BA\_Project

Satyam Prakash

11 December 2017

## Project

The aim is to build a Logistic Regression Model in order to predict a categorical variable named as default10yr in creditset.csv file. The predictors are age, income and loan.

## Solution

In order to predict a categorical variable using Logistic Regression we have to first train our model and then test it by splitting the dataset into 2 parts

# Importing the Dataset

# creditset< read.csv("file.choose()"")  
# Alternatively, we can use the import dataset feature in R  
creditset<- read.csv("G:/R language/creditset.csv")  
View(creditset)  
dim(creditset)

## [1] 2000 6

names(creditset)

## [1] "clientid" "income" "age" "loan" "LTI"   
## [6] "default10yr"

We can se that there are a total of 2000 oberservations namely clientid, income,age, loan, LTI, default10yr out of which we need to predict response variable Default10yr which is a categorical variable hence we have to make a Linear Regression Model

library(caTools)

## Warning: package 'caTools' was built under R version 3.4.3

table(creditset$default10yr)

##   
## 0 1   
## 1717 283

set.seed(2)  
split<-sample.split(creditset,SplitRatio = 0.75)  
split

## [1] TRUE TRUE TRUE TRUE FALSE FALSE

traindata<- subset(creditset,split=="TRUE")  
dim(traindata)

## [1] 1334 6

testdata<- subset(creditset, split== "FALSE")  
dim(testdata)

## [1] 666 6

names(creditset)

## [1] "clientid" "income" "age" "loan" "LTI"   
## [6] "default10yr"

We now check the actual counts of 0’s and 1’s of default110yr in creditset, testting dataset and training datset. by keepin 0 as a non defaulter and 1 as a defaulter

table(creditset$default10yr) #counts the number of zero's and non zero's in creditset

##   
## 0 1   
## 1717 283

table(traindata$default10yr) #counts the number of zero's and one's in training dataset

##   
## 0 1   
## 1133 201

table(testdata$default10yr) #counts the number of zero's and ones in testing dataset

##   
## 0 1   
## 584 82

### Building Logistic Regression Model

model<- glm(default10yr~income+age+loan, data=creditset,family=binomial)  
model

##   
## Call: glm(formula = default10yr ~ income + age + loan, family = binomial,   
## data = creditset)  
##   
## Coefficients:  
## (Intercept) income age loan   
## 9.9414690 -0.0002434 -0.3493025 0.0017343   
##   
## Degrees of Freedom: 1999 Total (i.e. Null); 1996 Residual  
## Null Deviance: 1631   
## Residual Deviance: 440.7 AIC: 448.7

Hence our Logit function predictors are as follows  
   
#Constant Term is 9.9414690  
#Coefficient for income is -0.0002434  
#Coefficient for age is -0.3493025  
#Coefficient for loan is 0.0017343

# Our Logit Function becomes

#logit(odds) =logit(default10yr) = 9.9414690 +((-0.0002434)\* income) + ((-0.3493025)\* age) + ((0.0017343)\* loan)  
  
#p(default10yr) = e^(9.9414690 +((-0.0002434)\* income) + ((-0.3493025)\* age) + ((0.0017343)\* loan)) / 1+e^(9.9414690 +((-0.0002434)\* income) + ((-0.3493025)\* age) + ((0.0017343)\* loan))

### Optimizing our Model

#Trying to build different set of predictors in order to optimise the model  
model1 <-glm(default10yr~income,data=traindata,family=binomial)  
summary(model1)

##   
## Call:  
## glm(formula = default10yr ~ income, family = binomial, data = traindata)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.5958 -0.5813 -0.5674 -0.5523 1.9862   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.897e+00 2.536e-01 -7.480 7.44e-14 \*\*\*  
## income 3.687e-06 5.292e-06 0.697 0.486   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1130.9 on 1333 degrees of freedom  
## Residual deviance: 1130.4 on 1332 degrees of freedom  
## AIC: 1134.4  
##   
## Number of Fisher Scoring iterations: 4

model2 <-glm(default10yr~age, data=traindata, family=binomial)  
summary(model2)

