RBrownBag Nr. 4 Regression

Credit Card

Gary R. Seamans

March 28, 2018

Contents

List of Figures

List of Tables

Problem Description

Lending that results in default is very costly and for this dataset I'll use logistic regression for determining the probability of default.

- Use duration, amount, installment, and age in this analysis, along with loan history, purpose, and rent.
- Use random selection for 900 cases to train the program, and then the other 100 cases will be used for testing.
- Test the program, and describe the results.

Introduction and Setup

This example was written in RMarkdown and the PDF was generated using Pandoc ("Pandoc - About pandoc," n.d.) and Latex ("Introduction to LaTeX," n.d.). Paragraph spacing was set to 1.5 lines.

For this problem the German credit data from ("UCI Machine Learning Repository," n.d.) was used. Each step of the R processing is described.

Load the Data

The first step is to load, and if necessary clean, the German Credit Data. The data is in a CSV file *germancredit.csv* and contains a header row.

```
gCreditData <- read.csv("germancredit.csv")
```

Once the data was loaded I took a look at the types of data using the *str()* function. I also used a custom function to turn the *str()* output into a table that I could manipulate with *xtable()*. The custom function, *strtable()* was downloaded from the **R-Bloggers** site ("Str Implementation for Data Frames | R-bloggers," n.d.). There are 21 variables in the German Credit card data.

```
print(xtable(print(strtable(gCreditData))), include.rownames = FALSE)
```

Table 1: German Credit Card Data Types

variable	class	levels	examples
Default	integer		0, 1, 0, 0, ...
checkingstatus1	Factor w/ 4 levels	"A11", "A12", "A13", "A14"	"A11", "A12", "A14",...
duration	integer		6, 48, 12, 42, ...
history	Factor w/ 5 levels	"A30", "A31", "A32", "A33", ...	"A34", "A32", "A34", ...
purpose	Factor w/ 10 levels	"A40", "A41", "A410", "A42", ...	"A43", "A43", "A46", ...
amount	integer		1169, 5951, 2096, 7882, ...
savings	Factor w/ 5 levels	"A61", "A62", "A63", "A64", ...	"A65", "A61", "A61", ...
employ	Factor w/ 5 levels	"A71", "A72", "A73", "A74", ...	"A75", "A73", "A74", ...
installment	integer		4, 2, 2, 2, ...
status	Factor w/ 4 levels	"A91", "A92", "A93", "A94"	"A93", "A92", "A93", ...
others	Factor w/ 3 levels	"A101", "A102", "A103"	"A101", "A101", "A101", ...
residence	integer		4, 2, 3, 4, ...
property	Factor w/ 4 levels	"A121", "A122", "A123", "A124"	"A121", "A121", ...
age	integer		67, 22, 49, 45, ...
otherplans	Factor w/ 3 levels	"A141", "A142", "A143"	"A143", "A143", "A143", ...
housing	Factor w/ 3 levels	"A151", "A152", "A153"	"A152", "A152", "A152", ...
cards	integer		2, 1, 1, 1, ...
job	Factor w/ 4 levels	"A171", "A172", "A173", "A174"	"A173", "A173", ...
liable	integer		1, 1, 2, 2, ...
tele	Factor w/ 2 levels	"A191", "A192"	"A192", "A191", "A191", ...
foreign	Factor w/ 2 levels	"A201", "A202"	"A201", "A201", "A201", ...

Prepare the data

Preparing the data involved reducing the number of parameters, per the problem description, and getting the data ready to be used in a logistic regression model.

Reducing the Parameters

The first step was to reduce the number of parameters to 8, versus 21. To accomplish this I used the R *subset()* function.

```
## Specify the names of the columns to include in the subset to be processed
gCDSubsetNames <- c("Default","duration","amount","installment","age","history","purpose","housing")

## Create the subset that will be processed
gCDSubset <- gCreditData[gCDSubsetNames]
```

Checking for missing Values

```
## Check for missing data
print(xtable(data.frame(as.list(sapply(gCDSubset, function(x)
  sum(is.na(x)))))), include.rownames = FALSE)
```

There were no missing values, as can be seen below.

Table 2: Check for Missing Values

Default	duration	amount	installment	age	history	purpose	housing
0	0	0	0	0	0	0	0

I could also use the *describe* function from the *Hmsic* package for the same purpose and to get additional insights into the data, but for now I'm mostly concerned with any missing values there might be.

