 0.031945
## 3 0.014608      2   0.93356 0.94978 0.031744
## 4 0.013553      3   0.91895 0.93920 0.031601
## 5 0.010074      5   0.89185 0.91931 0.031795
## 6 0.010000      7   0.87170 0.92498 0.032595
```

```
plotcp(model_rpart) ## plot the cp values VS relatively errors
```



Pruning the tree using the least error founds in cp values VS relatively errors plot

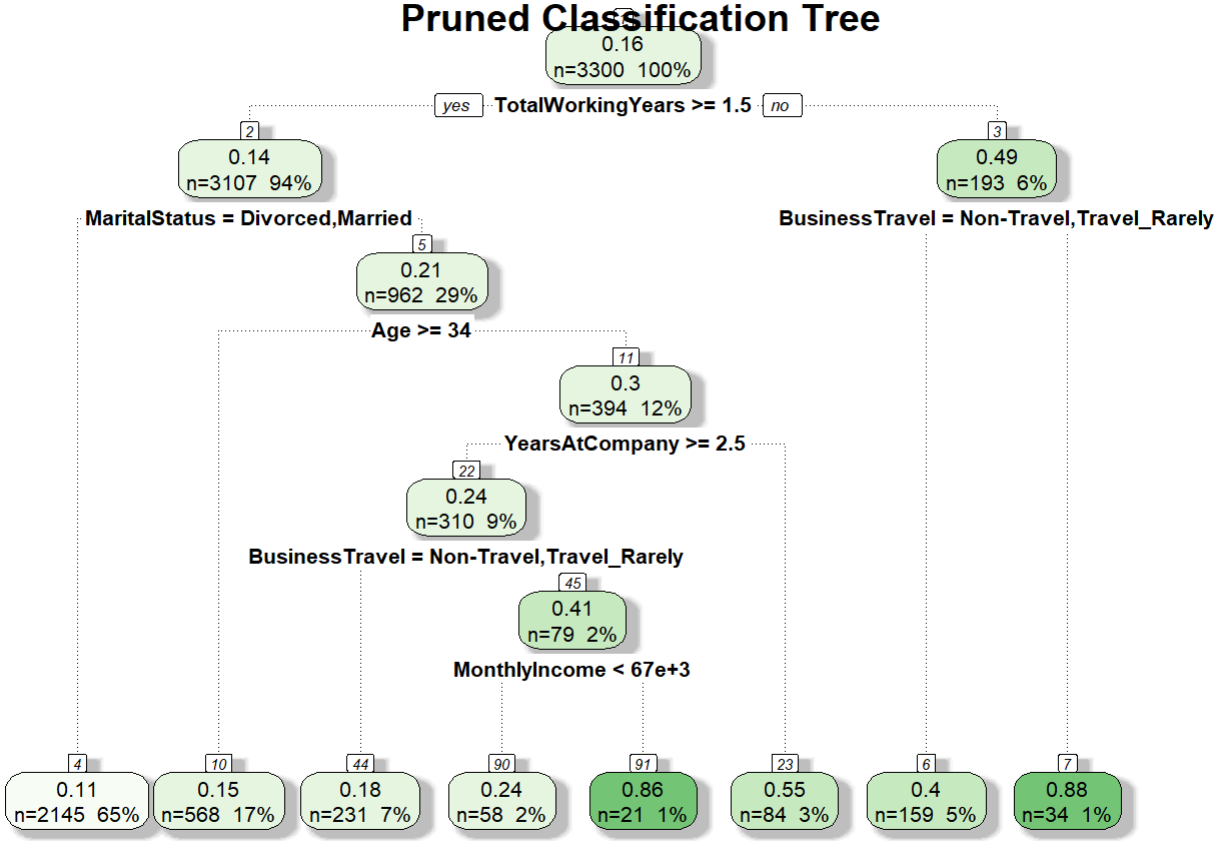
```
###Pruning the tree using cp = 0.012 which has less error
model_rpart_manual <- rpart(Attrition~.,
                     data = training,
                     control =rpart.control(cp = 0.012))
rpart.plot(model_rpart_manual)  ## plot the tree
```



The tree is not performing good. The model must have a minimum error factor with better cp as well

purning the tree