##   
## Call:  
## glm(formula = default10yr ~ age, family = binomial, data = traindata)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.39150 -0.48762 -0.21609 -0.09672 2.02203   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 3.06921 0.32098 9.562 <2e-16 \*\*\*  
## age -0.14283 0.01076 -13.278 <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: 1130.91 on 1333 degrees of freedom  
## Residual deviance: 809.99 on 1332 degrees of freedom  
## AIC: 813.99  
##   
## Number of Fisher Scoring iterations: 6

model3<- glm(default10yr~loan, data=traindata, family=binomial)  
summary(model3)

##   
## Call:  
## glm(formula = default10yr ~ loan, family = binomial, data = traindata)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6076 -0.5434 -0.3574 -0.2567 2.3937   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.667e+00 1.981e-01 -18.51 <2e-16 \*\*\*  
## loan 3.450e-04 2.748e-05 12.56 <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: 1130.91 on 1333 degrees of freedom  
## Residual deviance: 941.76 on 1332 degrees of freedom  
## AIC: 945.76  
##   
## Number of Fisher Scoring iterations: 5

model4<- glm(default10yr~age+income,data=traindata, family=binomial)  
summary(model4)

##   
## Call:  
## glm(formula = default10yr ~ age + income, family = binomial,   
## data = traindata)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.39233 -0.48738 -0.21597 -0.09667 2.02211   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 3.065e+00 4.347e-01 7.050 1.79e-12 \*\*\*  
## age -1.428e-01 1.077e-02 -13.266 < 2e-16 \*\*\*  
## income 8.987e-08 6.139e-06 0.015 0.988   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1130.91 on 1333 degrees of freedom  
## Residual deviance: 809.99 on 1331 degrees of freedom  
## AIC: 815.99  
##   
## Number of Fisher Scoring iterations: 6

model5<- glm(default10yr~age+loan, data=traindata, family= binomial)  
summary(model5)

##   
## Call:  
## glm(formula = default10yr ~ age + loan, family = binomial, data = traindata)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.22066 -0.28396 -0.08938 -0.01918 2.60955   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.904e+00 3.987e-01 4.776 1.79e-06 \*\*\*  
## age -2.147e-01 1.654e-02 -12.986 < 2e-16 \*\*\*  
## loan 6.128e-04 4.831e-05 12.686 < 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: 1130.91 on 1333 degrees of freedom  
## Residual deviance: 521.43 on 1331 degrees of freedom  
## AIC: 527.43  
##   
## Number of Fisher Scoring iterations: 7

model6<- glm(default10yr~income+loan, data= traindata, family= binomial)  
summary(model6)

##   
## Call:  
## glm(formula = default10yr ~ income + loan, family = binomial,   
## data = traindata)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6818 -0.5102 -0.2789 -0.1034 2.2054   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.429e+00 2.842e-01 -5.028 4.97e-07 \*\*\*  
## income -9.240e-05 1.074e-05 -8.607 < 2e-16 \*\*\*  
## loan 6.568e-04 5.181e-05 12.677 < 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: 1130.91 on 1333 degrees of freedom  
## Residual deviance: 842.77 on 1331 degrees of freedom  
## AIC: 848.77  
##   
## Number of Fisher Scoring iterations: 6

#Among all the models that we used, we found out that the AIC value has not decreased hence we can say that age, loan and income all are good predictors, so we use an optimised model for predicting our final model

### Predictions of the given Dataset

library(CARS)

## Warning: package 'CARS' was built under R version 3.4.3

res<- predict(model, traindata, type="response")  
head(res,n=5)