Data Modifications

The next steps were to convert the factor names into something more readable than the default names provided.

```
## Default has just two values so lets change it to a factor
gCDSubset$Default <- as.factor(gCDSubset$Default)

## Rename the fields to make it more readable
gCDSubset$Default <- mapvalues(gCDSubset$Default, from = c("0","1"), to = c("No", "Yes"))
gCDSubset$history <- mapvalues(gCDSubset$history, from = c("A30","A31", "A32", "A33","A34"),
          to = c("noCreditsTaken","paidBackDuly", "paidBackUntilNow", "delayInPayback","criticalAccount
```

```
gCDSubset$purpose <- mapvalues(gCDSubset$purpose,
          from = c("A40","A41","A42","A43","A44","A45","A46","A48","A49", "A410"),
          to = c("newCar","usedCar","furnitureEquip","radioTV",
                 "appliances","repairs","education","retraining","business", "other"))
gCDSubset$housing <- mapvalues(gCDSubset$housing, from = c("A151","A152","A153"), to = c("rent","own",
Then I took a look at the data types using the same method as prviously.
```

```
print(xtable(print(strtable(gCDSubset))), include.rownames = FALSE)
```

Table 3: Subset German Credit Card Data Types

variable	class	levels	examples
Default	Factor w/ 2 levels	"No", "Yes"	"No", "Yes", "No", "No", ...
duration	integer		6, 48, 12, 42, ...
amount	integer		1169, 5951, 2096, 7882, ...
installment	integer		4, 2, 2, 2, ...
age	integer		67, 22, 49, 45, ...
history	Factor w/ 5 levels	"noCreditsTaken","paidBackDuly", "paidBackUntilNow", ...	"criticalAccount", "paidBackUntilNow", "criticalAccount", ...
purpose	Factor w/ 10 levels	"newCar","usedCar","other", "furnitureEquip", ...	"radioTV", "radioTV", "education", "furnitureEquip", ...
housing	Factor w/ 3 levels	"rent", "own", "free"	"own", "own", "own", "free", ...

Table 3 looks much more readable with the modifications to the factor names.

Create the Training and Test datasets

Next I selected 900 random rows for the training set and used the remainder for the test set. The indices to use for the traning set were selected first then used to generate both the training and test sets.

```
## Select 900 random row numbers for the training set
trainIndices <- sample(1:nrow(gCDSubset), 900)

## Create the training set using the trainIndices
gCDTrain <- gCDSubset[trainIndices,]
nrow(gCDTrain)
[1] 900
## Create the test set, 100 rows, using the non-training set rows
gCDTest <- gCDSubset[-trainIndices,]
nrow(gCDTest)
[1] 100
```

Creating the Logistic Regression Model

The next step is to create the logistic regression model using the training data and examine the coefficients table.

```
gCDTrainGlm <- glm(Default ~ duration + amount + installment + age + history +
                 purpose + housing, family = binomial(link ='logit'), data = gCDTrain)
print(xtable(summary(gCDTrainGlm)$coef, caption = "gCDTrainGlm Coefficients"))
print(xtable(exp(cbind(Odds = coef(gCDTrainGlm), confint(gCDTrainGlm))), caption = "Odds and CI"))
```

Table 4: gCDTrainGlm Coefficients

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.39	0.55	0.70	0.48
duration	0.02	0.01	2.50	0.01
amount	0.00	0.00	2.34	0.02
installment	0.25	0.08	3.14	0.00
age	-0.02	0.01	-2.44	0.01
history.paidBackDuly	0.43	0.51	0.83	0.41
history.paidBackUntilNow	-1.02	0.38	-2.68	0.01
history.delayInPayback	-1.16	0.44	-2.65	0.01
history.criticalAccount	-1.90	0.41	-4.68	0.00
purpose.usedCar	-1.57	0.34	-4.58	0.00
purpose.other	-0.85	0.73	-1.17	0.24
purpose.furnitureEquip	-0.49	0.24	-2.05	0.04
purpose.radioTV	-0.97	0.23	-4.18	0.00
purpose.appliances	-0.35	0.68	-0.52	0.60
purpose.repairs	-0.35	0.55	-0.65	0.52
purpose.education	0.12	0.37	0.32	0.75
purpose.retraining	-2.15	1.12	-1.92	0.05
purpose.business	-0.66	0.31	-2.16	0.03
housing.own	-0.56	0.21	-2.69	0.01
housing.free	-0.07	0.32	-0.22	0.83