```
ptree<- prune(model_rpart,
              cp= model_rpart$cptable[which.min(model_rpart$cptable[,"rel error"]),"CP"])
fancyRpartPlot(ptree,
               uniform=TRUE,
               main="Pruned Classification Tree")
```




```
Control=trainControl(method= "repeatedcv",number=10,repeats=10,classProbs=TRUE,summaryFunction = multiClassSummary)
```

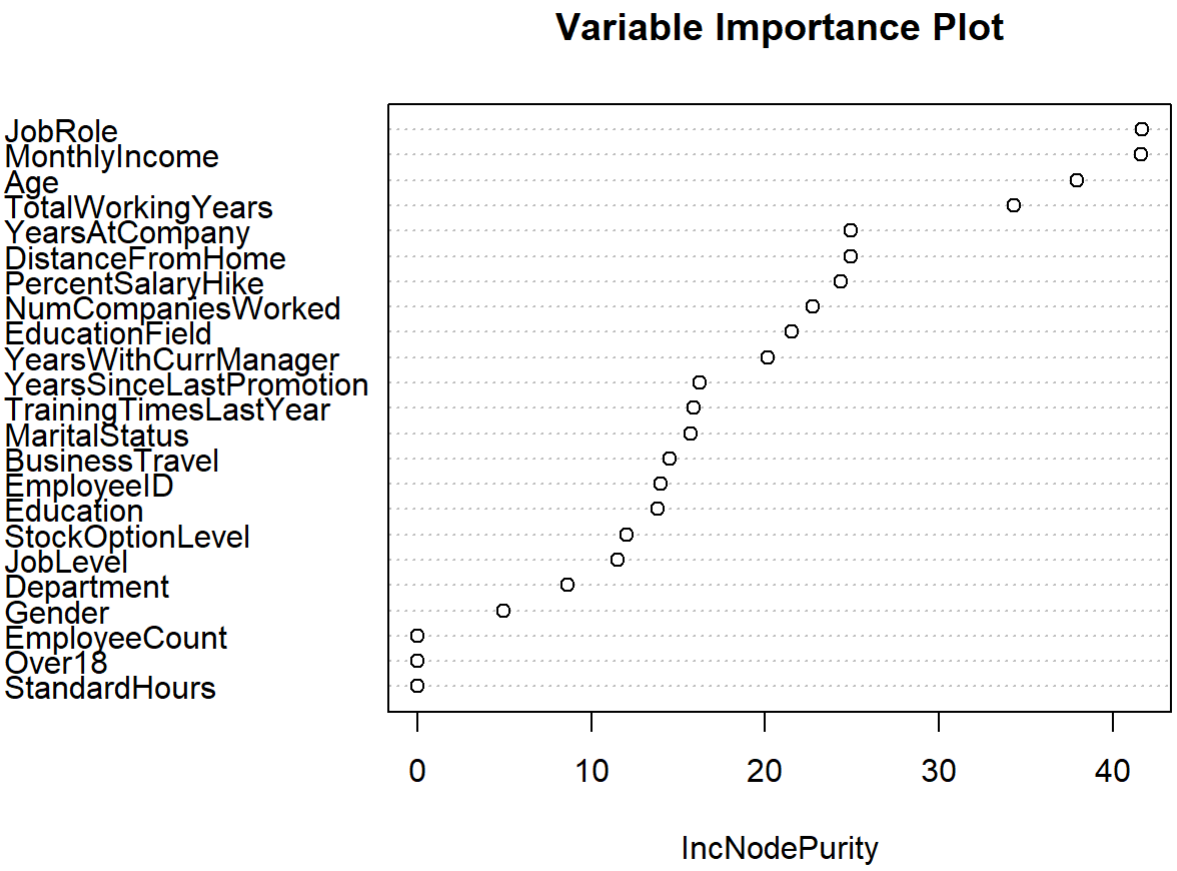
Using different parameters, this tree has balanced error and cp value as well . This tree has the optimal number of splits with seven nodes and eight terminals .

###Variable Importance using randomForest

