Random Forest and Logistic Regression

Suhail Shaikh

12/19/2019

Q.) What is the proportion of “Good” to “Bad” cases? Obtain descriptions of the predictor (independent) variables mean, standard deviations, etc. for real-values attributes, frequencies of di???erent category values. Look at the relationship of the input variables with the Target variable. Anything noteworthy in the data? Please include support (graphs, hypothesis testing, etc) for your observations

```
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(readxl)
GermanCred <- read_excel("German Credit.xls")

##changing Data types of all the following attribute to factor..
GermanCred <-mutate(GermanCred,
                       CHK_ACCT = as.factor(CHK_ACCT),
                       HISTORY = as.factor(HISTORY),
                       NEW_CAR = as.factor(NEW_CAR),
                       USED_CAR = as.factor(USED_CAR),
                       FURNITURE = as.factor(FURNITURE),
                       EDUCATION = as.factor(EDUCATION),
                       RETRAINING = as.factor(RETRAINING),
                       SAV_ACCT= as.factor(SAV_ACCT),
                       EMPLOYMENT = as.factor(EMPLOYMENT),
                       MALE_DIV=as.factor(MALE_DIV),
                       MALE_SINGLE = as.factor(MALE_SINGLE),
                       MALE_MAR_or_WID = as.factor(MALE_MAR_or_WID),
                       GUARANTOR = as.factor(GUARANTOR),
                       PRESENT_RESIDENT = as.factor(PRESENT_RESIDENT),
```

```r
                        REAL_ESTATE = as.factor(REAL_ESTATE),
                        PROP_UNKN_NONE =as.factor(PROP_UNKN_NONE),
                        OTHER_INSTALL = as.factor(OTHER_INSTALL),
                        RENT = as.factor(RENT),
                        OWN_RES = as.factor(OWN_RES),
                        JOB = as.factor(JOB),
                        TELEPHONE = as.factor(TELEPHONE),
                        FOREIGN = as.factor(FOREIGN),
                        RESPONSE = as.factor(RESPONSE))
#Proportion of good to bad cases
good_bad <- table(GermanCred$RESPONSE)
prop.table(good_bad)
```

```
##
##   0   1
## 0.3 0.7
```

```r
#Description of Independent Variables
library(psych)
describe(GermanCred)
```

```
##                    vars    n    mean      sd median trimmed     mad min
## OBS#                  1 1000  500.50  288.82  500.5  500.50  370.65   1
## CHK_ACCT*             2 1000    2.58    1.26    2.0    2.60    1.48   1
## DURATION              3 1000   20.90   12.06   18.0   19.47    8.90   4
## HISTORY*              4 1000    3.54    1.08    3.0    3.59    0.00   1
## NEW_CAR*              5 1000    1.23    0.42    1.0    1.17    0.00   1
## USED_CAR*             6 1000    1.10    0.30    1.0    1.00    0.00   1
## FURNITURE*            7 1000    1.18    0.39    1.0    1.10    0.00   1
## RADIO/TV              8 1000    0.28    0.45    0.0    0.22    0.00   0
## EDUCATION*            9 1000    1.05    0.22    1.0    1.00    0.00   1
## RETRAINING*          10 1000    1.10    0.30    1.0    1.00    0.00   1
## AMOUNT               11 1000 3271.26 2822.74 2319.5 2754.57 1627.15 250
## SAV_ACCT*            12 1000    2.10    1.58    1.0    1.88    0.00   1
## EMPLOYMENT*          13 1000    3.38    1.21    3.0    3.43    1.48   1
## INSTALL_RATE         14 1000    2.97    1.12    3.0    3.09    1.48   1
## MALE_DIV*            15 1000    1.05    0.22    1.0    1.00    0.00   1
## MALE_SINGLE*         16 1000    1.55    0.50    2.0    1.56    0.00   1
## MALE_MAR_or_WID*     17 1000    1.09    0.29    1.0    1.00    0.00   1
## CO-APPLICANT         18 1000    0.04    0.20    0.0    0.00    0.00   0
## GUARANTOR*           19 1000    1.05    0.22    1.0    1.00    0.00   1
## PRESENT_RESIDENT*    20 1000    2.85    1.10    3.0    2.93    1.48   1
## REAL_ESTATE*         21 1000    1.28    0.45    1.0    1.23    0.00   1
## PROP_UNKN_NONE*      22 1000    1.15    0.36    1.0    1.07    0.00   1
## AGE                  23 1000   35.55   11.38   33.0   34.17   10.38  19
## OTHER_INSTALL*       24 1000    1.19    0.39    1.0    1.11    0.00   1
## RENT*                25 1000    1.18    0.38    1.0    1.10    0.00   1
## OWN_RES*             26 1000    1.71    0.45    2.0    1.77    0.00   1
## NUM_CREDITS          27 1000    1.41    0.58    1.0    1.33    0.00   1
## JOB*                 28 1000    2.90    0.65    3.0    2.91    0.00   1
```

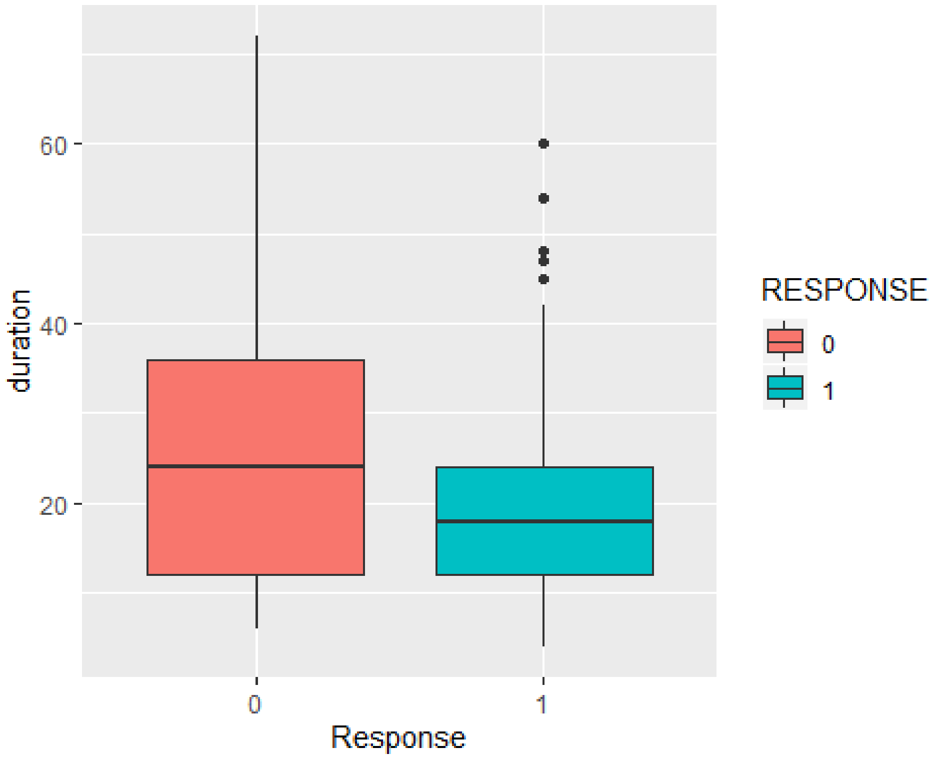
```
## NUM_DEPENDENTS      29 1000    1.16    0.36    1.0    1.07    0.00   1
## TELEPHONE*          30 1000    1.40    0.49    1.0    1.38    0.00   1
## FOREIGN*            31 1000    1.04    0.19    1.0    1.00    0.00   1
## RESPONSE*           32 1000    1.70    0.46    2.0    1.75    0.00   1
##                       max range  skew kurtosis    se
## OBS#                 1000   999  0.00    -1.20  9.13
## CHK_ACCT*               4     3  0.01    -1.66  0.04
## DURATION               72    68  1.09     0.90  0.38
## HISTORY*                5     4 -0.01    -0.59  0.03
## NEW_CAR*                2     1  1.25    -0.43  0.01
## USED_CAR*               2     1  2.61     4.81  0.01
## FURNITURE*              2     1  1.65     0.74  0.01
## RADIO/TV                1     1  0.98    -1.04  0.01
## EDUCATION*              2     1  4.12    15.02  0.01
## RETRAINING*             2     1  2.72     5.40  0.01
## AMOUNT              18424 18174  1.94     4.25 89.26
## SAV_ACCT*               5     4  1.01    -0.69  0.05
## EMPLOYMENT*             5     4 -0.12    -0.94  0.04
## INSTALL_RATE            4     3 -0.53    -1.21  0.04
## MALE_DIV*               2     1  4.12    15.02  0.01
## MALE_SINGLE*            2     1 -0.19    -1.96  0.02
## MALE_MAR_or_WID*        2     1  2.82     5.95  0.01
## CO-APPLICANT            1     1  4.62    19.39  0.01
## GUARANTOR*              2     1  4.03    14.25  0.01
## PRESENT_RESIDENT*       4     3 -0.27    -1.38  0.03
## REAL_ESTATE*            2     1  0.97    -1.07  0.01
## PROP_UNKN_NONE*         2     1  1.91     1.67  0.01
## AGE                    75    56  1.02     0.58  0.36
## OTHER_INSTALL*          2     1  1.61     0.60  0.01
## RENT*                   2     1  1.67     0.80  0.01
## OWN_RES*                2     1 -0.94    -1.12  0.01
## NUM_CREDITS             4     3  1.27     1.58  0.02
## JOB*                    4     3 -0.37     0.49  0.02
## NUM_DEPENDENTS          2     1  1.90     1.63  0.01
## TELEPHONE*              2     1  0.39    -1.85  0.02
## FOREIGN*                2     1  4.90    22.02  0.01
## RESPONSE*               2     1 -0.87    -1.24  0.01
```

```r
library(ggplot2)
```

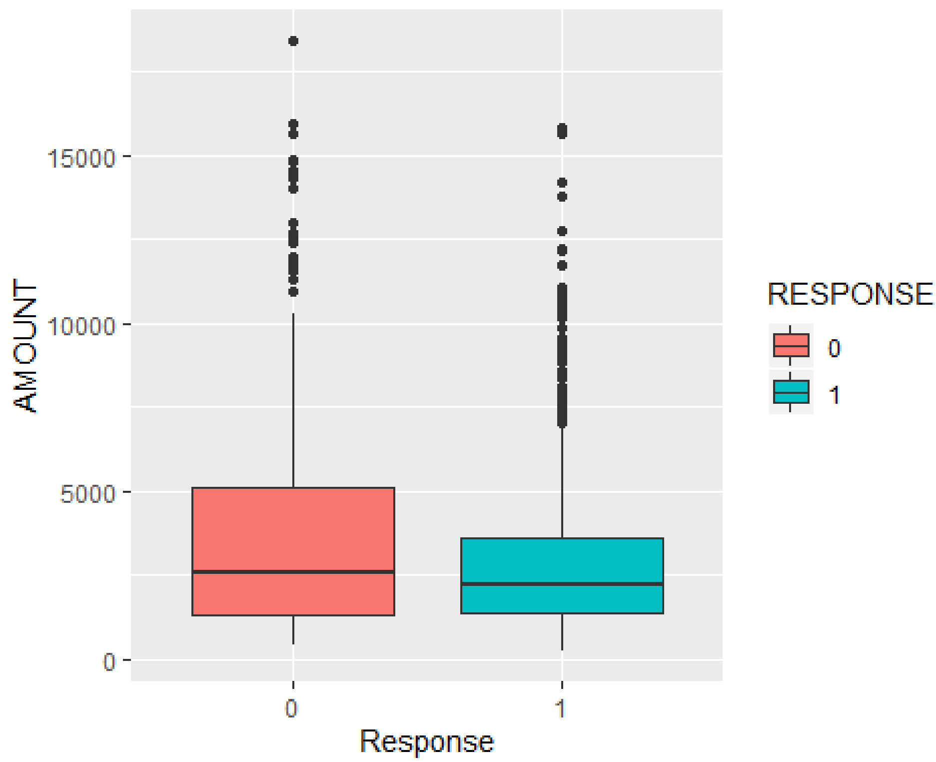
```
##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```r
###we are checking relation of individual independent variable with dependent variables.
### using Histograms, Barplots and Box plots..
library(ggplot2)
```

