Preject 2 - Report

Bravo

12/11/2018

Packages

```
library(ResourceSelection)

## ResourceSelection 0.3-2   2017-02-28

library(pROC)

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

library(pscl)

## Classes and Methods for R developed in the
## Political Science Computational Laboratory
## Department of Political Science
## Stanford University
## Simon Jackman
## hurdle and zeroinfl functions by Achim Zeileis

library(ISLR)
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(ROCR)

## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'

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

library(bestglm)

## Loading required package: leaps

library(openintro)

## Please visit openintro.org for free statistics materials

##
## Attaching package: 'openintro'

## The following objects are masked from 'package:datasets':
##
##     cars, trees

library(leaps)
library(tidyr)
library(plyr)

## ------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, th
en dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

library(rpart)
library(rpart.plot)
library(car)

## Loading required package: carData

##
## Attaching package: 'car'
## The following object is masked from 'package:openintro':
##
##     densityPlot

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

Question: How to improve customer retention of bank.

data resource: Kaggle

Import the data

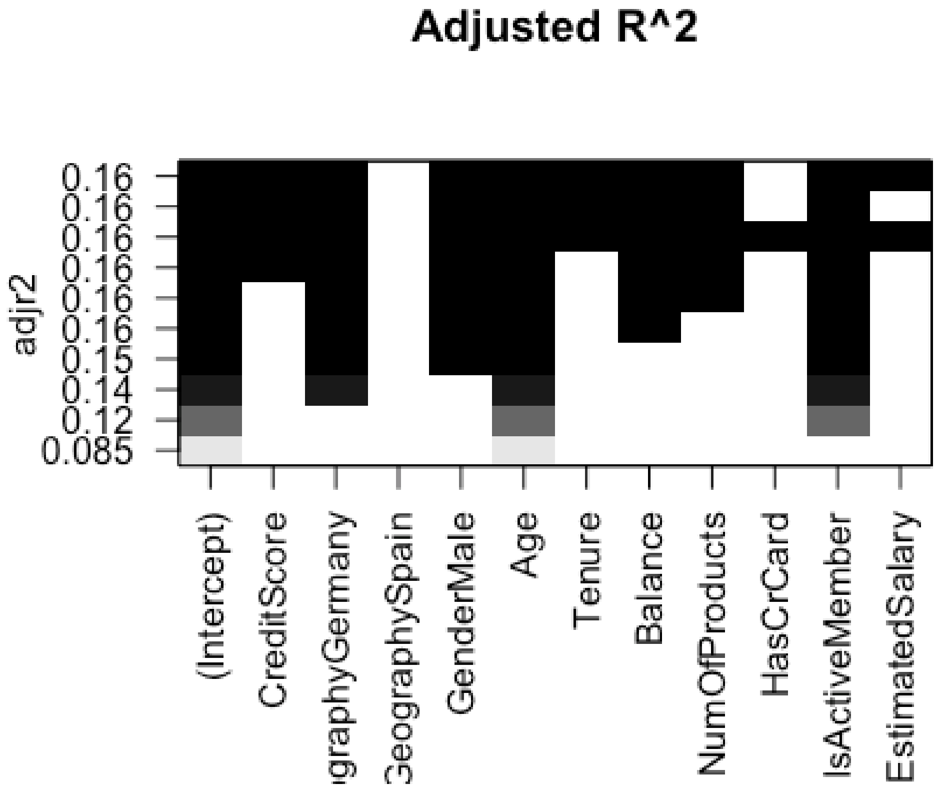
```
bank <- read.csv('Churn_Modelling.csv')
attach(bank)
#The first 3 colomns is not important:RowNumber,CustomerId,SurnameN
bank=bank[,c(-1,-2,-3)]
```

Divide the data set into test and train

```
#We are going to go with a 80% train 20% test appoarch. Not that the common a
nd space tells R to include the entire data.frame
train <- bank[1:8000, ]
test <- bank[8001:10000, ]
```

Subsetting - forward selection to search

```
reg.forward <- regsubsets(Exited~., data = train, nvmax = 10, nbest = 1, meth
od = "forward")
plot(reg.forward, scale = "adjr2", main = "Adjusted R^2")
```



```
reg.forward
```

```
## Subset selection object
## Call: regsubsets.formula(Exited ~ ., data = train, nvmax = 10, nbest = 1,
##     method = "forward")
## 11 Variables  (and intercept)
##                  Forced in Forced out
## CreditScore          FALSE      FALSE
## GeographyGermany     FALSE      FALSE
## GeographySpain       FALSE      FALSE
## GenderMale           FALSE      FALSE
## Age                  FALSE      FALSE
## Tenure               FALSE      FALSE
## Balance              FALSE      FALSE
## NumOfProducts        FALSE      FALSE
## HasCrCard            FALSE      FALSE
## IsActiveMember       FALSE      FALSE
## EstimatedSalary      FALSE      FALSE
## 1 subsets of each size up to 10
## Selection Algorithm: forward
```

```
#from the plot, we would like to select CreditScore, Geography, Gender, Age,
Balance, IsActiveMember
```

Exploratary Data Analysis

```
#About the whole data
head(bank)
```

```
##   CreditScore Geography Gender Age Tenure   Balance NumOfProducts
## 1         619    France Female  42      2      0.00             1
## 2         608     Spain Female  41      1  83807.86             1
## 3         502    France Female  42      8 159660.80             3
## 4         699    France Female  39      1      0.00             2
## 5         850     Spain Female  43      2 125510.82             1
## 6         645     Spain   Male  44      8 113755.78             2
##   HasCrCard IsActiveMember EstimatedSalary Exited
## 1         1              1       101348.88      1
## 2         0              1       112542.58      0
## 3         1              0       113931.57      1
## 4         0              0        93826.63      0
## 5         1              1        79084.10      0
## 6         1              0       149756.71      1
```

```
tail(bank, 6)
```

```
##       CreditScore Geography Gender Age Tenure   Balance NumOfProducts
## 9995          800    France Female  29      2      0.00             2
## 9996          771    France   Male  39      5      0.00             2
## 9997          516    France   Male  35     10  57369.61             1
## 9998          709    France Female  36      7      0.00             1
## 9999          772   Germany   Male  42      3  75075.31             2
## 10000         792    France Female  28      4 130142.79             1
##       HasCrCard IsActiveMember EstimatedSalary Exited
## 9995          0              0       167773.55      0
## 9996          1              0        96270.64      0
## 9997          1              1       101699.77      0
## 9998          0              1        42085.58      1
## 9999          1              0        92888.52      1
## 10000         1              0        38190.78      0
```

```
str(bank)
```

```
## 'data.frame':    10000 obs. of  11 variables:
##  $ CreditScore    : int  619 608 502 699 850 645 822 376 501 684 ...
##  $ Geography      : Factor w/ 3 levels "France","Germany",..: 1 3 1 1 3 3
1 2 1 1 ...
##  $ Gender         : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 2 2 1 2
2 ...
##  $ Age            : int  42 41 42 39 43 44 50 29 44 27 ...
##  $ Tenure         : int  2 1 8 1 2 8 7 4 4 2 ...
##  $ Balance        : num  0 83808 159661 0 125511 ...
##  $ NumOfProducts  : int  1 1 3 2 1 2 2 4 2 1 ...
##  $ HasCrCard      : int  1 0 1 0 1 1 1 1 0 1 ...
##  $ IsActiveMember : int  1 1 0 0 1 0 1 0 1 1 ...
##  $ EstimatedSalary: num  101349 112543 113932 93827 79084 ...
##  $ Exited         : int  1 0 1 0 0 1 0 1 0 0 ...
```