```
model_rf <- randomForest(Attrition~.,data = training)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

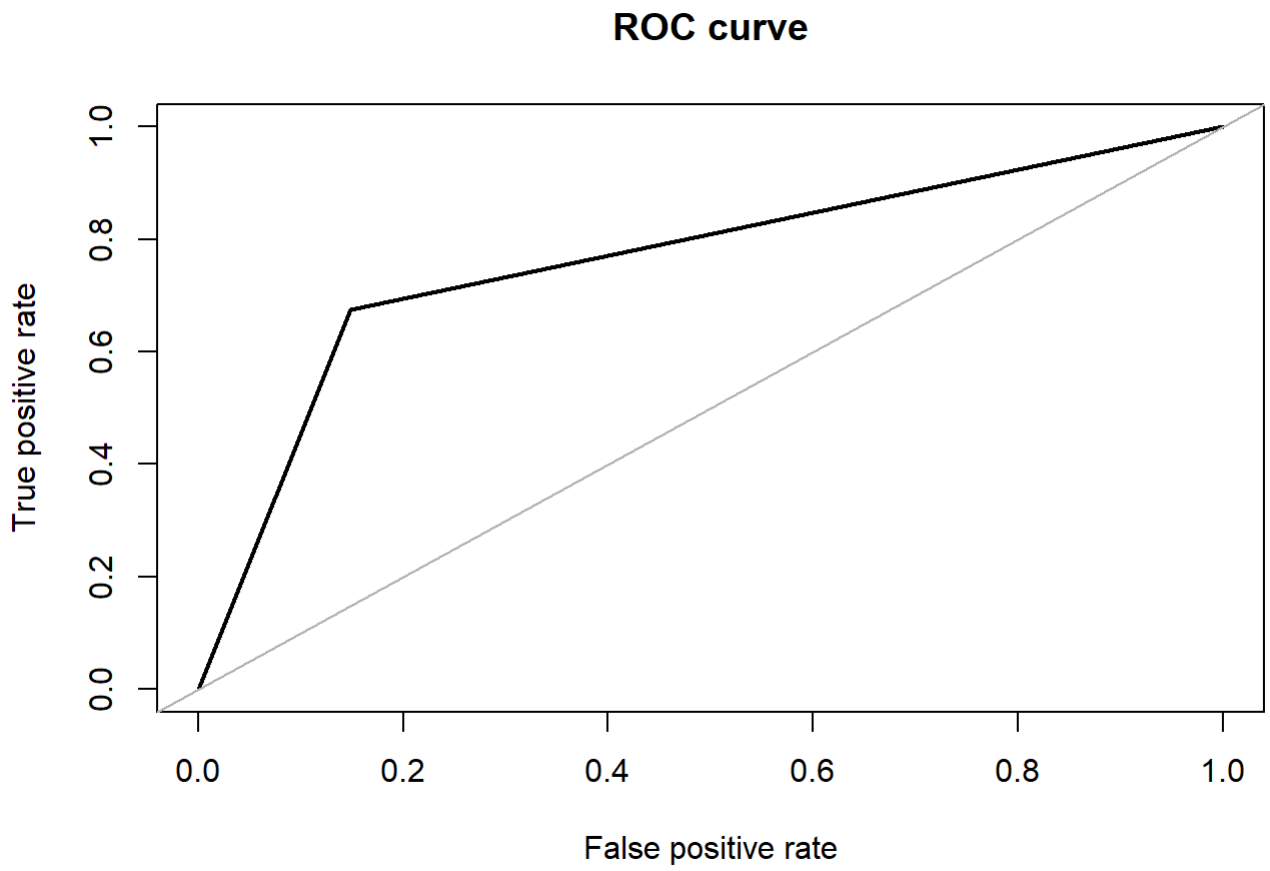
```
varImpPlot(model_rf,main = "Variable Importance Plot")
```



This plot shows the variables importance that affect the dependent variable attrition.