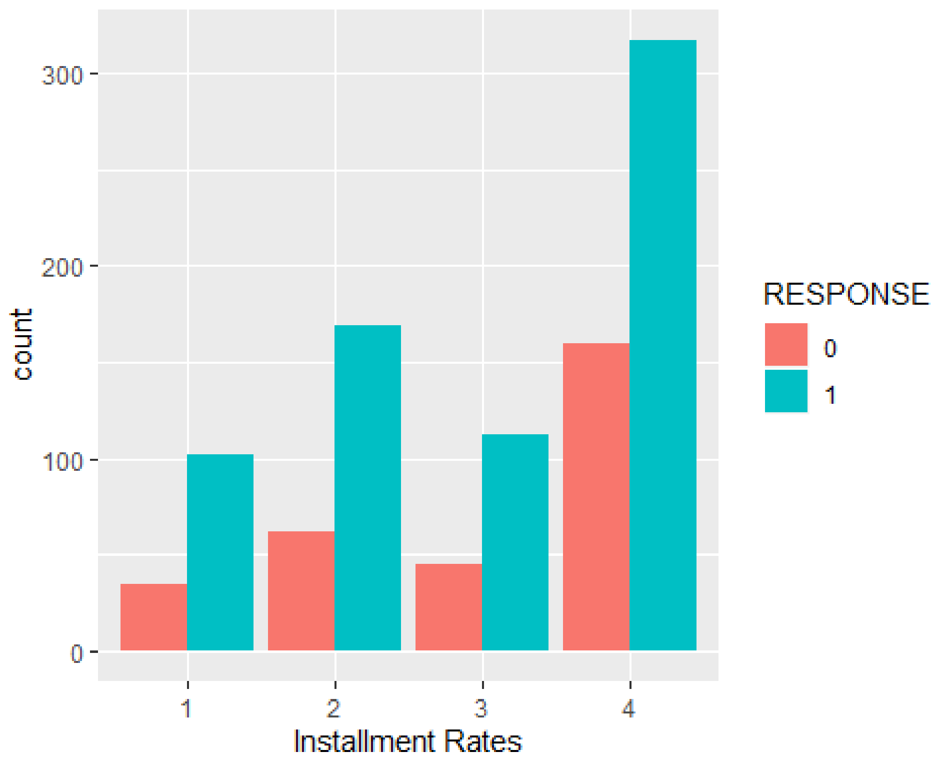
```
#relation between duration and response
ggplot(GermanCred, aes(x = RESPONSE, y= DURATION, fill =
RESPONSE))+geom_boxplot()+xlab("Response")+ylab("duration")
```



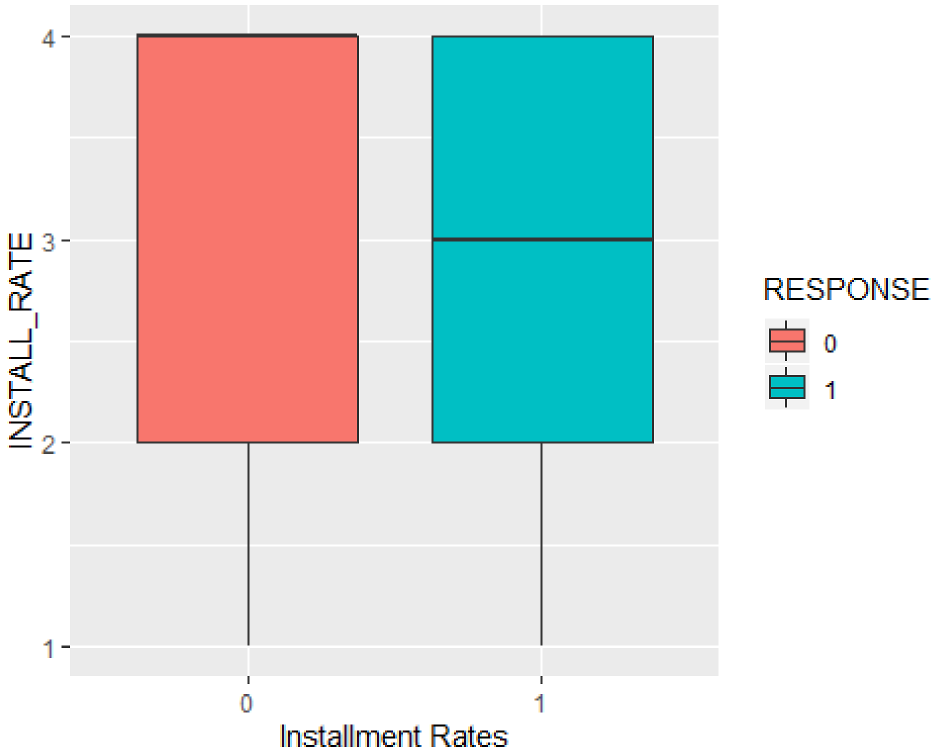

```
# above box plot shows that Median AGE for good response is less than bad
responses. It indicates a relation with dependent variable.

# relation between amount
ggplot(GermanCred, aes(x = RESPONSE, y= AMOUNT, fill =
RESPONSE))+geom_boxplot()+xlab("Response")+ylab("AMOUNT")
```



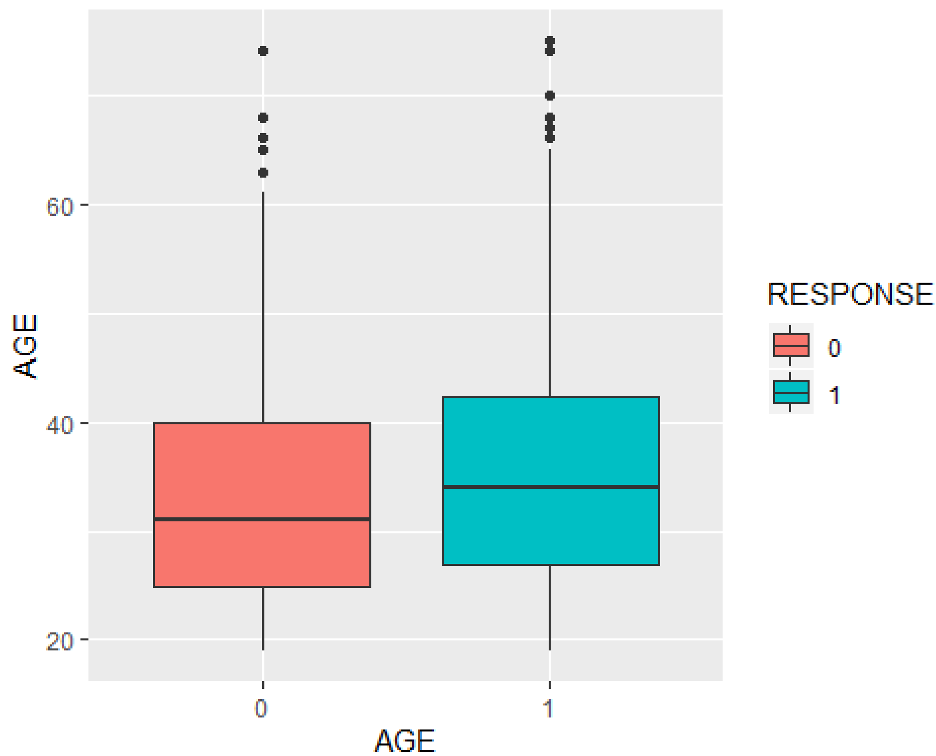

```
# not much of difference to be observed from Box plot for amount variable.
# installment rate

ggplot(GermanCred, aes(factor(INSTALL_RATE), ..count..))+
geom_bar(aes(fill=RESPONSE),position = "dodge")+xlab("Installment Rates")
```



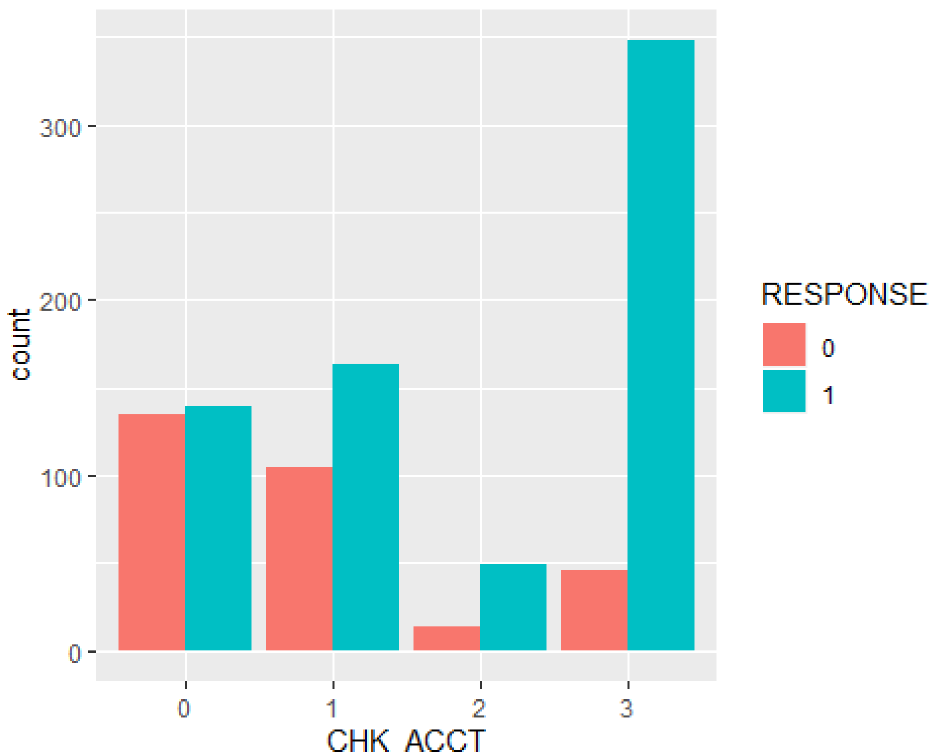

```r
# Barplot for Install Rate shows huge variation, For Installment rates 2,4
there is high percentage for Good responses.
ggplot(GermanCred, aes(x = RESPONSE, y = INSTALL_RATE, fill = RESPONSE ))+
geom_boxplot()+xlab("Installment Rates")
```