```
#About the dependent variable
table(Exited)
```

```
## Exited
##    0    1 
## 7963 2037
```

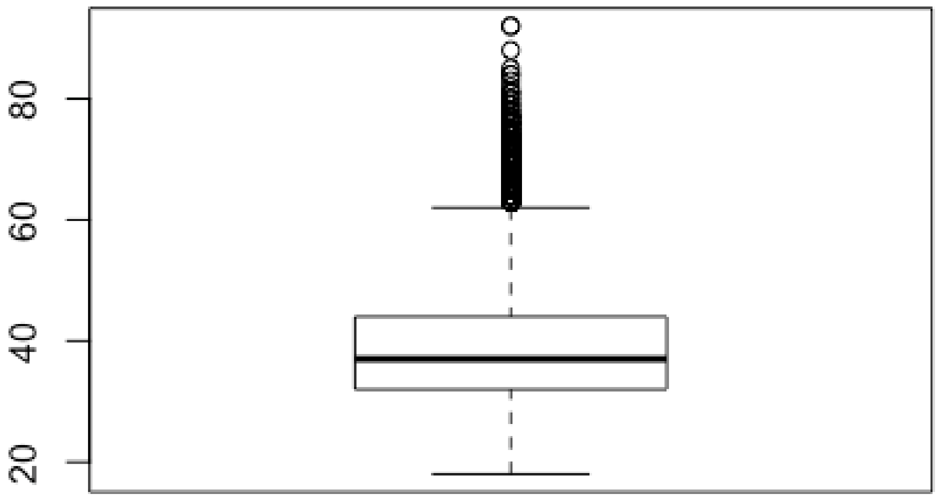
```
str(Exited)
```

```
##  int [1:10000] 1 0 1 0 0 1 0 1 0 0 ...
```

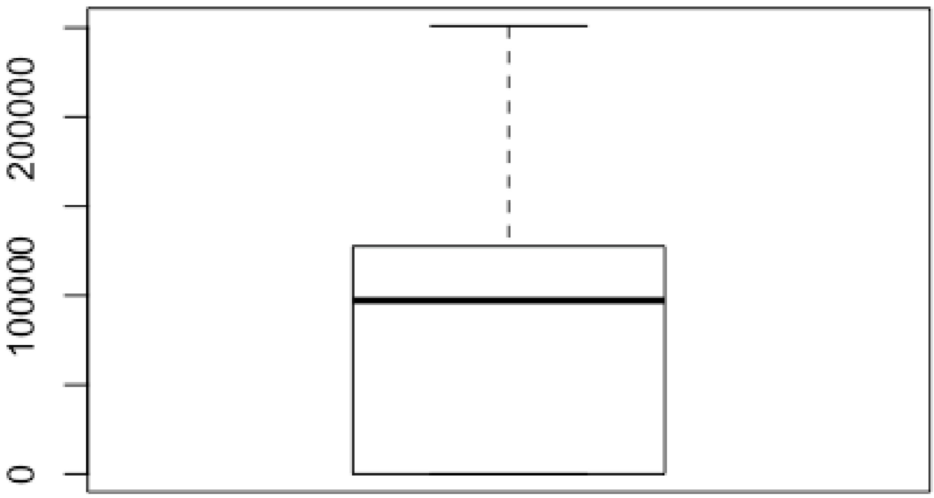
```
#Overview of all the variables
summary(bank)
```

