project.R

Saloni

2019-11-14

```
#Name: Saloni Mishra
#Purpose: Class project
# #install.packages("tidyr")
# install.packages("lmtest", repos = "http://cran.us.r-project.org")
# install.packages("devtools")
# install.packages("tidyverse")
# install.packages("caret")
# install.packages("car")
# install.packages("hrbrthemes")
# install.packages("olsrr")
# install.packages("ROCR")
# install.packages("pROC")
# install.packages("ggplot2")
library(pROC)
```

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

```
##
## Attaching package: 'pROC'
```

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

```
library(ROCR)
```

```
## Loading required package: gplots
```

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

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

```
library(olsrr)
```

```
##
## Attaching package: 'olsrr'
```

```
## The following object is masked from 'package:datasets':
##
##     rivers
```

```
library(hrbrthemes)

## NOTE: Either Arial Narrow or Roboto Condensed fonts are required to use
these themes.

##       Please use hrbrthemes::import_roboto_condensed() to install Roboto
Condensed and

##       if Arial Narrow is not on your system, please see
http://bit.ly/arialnarrow

library(tidyverse)

## -- Attaching packages ---------------------------------------------
tidyverse 1.2.1 --

## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------
tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

library(caret)

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift

library(car)

## Loading required package: carData

##
## Attaching package: 'car'

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

## The following object is masked from 'package:purrr':
##
##     some

library(VIF)
```

```
##
## Attaching package: 'VIF'

## The following object is masked from 'package:car':
##
##     vif

library(MASS)

##
## Attaching package: 'MASS'

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

## The following object is masked from 'package:olsrr':
##
##     cement

library(ggplot2)
library("readxl")
library(dplyr)
library(RColorBrewer)
library(reshape2)

##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths

library(ISLR)
library(ggplot2)
library(digest)
library(tidyr)
library(ggplot2)
library(dplyr)
getPalette = colorRampPalette(brewer.pal(12, "Set1"))

## Warning in brewer.pal(12, "Set1"): n too large, allowed maximum for
palette Set1 is 9
## Returning the palette you asked for with that many colors

#read data
my_data<-read_excel("C:\\Users\\Arvind\\Downloads\\default of credit card
clients.xls")

## New names:
## * `` -> ...1
```

```r
View(my_data)
#str(my_data)
colnames(my_data)<-my_data[1,]
mydata<-data.frame(apply(my_data[-1,], 2, as.numeric))
View(mydata)
# Creating Variable of different levels to two level
mydata$PAY_0<-ifelse(mydata$PAY_0<=0,1,0)
mydata$PAY_2<-ifelse(mydata$PAY_2<=0,1,0)
mydata$PAY_3<-ifelse(mydata$PAY_3<=0,1,0)
mydata$PAY_4<-ifelse(mydata$PAY_4<=0,1,0)
mydata$PAY_5<-ifelse(mydata$PAY_5<=0,1,0)
mydata$PAY_6<-ifelse(mydata$PAY_6<=0,1,0)
str(mydata)
```

```
## 'data.frame':    30000 obs. of  25 variables:
##  $ ID                        : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ LIMIT_BAL                 : num  20000 120000 90000 50000 50000 50000
500000 100000 140000 20000 ...
##  $ SEX                       : num  2 2 2 2 1 1 1 2 2 1 ...
##  $ EDUCATION                 : num  2 2 2 2 2 1 1 2 3 3 ...
##  $ MARRIAGE                  : num  1 2 2 1 1 2 2 2 1 2 ...
##  $ AGE                       : num  24 26 34 37 57 37 29 23 28 35 ...
##  $ PAY_0                     : num  0 1 1 1 1 1 1 1 1 1 ...
##  $ PAY_2                     : num  0 0 1 1 1 1 1 1 1 1 ...
##  $ PAY_3                     : num  1 1 1 1 1 1 1 1 0 1 ...
##  $ PAY_4                     : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ PAY_5                     : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ PAY_6                     : num  1 0 1 1 1 1 1 1 1 1 ...