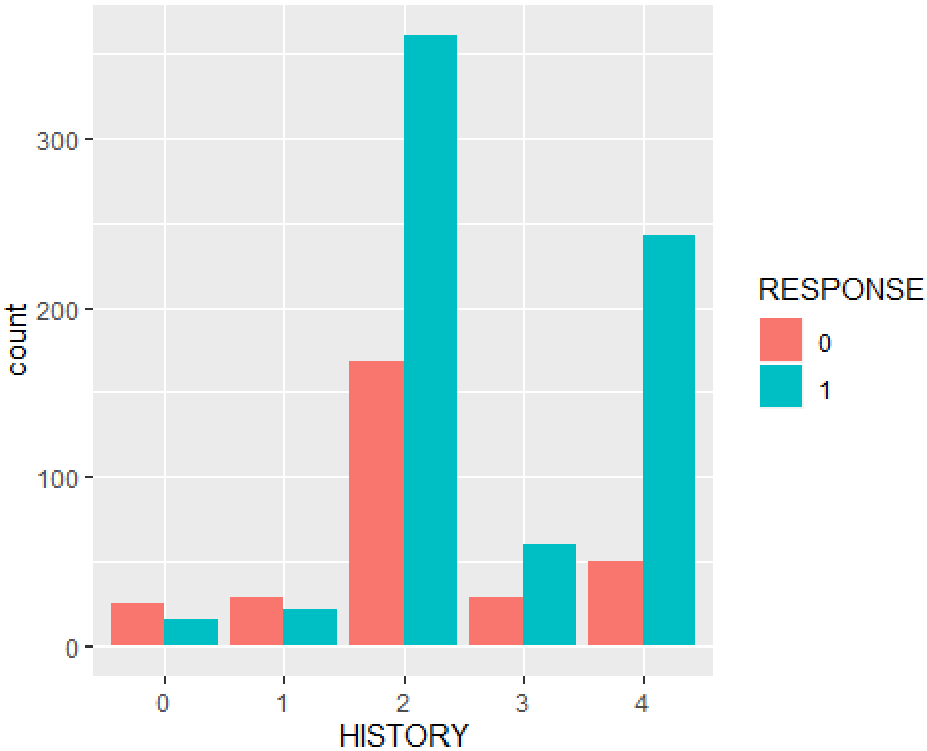

```
#Median installment rate for good response is far superior than bad
ones....this is definately a significant variable.....
# Age
ggplot(GermanCred, aes(x = RESPONSE, y = AGE, fill = RESPONSE ))+
geom_boxplot()+xlab("AGE")
```



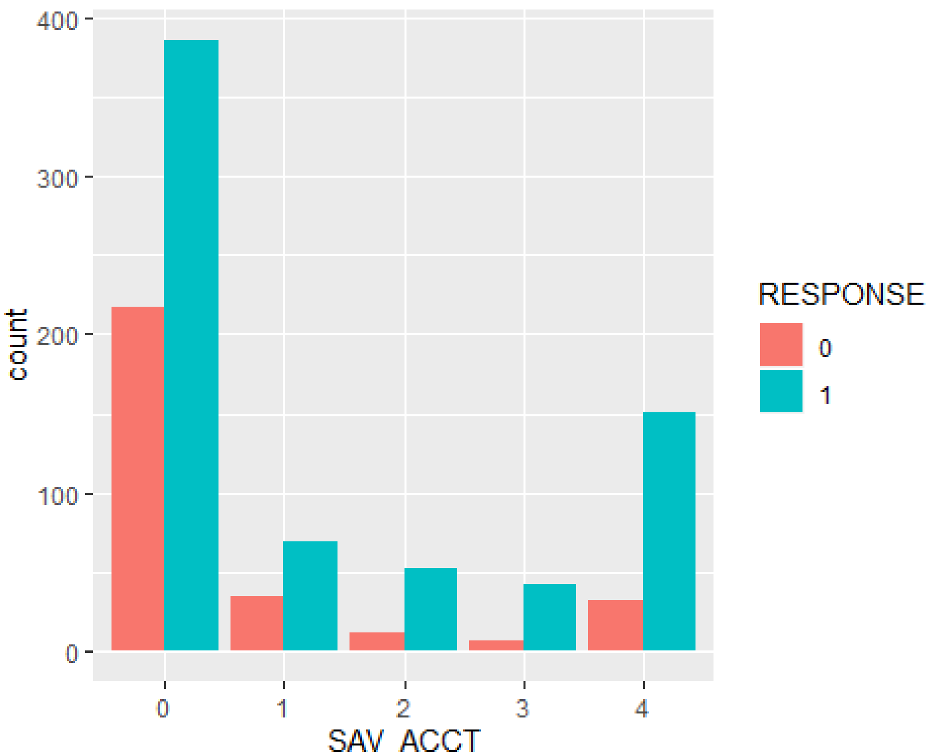

```
###median age for good responses are greater than bad responses....variable
looking significant....

#categorical variable  relations

ggplot(GermanCred, aes(CHK_ACCT,..count..))+geom_bar(aes(fill=RESPONSE),
position="dodge")
```



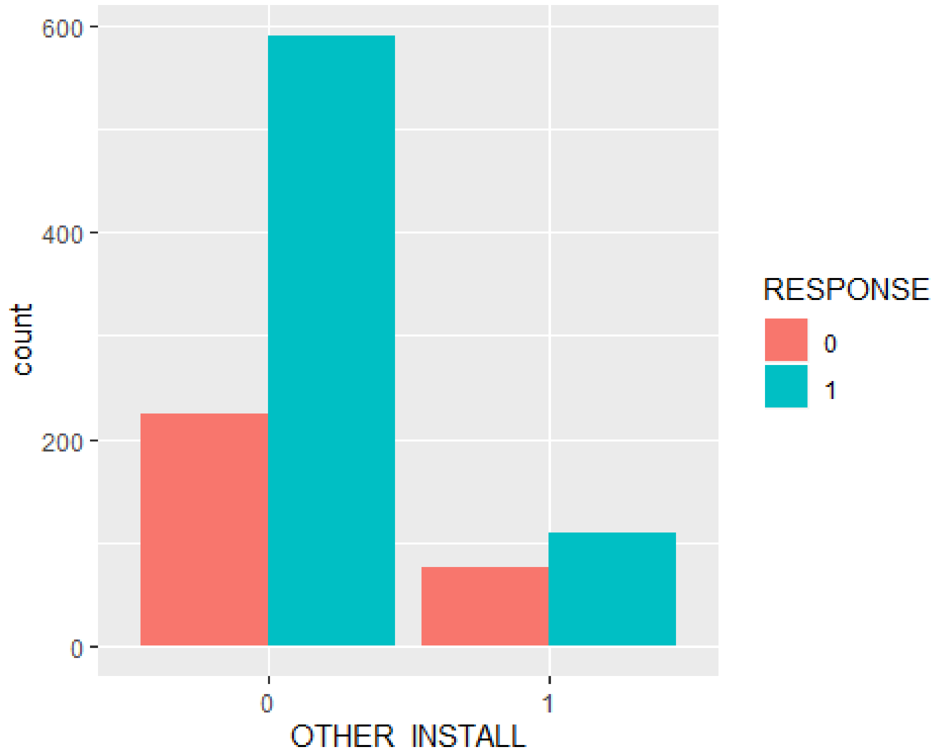

```
## huge variation observed for good responses for CHK_ACC value = 3.
definately a significant variable

#credit history
ggplot(GermanCred, aes(HISTORY,..count..))+geom_bar(aes(fill=RESPONSE),
position="dodge")
```



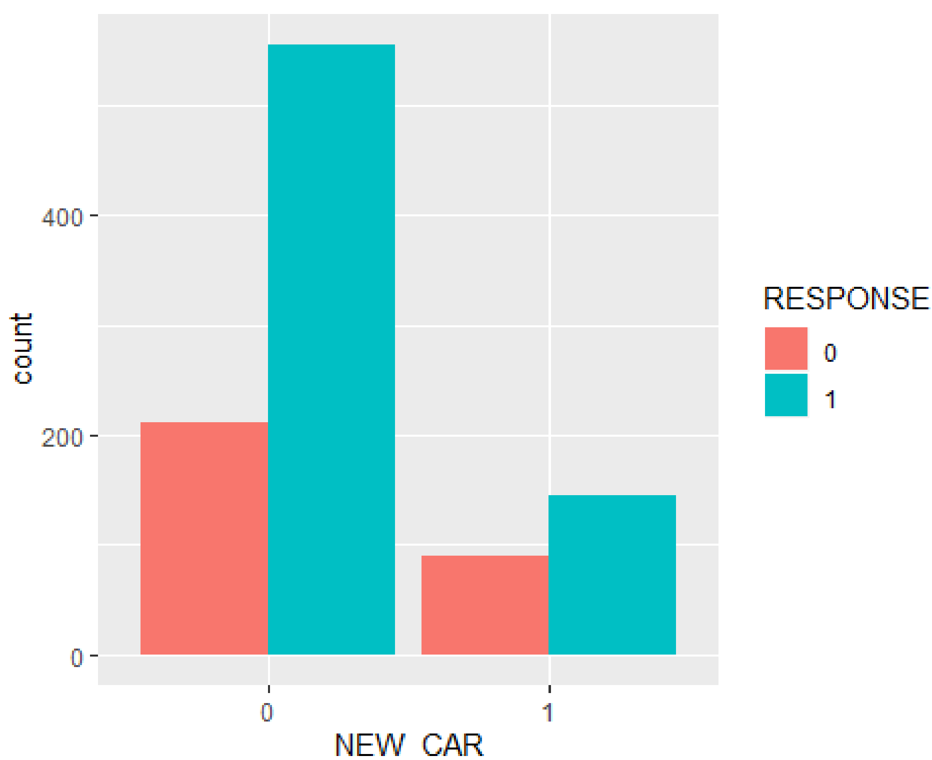

```
#variation observed at history = 2,4. good responses outnumbers bad responses
at these two levels....definately a significant variable.

#saving account
ggplot(GermanCred, aes(SAV_ACCT,..count..))+geom_bar(aes(fill=RESPONSE),
position="dodge")
```