```
##   CreditScore      Geography       Gender          Age       
##  Min.   :350.0   France :5014   Female:4543   Min.   :18.00  
##  1st Qu.:584.0   Germany:2509   Male  :5457   1st Qu.:32.00  
##  Median :652.0   Spain  :2477                 Median :37.00  
##  Mean   :650.5                                Mean   :38.92  
##  3rd Qu.:718.0                                3rd Qu.:44.00  
##  Max.   :850.0                                Max.   :92.00  
##      Tenure           Balance       NumOfProducts    HasCrCard     
##  Min.   : 0.000   Min.   :     0   Min.   :1.00   Min.   :0.0000  
##  1st Qu.: 3.000   1st Qu.:     0   1st Qu.:1.00   1st Qu.:0.0000  
##  Median : 5.000   Median : 97199   Median :1.00   Median :1.0000  
##  Mean   : 5.013   Mean   : 76486   Mean   :1.53   Mean   :0.7055  
##  3rd Qu.: 7.000   3rd Qu.:127644   3rd Qu.:2.00   3rd Qu.:1.0000  
##  Max.   :10.000   Max.   :250898   Max.   :4.00   Max.   :1.0000  
##  IsActiveMember   EstimatedSalary         Exited      
##  Min.   :0.0000   Min.   :    11.58   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.: 51002.11   1st Qu.:0.0000  
##  Median :1.0000   Median :100193.91   Median :0.0000  
##  Mean   :0.5151   Mean   :100090.24   Mean   :0.2037  
##  3rd Qu.:1.0000   3rd Qu.:149388.25   3rd Qu.:0.0000  
##  Max.   :1.0000   Max.   :199992.48   Max.   :1.0000
```

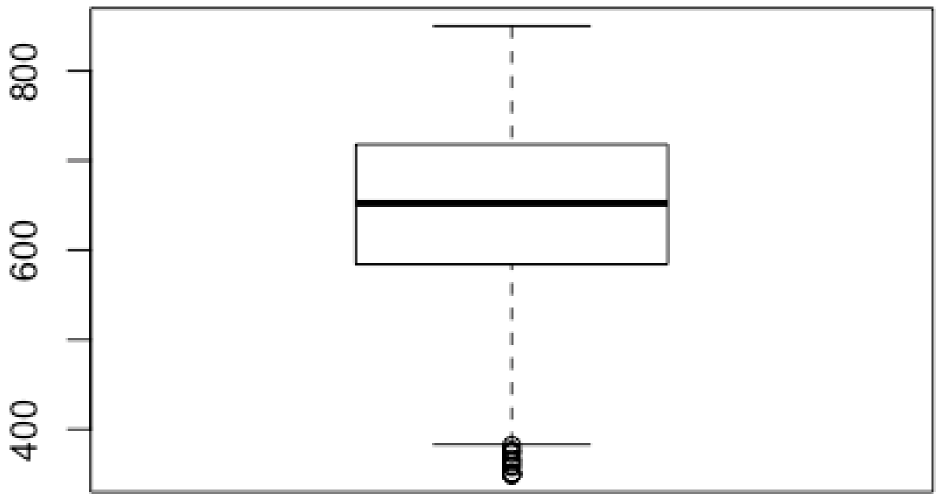
```
#boxplot of the variables
#age
box_age<- boxplot(Age)
```



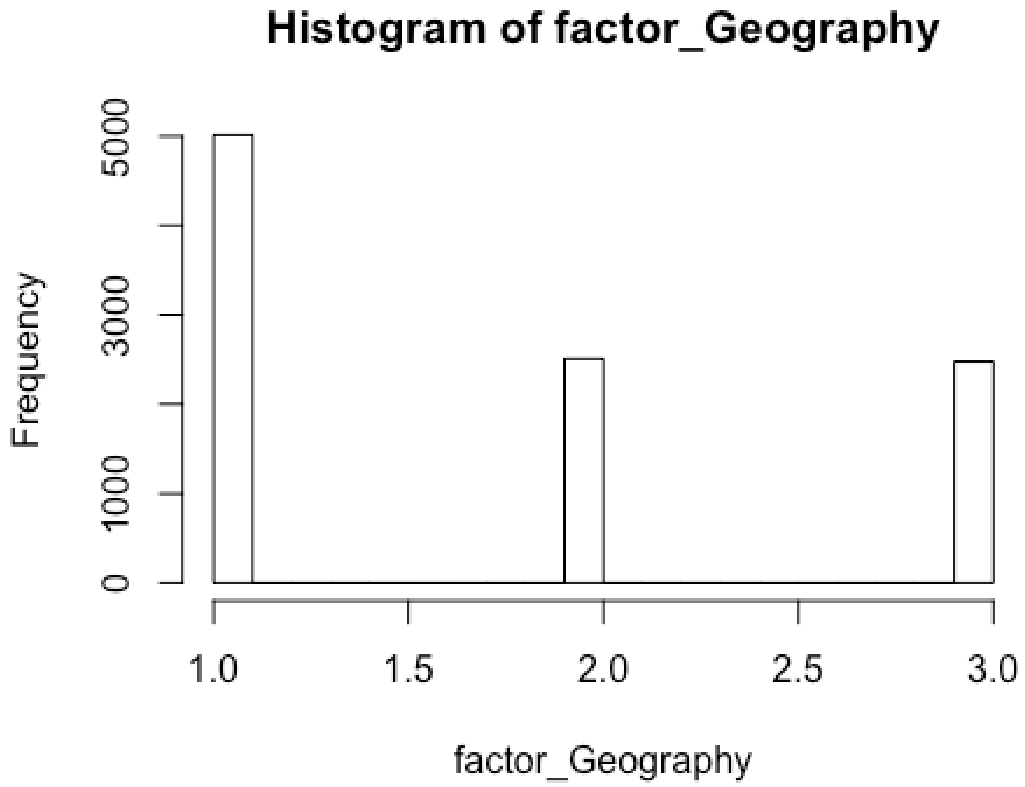
```
#balance
box_balance <- boxplot(Balance)
```



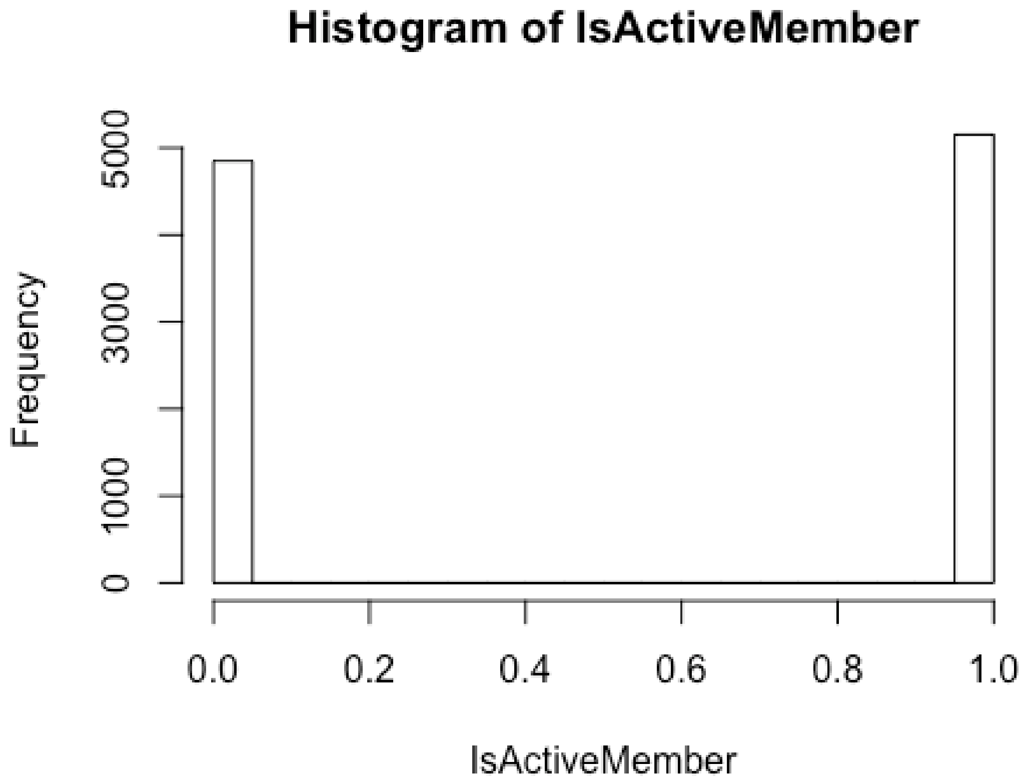
```
#credit score
box_creditscore<-boxplot(CreditScore)
```



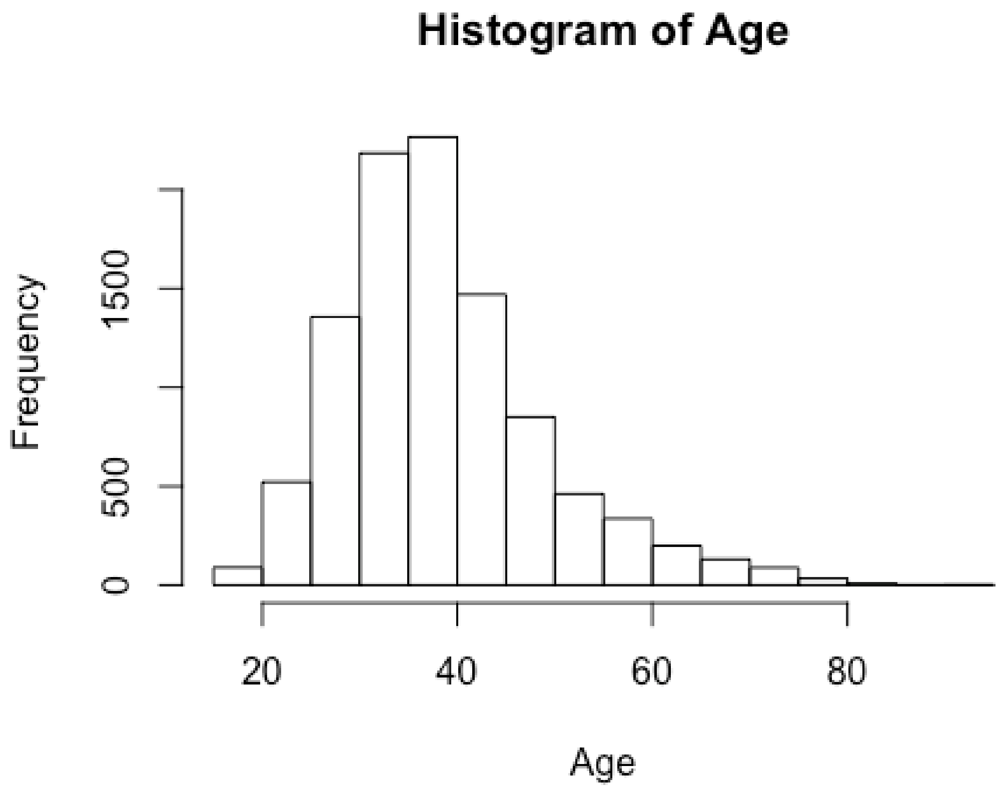
```
#hist of the numeric variable
#Geography
factor_Geography <- as.integer(Geography)
#1 is France, 2 is Germany, 3 is Spain
hist(factor_Geography)
```