## 1 2 3 4 7   
## 3.008809e-06 2.071455e-02 5.339626e-06 2.877104e-03 2.167951e-01

#Confusion Matrix  
 #Threshold Probe is taken as 0.5 by default  
  
tabtrain<-table(ActualValue=traindata$default10yr, predictiveValue=res>=0.5)  
tabtrain

## predictiveValue  
## ActualValue FALSE TRUE  
## 0 1107 26  
## 1 42 159

accuracytrain<- sum(diag(tabtrain)/sum(tabtrain))  
accuracytrain

## [1] 0.9490255

Hence model built is accurate upto 95% meaning that the accuracy in the prediction was 95% correct

### Predicting Probabilities for the testing dataset

#Checking for accuracy with testing dataset  
#Finding the Predicted Probabilities associated with observations using our model for testing dataset  
res1<- predict(model,testdata ,type="response")  
head(res1)

## 5 6 11 12 17   
## 9.142142e-01 9.518862e-08 8.329547e-04 6.796359e-05 4.735863e-04   
## 18   
## 9.352335e-08

#Confusion Matrix for our results when compared with predicted values for testing dataset.  
tabtest<-table(ActualValue=testdata$default10yr, predictiveValue=res1>=0.5) #with probability cutoff= 0.5  
tabtest

## predictiveValue  
## ActualValue FALSE TRUE  
## 0 565 19  
## 1 15 67

accuracytest<- sum(diag(tabtest)/sum(tabtest))  
accuracytest

## [1] 0.9489489

Our Model is 95% accurate

### ROC Curve

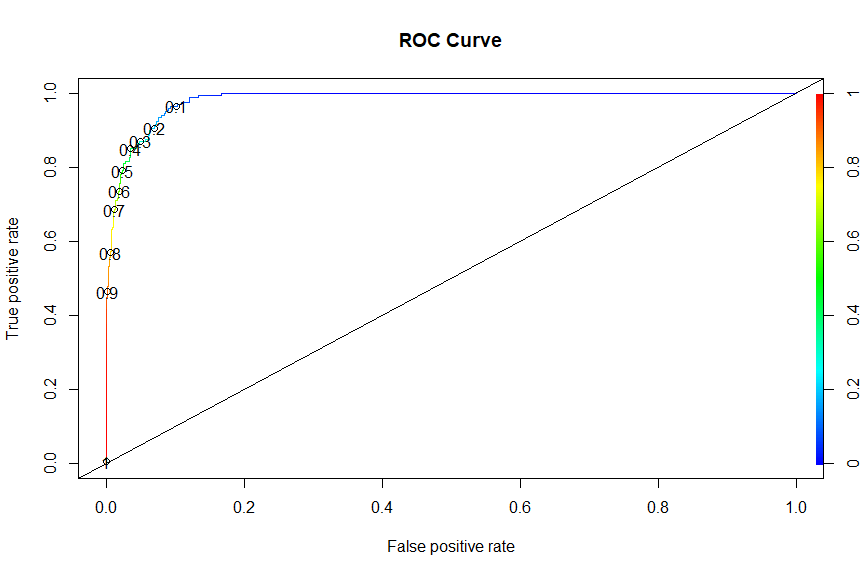
## Warning: package 'ROCR' was built under R version 3.4.3

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.4.3

##   
## Attaching package: 'gplots'

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



#Now we check the confusion matrix for different probabilities of our Dataset  
table(ActualValue=traindata$default10yr,PredictedValue=res>=0.3)

## PredictedValue  
## ActualValue FALSE TRUE  
## 0 1078 55  
## 1 26 175

table(ActualValue=traindata$default10yr,PredictedValue=res>=0.4)

## PredictedValue  
## ActualValue FALSE TRUE  
## 0 1093 40  
## 1 30 171

table(ActualValue=traindata$default10yr,PredictedValue=res>=0.5)