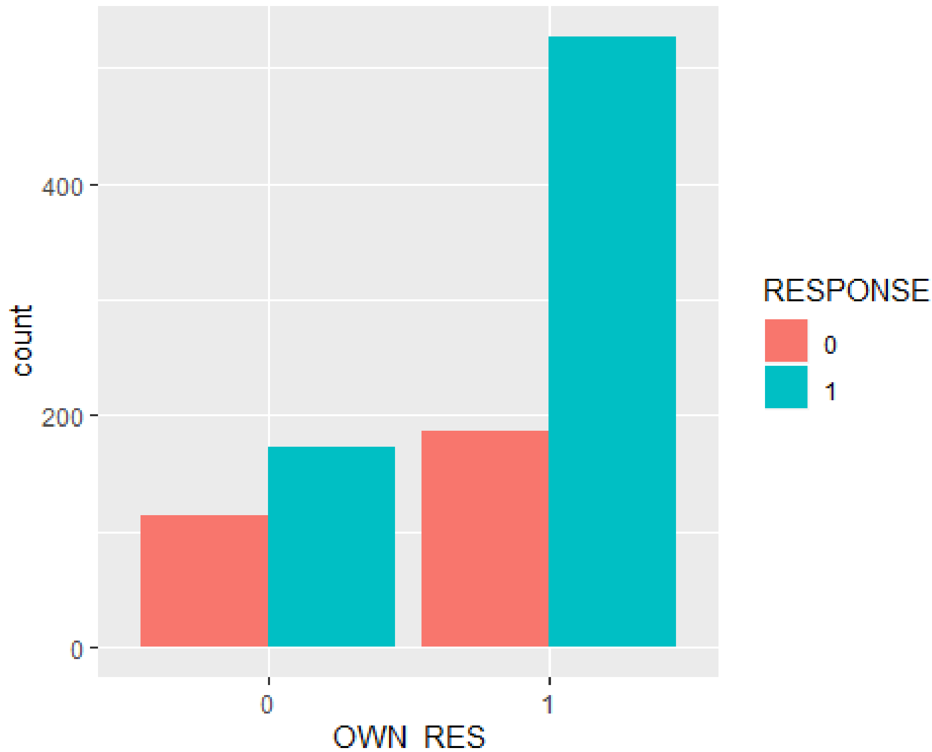

```
#good responses outnumbers bad responses for Sav_ACC level zero. We can
consider this for significance level.


#other install
ggplot(GermanCred, aes(OTHER_INSTALL,..count..))+geom_bar(aes(fill=RESPONSE),
position="dodge")
```




```r
#New_car
ggplot(GermanCred, aes(NEW_CAR,..count..))+geom_bar(aes(fill=RESPONSE),
position="dodge")
```



```r
##people with no car has good response compared to people with car.
#own residence
ggplot(GermanCred, aes(OWN_RES,..count..))+geom_bar(aes(fill=RESPONSE),
position="dodge")
```




```r
#People wix`th own residence are more efficient at paying loans than no
house.
#building Logistics regression model

#building linear regression model on Numerical variables.
attach(GermanCred)

library(caTools)
# splitting data in test and train
sample.split(RESPONSE,SplitRatio = .7) -> split_index
train_d <- subset(GermanCred, split_index == T)
test_d <- subset(GermanCred, split_index == F)

#Building logistics regression to find out impactful independent variable

#install Rcmdr for logistics regression
library(Rcmdr)

## Loading required package: splines

## Loading required package: RcmdrMisc
```

```
## Loading required package: car

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:psych':
##
##     logit

## The following object is masked from 'package:dplyr':
##
##     recode

## Loading required package: sandwich

## Loading required package: effects

## Registered S3 methods overwritten by 'lme4':
##   method                          from
##   cooks.distance.influence.merMod car
##   influence.merMod                car
##   dfbeta.influence.merMod         car
##   dfbetas.influence.merMod        car

## lattice theme set by effectsTheme()
## See ?effectsTheme for details.

## The Commander GUI is launched only in interactive sessions

##
## Attaching package: 'Rcmdr'

## The following object is masked from 'package:base':
##
##     errorCondition

# logistics regression model on numeric data variables with train data
attach(GermanCred)

## The following objects are masked from GermanCred (pos = 11):
##
##     AGE, AMOUNT, CHK_ACCT, CO-APPLICANT, DURATION, EDUCATION,
##     EMPLOYMENT, FOREIGN, FURNITURE, GUARANTOR, HISTORY,
##     INSTALL_RATE, JOB, MALE_DIV, MALE_MAR_or_WID, MALE_SINGLE,
##     NEW_CAR, NUM_CREDITS, NUM_DEPENDENTS, OBS#, OTHER_INSTALL,
##     OWN_RES, PRESENT_RESIDENT, PROP_UNKN_NONE, RADIO/TV,
##     REAL_ESTATE, RENT, RESPONSE, RETRAINING, SAV_ACCT, TELEPHONE,
##     USED_CAR

# logistics regression model on numeric data variables with German data
```

```
  #full = glm(RESPONSE~., family = binomial(logit), data = GermanCred)
  #final = stepwise(full, direction = "forward", criterion = "BIC", data =
GermanCred)
#summary(final)

#BIC Method selects variables like:
#Coefficients:
#        Estimate Std. Error z value Pr(>|z|)
#(Intercept)         0.49704    0.23735   2.094 0.036249 *
 #       CHK_ACCT[T.1]       0.49078    0.18708   2.623 0.008708 **
  #      CHK_ACCT[T.2]       1.19147    0.33869   3.518 0.000435 ***
   #     CHK_ACCT[T.3]       1.98522    0.20979   9.463  < 2e-16 ***
    #    DURATION           -0.04357    0.00658  -6.621 3.56e-11 ***
     #   USED_CAR[T.1]       1.05581    0.31729   3.328 0.000876 ***
     #   OWN_RES[T.1]        0.49442    0.17074   2.896 0.003782 **
     #   OTHER_INSTALL[T.1] -0.68440    0.19057  -3.591 0.000329 ***
      #  GUARANTOR[T.1]      1.09959    0.39365   2.793 0.005217 **
       # MALE_SINGLE[T.1]    0.48216    0.16027   3.009 0.002625 **
        #NEW_CAR[T.1]       -0.49876    0.18139  -2.750 0.005967 **

#BIC Method selects variables like: CHK_ACCT, DURATION,
INSTALL_RATE,OTHER_INSTALL


#Logistic regression through AIC variable selection
  #full = glm(RESPONSE~., family = binomial(logit), data = GermanCred)
  #final = stepwise(full, direction = "backward",data = GermanCred)
#summary(final)

#Coefficients:
 #                 Estimate Std. Error z value Pr(>|z|)
  #(Intercept)     1.602e+00  3.198e-01   5.010 5.45e-07 ***
   #CHK_ACCT1       5.290e-01  1.882e-01   2.810 0.004947 **
    #CHK_ACCT2       1.137e+00  3.403e-01   3.341 0.000836 ***
     #CHK_ACCT3       2.081e+00  2.110e-01   9.864  < 2e-16 ***
      #DURATION       -3.016e-02  8.324e-03  -3.624 0.000290 ***
       #NEW_CAR1       -5.344e-01  1.829e-01  -2.921 0.003486 **
        #USED_CAR1       9.819e-01  3.207e-01   3.062 0.002199 **
         #AMOUNT         -1.063e-04  3.821e-05  -2.781 0.005419 **
          #INSTALL_RATE   -2.661e-01  7.964e-02  -3.341 0.000834 ***
           #MALE_SINGLE1    6.398e-01  1.645e-01   3.889 0.000101 ***
            #GUARANTOR1      1.046e+00  3.902e-01   2.681 0.007335 **
             #OTHER_INSTALL1 -6.623e-01  1.915e-01  -3.458 0.000544 ***
              #AIC Method selects variables like: CHK_ACCT, DURATION,
USED_CAR,OTHER_INSTALL


#building Logistics regression model
full =
```

```r
glm(RESPONSE~CHK_ACCT+AMOUNT+HISTORY+INSTALL_RATE+DURATION+OTHER_INSTALL,
family = binomial, data = train_d)
#summary(full)



#using BIC method to select variable in LR
#final = stepwise(full, direction = "forward", criterion = "BIC", data =
train_d)
#summary(final)

# Logistics regression suggest that variable AGE and Duration are of most
significance among others.
p <- predict(full, type="response", test_d)
p.survive <- round(p)
#changing class of variable p.survive
p.survive <- as.factor(p.survive)

library(caret)

## Loading required package: lattice

confusionMatrix(p.survive, test_d$RESPONSE)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  31  15
##          1  59 195
##
##                Accuracy : 0.7533
##                  95% CI : (0.7005, 0.8011)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.02388
##
##                   Kappa : 0.3173
##
##  Mcnemar's Test P-Value : 5.773e-07
##
##             Sensitivity : 0.3444
##             Specificity : 0.9286
##          Pos Pred Value : 0.6739
##          Neg Pred Value : 0.7677
##              Prevalence : 0.3000
##          Detection Rate : 0.1033
##    Detection Prevalence : 0.1533
##       Balanced Accuracy : 0.6365
##
```

```
##        'Positive' Class : 0
##

#We also checked the accuracy of the logistice regression model on this data
for reference and made confusion matrix for the same.we used the same set of
variable which we got from BIC and AIC. This model fitted with 24% OOB error.
```

2.b)Divide the data randomly into training (60%) and test (40%) partitions, and develop the “best” classi???cation tree and random forest models to predict Good and Bad customers. Try to ???nd the best values of the parameters needed in your models. In your decision tree model, what are the best nodes for classifying “Good” applicants? Output rules corresponding to these. Please explain why you chose these nodes.

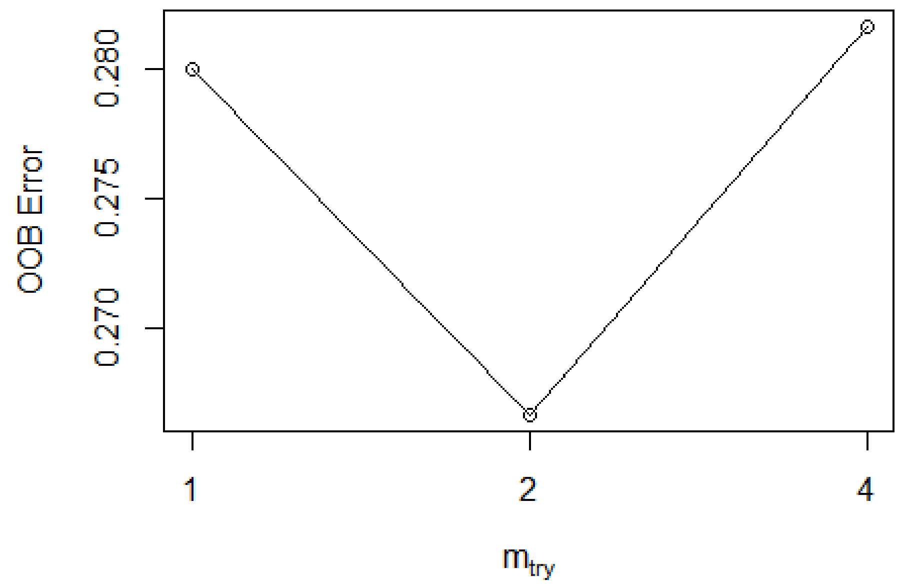
```
set.seed(2)
sample=sample(1:nrow(GermanCred),floor(nrow(GermanCred)*0.6))
train <-GermanCred[sample, ]
test <-GermanCred[-sample,]
nrow(train)

## [1] 600

nrow(test)

## [1] 400

#install.packages("randomForest")
library(randomForest)

## 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:psych':
##
##     outlier

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

#attributes found significant from logistic regression model, performed on
this data earlier.
#CHK_ACCT+AMOUNT+DURATION+INSTALL_RATE+OTHER_INSTALL+SAV_ACCT+HISTORY
model1 <-
```

```r
randomForest(RESPONSE~CHK_ACCT+AMOUNT+DURATION+INSTALL_RATE+OTHER_INSTALL+SAV
_ACCT+HISTORY+OWN_RES,data=train,ntree=500,mtry=2,importance=TRUE,proximity=T
RUE)
#model1

library(caret)

#Testing on Training dataset
#Predictions on training dataset
Predict_Train<-predict(model1,train,type = "class")

#Confusion matrix for evaluating the model on training dataset
confusionMatrix(Predict_Train,train$RESPONSE)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 158   1
##          1  18 423
##
##                Accuracy : 0.9683
##                  95% CI : (0.951, 0.9808)
##     No Information Rate : 0.7067
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9214
##
##  Mcnemar's Test P-Value : 0.0002419
##
##             Sensitivity : 0.8977
##             Specificity : 0.9976
##          Pos Pred Value : 0.9937
##          Neg Pred Value : 0.9592
##              Prevalence : 0.2933
##          Detection Rate : 0.2633
##    Detection Prevalence : 0.2650
##       Balanced Accuracy : 0.9477
##
##        'Positive' Class : 0
##

#The accuracy of Random Forest on Training data is 96.83%


#Testing on Testing dataset
#Predictions on training dataset
Predict_Test<-predict(model1,test,type = "class")
#Confusion matrix for evaluating the model on Testing dataset
confusionMatrix(Predict_Test,test$RESPONSE)
```