```
#IsActiveMember
hist(IsActiveMember)
```


Histogram of IsActiveMember
Frequency
0
1000
3000
5000
0.0
0.2
0.4
0.6
0.8
1.0
IsActiveMember

```
#Age
hist(Age)
```



Logistic regression model

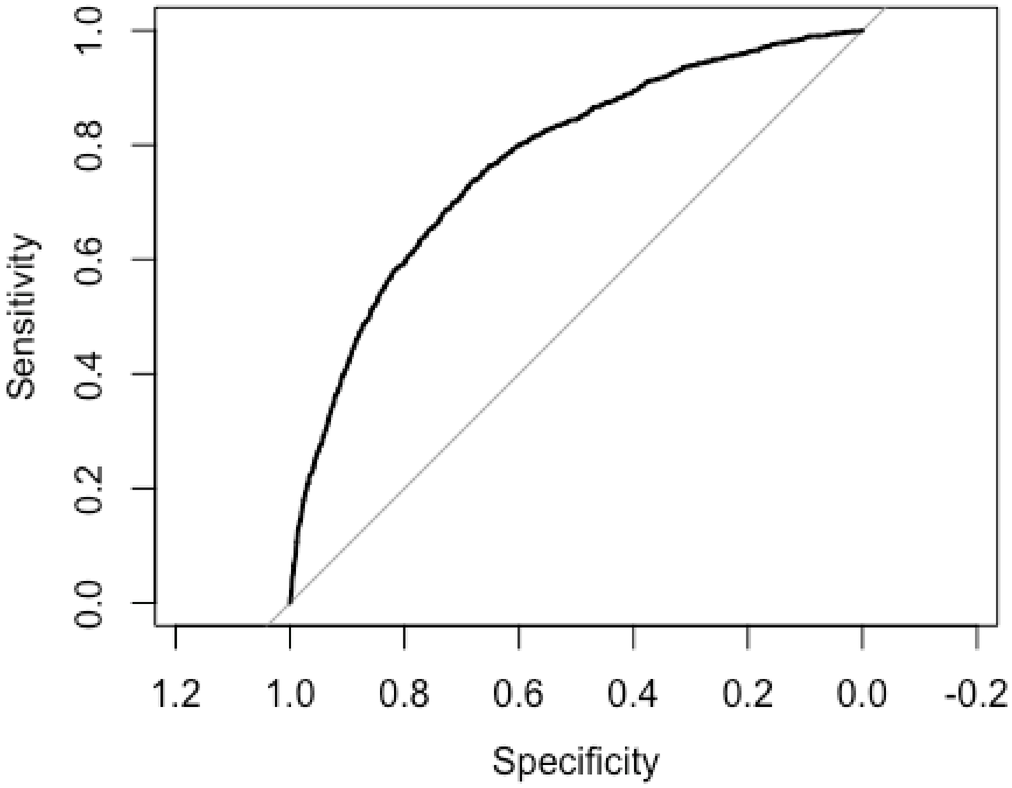
Choose CreditScore, Geography, Gender, Age, Balance, IsActiveMember

```
bank_log<-glm(Exited~CreditScore+Geography+Gender+Age+Balance+IsActiveMember,
 family = binomial(link = "logit"), data = train)

#Multicollinearity
vif(bank_log)

##                    GVIF Df GVIF^(1/(2*Df))
## CreditScore    1.001354  1        1.000677
## Geography      1.196809  2        1.045939
## Gender         1.003773  1        1.001885
## Age            1.091707  1        1.044848
## Balance        1.195835  1        1.093543
## IsActiveMember 1.086432  1        1.042321

#The results of GVIF^(1/(2*Df)) are lower that 2, so it seems like there is n
o correlation between the independent variables.

summary(bank_log)
```

```
## 
## Call:
## glm(formula = Exited ~ CreditScore + Geography + Gender + Age + 
##     Balance + IsActiveMember, family = binomial(link = "logit"), 
##     data = train)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -2.3786  -0.6553  -0.4496  -0.2582   2.9992  
## 
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)    
## (Intercept)      -3.717e+00  2.417e-01 -15.379  < 2e-16 ***
## CreditScore      -5.973e-04  3.134e-04  -1.906   0.0567 .  
## GeographyGermany  7.333e-01  7.548e-02   9.716  < 2e-16 ***
## GeographySpain    8.375e-03  7.861e-02   0.107   0.9152    
## GenderMale       -5.780e-01  6.100e-02  -9.476  < 2e-16 ***
## Age               7.545e-02  2.899e-03  26.022  < 2e-16 ***
## Balance           2.999e-06  5.521e-07   5.431 5.61e-08 ***
## IsActiveMember   -1.147e+00  6.502e-02 -17.637  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 8135.0  on 7999  degrees of freedom
## Residual deviance: 6832.4  on 7992  degrees of freedom
## AIC: 6848.4
## 
## Number of Fisher Scoring iterations: 5
```

```
#For every one unit change in creditscore, log odds of exited decreases by 5.973e-04
#For every one unit change in age, log odds of exited increases by 7.545e-02
#For every one unit change in balance, log odds of exited increases by 2.999e-06
#For rank uses France as a baseline so log odds increase by 7.333e-01 when moving from France to Germany
#For rank uses Female as a baseline so log odds decrease by 5.780e-01 when moving from Female to Male
#Residual Deviance: Reductions from 8135.0 to 6832.4, not great
#AIC: Akaike information criterion – Comparison between models, lower AIC is better

#Converting log odds coefficients to probabilities via exp() function
bank_log_ouput <- exp(coef(bank_log))
bank_log_ouput
```

```
##      (Intercept)      CreditScore GeographyGermany   GeographySpain 
##       0.02430666       0.99940291       2.08199990       1.00841011
##       GenderMale              Age          Balance   IsActiveMember
##       0.56099703       1.07836574       1.00000300       0.31765101

#For every unit increase in credit score the odds of being exited increase by
 a factor of 0.99940291 or the odds are 1 percent higher (subtract 1 and * 10
0).
#For every unit increase in age the odds of being exited increase by a factor
 of 1.078 or the odds are 7 percent higher (subtract 1 and * 100).
#For every unit increase in balance the odds of being exited increase by a fa
ctor of 1.0 or the odds are 0 percent higher (subtract 1 and * 100).