##  $ BILL_AMT1                 : num  3913 2682 29239 46990 8617 ...
##  $ BILL_AMT2                 : num  3102 1725 14027 48233 5670 ...
##  $ BILL_AMT3                 : num  689 2682 13559 49291 35835 ...
##  $ BILL_AMT4                 : num  0 3272 14331 28314 20940 ...
##  $ BILL_AMT5                 : num  0 3455 14948 28959 19146 ...
##  $ BILL_AMT6                 : num  0 3261 15549 29547 19131 ...
##  $ PAY_AMT1                  : num  0 0 1518 2000 2000 ...
##  $ PAY_AMT2                  : num  689 1000 1500 2019 36681 ...
##  $ PAY_AMT3                  : num  0 1000 1000 1200 10000 657 38000 0 432
0 ...
##  $ PAY_AMT4                  : num  0 1000 1000 1100 9000 ...
##  $ PAY_AMT5                  : num  0 0 1000 1069 689 ...
##  $ PAY_AMT6                  : num  0 2000 5000 1000 679 ...
##  $ default.payment.next.month: num  1 1 0 0 0 0 0 0 0 0 ...
```

```r
names <- c(3:5,7:12,25)
#Changing numeric to factor
mydata[,names] <- lapply(mydata[,names] , factor)
str(mydata)
```

```
## 'data.frame':    30000 obs. of  25 variables:
##  $ ID                        : num  1 2 3 4 5 6 7 8 9 10 ...
```

```
##  $ LIMIT_BAL                  : num  20000 120000 90000 50000 50000 50000
500000 100000 140000 20000 ...
##  $ SEX                        : Factor w/ 2 levels "1","2": 2 2 2 2 1 1 1 2
2 1 ...
##  $ EDUCATION                  : Factor w/ 7 levels "0","1","2","3",..: 3 3
3 3 3 2 2 3 4 4 ...
##  $ MARRIAGE                   : Factor w/ 4 levels "0","1","2","3": 2 3 3 2
2 3 3 3 2 3 ...
##  $ AGE                        : num  24 26 34 37 57 37 29 23 28 35 ...
##  $ PAY_0                      : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2
2 2 ...
##  $ PAY_2                      : Factor w/ 2 levels "0","1": 1 1 2 2 2 2 2 2
2 2 ...
##  $ PAY_3                      : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2
1 2 ...
##  $ PAY_4                      : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2
2 2 ...
##  $ PAY_5                      : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2
2 2 ...
##  $ PAY_6                      : Factor w/ 2 levels "0","1": 2 1 2 2 2 2 2 2
2 2 ...
##  $ BILL_AMT1                  : num  3913 2682 29239 46990 8617 ...
##  $ BILL_AMT2                  : num  3102 1725 14027 48233 5670 ...
##  $ BILL_AMT3                  : num  689 2682 13559 49291 35835 ...
##  $ BILL_AMT4                  : num  0 3272 14331 28314 20940 ...
##  $ BILL_AMT5                  : num  0 3455 14948 28959 19146 ...
##  $ BILL_AMT6                  : num  0 3261 15549 29547 19131 ...
##  $ PAY_AMT1                   : num  0 0 1518 2000 2000 ...
##  $ PAY_AMT2                   : num  689 1000 1500 2019 36681 ...
##  $ PAY_AMT3                   : num  0 1000 1000 1200 10000 657 38000 0 432
0 ...
##  $ PAY_AMT4                   : num  0 1000 1000 1100 9000 ...
##  $ PAY_AMT5                   : num  0 0 1000 1069 689 ...
##  $ PAY_AMT6                   : num  0 2000 5000 1000 679 ...
##  $ default.payment.next.month: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1
1 1 ...