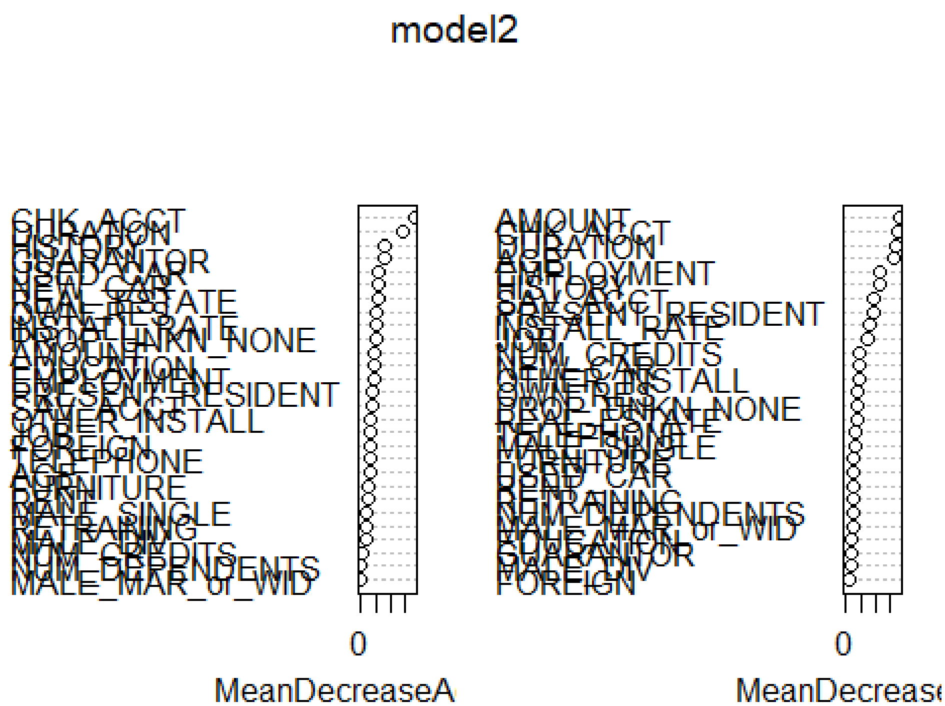
```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   0   1
##          0  53  31
##          1  71 245
##                                           
##                Accuracy : 0.745           
##                  95% CI : (0.6993, 0.787) 
##     No Information Rate : 0.69            
##     P-Value [Acc > NIR] : 0.0091786       
##                                           
##                   Kappa : 0.3458          
##                                           
##  Mcnemar's Test P-Value : 0.0001127       
##                                           
##             Sensitivity : 0.4274          
##             Specificity : 0.8877          
##          Pos Pred Value : 0.6310          
##          Neg Pred Value : 0.7753          
##              Prevalence : 0.3100          
##          Detection Rate : 0.1325          
##    Detection Prevalence : 0.2100          
##       Balanced Accuracy : 0.6576          
##                                           
##        'Positive' Class : 0               
## 
```

```
#The accuracy of Random Forest on Training data is 74.5%

#used to find optimal value of Mtry
t <-
tuneRF(x=train[,c("CHK_ACCT","AMOUNT","DURATION","INSTALL_RATE","OTHER_INSTAL
L","SAV_ACCT","HISTORY","OWN_RES")],y=train$RESPONSE,
             stepFactor = 0.5,
             plot=TRUE,
           trace=TRUE,
             ntreetry=300,
             doBest = TRUE,
           improve=0.05)
```

```
## mtry = 2  OOB error = 26.67% 
## Searching left ...
## mtry = 4     OOB error = 28.17% 
## -0.05625 0.05 
## Searching right ...
## mtry = 1     OOB error = 28% 
## -0.05 0.05
```




```
#The mtry=2 has minimum OOB error. So selecting mtry=2 as tuning parameter

#Checking Importance of Attributes Created RF model "model2" with all
attributes.
train_New <- train[,c(-1,-8,-18)]
model2 <-
randomForest(RESPONSE~.,data=train_New,ntree=500,mtry=2,importance=TRUE,proxi
mity=TRUE)

varImpPlot(model2,sort=T) #Graph of Values of important variables
```




```
importance(model2) #values as per importance
```

```
##                          0          1 MeanDecreaseAccuracy
## CHK_ACCT        18.1997015 11.3915001           18.2024988
## DURATION         8.6631241 11.9583987           14.2692674
## HISTORY          5.9123404  5.7073808            7.9941132
## NEW_CAR          6.8194617  1.9515051            5.8612132
## USED_CAR         5.4091472  3.5947637            5.9151956
## FURNITURE        1.2727072  2.2053857            2.7455422
## EDUCATION        3.1808118  3.5564387            4.4458788
## RETRAINING       0.9634625  1.1859700            1.6522583
## AMOUNT           1.3336485  4.9064268            4.8643973
## SAV_ACCT         4.9775443  1.3644958            3.7853011
## EMPLOYMENT       3.1905026  2.8378205            4.3274597
## INSTALL_RATE     3.8443252  3.9266768            5.3548108
## MALE_DIV         1.1088791  1.1495705            1.6087860
## MALE_SINGLE      1.2661966  1.2747402            1.8222525
## MALE_MAR_or_WID -1.5844231  1.2951532            0.1302151
## GUARANTOR        4.1458715  6.8438529            7.6383978
## PRESENT_RESIDENT 6.4163508  0.7433083            4.1698462
## REAL_ESTATE      4.3882967  3.9591669            5.7272668
## PROP_UNKN_NONE   3.4490437  3.2476812            5.1337379
## AGE              3.5582221  1.3120353            3.0273552
## OTHER_INSTALL    0.3862409  4.1556892            3.4820177
## RENT             0.7705904  2.6247394            2.6627236
## OWN_RES          3.1146844  4.0878792            5.4518207
## NUM_CREDITS      2.3266122 -1.0026551            0.5632586
```

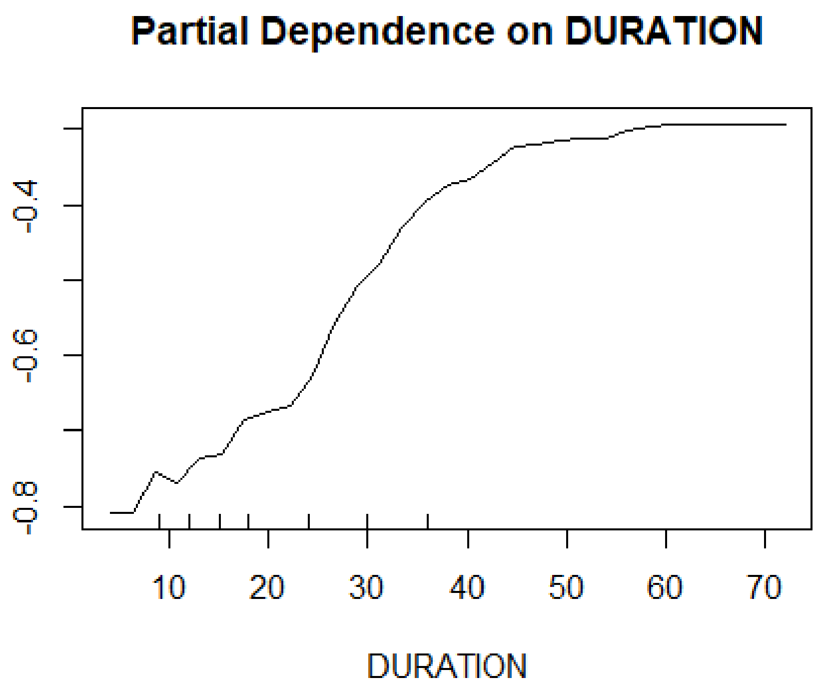
```
## JOB                1.8247024  2.7957782            3.2782401
## NUM_DEPENDENTS   -0.1963926  0.3345488            0.1558448
## TELEPHONE         4.3425851  0.5193297            3.0837086
## FOREIGN           2.2083229  2.6740876            3.2474231
##                  MeanDecreaseGini
## CHK_ACCT                17.698371
## DURATION                16.955990
## HISTORY                 10.959582
## NEW_CAR                  4.219514
## USED_CAR                 2.535322
## FURNITURE                2.664357
## EDUCATION                1.981568
## RETRAINING               2.404004
## AMOUNT                  18.235969
## SAV_ACCT                 9.517223
## EMPLOYMENT              11.619938
## INSTALL_RATE             7.944044
## MALE_DIV                 1.815890
## MALE_SINGLE              3.371825
## MALE_MAR_or_WID          2.154219
## GUARANTOR                1.944446
## PRESENT_RESIDENT         9.500363
## REAL_ESTATE              3.840704
## PROP_UNKN_NONE           3.936886
## AGE                     15.970416
## OTHER_INSTALL            4.189318
## RENT                     2.463483
## OWN_RES                  4.035579
## NUM_CREDITS              4.403636
## JOB                      7.481976
## NUM_DEPENDENTS           2.258755
## TELEPHONE                3.447204
## FOREIGN                  1.397987
```

```
#Since meanDecreaseAccuracy is highest for variables CHECKING_Acount,
Duration,History,Guarantor,Used car, New_car, Own_residence these are more
important vattributes as if these variables are removed decrease in accuracy
will be highest.
#Since meanDecreaseGini is highest for variables Amount,CHECKING_Acount,
Duration,Age,Employment,Savg_Act these are more important vattributes as if
these variables are removed the terminal nodes will not be pure.

#Which predictor variables are actually used in randomForest
varUsed(model2)
```

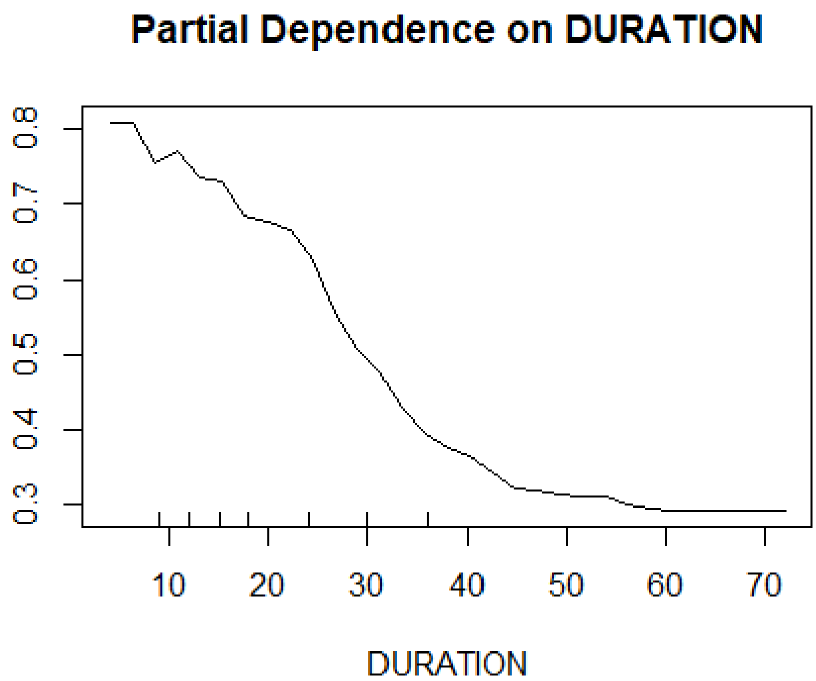
```
##  [1] 2962 3594 2790 1555 1043 1382  799 1138 4155 2672 3172 2855  851 1740
## [15] 1137  802 2933 1522 1275 3867 1661 1159 1456 2175 2550 1302 1779  744
```

```
#This gives the importance of attributes in making RandomForest model2.
Higher the value more importance is the attribute.
```

