Logistic_Regression.R

```
## Logistic Regression


##Calling library
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.6.2

#install.packages("cowplot", lib="/Library/Frameworks/R.framework/Versions/3.
5/Resources/library")
library(cowplot)

## Warning: package 'cowplot' was built under R version 3.6.3

##
## ********************************************************

## Note: As of version 1.0.0, cowplot does not change the

##   default ggplot2 theme anymore. To recover the previous

##   behavior, execute:
##   theme_set(theme_cowplot())

## ********************************************************

## Few packages for confusion matrix. Lets look at them one by one
#install.packages("regclass", lib="/Library/Frameworks/R.framework/Versions/3
.5/Resources/library")
library(regclass)

## Warning: package 'regclass' was built under R version 3.6.3

## Loading required package: bestglm

## Warning: package 'bestglm' was built under R version 3.6.3

## Loading required package: leaps

## Warning: package 'leaps' was built under R version 3.6.3

## Loading required package: VGAM

## Warning: package 'VGAM' was built under R version 3.6.3

## Loading required package: stats4

## Loading required package: splines
```

```
## Loading required package: rpart

## Loading required package: randomForest

## Warning: package 'randomForest' was built under R version 3.6.3

## 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

## Important regclass change from 1.3:
## All functions that had a . in the name now have an _
## all.correlations -> all_correlations, cor.demo -> cor_demo, etc.
```

```r
#install.packages("caret", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3

## Loading required package: lattice

## Warning: package 'lattice' was built under R version 3.6.2

##
## Attaching package: 'lattice'

## The following object is masked from 'package:regclass':
##
##     qq

##
## Attaching package: 'caret'

## The following object is masked from 'package:VGAM':
##
##     predictors
```

```r
#install.packages("e1071", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.6.3
```

```r
#install.packages("pROC", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 3.6.3

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

##
## Attaching package: 'pROC'

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

```
#Loading Dataset
bank=read.csv("C:/Users/Deepal/Desktop/MVA/bank.csv",fill=TRUE)
View(bank)
attach(bank)
head(bank)
```

```
##   age        job marital education default balance housing loan contact
## 1  59     admin. married secondary      no    2343     yes   no unknown
## 2  56     admin. married secondary      no      45      no   no unknown
## 3  41 technician married secondary      no    1270     yes   no unknown
## 4  55   services married secondary      no    2476     yes   no unknown
## 5  54     admin. married  tertiary      no     184      no   no unknown
## 6  42 management  single  tertiary      no       0     yes  yes unknown
##   day month duration campaign pdays previous poutcome deposit
## 1   5   may     1042        1    -1        0  unknown     yes
## 2   5   may     1467        1    -1        0  unknown     yes
## 3   5   may     1389        1    -1        0  unknown     yes
## 4   5   may      579        1    -1        0  unknown     yes
## 5   5   may      673        2    -1        0  unknown     yes
## 6   5   may      562        2    -1        0  unknown     yes
```

```
str(bank)
```

```
## 'data.frame':    11162 obs. of  17 variables:
##  $ age      : int  59 56 41 55 54 42 56 60 37 28 ...
##  $ job      : Factor w/ 12 levels "admin.","blue-collar",..: 1 1 10 8 1 5
5 6 10 8 ...
##  $ marital  : Factor w/ 3 levels "divorced","married",..: 2 2 2 2 2 3 2 1
2 3 ...
##  $ education: Factor w/ 4 levels "primary","secondary",..: 2 2 2 2 3 3 3 2
2 2 ...
##  $ default  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ balance  : int  2343 45 1270 2476 184 0 830 545 1 5090 ...
##  $ housing  : Factor w/ 2 levels "no","yes": 2 1 2 2 1 2 2 2 2 2 ...
##  $ loan     : Factor w/ 2 levels "no","yes": 1 1 1 1 1 2 2 1 1 1 ...
##  $ contact  : Factor w/ 3 levels "cellular","telephone",..: 3 3 3 3 3 3 3
3 3 3 ...
##  $ day      : int  5 5 5 5 5 5 6 6 6 6 ...
##  $ month    : Factor w/ 12 levels "apr","aug","dec",..: 9 9 9 9 9 9 9 9 9
9 ...
```