```

```r
View(mydata)

#summary(mydata)
# for randomization
set.seed(100)
mydata1<-mydata[,c(-1,-14:-18)]
#View(mydata)
rows <- sample(nrow(mydata1))
mydata1 <- mydata1[rows, ]
# spliting (70%): split for training and test dataset
split <- round(nrow(mydata1) * .7)
train <- mydata1[1:split, ]
test <- mydata1[(split + 1):nrow(mydata1), ]
```

```r
rownames(train)<- seq(length=nrow(train))
rownames(test)<- seq(length=nrow(test))
View(mydata1)
########################################
########################################
# All Subsets Regression
# Fitting logistic model

glm.train <- glm(train$default.payment.next.month ~ ., train, family =
"binomial")
glm.ptrain=predict(glm.train, type="response")
train_res<-train
train_res$Prob<-glm.ptrain
View(train_res)



train_res$default.payment<-ifelse(glm.ptrain>0.3,"1","0")
#lvs <- c("1", "0")
#levels = rev(lvs)
#table(glm.ptrain)
#View(train_res)
conf_matrix<-
table(train_res$default.payment,train_res$default.payment.next.month)
conf_matrix

##
##         0     1
##   0 14032  2253
##   1  2258  2457

summary(glm.train)

##
## Call:
## glm(formula = train$default.payment.next.month ~ ., family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7759  -0.5996  -0.5258  -0.3302   3.1128
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.199e+01  9.066e+01  -0.132 0.894780
## LIMIT_BAL   -1.533e-06  1.851e-07  -8.280  < 2e-16 ***
## SEX2        -1.141e-01  3.746e-02  -3.045 0.002330 **
## EDUCATION1   1.085e+01  9.066e+01   0.120 0.904763
## EDUCATION2   1.086e+01  9.066e+01   0.120 0.904604
## EDUCATION3   1.081e+01  9.066e+01   0.119 0.905056
## EDUCATION4   9.972e+00  9.066e+01   0.110 0.912408
```