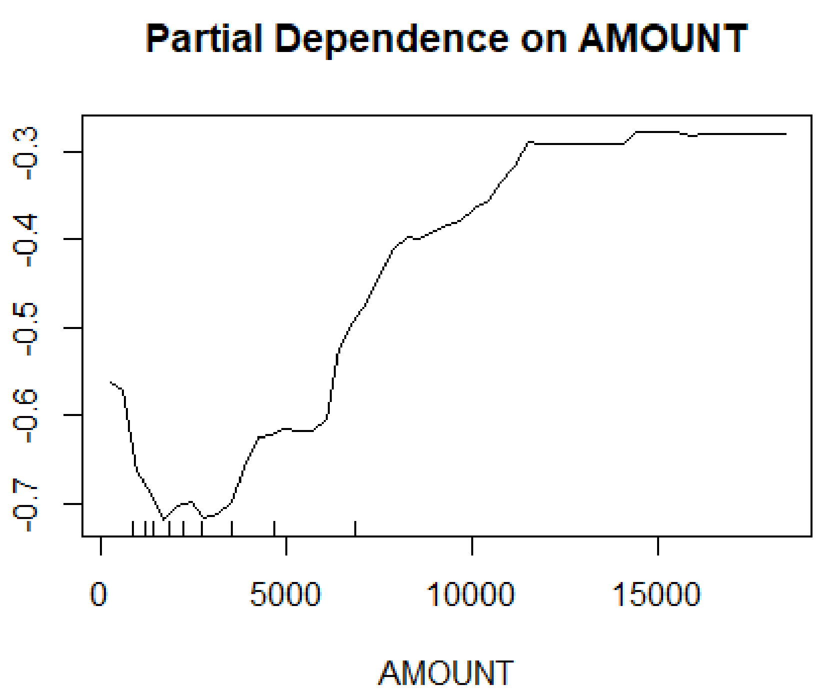
```r
#Partial dependence plot
#How the value of Response depend on Numerical values like shown below.
train_New <-as.data.frame(train_New)
#This graph shows that the probability of getting class zero increases with
increase in Duration.
partialPlot(model2,train_New,DURATION,"0")
```



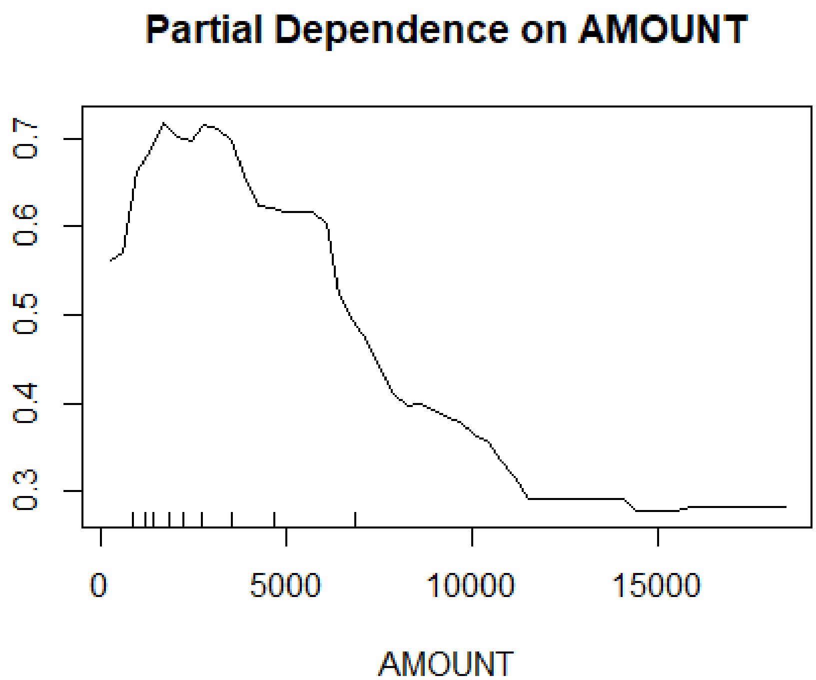

```r
#This graph shows that the probability of getting class One decreases with
increase in Duration.
partialPlot(model2,train_New,DURATION,"1")
```



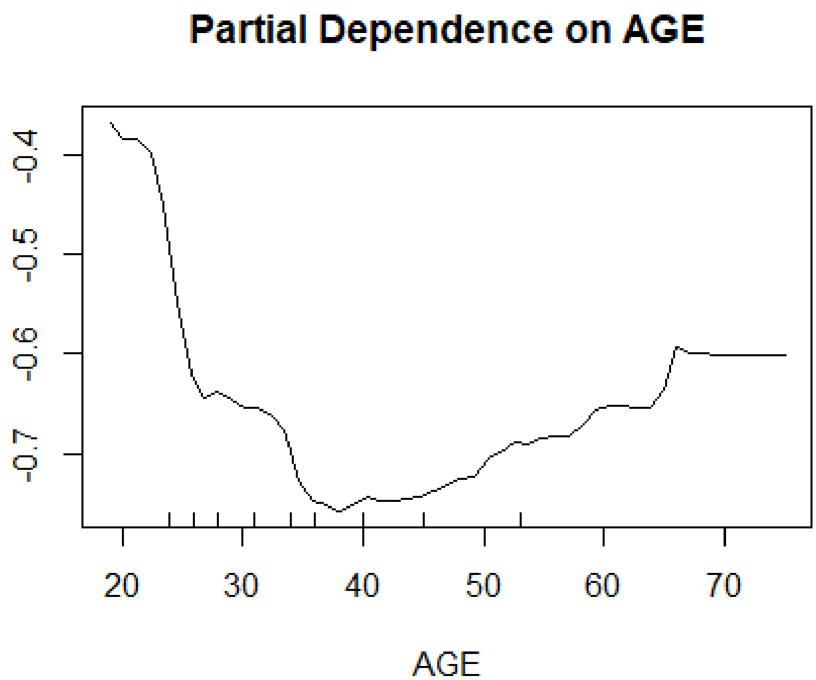

```
#This graph shows that the probability of getting class zero increases with
increase in Amount after the amount of 5000.
partialPlot(model2,train_New,AMOUNT,"0")
```



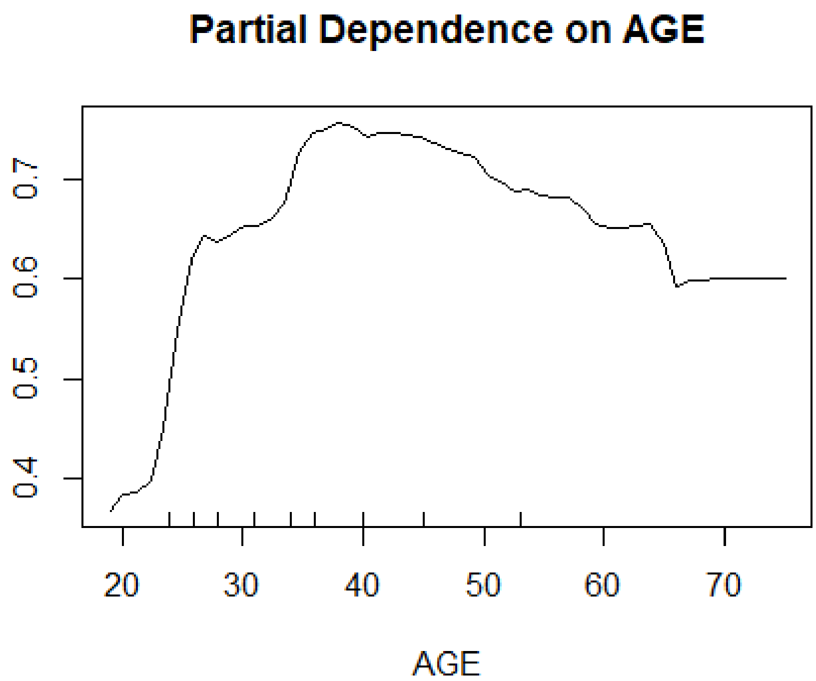
```r
#This graph shows that the probability of getting class one increass till amount=5000 and decreases with increase in Amount after the amount of 5000.
partialPlot(model2,train_New,AMOUNT,"1")
```



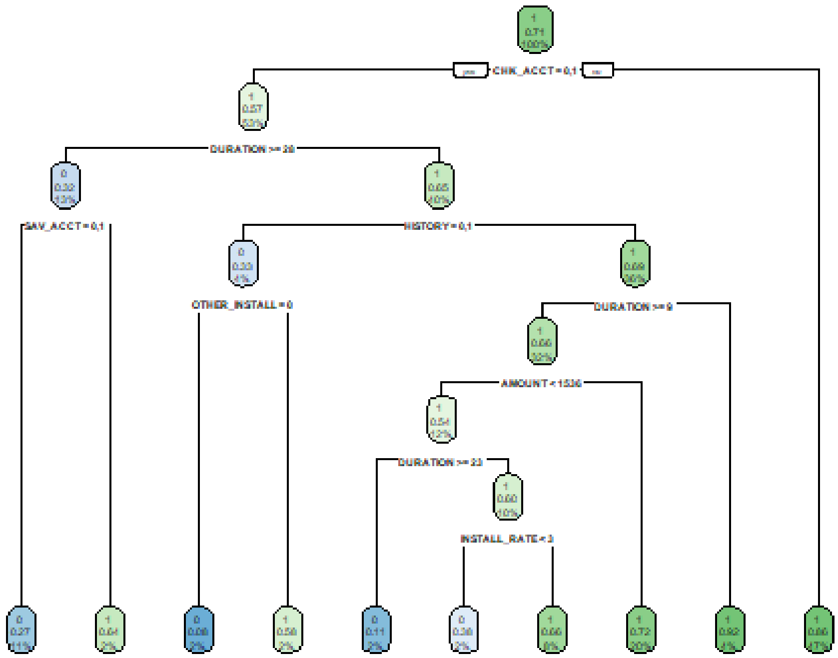

```r
#This graph shows that the probability of getting class zero decreses till Age=40.So the people who are less than 40 are  good creditos and increases with increase in Age after the age of 40.So people who are above age 40 are not good creditors as probability of class zero increases.
partialPlot(model2,train_New,AGE,"0")
```




```
#This graph shows that the probability of getting class one increases till
Age=40.So the people who are less than 40 are  good creditos and decreases
with increase in Age after the age of 40.So people who are above age 40 are
not good creditors as probability of class one decreases.
partialPlot(model2,train_New,AGE,"1")
```




```r
#Extracting single tree in RF
 #getTree(model2,1,labelVar = TRUE)
  #1st RF tree. -1 indicates its terminal node and has prediction non-NA
value.


#Making Decision tree
library(rpart)
library(rpart.plot)
model_tree <-
rpart(RESPONSE~CHK_ACCT+AMOUNT+DURATION+INSTALL_RATE+OTHER_INSTALL+SAV_ACCT+H
ISTORY+OWN_RES,data=train,method="class")
#summary(model_tree)

##Plotting decision tree
rpart.plot(model_tree)
```



```
#The support for predicting class One is happening when we are splitting on
attributes like Checking_Act, Duration and Own residence. Support and
confidence for same are:
  #Checking_Act : support=44%, confidence=0.86.Also support=56%,
confidence=0.57
    #Duration : support=46% confidence=0.51
       #Own residence: support=28%, confidence=0.59

#Predicting on train data
Predict_Train_tree <- predict(model_tree,train,type ="class")

#Confusion Matrix for evaluating the model on training dataset
#The accuracy of decision tree model model_tree is 79%.
confusionMatrix(Predict_Train_tree,train$RESPONSE)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  75  25
##          1 101 399
##
##                Accuracy : 0.79
##                  95% CI : (0.7552, 0.8219)
##     No Information Rate : 0.7067
##     P-Value [Acc > NIR] : 2.357e-06
##
```

```
##                   Kappa : 0.4202
##
##  Mcnemar's Test P-Value : 2.365e-11
##
##             Sensitivity : 0.4261
##             Specificity : 0.9410
##          Pos Pred Value : 0.7500
##          Neg Pred Value : 0.7980
##              Prevalence : 0.2933
##          Detection Rate : 0.1250
##    Detection Prevalence : 0.1667
##       Balanced Accuracy : 0.6836
##
##        'Positive' Class : 0
##
```

```
#Test data
#Predicting on test data
Predict_Test_tree <- predict(model_tree,test,type ="class")

#Confusion Matrix for evaluating the model on training dataset
#The accuracy of decision tree model model_tree is 73.5%.
confusionMatrix(Predict_Test_tree,test$RESPONSE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  45  27
##          1  79 249
##
##                Accuracy : 0.735
##                  95% CI : (0.6889, 0.7776)
##     No Information Rate : 0.69
##     P-Value [Acc > NIR] : 0.02797
##
##                   Kappa : 0.2997
##
##  Mcnemar's Test P-Value : 7.287e-07
##
##             Sensitivity : 0.3629
##             Specificity : 0.9022
##          Pos Pred Value : 0.6250
##          Neg Pred Value : 0.7591
##              Prevalence : 0.3100
##          Detection Rate : 0.1125
##    Detection Prevalence : 0.1800
##       Balanced Accuracy : 0.6325
##
```

```
##        'Positive' Class : 0
##

#Comparing Decision tree and Random Forest
#The accuracy of the training data on Random Forest is 95.83% and accuracy of training data on Decision Tree is 80.5%
#The accuracy of the testing data on Random Forest is 73.5% and accuracy of testing data on Decision Tree is 70.5%
```