```
##  $ duration : int  1042 1467 1389 579 673 562 1201 1030 608 1297 ...
##  $ campaign : int  1 1 1 1 2 2 1 1 1 3 ...
##  $ pdays    : int  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ previous : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome : Factor w/ 4 levels "failure","other",..: 4 4 4 4 4 4 4 4 4 4
...
##  $ deposit  : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...
```

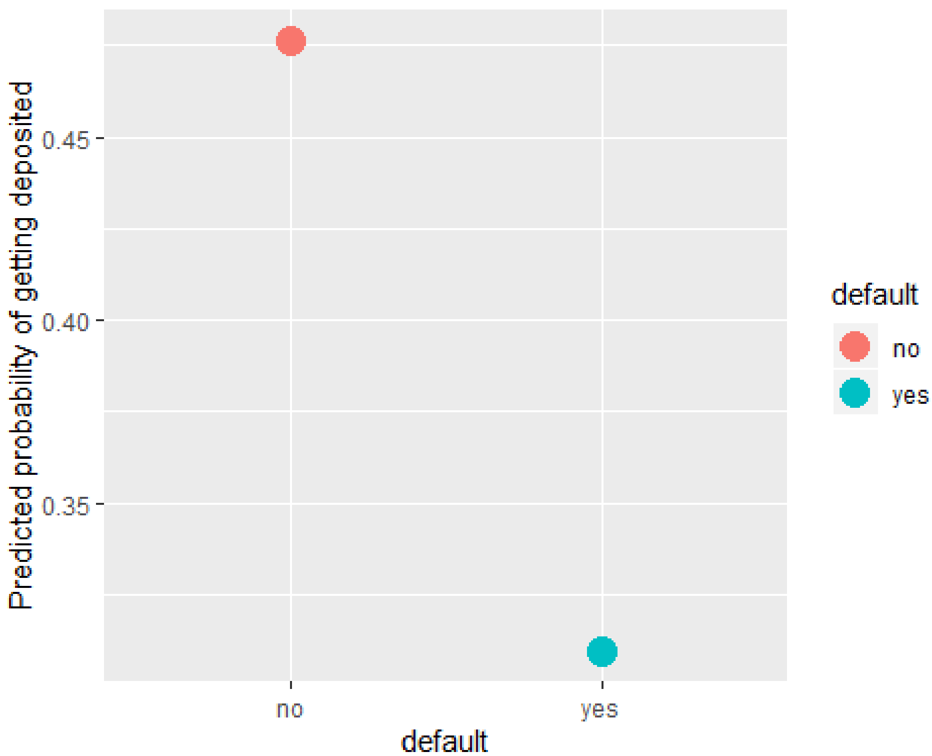
```
# This shows that we need to tell R which columns contain factors it also sho
ws us that there are some missing values. There are "?"s in the dataset.
# These are in the "ca" and "thal" columns. First, convert "?"s to NAs...
bank[bank == "?"] <- NA

## For some logistic regression we'll create a very simple model that uses de
posit to predict default
xtabs(~ deposit + default, data=bank)
```

```
##        default
## deposit   no  yes
##     no  5757  116
##     yes 5237   52
```

```
#Customer who doesnot fall into default category are the one bank should targ
et the most.

logistic_simple <- glm(deposit ~ default, data=bank, family="binomial")
summary(logistic_simple)
```

```
## 
## Call:
## glm(formula = deposit ~ default, family = "binomial", data = bank)
## 
## Deviance Residuals: 
##    Min      1Q  Median      3Q     Max  
## -1.137  -1.137  -1.137   1.218   1.531  
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -0.09467    0.01910  -4.958 7.14e-07 ***
## defaultyes  -0.70768    0.16798  -4.213 2.52e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 15443  on 11161  degrees of freedom
## Residual deviance: 15424  on 11160  degrees of freedom
## AIC: 15428
## 
## Number of Fisher Scoring iterations: 4
```

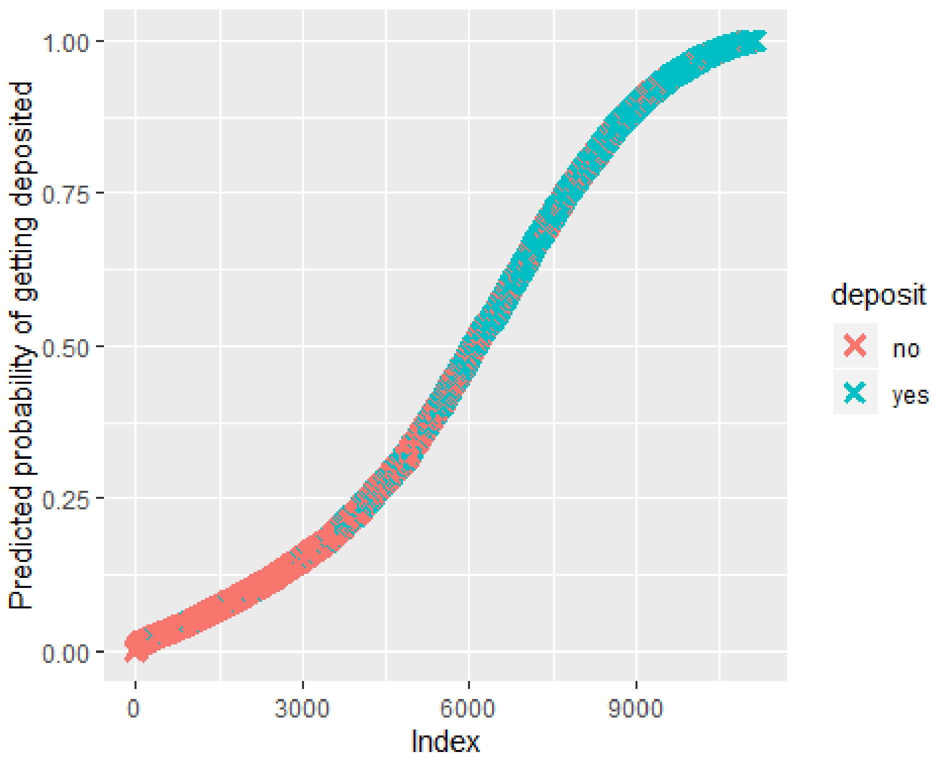
```r
Nodeposit.log.odds <- log(5237 / 5757)
Nodeposit.log.odds