```
## EDUCATION5   9.592e+00  9.066e+01   0.106 0.915739
## EDUCATION6   1.096e+01  9.066e+01   0.121 0.903741
## MARRIAGE1    2.312e+00  1.060e+00   2.181 0.029169 *
## MARRIAGE2    2.156e+00  1.060e+00   2.033 0.042039 *
## MARRIAGE3    2.243e+00  1.073e+00   2.090 0.036579 *
## AGE          4.220e-03  2.279e-03   1.852 0.064084 .
## PAY_01      -1.274e+00  5.033e-02 -25.324  < 2e-16 ***
## PAY_21      -1.967e-01  6.864e-02  -2.865 0.004165 **
## PAY_31      -4.276e-01  6.734e-02  -6.350 2.15e-10 ***
## PAY_41      -2.494e-01  7.492e-02  -3.328 0.000873 ***
## PAY_51      -2.346e-01  8.231e-02  -2.850 0.004373 **
## PAY_61      -3.598e-01  7.039e-02  -5.111 3.20e-07 ***
## BILL_AMT1    1.430e-06  2.910e-07   4.914 8.92e-07 ***
## PAY_AMT1    -6.887e-06  2.093e-06  -3.290 0.001003 **
## PAY_AMT2    -7.154e-06  1.987e-06  -3.600 0.000318 ***
## PAY_AMT3    -2.395e-06  1.632e-06  -1.468 0.142138
## PAY_AMT4    -5.547e-06  1.980e-06  -2.801 0.005089 **
## PAY_AMT5    -1.418e-06  1.561e-06  -0.908 0.363807
## PAY_AMT6    -9.770e-07  1.376e-06  -0.710 0.477585
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22356  on 20999  degrees of freedom
## Residual deviance: 18980  on 20974  degrees of freedom
## AIC: 19032
##
## Number of Fisher Scoring iterations: 11

#senstivity
a<-sensitivity(conf_matrix)
b<-specificity(conf_matrix)
a

## [1] 0.8613874

b

## [1] 0.5216561

#TEST