2.c) Which model is a better model? Why?

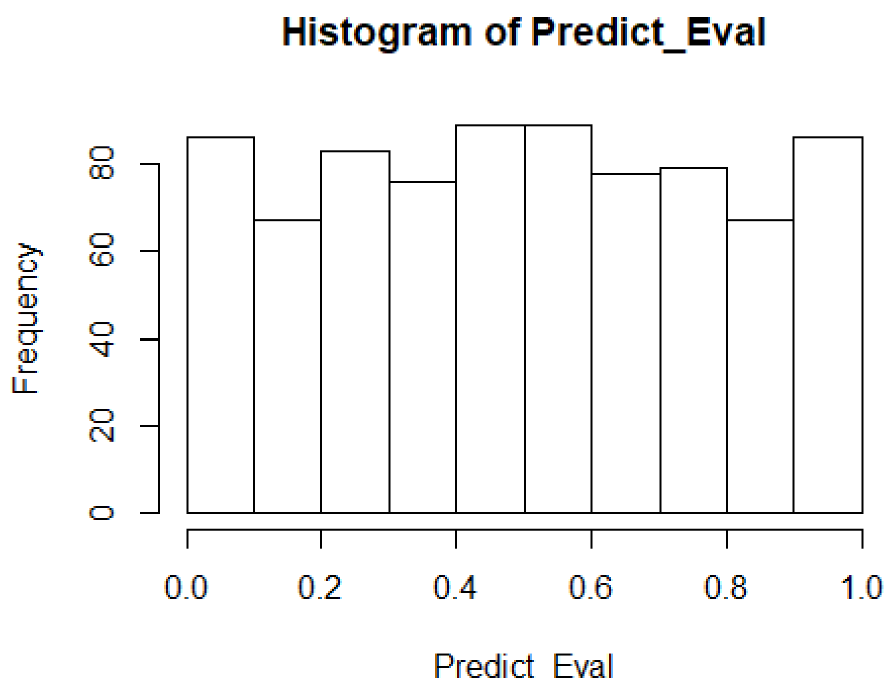
```
tab_RF<- table(Predict_Test,test$RESPONSE)
#Profit for Random Forest Model
Profit_RF <- -500*tab_RF[2,1]+100*tab_RF[2,2]
Profit_RF

## [1] -11000

tab_tree<- table(Predict_Test_tree,test$RESPONSE)
#Profit for Decision Tree Model
Profit_tree <- -500*tab_tree[2,1]+100*tab_tree[2,2]
Profit_tree

## [1] -14600

#Comparing Decision tree and Random Forest
  #The profit for the Random Forest model is -10900 DM whereas the profit for the Decision tree model is -14600.
    #As profit for the Random Forest model is more than the Decision tree we will choose Random Forest model.
```

2.d) The classes returned by your models are based on the cutooff point of 0.5. Can you improve the performance your model by changing this cutooff. Explain how you approach this.

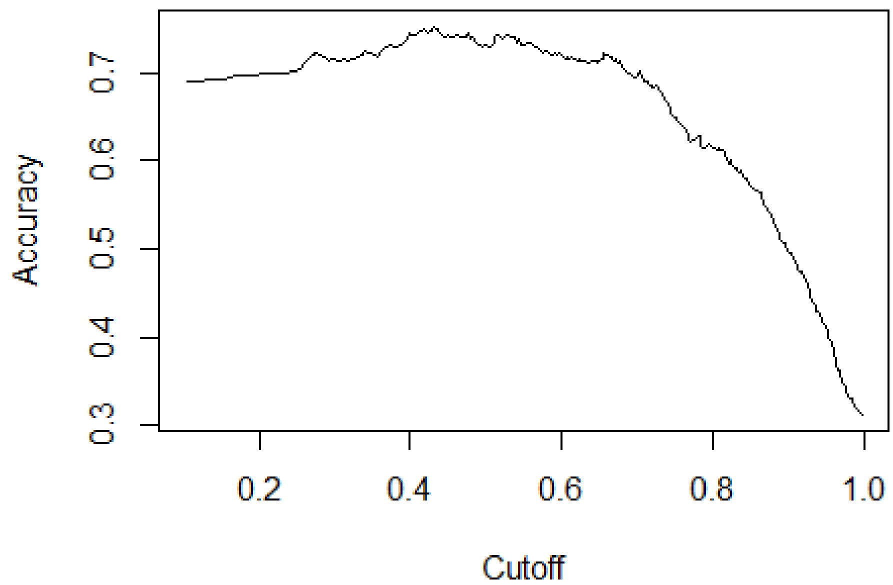
```
library(randomForest)
model3_RF <-
randomForest(RESPONSE~CHK_ACCT+AMOUNT+DURATION+OTHER_INSTALL+SAV_ACCT+HISTORY+OWN_RES+EMPLOYMENT+AGE,data=train,ntree=500,mtry=2,importance=TRUE,proximity=TRUE)


#Testing on Training dataset
#Predictions on training dataset
Predict_Train_Model3_RF<-predict(model3_RF,train,type = "class")

tab <-table(Predict_Train_Model3_RF,train$RESPONSE)
tab

##
## Predict_Train_Model3_RF   0   1
```

```
##                         0 171   0
##                         1   5 424

#Misclassification Error
  #1-sum(diag(tab))/sum(tab)
    #The misclassification error on training dataset is just 0.833%.

#confusionMatrix(Predict_Train_Model3_RF,train$RESPONSE)
  #Accuracy of 99.17

#Testing on Testing dataset
#Predictions on Testing dataset
Predict_Test_Model3_RF<-predict(model3_RF,test,type = "class")

tab_test <-table(Predict_Test_Model3_RF,test$RESPONSE)
tab_test

##
## Predict_Test_Model3_RF   0   1
##                        0  53  37
##                        1  71 239

#Misclassification Error
1-sum(diag(tab))/sum(tab)

## [1] 0.008333333

 #The misclassification error on training dataset is just 0.833%.

#confusionMatrix(Predict_Test_Model3_RF,test$RESPONSE)
  #Accuracy of 73.25


#Model Performance Evaluation
library(ROCR)

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

Predict_Eval<-predict(model3_RF,test,type = "prob")

#Comparing predicted and actual response for 1st six rows
  #head(Predict_Eval)
    #head(train$RESPONSE)
```

```r
hist(Predict_Eval)
```

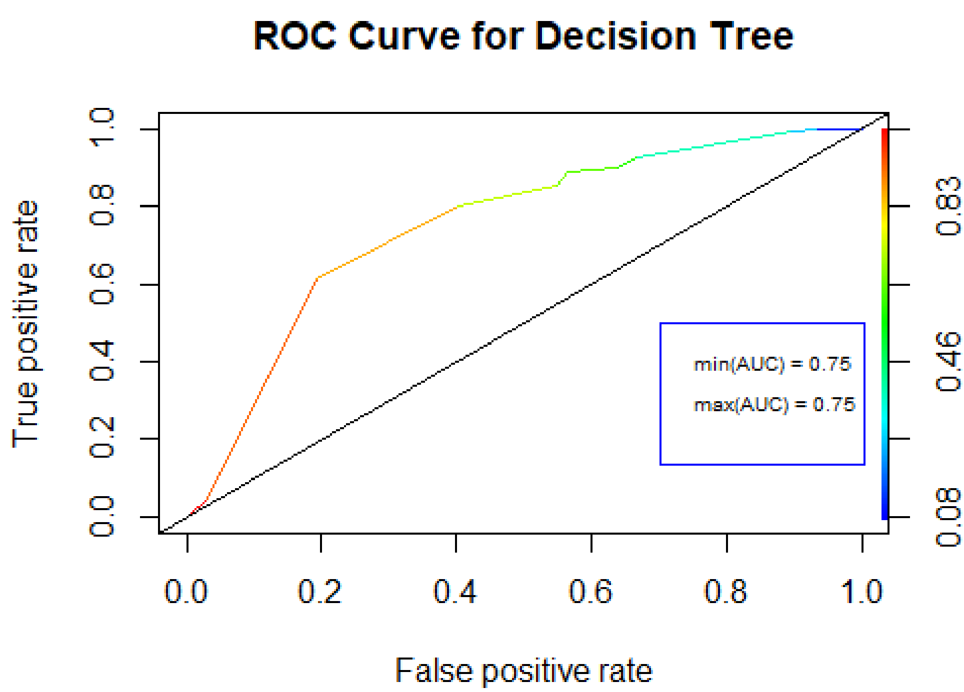



```r
#It shows the frequency of Predicted probability. So the frequency of
predicting the probability between 0.4 and 0.6 is maximum.
#Currently the model is built using the treshold of 0.5 but if we use cut_off
0.4 or 0.6 we may have another type of classification,accuracy and
misclassification will change.


Predict_Eval_New <-prediction(Predict_Eval[,2],test$RESPONSE) #Subset is
added on Predict_Eval so that only the column with class 1 is selected and
not both columns with class 1 and 0.
eval <- performance(Predict_Eval_New ,"acc") #acc stands for accuracy values
plot(eval)
```