#Test our models goodness of fit
hoslem.test(train$Exited, fitted(bank_log))

##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  train$Exited, fitted(bank_log)
## X-squared = 13.407, df = 8, p-value = 0.0986

#how well our data fits the model. Specifically, the HL test calculates if th
e observed event rates match the expected event rates in population subgroups
.
#p-value is 0.0986, above 0.05 better

#First we want to use the data to predict the likelihood that each customer w
ill be exited
prob <- plogis(predict(bank_log, type = c("response")))
head(prob)

##         1         2         3         4         5         6
## 0.5281128 0.5332221 0.6007770 0.5579602 0.5371784 0.5662958

bank_roc <- roc(Exited~prob,data=train)
plot(bank_roc)
```



```
#Receiver Operating Characteristic: Measure of the Sensitivity (true positive
) and Specificity (false positives) of the Model
#Look good

#Psuedo R
pR2(bank_log)

##           llh       llhNull            G2      McFadden          r2ML 
## -3416.1834340 -4067.5185704  1302.6702728     0.1601308     0.1502676 
##          r2CU 
##     0.2354265
```

Use the model to predict Exited on the data

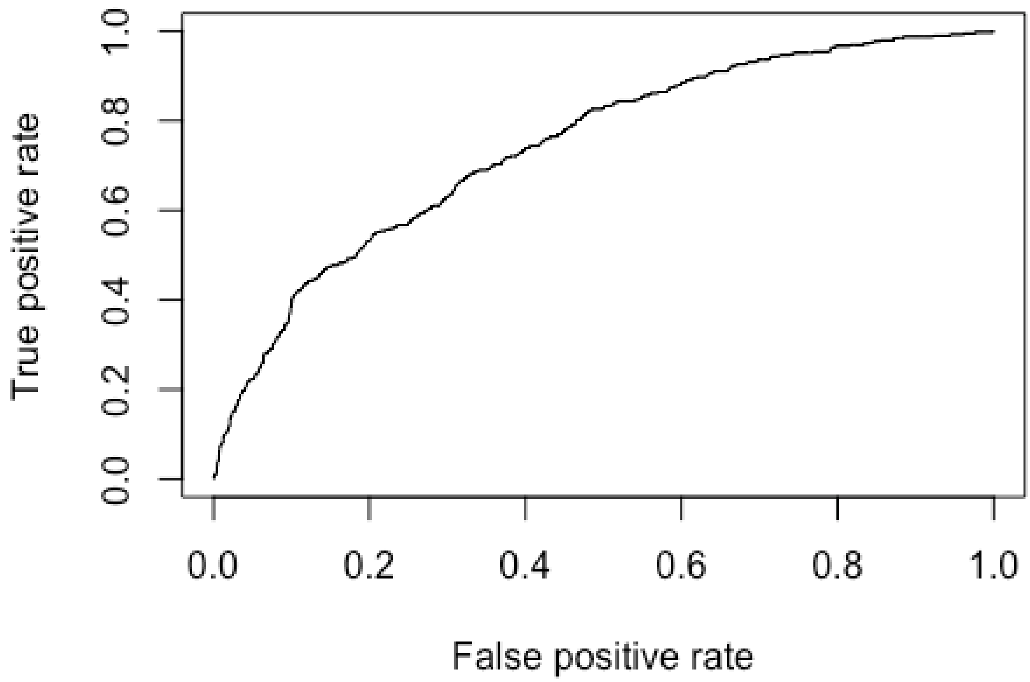
```
predict.model2 <- predict.glm(bank_log,test,type='response')
head(predict.model2)

##       8001       8002       8003       8004       8005       8006 
## 0.10612591 0.09156739 0.13616299 0.23535508 0.08210632 0.23664920

predict.model2.1 <- ifelse(predict.model2 > 0.5,1,0)

#essentially we are creating percentage likelihood of Exited for each value,
above 50% we are saying it's more likely to occur.
head(predict.model2.1)

## 8001 8002 8003 8004 8005 8006
##    0    0    0    0    0    0

test %>% group_by(Exited) %>% summarise(no_rows = length(Exited))

##   no_rows
## 1    2000

#So we only had to correctly identify 390 Yes factors, which is 19.5%  of the
 total factor, not very small
```

hit rate

```
model3hit <- mean(predict.model2.1!=test$Exited)
hitrate <- 1-model3hit
hitrate

## [1] 0.809

#hit rate of our model is 0.809, this means our model is good. the higher the
 better
```

Do a roc curve, just to confirm our model, using a different package then our first example

```
#In order to use the package we first have to set the prediction
newpredict <- prediction(predict.model2,test$Exited)
#Next we want to measure true possitives which is "tpr" and also False Positi
ves "fpr"
newpredict.performance <- performance(newpredict, measure = "tpr",x.measure =
 "fpr")
#then we plot these two measures
plot(newpredict.performance)
```



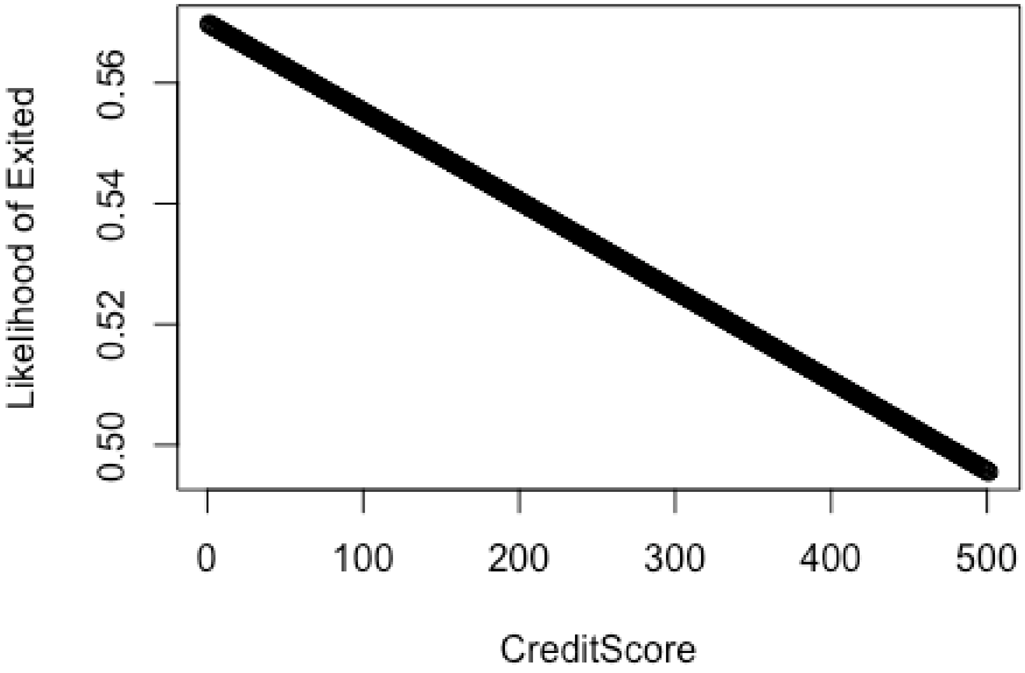
get the AUC again using the performance function