glm.ptest<-predict(glm.train, test, type="response")
test_res<-test
test_res$Prob<-glm.ptest
test_res$Default_Predict<-ifelse(glm.ptest>0.3,"1","0")
conf_matrix2<-
table(test_res$default.payment.next.month,test_res$Default_Predict)
conf_matrix2
```

```
##
##        0    1
##   0 6072 1002
##   1  883 1043
```

```
#dimnames(conf_matrix2)[[1]] = c("1","0")
#conf_matrix3<-conf_matrix2[order(conf_matrix2[,1]),]
#senstivity
d<-sensitivity(conf_matrix2)
e<-specificity(conf_matrix2)
d
```

```
## [1] 0.873041
```

```
e
```

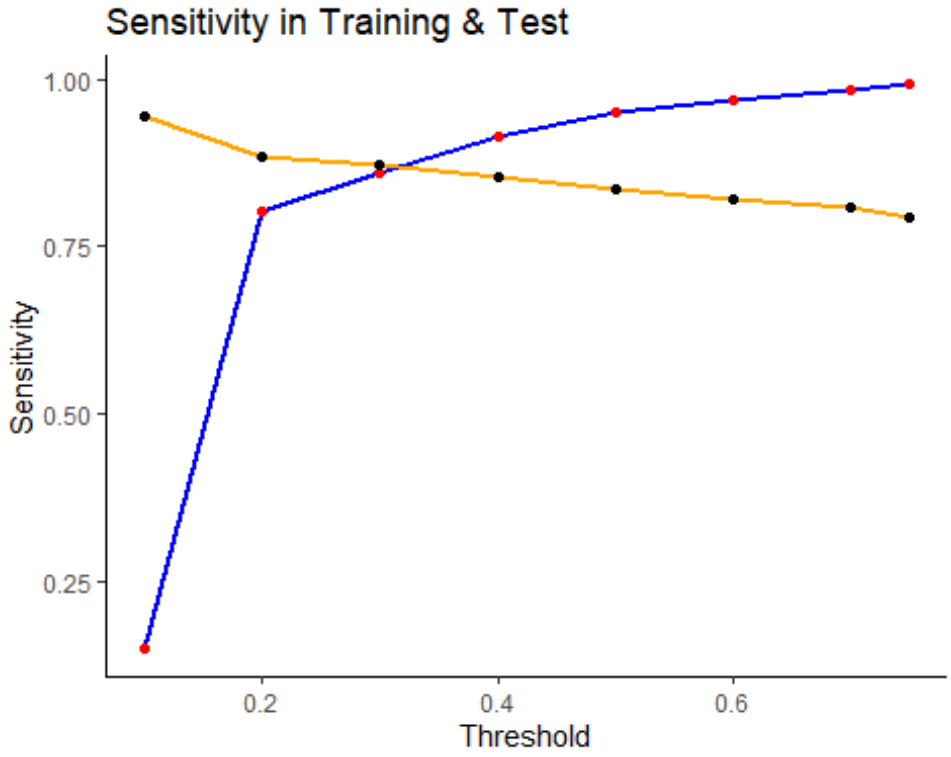
```
## [1] 0.5100244
```

```
threshold<-function(threshold=0.2){
  train_res$default.payment<-ifelse(glm.ptrain>threshold,"1","0")
  conf_matrix<-
table(train_res$default.payment,train_res$default.payment.next.month)
  a1<-sensitivity(conf_matrix)
  b1<-specificity(conf_matrix)
  #TEST
  test_res$Default_Predict<-ifelse(glm.ptest>threshold,"1","0")
  conf_matrix2<-
table(test_res$default.payment.next.month,test_res$Default_Predict)
  d1<-sensitivity(conf_matrix2)
  e1<-specificity(conf_matrix2)

  output<-c(a1,b1,threshold,d1,e1)
  return(output)
}

a1<-threshold(0.1)
a2<-threshold(0.2)
a3<-threshold(0.3)
a4<-threshold(0.4)
a5<-threshold(0.5)
a6<-threshold(0.6)
a7<-threshold(0.7)
a8<-threshold(0.75)


imp<-data.frame(rbind(a1,a2,a3,a4,a5,a6,a7,a8))
imp
```

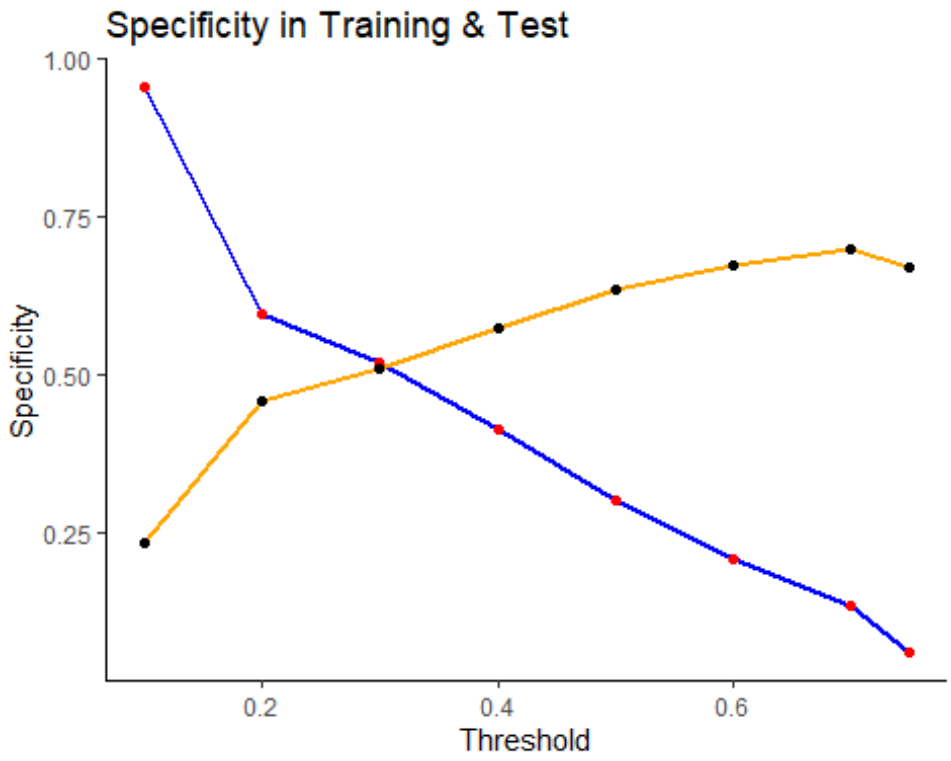
```
##           X1         X2   X3        X4        X5
## a1 0.1492327 0.95647558 0.10 0.9447964 0.2362255
## a2 0.8037446 0.59660297 0.20 0.8849157 0.4580123
```

```