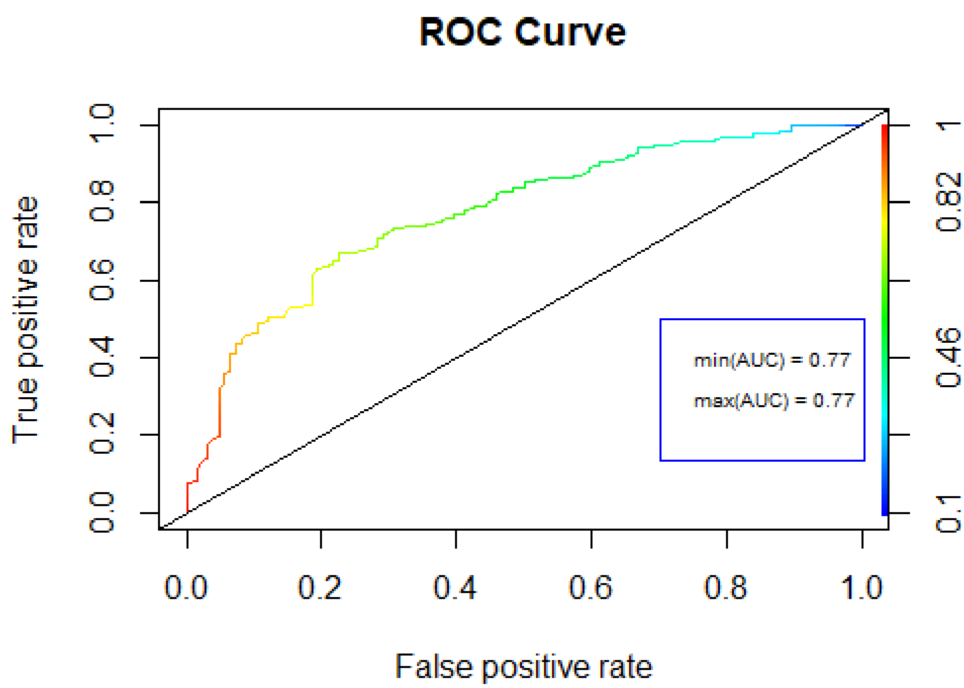

```
#It shows that the accuracy is maximum around cutoff 0.4 to 0.6 and after
treshold of 0.6 the accuracy decreases.


#Identify best cut-off value and accuracy at that value
max <-which.max(slot(eval,"y.values")[[1]])
acc <- slot(eval,"y.values")[[1]][max]
cut_off <- slot(eval,"x.values")[[1]][max]

#Here it shows that the Accuracy will be maximum at cut-off of 0.432 and
maximum accuracy is 75.25%.
#Hence the performance(Accuracy) of the model can be increased if we change
the cut-off frequency (treshold) to 0.432.
print(c(Accuracy=acc,Cutoff=cut_off))

##   Accuracy Cutoff.199
##     0.7525     0.4320

#ROC Curve for Decision Tree
Predict_Test_tree_new  <- predict(model_tree,test,type ="prob")
pred_tree <- prediction(Predict_Test_tree_new[,2],test$RESPONSE)
roc_tree <- performance(pred_tree,"tpr","fpr")
plot(roc_tree,
     colorize=T,
     main="ROC Curve for Decision Tree"
)
abline(a=0, b=1)
```

```
auc_tree <-performance(pred_tree,"auc")
auc_tree <-unlist(slot(auc_tree,"y.values"))
auc_tree

## [1] 0.7524398

#The area under the AUC curve is 75.24% which is more compared to the random
curve which has the AUC of 0.5.
#legend(0.6,0.2,auc,title="AUC",cex=1)

minauc_t <- min(round(auc_tree, digits = 2))
maxauc_t <- max(round(auc_tree, digits = 2))
minauct_t <- paste(c("min(AUC) = "), minauc_t, sep = "")
maxauct_t <- paste(c("max(AUC) = "), maxauc_t, sep = "")

legend(0.7, 0.5, c(minauct_t, maxauct_t, "\n"), border = "white", cex = 0.6,
box.col = "blue")
abline(a= 0, b=1)
```



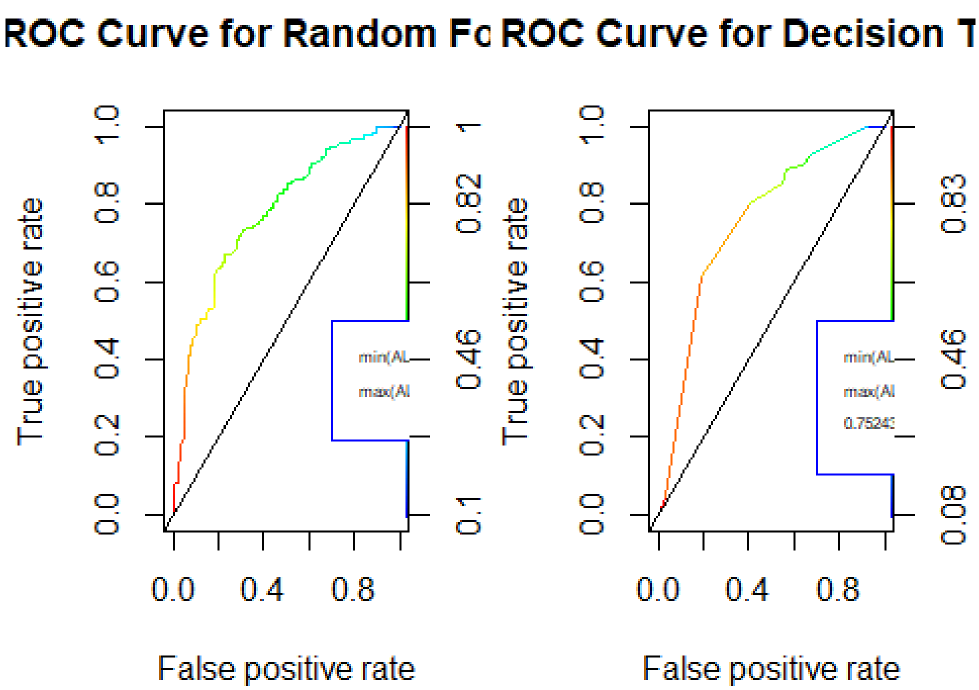

```
#ROC Curve for Random Forest
pred <- prediction(Predict_Eval[,2],test$RESPONSE)
roc <- performance(pred,"tpr","fpr")
plot(roc,
     colorize=T,
     main="ROC Curve"
     )
```

```
abline(a=0, b=1)


auc <-performance(pred,"auc")
auc <-unlist(slot(auc,"y.values"))
auc

## [1] 0.7747195

#The area under the AUC curve is 77.4% which is more compared to the random
curve which has the AUC of 0.5.
#legend(0.6,0.2,auc,title="AUC",cex=1)

minauc = min(round(auc, digits = 2))
maxauc = max(round(auc, digits = 2))
minauct = paste(c("min(AUC) = "), minauc, sep = "")
maxauct = paste(c("max(AUC) = "), maxauc, sep = "")
legend(0.7, 0.5, c(minauct, maxauct, "\n"), border = "white", cex = 0.6,
box.col = "blue")
abline(a= 0, b=1)
```




```
#Comparing the ROC curves for Random Forest and Decision tree
par(mfrow=c(1,2))
plot(roc,
     colorize=T,
     main="ROC Curve for Random Forest"
)
```

```r
abline(a=0, b=1)
legend(0.7, 0.5, c(minauct, maxauct, "\n"), border = "white", cex = 0.5,
box.col = "blue")


plot(roc_tree,
     colorize=T,
     main="ROC Curve for Decision Tree"
)
abline(a=0, b=1)
legend(0.7, 0.5, c(minauct_t, maxauct_t,auc_tree, "\n"), border = "white",
cex = 0.5, box.col = "blue")
```




```r
#Eucledian Function
Eucledian_function<- function(x,y,p) {

  d<-sqrt(( (x-0)^2  ) +  ( (y-1  )^2 ))
  index<-which(d==min(d))
  c(recall = y[[index]],specificity= 1-x[[index]] ,cutoff = p[[index]])
}

roc@alpha.values

## [[1]]
##   [1]   Inf 0.996 0.994 0.988 0.984 0.982 0.978 0.976 0.974 0.970 0.968
##  [12] 0.966 0.964 0.962 0.960 0.958 0.956 0.954 0.952 0.950 0.948 0.946
##  [23] 0.944 0.942 0.938 0.936 0.934 0.932 0.928 0.926 0.924 0.922 0.920
```

```
##  [34] 0.918 0.916 0.912 0.910 0.908 0.904 0.902 0.898 0.896 0.890 0.888
##  [45] 0.886 0.884 0.882 0.880 0.876 0.874 0.870 0.868 0.866 0.864 0.862
##  [56] 0.856 0.848 0.846 0.844 0.838 0.836 0.834 0.832 0.826 0.824 0.822
##  [67] 0.820 0.818 0.816 0.814 0.812 0.808 0.806 0.798 0.794 0.792 0.790
##  [78] 0.788 0.784 0.782 0.780 0.776 0.772 0.770 0.768 0.766 0.764 0.762
##  [89] 0.758 0.756 0.754 0.750 0.746 0.744 0.742 0.740 0.738 0.736 0.734
## [100] 0.732 0.730 0.728 0.724 0.722 0.720 0.718 0.716 0.714 0.708 0.706
## [111] 0.704 0.700 0.698 0.694 0.692 0.690 0.684 0.678 0.676 0.674 0.672
## [122] 0.670 0.668 0.666 0.662 0.660 0.656 0.654 0.648 0.646 0.644 0.642
## [133] 0.638 0.636 0.628 0.624 0.622 0.616 0.612 0.610 0.608 0.606 0.602
## [144] 0.598 0.596 0.594 0.592 0.588 0.582 0.580 0.574 0.572 0.570 0.566
## [155] 0.562 0.558 0.556 0.552 0.546 0.542 0.540 0.538 0.528 0.524 0.522
## [166] 0.518 0.516 0.512 0.510 0.506 0.504 0.500 0.496 0.492 0.488 0.484
## [177] 0.482 0.480 0.478 0.476 0.474 0.472 0.460 0.458 0.452 0.450 0.446
## [188] 0.442 0.440 0.436 0.432 0.428 0.424 0.420 0.418 0.414 0.410 0.408
## [199] 0.406 0.400 0.398 0.396 0.392 0.388 0.380 0.374 0.368 0.364 0.362
## [210] 0.358 0.356 0.352 0.346 0.340 0.338 0.336 0.330 0.326 0.316 0.314
## [221] 0.304 0.298 0.292 0.286 0.280 0.274 0.272 0.268 0.266 0.262 0.260
## [232] 0.258 0.254 0.250 0.232 0.170 0.154 0.104
```

```
mapply(Eucledian_function,roc@x.values,roc@y.values,roc@alpha.values)
```

```
##                  [,1]
## recall      0.6702899
## specificity 0.7741935
## cutoff      0.7040000
```

```
#In order to decide the best cut-off point we calculated the distance of all
points on the roc curve from (1,0) i.e Recall=1 and False Positive=0.
#We found out that the cutoff=0.156 has the least distance from the best
point, so the True positive will be maximum,false positive will be minimum
and hence profit will be maximum at this cutoff. Hence the performance of
model can be improved by changing the cut-off to 0.156.
```

2.e) Summarize your Finndings.

```
#Since the Random Forest algorithm is based on the Bootstrap Aggregation, the
Random forest make a set of decision trees with attributes at nodes such that
there is minimum correlation between the decision trees which eventually
helps in giving the different outputs and hence the variance is minimized. In
addition to it, the decision tree is based on the specific set of rules(like
Info gain or gini index) but in the random forest since the process of
splitting and root nodes finding is random which gives different outputs
without much correlation. Also, Random forest avoids overfitting. Hence the
Random Forest is preferred compared to decision trees.We have found similar
result over here as well which is backed by accuracy figures for all the
three models, however accuracy is debatable matric to determine effectiveness
of a model. This case asks for a model which minimizes cost and maximizes
profit. We have calculated profit matric for all the models and concluded
that random forest model is best of all available models to predict profit
component in terms of opportunity cost.
```

```
#Accuracy:
#Random Forest:74.5%
#Decision Tree: 70.5%
#Logistic regression: 74%
```