```
AUC <- performance(newpredict, measure = "auc")
AUC

## An object of class "performance"
## Slot "x.name":
## [1] "None"
##
## Slot "y.name":
## [1] "Area under the ROC curve"
##
## Slot "alpha.name":
## [1] "none"
##
## Slot "x.values":
## list()
##
## Slot "y.values":
## [[1]]
## [1] 0.7418554
##
##
```

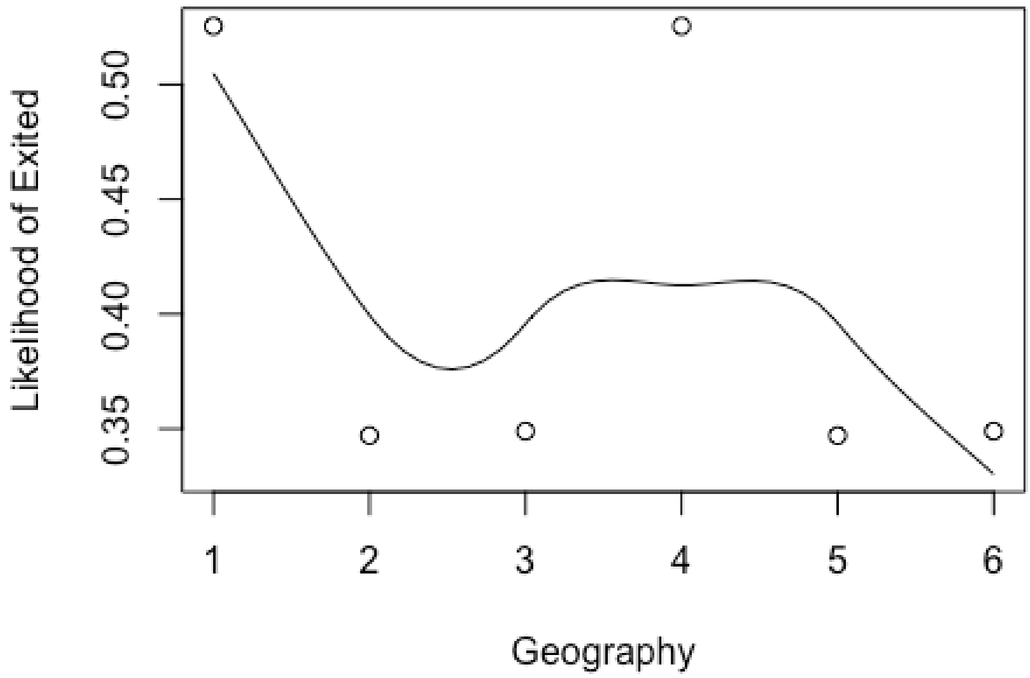
```
## Slot "alpha.values":
## list()

#the result is 0.74, good
```

Use this model to actually predict something

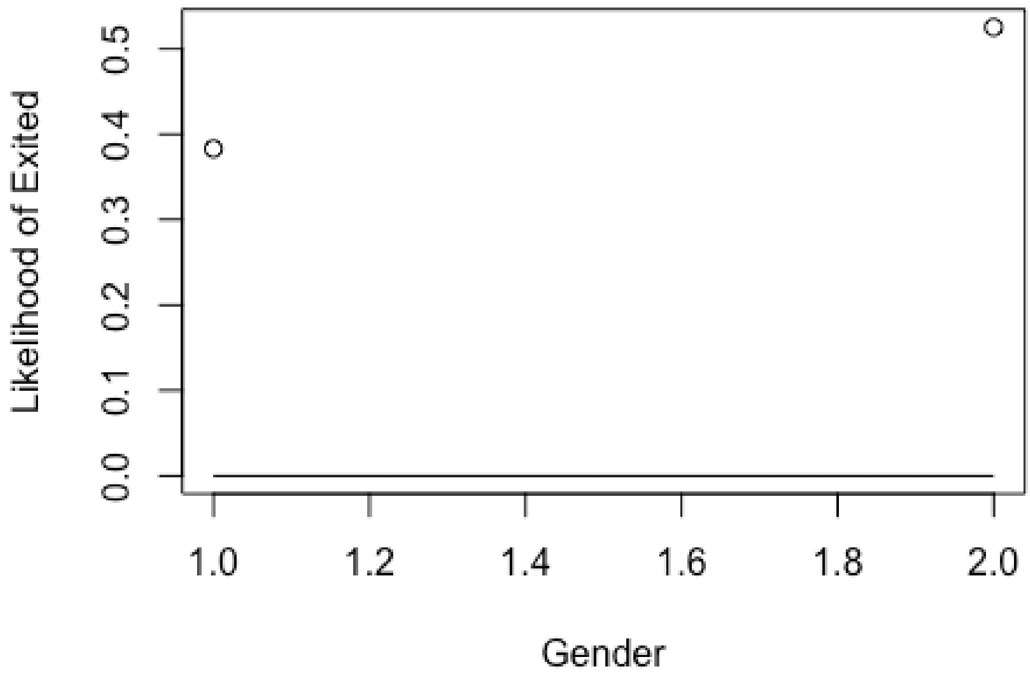
```
actual <- data.frame(CreditScore=650,Geography='Germany',Gender='Female',Age=
43,Balance=76486,IsActiveMember=0)
actualpredict <- predict(bank_log,actual,type='response')
actualpredict

##        1
## 0.525391

#prediction about creditscore
new.data.test1 <- data.frame(CreditScore=350:850,Geography='Germany',Gender='
Female',Age=43,Balance=76486,IsActiveMember=0)
new.data.test.pred1 <- predict(bank_log,new.data.test1, type = "response")
head(new.data.test.pred1)

##         1         2         3         4         5         6
## 0.5697503 0.5696039 0.5694575 0.5693110 0.5691646 0.5690181

scatter.smooth(new.data.test.pred1, xlab = "CreditScore", ylab = "Likelihood
of Exited")
```



```
#prediction about Geography
new.data.test2 <- data.frame(CreditScore=650,Geography=rep(c('Germany','France','Spain'),2),Gender='Female',Age=43,Balance=76486,IsActiveMember=0)
new.data.test.pred2 <- predict(bank_log,new.data.test2, type = "response")
head(new.data.test.pred2)