## [1] -0.09466769

Yesdeposit.log.odds.ratio <- log((52 / 116) / (5237/5757))
Yesdeposit.log.odds.ratio

## [1] -0.7076788

## Now calculate the overall "Pseudo R-squared" and its p-value
ll.null <- logistic_simple$null.deviance/-2
ll.proposed <- logistic_simple$deviance/-2
ll.null

## [1] -7721.624

ll.proposed

## [1] -7712.103

## McFadden's Pseudo R^2 = [ LL(Null) - LL(Proposed) ] / LL(Null)
(ll.null - ll.proposed) / ll.null

## [1] 0.001233108

## chi-square value = 2*(LL(Proposed) - LL(Null))
## p-value = 1 - pchisq(chi-square value, df = 2-1)
1 - pchisq(2*(ll.proposed - ll.null), df=1)

## [1] 1.277933e-05

##
predicted.data <- data.frame(probability.of.deposit=logistic_simple$fitted.va
lues,default=bank$default)
head(predicted.data)

##   probability.of.deposit default
## 1              0.4763507      no
## 2              0.4763507      no
## 3              0.4763507      no
## 4              0.4763507      no
## 5              0.4763507      no
## 6              0.4763507      no

## We can plot the data...
ggplot(data=predicted.data, aes(x=default, y=probability.of.deposit)) +
  geom_point(aes(color=default), size=5) +
  xlab("default") +
  ylab("Predicted probability of getting deposited")
```




```
## Since there are only two probabilities (one for default and one for not de
fault),
## we can use a table to summarize the predicted probabilities.
xtabs(~ probability.of.deposit + default, data=predicted.data)

##                         default
## probability.of.deposit    no   yes
##       0.309523809523999     0   168
##       0.476350736765514 10994     0

## Now we will use all of the data available for prediction. This is not the
best way to do this
logistic <- glm(deposit ~ ., data=bank, family="binomial")
summary(logistic)

##
## Call:
## glm(formula = deposit ~ ., family = "binomial", data = bank)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.9952  -0.5996  -0.2117   0.6148   2.8620
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -7.870e-01  2.685e-01  -2.931 0.003383 **
## age                  -7.645e-04  3.204e-03  -0.239 0.811405
```

```
## jobblue-collar     -3.314e-01  1.043e-01  -3.179 0.001480 **
## jobentrepreneur    -3.955e-01  1.762e-01  -2.244 0.024818 *
## jobhousemaid       -4.757e-01  1.911e-01  -2.490 0.012790 *
## jobmanagement      -2.683e-01  1.075e-01  -2.496 0.012563 *
## jobretired          2.972e-01  1.474e-01   2.017 0.043693 *
## jobself-employed   -4.298e-01  1.618e-01  -2.657 0.007886 **
## jobservices        -2.835e-01  1.205e-01  -2.352 0.018673 *
## jobstudent          5.907e-01  1.763e-01   3.351 0.000805 ***
## jobtechnician      -1.567e-01  9.935e-02  -1.578 0.114676
## jobunemployed      -1.169e-01  1.673e-01  -0.698 0.484879
## jobunknown         -3.942e-01  3.446e-01  -1.144 0.252691
## maritalmarried     -1.800e-01  8.566e-02  -2.101 0.035607 *
## maritalsingle       7.670e-02  9.853e-02   0.778 0.436313
## educationsecondary  2.053e-01  9.281e-02   2.212 0.026969 *
## educationtertiary   4.631e-01  1.093e-01   4.236 2.27e-05 ***
## educationunknown    2.640e-01  1.506e-01   1.753 0.079628 .
## defaultyes         -8.455e-03  2.215e-01  -0.038 0.969556
## balance             2.799e-05  8.516e-06   3.287 0.001012 **
## housingyes         -7.001e-01  6.217e-02 -11.260  < 2e-16 ***
## loanyes            -5.019e-01  8.381e-02  -5.988 2.13e-09 ***
## contacttelephone   -5.330e-02  1.080e-01  -0.494 0.621644
## contactunknown     -1.555e+00  9.665e-02 -16.091  < 2e-16 ***
## day                 3.741e-03  3.541e-03   1.056 0.290813
## monthaug           -8.185e-01  1.109e-01  -7.378 1.61e-13 ***
## monthdec            1.373e+00  3.706e-01   3.706 0.000211 ***
## monthfeb           -1.675e-01  1.277e-01  -1.311 0.189690
## monthjan           -1.239e+00  1.671e-01  -7.414 1.22e-13 ***
## monthjul           -9.824e-01  1.122e-01  -8.753  < 2e-16 ***
## monthjun            2.854e-01  1.327e-01   2.151 0.031511 *
## monthmar            2.030e+00  2.289e-01   8.868  < 2e-16 ***
## monthmay           -6.584e-01  1.068e-01  -6.165 7.03e-10 ***
## monthnov           -9.556e-01  1.207e-01  -7.915 2.47e-15 ***
## monthoct            1.080e+00  1.762e-01   6.128 8.92e-10 ***
## monthsep            9.350e-01  1.994e-01   4.688 2.75e-06 ***
## duration            5.469e-03  1.244e-04  43.978  < 2e-16 ***
## campaign           -9.119e-02  1.362e-02  -6.696 2.14e-11 ***
## pdays              -8.934e-05  4.300e-04  -0.208 0.835407
## previous            1.731e-02  1.421e-02   1.218 0.223360
## poutcomeother       8.847e-02  1.331e-01   0.665 0.506246
## poutcomesuccess     2.227e+00  1.416e-01  15.731  < 2e-16 ***
## poutcomeunknown    -2.768e-01  1.376e-01  -2.012 0.044259 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15443.2  on 11161  degrees of freedom
## Residual deviance:  9110.7  on 11119  degrees of freedom
## AIC: 9196.7
```