Deviance and AIC

```
Null deviance: 1096.15  on 899  degrees of freedom
Residual deviance:  944.48  on 880  degrees of freedom
AIC: 984.48
```

Examining the *z value* and *Pr(>|z|)* there appear to be a number of parameters, highlighted in red, that are not significantly contributing to the model. I could remove just some of the factor levels or I could combine the insignificant levels into a new level, but that could bias the outcome so I'll check the fit using the chi-square, and p-value.

Checking the Model Fit

```
## Check chi-square, df, and p-value
with(gCDTrainGlm, null.deviance - deviance)
[1] 151.6648 # chi-square
with(gCDTrainGlm, df.null -df.residual)
[1] 19       # degrees of freedom
with(gCDTrainGlm, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = FALSE))
[1] 1.045694e-22 # p-value
```

With a chi-square of 151.66 on 19 degress of freedom and a p-value < 0.0001 the fit seems acceptable so I'll use this model for my predictions. Next I'll print the analysis of deviance along with the Confidence Intervals and Odds table.

```
print(xtable(anova(gCDTrainGlm, test="Chisq"), caption = "Analysis of Deviance"))
print(xtable(exp(cbind(Odds = coef(gCDTrainGlm), confint(gCDTrainGlm))),
             caption = "Odds and CI"))
```

Table 5: Analysis of Deviance

	Df	Deviance	Resid. Df	Resid. Dev	Pr(>Chi)
NULL			899	1096.15	
duration	1	30.18	898	1065.97	0.0000
amount	1	1.23	897	1064.74	0.2679
installment	1	4.99	896	1059.75	0.0255
age	1	10.06	895	1049.69	0.0015
history	4	55.39	891	994.30	0.0000
purpose	9	40.72	882	953.58	0.0000
housing	2	9.10	880	944.48	0.0106

Above we can see the effect of the added parameters on the residual deviance for each parameter added compared to the *Null/Intercept*. Each of the added parameters affects the odds of default albeit the impact of *amount* is trivial.

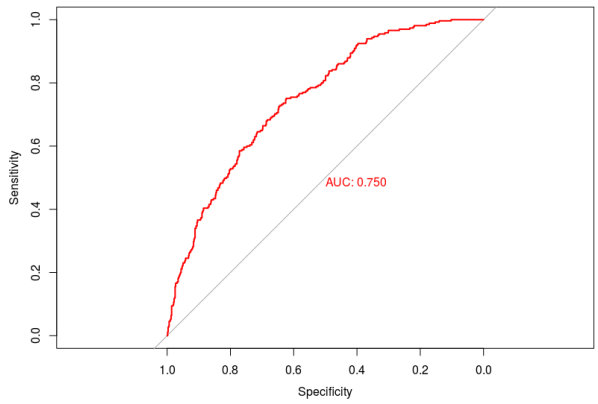
Table 6: Odds and CI

	Odds	2.5 %	97.5 %
(Intercept)	1.47	0.50	4.40
duration	1.02	1.00	1.04
amount	1.00	1.00	1.00
installment	1.29	1.10	1.51
age	0.98	0.96	1.00
historypaidBackDuly	1.53	0.56	4.22
historypaidBackUntilNow	0.36	0.17	0.75
historydelayInPayback	0.31	0.13	0.73
historycriticalAccount	0.15	0.07	0.33
purposeusedCar	0.21	0.10	0.40
purposeother	0.43	0.09	1.72
purposefurnitureEquip	0.61	0.38	0.98
purposeradioTV	0.38	0.24	0.60
purposeappliances	0.70	0.17	2.52
purposerepairs	0.70	0.22	1.97
purposeeducation	1.13	0.54	2.31
purposeretraining	0.12	0.01	0.74
purposebusiness	0.51	0.28	0.93
housingown	0.57	0.38	0.86
housingfree	0.93	0.49	1.76

As another test I plotted the Receiver-operating characteristic (ROC curve). While this is more valuable when comparing models it can still provide insight as to the predictive power of the model.

```
plot.roc(gCDTrain$Default, fitted(gCDTrainGlm), print.auc = TRUE, col = "red")
Data: fitted(gCDTrainGlm) in 632 controls (gCDTrain$Default No) < 268 cases (gCDTrain$Default Yes).
Area under the curve: 0.7497
```

Figure 1: ROC Curve for Default



The straight line represents the curve for an *uninformative* model. The greater the area under the curve the better the predictive value of the model. In this instance it shows a reasonable balance between sensitivity and specificity. Approximately 75% of the cases were under the curve so I would expect close to 75% predictive accuracy.