##         1         2         3         4         5         6 
## 0.5253910 0.3471303 0.3490307 0.5253910 0.3471303 0.3490307

scatter.smooth(new.data.test.pred2, xlab = "Geography", ylab = "Likelihood of Exited")
```

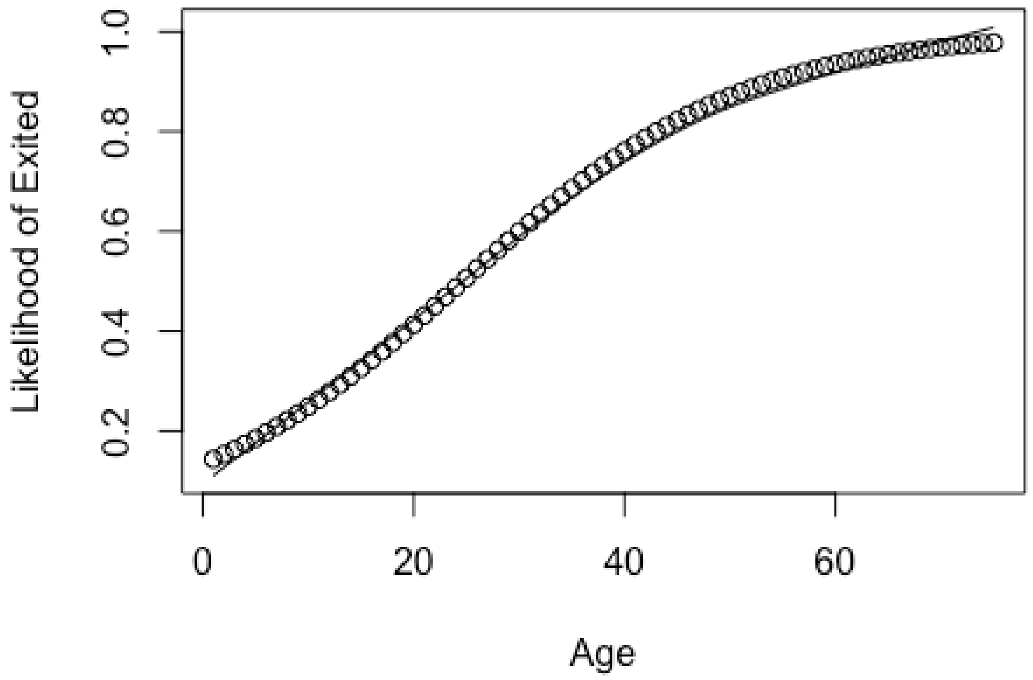


```
#prediction about Gender
new.data.test3 <- data.frame(CreditScore=650,Geography='Germany',Gender=c('Ma
le','Female'),Age=43,Balance=76486,IsActiveMember=0)
new.data.test.pred3 <- predict(bank_log,new.data.test3, type = "response")
head(new.data.test.pred3)

##         1         2
## 0.3831054 0.5253910

scatter.smooth(new.data.test.pred3, xlab = "Gender", ylab = "Likelihood of Ex
ited")

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger
```

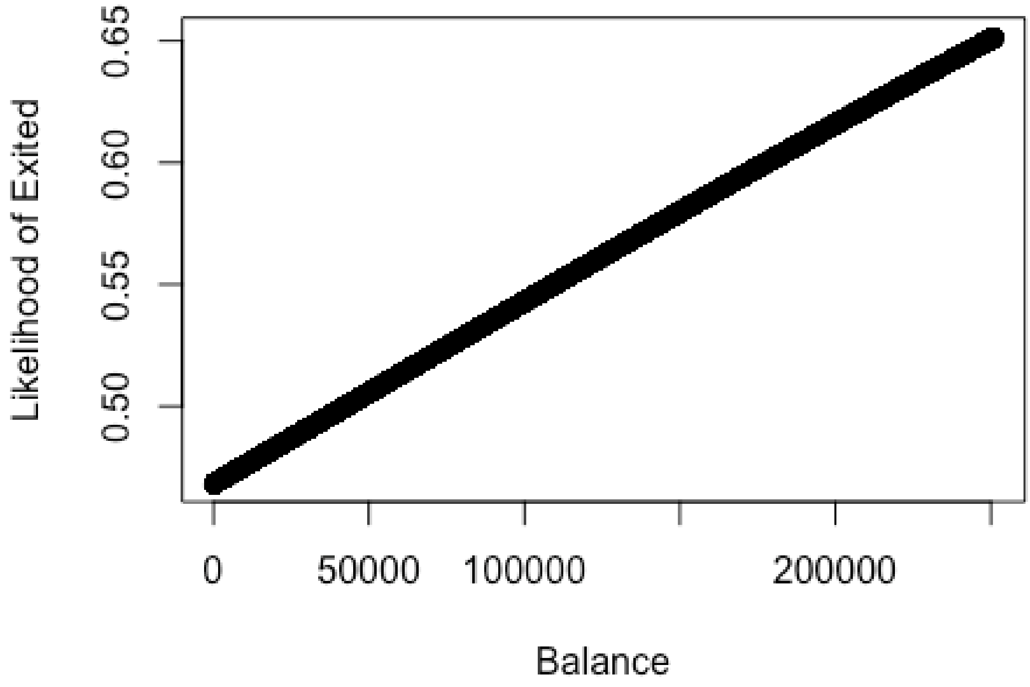


```
#prediction about Age
new.data.test4 <- data.frame(CreditScore=650,Geography='Germany',Gender='Fema
le',Age=18:92,Balance=76486,IsActiveMember=0)
```

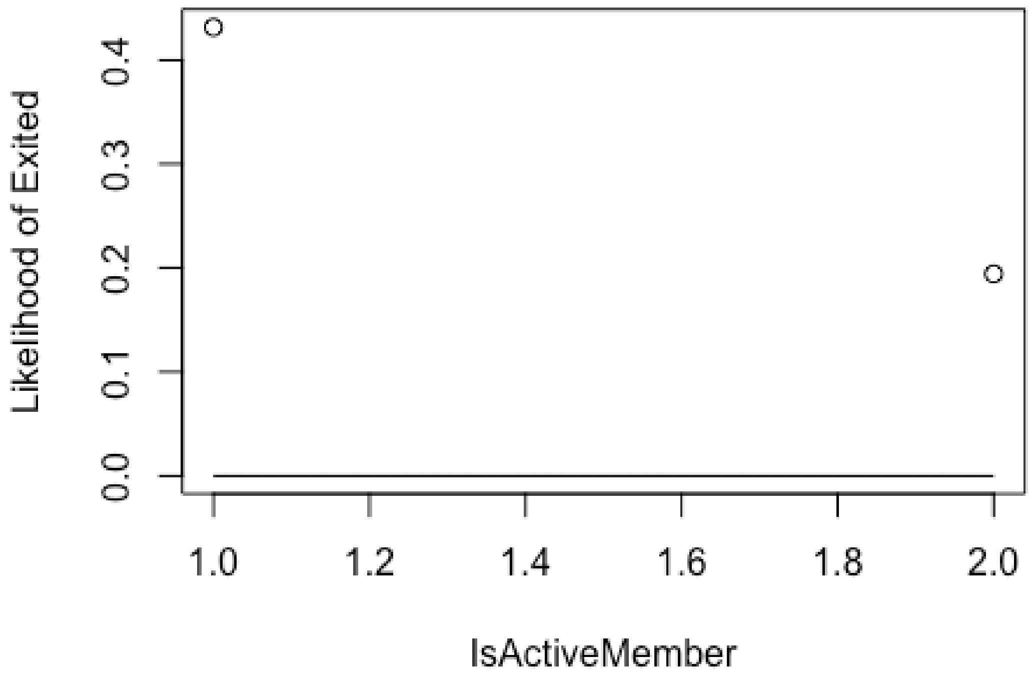
```
new.data.test.pred4 <- predict(bank_log,new.data.test4, type = "response")
head(new.data.test.pred4)