```
##
## Number of Fisher Scoring iterations: 5

## Now calculate the overall "Pseudo R-squared" and its p-value
ll.null <- logistic$null.deviance/-2
ll.proposed <- logistic$deviance/-2

## McFadden's Pseudo R^2 = [ LL(Null) - LL(Proposed) ] / LL(Null)
(ll.null - ll.proposed) / ll.null

## [1] 0.4100529

## The p-value for the R^2
1 - pchisq(2*(ll.proposed - ll.null), df=(length(logistic$coefficients)-1))

## [1] 0

## now we can plot the data
predicted.data <- data.frame(probability.of.deposit=logistic$fitted.values,de
posit=bank$deposit)
predicted.data <- predicted.data[order(predicted.data$probability.of.deposit,
decreasing=FALSE),]
predicted.data$rank <- 1:nrow(predicted.data)

## Lastly, we can plot the predicted probabilities for each sample having
ggplot(data=predicted.data, aes(x=rank, y=probability.of.deposit)) +
  geom_point(aes(color=deposit), alpha=1, shape=4, stroke=2) +
  xlab("Index") +
  ylab("Predicted probability of getting deposited")
```

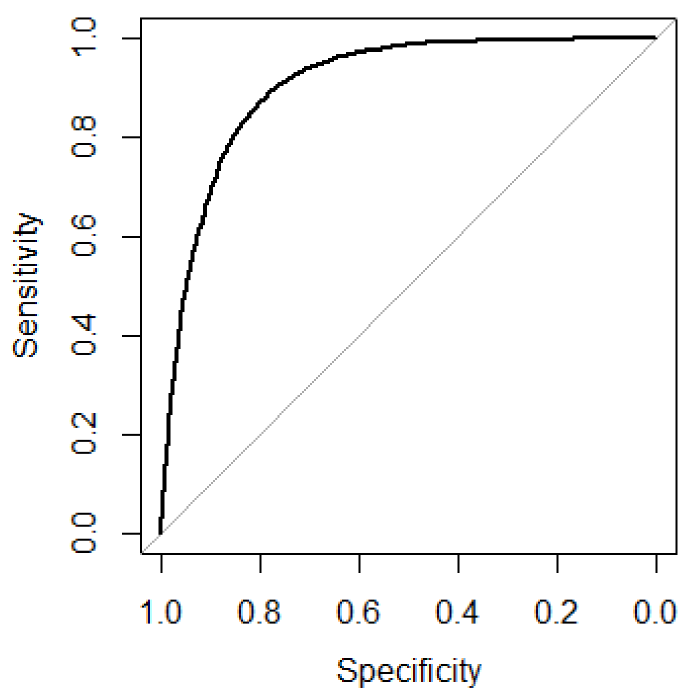



```
#confusion matrix
confusion_matrix(logistic)

##              Predicted no Predicted yes Total
## Actual no            5021           852  5873
## Actual yes           1074          4215  5289
## Total                6095          5067 11162

pdata <- predict(logistic,newdata=bank,type="response" )
head(pdata)

##         1         2         3         4         5         6
## 0.8448000 0.9905914 0.9682944 0.2470517 0.6192912 0.2091538

head(bank$deposit)

## [1] yes yes yes yes yes yes
## Levels: no yes

pdataF <- as.factor(ifelse(test=as.numeric(pdata>0.5) == 0, yes="Deposit", no
="NotDeposit"))


roc(bank$deposit,logistic$fitted.values,plot=TRUE)