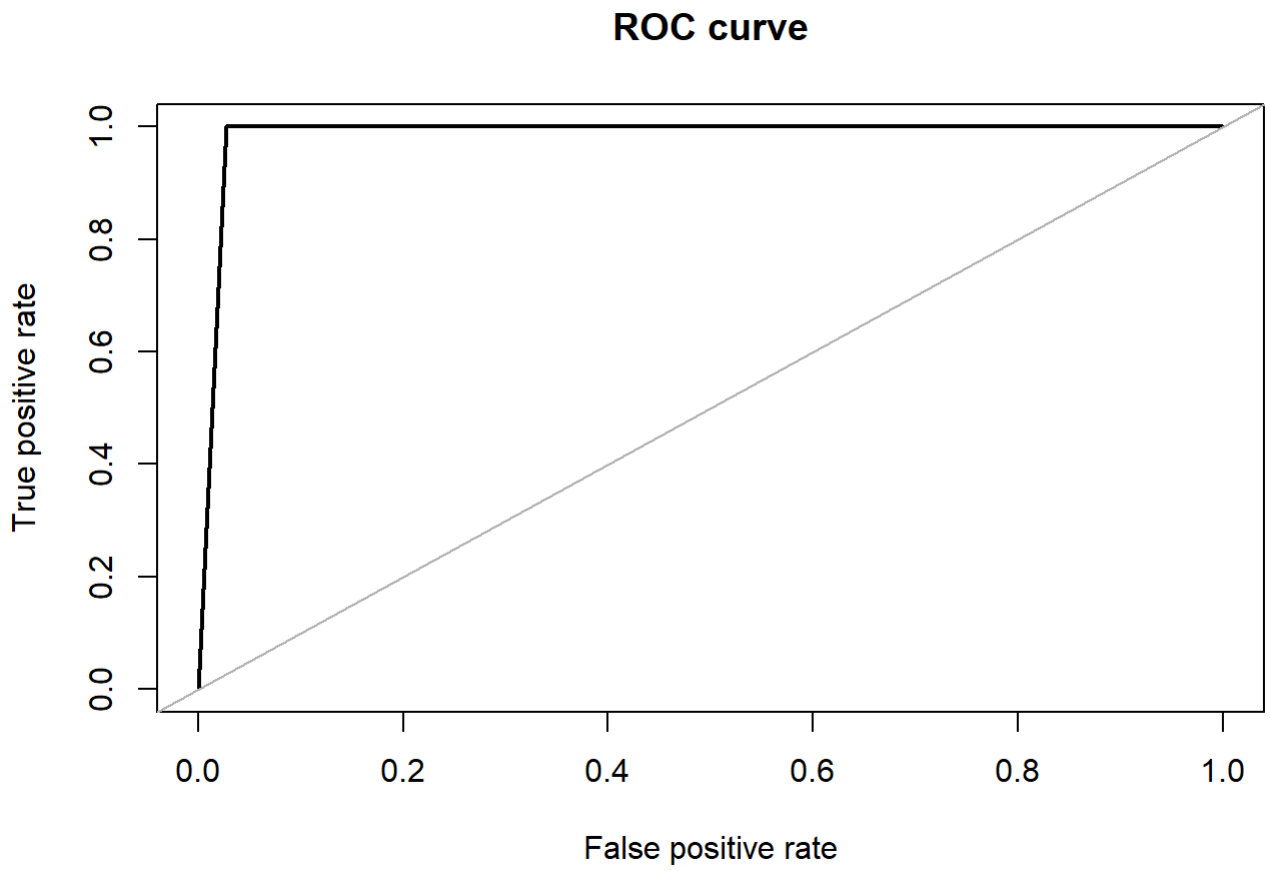
###Performance evaluation for the models

```
# predict Rpart
pred_rpart <- predict(model_rpart,newdata = testing)
pred_rpart<-floor(pred_rpart+0.5)
roc.curve(as.factor(pred_rpart),testing$Attrition) # 76.9%
```




```
## Area under the curve (AUC): 0.764
```

```
# predict random forest
pred_rf <- predict(model_rf,newdata = testing)
pred_rf<-floor(pred_rf+0.5)
roc.curve(as.factor(pred_rf),testing$Attrition) # 99%
```




```
## Area under the curve (AUC): 0.987
```