##         1         2         3         4         5         6
## 0.1437464 0.1532845 0.1633347 0.1739086 0.1850156 0.1966631

scatter.smooth(new.data.test.pred4, xlab = "Age", ylab = "Likelihood of Exite
d")
```



```
#prediction about Balance
new.data.test5 <- data.frame(CreditScore=650,Geography='Germany',Gender='Fema
le',Age=43,Balance=0:250898,IsActiveMember=0)
new.data.test.pred5 <- predict(bank_log,new.data.test5, type = "response")
head(new.data.test.pred5)

##         1         2         3         4         5         6
## 0.4681198 0.4681206 0.4681213 0.4681221 0.4681228 0.4681236

scatter.smooth(new.data.test.pred5, xlab = "Balance", ylab = "Likelihood of E
xited")
```



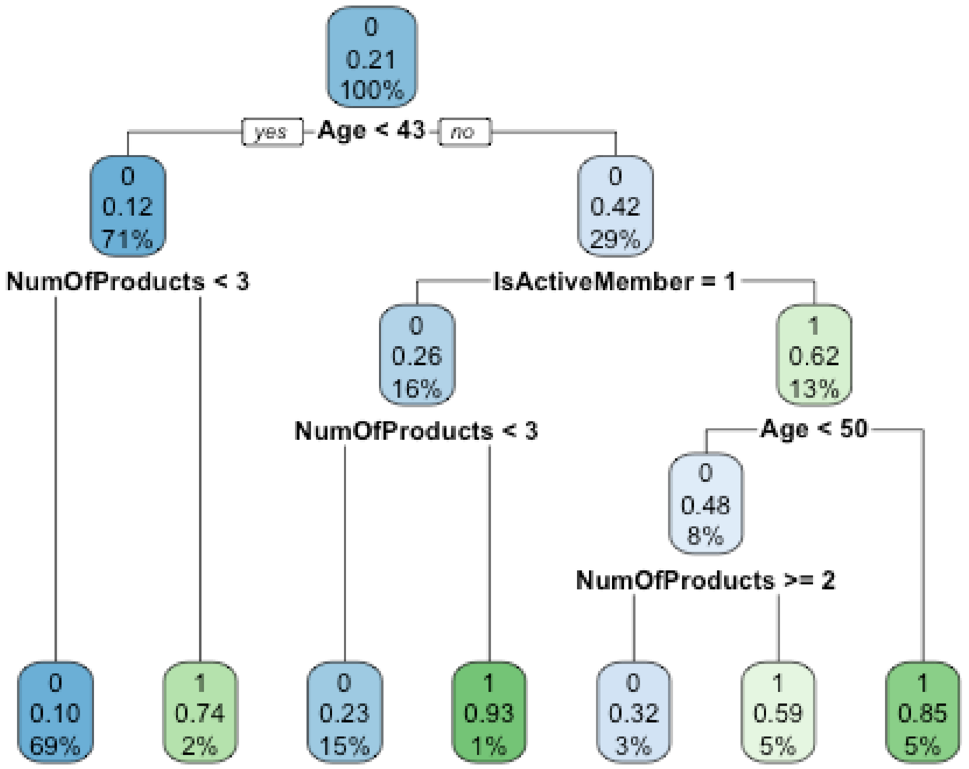
```
#prediction about IsActiveMember
new.data.test6 <- data.frame(CreditScore=650,Geography='Germany',Gender='Fema
le',Age=38,Balance=76486,IsActiveMember=as.integer(0:1))
new.data.test.pred6 <- predict(bank_log,new.data.test6, type = "response")
head(new.data.test.pred6)

##         1         2
## 0.4315373 0.1942881

scatter.smooth(new.data.test.pred6, xlab = "IsActiveMember", ylab = "Likeliho
od of Exited")

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : pseudoinverse used at 0.995

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : neighborhood radius 0.005
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : at 2.005

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : radius 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : There are other near singularities as well. 2.5e-05

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## FALSE, : zero-width neighborhood. make span bigger
```



Decision Tree

```
str(bank)
```

```
## 'data.frame':    10000 obs. of  11 variables:
##  $ CreditScore    : int  619 608 502 699 850 645 822 376 501 684 ...
##  $ Geography      : Factor w/ 3 levels "France","Germany",..: 1 3 1 1 3 3
1 2 1 1 ...
##  $ Gender         : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 2 2 1 2
2 ...
##  $ Age            : int  42 41 42 39 43 44 50 29 44 27 ...
##  $ Tenure         : int  2 1 8 1 2 8 7 4 4 2 ...
##  $ Balance        : num  0 83808 159661 0 125511 ...
##  $ NumOfProducts  : int  1 1 3 2 1 2 2 4 2 1 ...
##  $ HasCrCard      : int  1 0 1 0 1 1 1 1 0 1 ...
##  $ IsActiveMember : int  1 1 0 0 1 0 1 0 1 1 ...
##  $ EstimatedSalary: num  101349 112543 113932 93827 79084 ...
##  $ Exited         : int  1 0 1 0 0 1 0 1 0 0 ...

#Build the model
# Train the tree with the rpart() function.
# We'll need to set the seed to make the results reproducible.
set.seed(1)
bank_tree_gini = rpart(Exited~.,
                           method = "class",
                           data = train,
                       control = rpart.control(maxdepth = 4))
rpart.plot(bank_tree_gini)
```



```r
# Let's use the "predict" function to test our our model and then
# evaluate the accuracy of the results.
bank_model = predict(bank_tree_gini, type = "class")

# Let's compare the results to the actual data.
bank_matrix = table(bank_model, train$Exited)
bank_matrix

##
## bank_model    0    1
##          0 6094  936
##          1  259  711

table(bank_model)

## bank_model
##    0    1
## 7030  970

# The error rate is defined as a classification of "Pregnant" when
# this is not the case, and vice versa. It's the sum of all the
# values where a column contains the opposite value of the row.
sum(bank_matrix[row(bank_matrix) != col(bank_matrix)])
```

```
## [1] 1195

# 1195


# The error rate divides this figure by the total number of data points
# for which the forecast is created.
sum(bank_matrix)

## [1] 8000

# 8000

# Let's use these values in 1 calculation.
bank_error_rate = sum(bank_matrix[row(bank_matrix) != col(bank_matrix)]) /
                  sum(bank_matrix)


paste0("Real error rate is: ", bank_error_rate * 100, "%")

## [1] "Real error rate is: 14.9375%"

# "Real error rate is: 14.9375%"

bank_model<- as.numeric(bank_model)

predict.model2.1 <- ifelse(bank_model > 0.5,1,0)
#essentially we are creating percentage likelihood of Exited for each value,
above 50% we are saying it's more likely to occur.
head(predict.model2.1)

## [1] 1 1 1 1 1 1

model4hit <- mean(predict.model2.1!=test$Exited)
model4hit

## [1] 0.805

#80.5%, absolutely higher than the one on logsitic regression.
```

Conclusion and solution:

1, The bank should hold promotional activities to the old, particularly age is older than 43.

2, Create more useful products and enable customers to have more accounts, better than 3.

3, Focus on German and Female, compared to France, Spain and Male.

4, Establish a good membership system to make more active members.