## Setting levels: control = no, case = yes

## Setting direction: controls < cases
```

```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
plot = TRUE)
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Area under the curve: 0.905
```

```r
par(pty = "s")
roc(bank$deposit,logistic$fitted.values,plot=TRUE)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

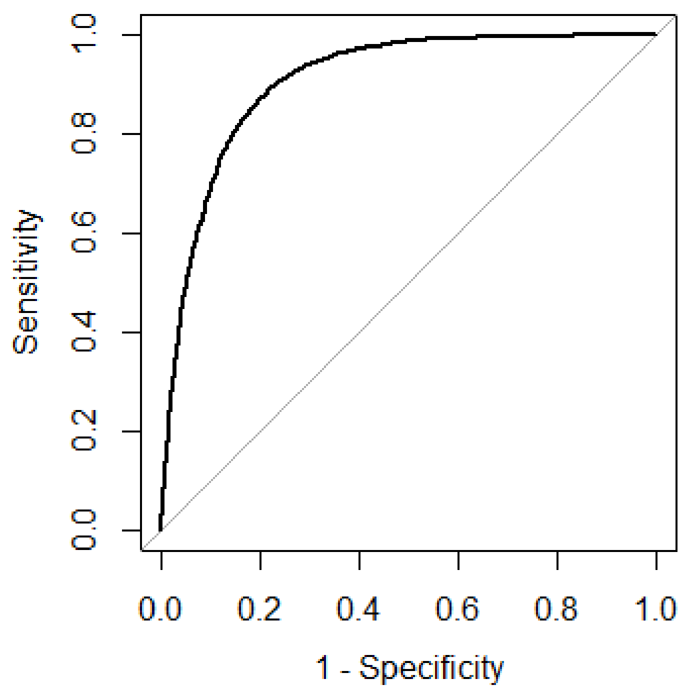



```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
plot = TRUE)
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Area under the curve: 0.905
```

```r
## NOTE: By default, roc() uses specificity on the x-axis and the values rang
e
## from 1 to 0. This makes the graph look like what we would expect, but the
```

```r
## x-axis itself might induce a headache. To use 1-specificity (i.e. the
## False Positive Rate) on the x-axis, set "legacy.axes" to TRUE.
roc(bank$deposit,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE)
```

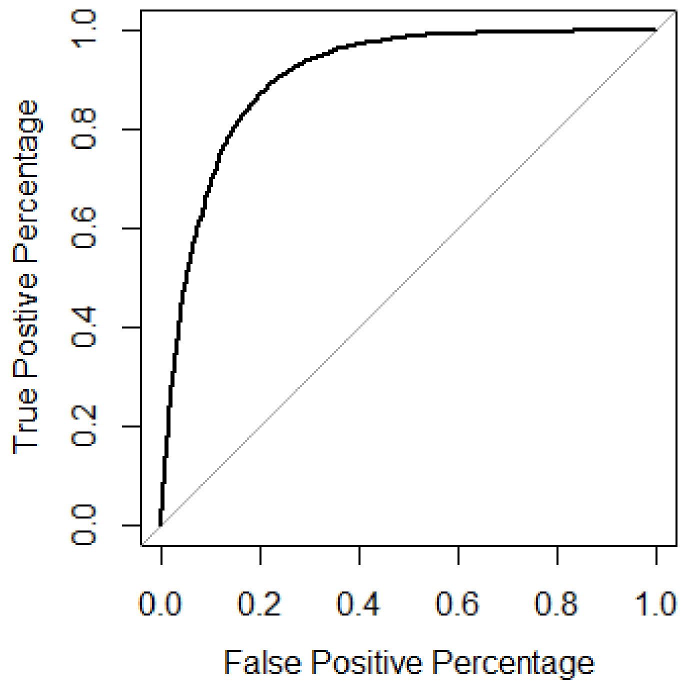
```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```




```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
plot = TRUE, legacy.axes = TRUE)
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Area under the curve: 0.905
```

```r
roc(bank$deposit,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive Percentage")
```

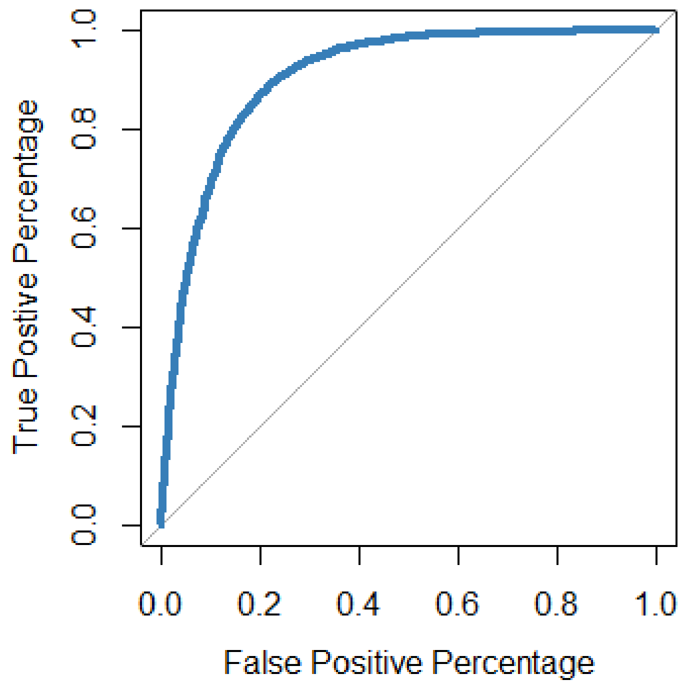
```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```




```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage",     ylab
= "True Postive Percentage")
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Area under the curve: 0.905
```

```
roc(bank$deposit,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=
4)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```