Training the Model

Now that I've determined the model is sufficient I'll fit/train the model and use it to predict the *Default* values in the training set.

```
## Train the model
mfit <- train(Default ~ duration + amount + installment + age + history +
                purpose + housing, family = "binomial", method = "glm", data = gCDTrain)
```

Predicting Default

In this section I'll use the trained model to predict *Default* on the test dataset.

```
## Generate predictions for Default from the trained model
pred <- predict(mfit, newdata=gCDTest)

## Assess the accuracy of predictions
accuracy <- table(pred, gCDTest[,"Default"])
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  104  28
       Yes  10   7

               Accuracy : 0.745
                 95% CI : (0.6672, 0.8128)
    No Information Rate : 0.7651
    P-Value [Acc > NIR] : 0.75349

                  Kappa : 0.1366
 Mcnemar's Test P-Value : 0.00582

            Sensitivity : 0.9123
            Specificity : 0.2000
         Pos Pred Value : 0.7879
         Neg Pred Value : 0.4118
             Prevalence : 0.7651
         Detection Rate : 0.6980
   Detection Prevalence : 0.8859
      Balanced Accuracy : 0.5561

       'Positive' Class : No
[1] 0.72
```

This model had a 72% accuracy in predicting *Default* in the test data set. This is very close to what we would expect from the tests that were run on the *gCDTrainGlm* model. The classifier is now ready to to be used on additional data.

The classifier can be used on individual rows, or ranges of rows of data as demonstrated in the R code that follows. First I'll pick a subset of of the *gCDTest* data set.

```
print(xtable(gCDTest[5:10,]))
```

Table 7: gCDTest Excerpt

	Default	duration	amount	installment	age	history	purpose	housing
40	No	9	458	4	24	paidBackUntilNow	radioTV	own
45	Yes	48	6143	4	58	criticalAccount	usedCar	free
52	No	27	5965	1	30	delayInPayback	usedCar	own
55	Yes	36	2225	4	57	delayInPayback	newCar	free
56	No	6	783	1	26	paidBackDuly	newCar	own
69	Yes	36	1819	4	37	paidBackUntilNow	education	free

Now I'll run a prediction against this data and show the results in a table.

```
p1 <- predict(mfit, newdata=gCDTest[5:10,])
miniTest <- data_frame(p1,gCDTest[5:10,]$Default)
colnames(miniTest) <- c("Predict", "Test")
print(xtable(miniTest, caption = "Mini Predictive Test"))
```

Table 8: Mini Predictive Test

	Predict	Test
1	No	No
2	No	Yes
3	No	No
4	Yes	Yes
5	Yes	No
6	Yes	Yes

The correct *Default* predictions are highlighted in green and the incorrect predictions are highlighted in red.

Summary

Using the German Credit dataset a logistic model was created using a traing set of 900, out of 1000, records randomly selected from the data. The logistic regression model was assessed for fit. The assessment indicated that the model should have approximately 75% predictive accuracy. A classifier was then created from the model and run against the 100 records in the test dataset. The classifier was approximately 72% accurate in predicting *Default* on the test dataset. Examples of running the classifier against the entire test dataset and a subset were demonstrated. These same techniques could be used to assess new records as they are received.

References

Introduction to LaTeX. (n.d.). Retrieved from https://www.latex-project.org/about/

Pandoc - About pandoc. (n.d.). Retrieved from http://pandoc.org/

Str Implementation for Data Frames | R-bloggers. (n.d.). Retrieved from https://www.r-bloggers.com/str-implementation-for-data-frames/

UCI Machine Learning Repository: Statlog (German Credit Data) Data Set. (n.d.). Retrieved from https://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)