## PredictedValue  
## ActualValue FALSE TRUE  
## 0 1107 26  
## 1 42 159

#So, we see that 0.4 gives the best accuracy with least false negatives  
#Thus, the confusion matrix would be  
tabtrain0.4<- table(AxtualValue=traindata$default10yr,PredictedValue=res>=0.4)  
tabtrain

## predictiveValue  
## ActualValue FALSE TRUE  
## 0 1107 26  
## 1 42 159

#checking actual zero's and ones's in the given dataset  
table(traindata$default10yr)

##   
## 0 1   
## 1133 201

# Sensitivity and Specitivity for Accuracy

# Calculating Sensitivity and Specitivity for dataset training for p=0.4  
sensitivitytest<- tabtrain0.4[4]/(tabtrain0.4[2]+ tabtrain0.4[4])  
sensitivitytest

## [1] 0.8507463

specificitytest<- tabtrain0.4[1]/(tabtrain0.4[1]+ tabtrain0.4[3])  
specificitytest

## [1] 0.9646955

#Classification   
#Classification  
#classification Accuracy  
classification\_at<-(sensitivitytest+specificitytest)/2  
classification\_at

## [1] 0.9077209

#Overall Accuracy  
#(TruePositive+ TrueNegative)/ (TP+TN+FP+FN)  
  
accuracy\_tabtrain<- sum(diag(tabtrain0.4))/sum(tabtrain0.4)  
accuracy\_tabtrain

## [1] 0.9475262

### —————————————————————

# Verifying our model with testing dataset

# Checking Actual 0's and 1's in the tabtest dataset  
table(testdata$default10yr)

##   
## 0 1   
## 584 82

#The Confusion Matrix for the test will be  
tabtest0.4<- table(ActualVlaue=testdata$default10yr,PredictedValue=res1>=0.4)  
tabtest0.4

## PredictedValue  
## ActualVlaue FALSE TRUE  
## 0 559 25  
## 1 10 72

#Veryfying Our Model  
  
#Calculating Sensitivity  
sensitivitytest<- tabtest0.4[4]/(tabtest0.4[2]+ tabtest0.4[4])  
sensitivitytest

## [1] 0.8780488

specificitytest<- tabtest0.4[1]/(tabtest0.4[1]+ tabtest0.4[3])  
specificitytest

## [1] 0.9571918

#Classification Accuracy for group 0 is shown  
#Classification Accuracy for group 1 is shown  
classification\_at<-(sensitivitytest+specificitytest)/2  
classification\_at

## [1] 0.9176203

#Overall Accuracy  
#(TruePositive+ TrueNegative)/ (TP+TN+FP+FN)  
accuracy\_tabtrain<- sum(diag(tabtest0.4))/sum(tabtest0.4)  
accuracy\_tabtrain

## [1] 0.9474474

Our Logistic Regression is a good model because it was able to predict before data was given to it. The model is very less risky

#Calculating Area under ROC curve  
#install.packages("verification")  
library(verification)

## Warning: package 'verification' was built under R version 3.4.3

## Warning: package 'fields' was built under R version 3.4.3

## Warning: package 'spam' was built under R version 3.4.3

## Warning: package 'dotCall64' was built under R version 3.4.3

## Warning: package 'maps' was built under R version 3.4.3

## Warning: package 'CircStats' was built under R version 3.4.3

## Warning: package 'dtw' was built under R version 3.4.3

## Warning: package 'proxy' was built under R version 3.4.3

roc.area(traindata$default10yr,res)

## $A  
## [1] 0.98262  
##   
## $n.total  
## [1] 1334  
##   
## $n.events  
## [1] 201  
##   
## $n.noevents  
## [1] 1133  
##   
## $p.value  
## [1] 5.319821e-106

#The Area under ROC curve is as shown

#Calculating nagelkerke's R square  
#install.packages("fmsb")  
library(fmsb)  
NagelkerkeR2(model)

## $N  
## [1] 2000  
##   
## $R2  
## [1] 0.8043509