```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage",     ylab
= "True Postive Percentage", col = "#377eb8", lwd = 4)
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Area under the curve: 0.905
```

```
roc(bank$deposit,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=
4)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Percentage",     ylab
= "True Postive Percentage", col = "#377eb8", lwd = 4)
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Area under the curve: 0.905
```

```r
## If we want to find out the optimal threshold we can store the
## data used to make the ROC graph in a variable...
roc.info <- roc(bank$deposit, logistic$fitted.values, legacy.axes=TRUE)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```r
str(roc.info)
```

```
## List of 15
##  $ percent           : logi FALSE
##  $ sensitivities     : num [1:11163] 1 1 1 1 1 1 1 1 1 1 ...
##  $ specificities     : num [1:11163] 0 0.00017 0.000341 0.000511 0.000681
...
##  $ thresholds        : num [1:11163] -Inf 0.00025 0.000523 0.00085 0.00136
5 ...
##  $ direction         : chr "<"
##  $ cases             : Named num [1:5289] 0.845 0.991 0.968 0.247 0.619 ..
.
##   ..- attr(*, "names")= chr [1:5289] "1" "2" "3" "4" ...
##  $ controls          : Named num [1:5873] 0.339 0.117 0.661 0.126 0.199 ..
.
##   ..- attr(*, "names")= chr [1:5873] "5290" "5291" "5292" "5293" ...
##  $ fun.sesp          :function (thresholds, controls, cases, direction)
##  $ auc               : 'auc' num 0.905
##   ..- attr(*, "partial.auc")= logi FALSE
##   ..- attr(*, "percent")= logi FALSE
##   ..- attr(*, "roc")=List of 15
##   .. ..$ percent           : logi FALSE
##   .. ..$ sensitivities     : num [1:11163] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..$ specificities     : num [1:11163] 0 0.00017 0.000341 0.000511 0.0
00681 ...
##   .. ..$ thresholds        : num [1:11163] -Inf 0.00025 0.000523 0.00085 0
.001365 ...
##   .. ..$ direction         : chr "<"
##   .. ..$ cases             : Named num [1:5289] 0.845 0.991 0.968 0.247 0.
619 ...
##   .. .. ..- attr(*, "names")= chr [1:5289] "1" "2" "3" "4" ...
##   .. ..$ controls          : Named num [1:5873] 0.339 0.117 0.661 0.126 0.
199 ...
##   .. .. ..- attr(*, "names")= chr [1:5873] "5290" "5291" "5292" "5293" ...
##   .. ..$ fun.sesp          :function (thresholds, controls, cases, directi
on)
##   .. ..$ auc               : 'auc' num 0.905
##   .. .. ..- attr(*, "partial.auc")= logi FALSE
##   .. .. ..- attr(*, "percent")= logi FALSE
##   .. .. ..- attr(*, "roc")=List of 8
##   .. .. .. ..$ percent      : logi FALSE
##   .. .. .. ..$ sensitivities: num [1:11163] 1 1 1 1 1 1 1 1 1 1 ...
##   .. .. .. ..$ specificities: num [1:11163] 0 0.00017 0.000341 0.000511 0.
```