## a3 0.8613874 0.52165605 0.30 0.8730410 0.5100244
## a4 0.9146102 0.41486200 0.40 0.8533404 0.5761871
## a5 0.9511971 0.30063694 0.50 0.8354855 0.6353066
## a6 0.9697974 0.20976645 0.60 0.8220491 0.6732824
## a7 0.9839779 0.13418259 0.70 0.8083672 0.7012658
## a8 0.9933088 0.06157113 0.75 0.7947905 0.6705882
```

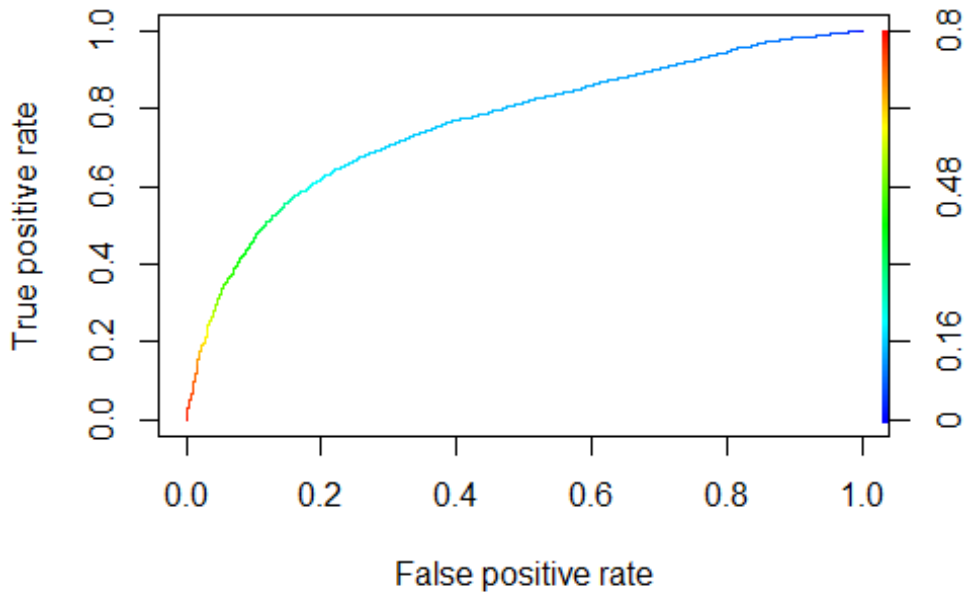
```r
ggplot(data=imp,aes(x=X3))+
  geom_line(aes(y=X1), color="Blue", size=1)+
  geom_line(aes(y=X4), color="Orange", size=1)+
  geom_point(y=imp$X1, color="red")+
  geom_point(y=imp$X4, color="black")+
  xlab("Threshold")+
  ylab("Sensitivity")+
  ggtitle("Sensitivity in Training & Test")+
  theme_classic()
```



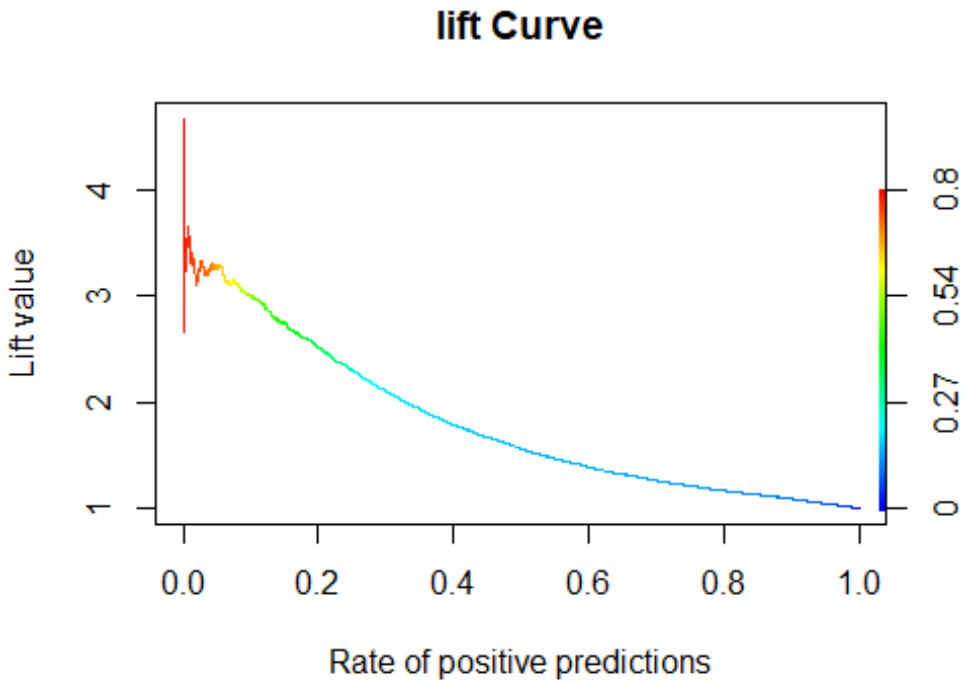

```r
ggplot(data=imp,aes(x=X3))+
  geom_line(aes(y=X2), color="Blue", size=1)+
  geom_line(aes(y=X5), color="Orange", size=1)+
  geom_point(y=imp$X2, color="red")+
  geom_point(y=imp$X5, color="black")+
  xlab("Threshold")+
  ylab("Specificity")+
  ggtitle("Specificity in Training & Test")+
  theme_classic()
```




```
# ROC plot

ROCpred<-prediction(test_res$Prob,test_res$default.payment.next.month)
ROCperf<-performance(ROCpred, 'tpr', 'fpr')
# ROCperf<-performance(ROCpred, 'auc')
# auc<-as.numeric(ROCperf@y.values)
# auc
#auc =0.768

plot(ROCperf, colorize = TRUE, text.adj = c(-0.2,1.7))
```




```r
#lift curve

perf=performance(ROCpred,"lift","rpp")

plot(perf,main="lift Curve",colorize=T)
```




```
#summary(glm.train)
#########################&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
#############################
```