```
000681 ...
##   .. .. .. ..$ thresholds   : num [1:11163] -Inf 0.00025 0.000523 0.00085
0.001365 ...
##   .. .. .. ..$ direction    : chr "<"
##   .. .. .. ..$ cases        : Named num [1:5289] 0.845 0.991 0.968 0.247 0
.619 ...
##   .. .. .. .. ..- attr(*, "names")= chr [1:5289] "1" "2" "3" "4" ...
##   .. .. .. ..$ controls     : Named num [1:5873] 0.339 0.117 0.661 0.126 0
.199 ...
##   .. .. .. .. ..- attr(*, "names")= chr [1:5873] "5290" "5291" "5292" "529
3" ...
##   .. .. .. ..$ fun.sesp     :function (thresholds, controls, cases, direct
ion)
##   .. .. .. ..- attr(*, "class")= chr "roc"
##   .. ..$ call              : language roc.default(response = bank$deposit,
predictor = logistic$fitted.values,      legacy.axes = TRUE)
##   .. ..$ original.predictor: Named num [1:11162] 0.845 0.991 0.968 0.247 0
.619 ...
##   .. .. ..- attr(*, "names")= chr [1:11162] "1" "2" "3" "4" ...
##   .. ..$ original.response : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2
2 2 2 ...
##   .. ..$ predictor         : Named num [1:11162] 0.845 0.991 0.968 0.247 0
.619 ...
##   .. .. ..- attr(*, "names")= chr [1:11162] "1" "2" "3" "4" ...
##   .. ..$ response          : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2
2 2 2 ...
##   .. ..$ levels            : chr [1:2] "no" "yes"
##   .. ..- attr(*, "class")= chr "roc"
##  $ call                : language roc.default(response = bank$deposit, predi
ctor = logistic$fitted.values,      legacy.axes = TRUE)
##  $ original.predictor: Named num [1:11162] 0.845 0.991 0.968 0.247 0.619 .
..
##   ..- attr(*, "names")= chr [1:11162] "1" "2" "3" "4" ...
##  $ original.response : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2
...
##  $ predictor         : Named num [1:11162] 0.845 0.991 0.968 0.247 0.619 .
..
##   ..- attr(*, "names")= chr [1:11162] "1" "2" "3" "4" ...
##  $ response          : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2
...
##  $ levels            : chr [1:2] "no" "yes"
##  - attr(*, "class")= chr "roc"

roc.df <- data.frame(tpp=roc.info$sensitivities*100, ## tpp = true positive p
ercentage
                     fpp=(1 - roc.info$specificities)*100, ## fpp = false pos
itive precentage
                     thresholds=roc.info$thresholds)
head(roc.df) ## head() will show us the values for the upper right-hand corne
r of the ROC graph, when the threshold is so low
```

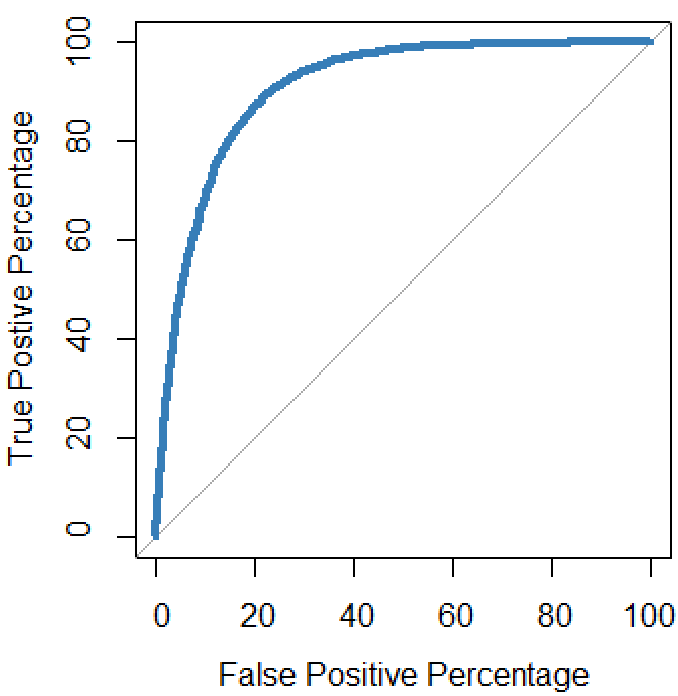
```
##   tpp       fpp   thresholds
## 1 100 100.00000         -Inf
## 2 100  99.98297 0.0002500226
## 3 100  99.96595 0.0005229565
## 4 100  99.94892 0.0008497192
## 5 100  99.93189 0.0013647133
## 6 100  99.91486 0.0018716406
```

```
tail(roc.df) ## tail() will show us the values for the lower left-hand corner
```

```
##              tpp        fpp thresholds
## 11158 0.07562866 0.01702707  0.9999996
## 11159 0.05672150 0.01702707  0.9999998
## 11160 0.03781433 0.01702707  0.9999999
## 11161 0.01890717 0.01702707  1.0000000
## 11162 0.01890717 0.00000000  1.0000000
## 11163 0.00000000 0.00000000        Inf
```

```
## now let's look at the thresholds between TPP 60% and 80%
#roc.df[roc.df$tpp > 60 & roc.df$tpp < 80,]
roc(bank$deposit,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=
4, percent=TRUE)
```

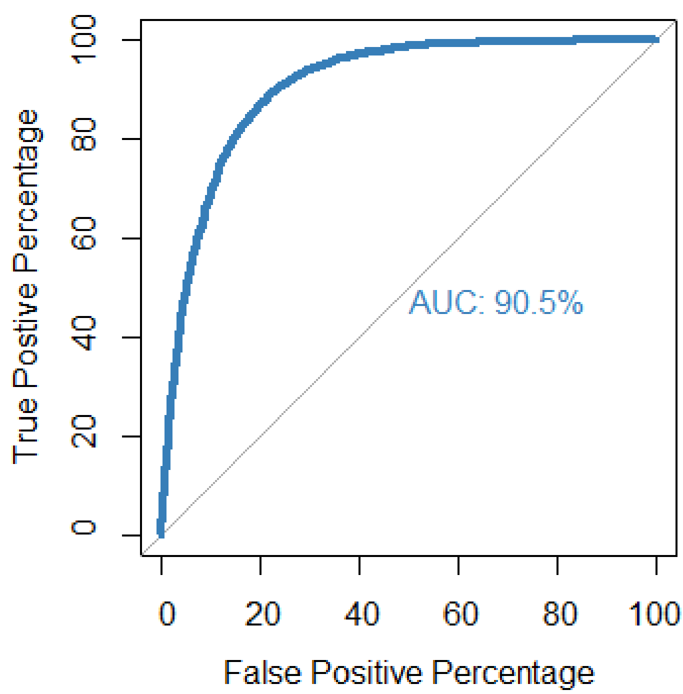
```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```



```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Perce
ntage",     ylab = "True Postive Percentage", col = "#377eb8", lwd = 4)
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Area under the curve: 90.5%
```

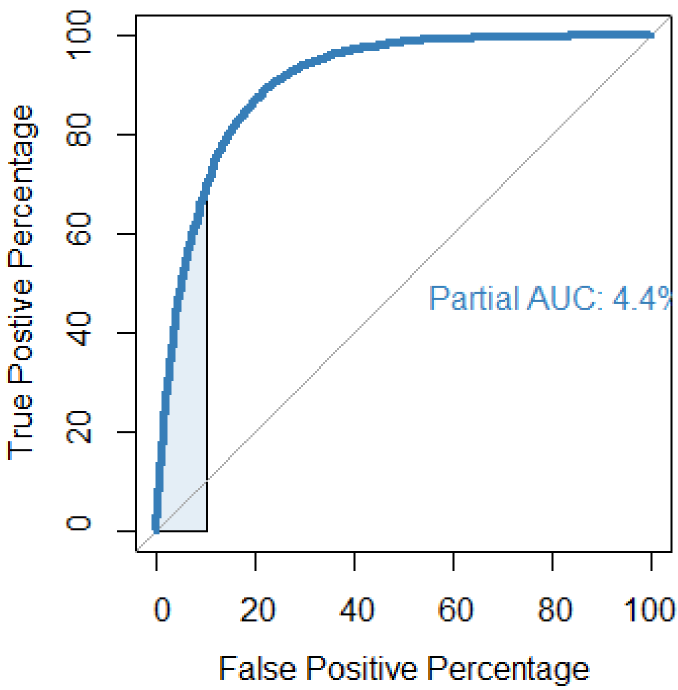
```
roc(bank$deposit,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=
4, percent=TRUE, print.auc=TRUE)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```




```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Perce
ntage",     ylab = "True Postive Percentage", col = "#377eb8", lwd = 4,     p
rint.auc = TRUE)
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Area under the curve: 90.5%
```

```r
roc(bank$deposit,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="Fa
lse Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=
4, percent=TRUE, print.auc=TRUE, partial.auc=c(100, 90), auc.polygon = TRUE,
auc.polygon.col = "#377eb822", print.auc.x=45)

## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```




```r
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic$fitted.values,
percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "False Positive Perce
ntage",     ylab = "True Postive Percentage", col = "#377eb8", lwd = 4,     p
rint.auc = TRUE, partial.auc = c(100, 90), auc.polygon = TRUE,     auc.polygo
n.col = "#377eb822", print.auc.x = 45)
##
## Data: logistic$fitted.values in 5873 controls (bank$deposit no) < 5289 cas
es (bank$deposit yes).
## Partial area under the curve (specificity 100%-90%): 4.391%

# Lets do two roc plots to understand which model is better
roc(bank$deposit, logistic_simple$fitted.values, plot=TRUE, legacy.axes=TRUE,
percent=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage
", col="#377eb8", lwd=4, print.auc=TRUE)

## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```
##
## Call:
## roc.default(response = bank$deposit, predictor = logistic_simple$fitted.va
lues,      percent = TRUE, plot = TRUE, legacy.axes = TRUE, xlab = "False Posi
tive Percentage",      ylab = "True Postive Percentage", col = "#377eb8", lwd
= 4,      print.auc = TRUE)
##
## Data: logistic_simple$fitted.values in 5873 controls (bank$deposit no) < 5
289 cases (bank$deposit yes).
## Area under the curve: 50.5%
```

```
# Lets add the other graph
plot.roc(bank$deposit, logistic$fitted.values, percent=TRUE, col="#4daf4a", l
wd=4, print.auc=TRUE, add=TRUE, print.auc.y=40)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```
legend("bottomright", legend=c("Simple", "Non Simple"), col=c("#377eb8", "#4d
af4a"), lwd=4) # Make it user friendly
```