Assignment 10 Part 1,2- Logistic Regressions

David Pahmer

2022-05-22

Logistic Regression

Fit a Logistic Regression Model to Thoracic Surgery Binary Dataset
Assignment Instructions: #### Fit a binary logistic regression model to the data set that predicts whether or not the patient survived for one year (the Risk1Y variable) after the surgery. Use the glm() function to perform the logistic regression. See Generalized Linear Models for an example. Include a summary using the summary() function in your results.

According to the summary, which variables had the greatest effect on the survival rate?

```
library(caTools)
library(farff)
setwd("C:/users/pahme/onedrive/documents/github/dsc520/data")
surgery.full <- readARFF("thoraricsurgery.arff")
```

To compute the accuracy of your model, use the dataset to predict the outcome variable. The percent of correct predictions is the accuracy of your model. What is the accuracy of your model?

```
## Parse with reader=readr : thoraricsurgery.arff
```

```
## Loading required package: readr
```

```
## header: 0.000000; preproc: 0.000000; data: 0.110000; postproc: 0.000000; total: 0.110000
```

```
str(surgery.full)
```

```
## 'data.frame':    470 obs. of  17 variables:
##  $ DGN    : Factor w/ 7 levels "DGN3","DGN2",..: 2 1 1 1 1 1 1 2 1 1 ...
##  $ PRE4   : num  2.88 3.4 2.76 3.68 2.44 2.48 4.36 3.19 3.16 2.32 ...
##  $ PRE5   : num  2.16 1.88 2.08 3.04 0.96 1.88 3.28 2.5 2.64 2.16 ...
##  $ PRE6   : Factor w/ 3 levels "PRZ2","PRZ1",..: 2 3 2 3 1 2 2 2 1 2 ...
##  $ PRE7   : Factor w/ 2 levels "T","F": 2 2 2 2 2 2 2 2 2 2 ...
##  $ PRE8   : Factor w/ 2 levels "T","F": 2 2 2 2 1 2 2 2 2 2 ...
##  $ PRE9   : Factor w/ 2 levels "T","F": 2 2 2 2 2 2 2 2 2 2 ...
##  $ PRE10  : Factor w/ 2 levels "T","F": 1 2 1 2 1 1 1 1 1 1 ...
##  $ PRE11  : Factor w/ 2 levels "T","F": 1 2 2 2 1 2 2 2 1 2 ...
##  $ PRE14  : Factor w/ 4 levels "OC11","OC14",..: 2 3 1 1 1 1 3 1 1 1 ...
##  $ PRE17  : Factor w/ 2 levels "T","F": 2 2 2 2 2 2 1 2 2 2 ...
##  $ PRE19  : Factor w/ 2 levels "T","F": 2 2 2 2 2 2 2 2 2 2 ...
##  $ PRE25  : Factor w/ 2 levels "T","F": 2 2 2 2 2 2 2 1 2 2 ...
##  $ PRE30  : Factor w/ 2 levels "T","F": 1 1 1 2 1 2 1 1 1 1 ...
##  $ PRE32  : Factor w/ 2 levels "T","F": 2 2 2 2 2 2 2 2 2 2 ...
##  $ AGE    : num  60 51 59 54 73 51 59 66 68 54 ...
```

```
##  $ Risk1Yr: Factor w/ 2 levels "T","F": 2 2 2 2 1 2 1 1 2 2 ...
```

```
# Without understanding the various predictors we can't tell which value should be the baseline value..
# However, we can determine the baseline for the outcome variable, and we want to switch them.
str(surgery.full$Risk1Yr)
```

```
##  Factor w/ 2 levels "T","F": 2 2 2 2 1 2 1 1 2 2 ...
```

```
surgery.full$Risk1Yr <- relevel(surgery.full$Risk1Yr, "F")
str(surgery.full$Risk1Yr)
```

```
##  Factor w/ 2 levels "F","T": 1 1 1 1 2 1 2 2 1 1 ...
```

```
summary(surgery.full)
```

```
##    DGN           PRE4            PRE5          PRE6     PRE7    PRE8    PRE9
##  DGN3:349   Min.   :1.440   Min.   : 0.960   PRZ2: 27   T: 31   T: 68   T: 31
##  DGN2: 52   1st Qu.:2.600   1st Qu.: 1.960   PRZ1:313   F:439   F:402   F:439
##  DGN4: 47   Median :3.160   Median : 2.400   PRZ0:130
##  DGN6:  4   Mean   :3.282   Mean   : 4.569
##  DGN5: 15   3rd Qu.:3.808   3rd Qu.: 3.080
##  DGN8:  2   Max.   :6.300   Max.   :86.300
##  DGN1:  1
##  PRE10   PRE11    PRE14     PRE17   PRE19   PRE25   PRE30   PRE32
##  T:323   T: 78   OC11:177   T: 35   T:  2   T:  8   T:386   T:  2
##  F:147   F:392   OC14: 17   F:435   F:468   F:462   F: 84   F:468
##                  OC12:257
##                  OC13: 19
##
##
##
##       AGE        Risk1Yr
##  Min.   :21.00   F:400
##  1st Qu.:57.00   T: 70
##  Median :62.00
##  Mean   :62.53
##  3rd Qu.:69.00
##  Max.   :87.00
##
```

```
# Split into training dataset (surgery) and testing or validating dataset (surgery.test)
rnd <- sample.split(c(1:470), SplitRatio = .75)
surgery <- subset(surgery.full,rnd==TRUE)
surgery.test <- subset(surgery.full, rnd==FALSE)


model.1 <- glm(Risk1Yr ~ DGN , data = surgery, family = binomial())
summary(model.1)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ DGN, family = binomial(), data = surgery)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1774  -0.4922  -0.4922  -0.4922   2.0837
##
```

```
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -2.0498     0.1940 -10.568  < 2e-16 ***
## DGNDGN2        1.0804     0.4038   2.676  0.00745 **
## DGNDGN4        0.2581     0.5205   0.496  0.62004
## DGNDGN6      -14.5162  1199.7724  -0.012  0.99035
## DGNDGN5        2.0498     0.7332   2.796  0.00518 **
## DGNDGN8      -14.5162  2399.5447  -0.006  0.99517
## DGNDGN1      -14.5162  2399.5447  -0.006  0.99517
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 287.70  on 351  degrees of freedom
## Residual deviance: 273.55  on 345  degrees of freedom
## AIC: 287.55
##
## Number of Fisher Scoring iterations: 15
```

```
model.2 <- glm(Risk1Yr ~ DGN + AGE , data = surgery, family = binomial())
summary(model.2)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ DGN + AGE, family = binomial(), data = surgery)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2233  -0.5290  -0.4889  -0.4589   2.2367
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -3.10351    1.17757  -2.636  0.00840 **
## DGNDGN2        1.11068    0.40612   2.735  0.00624 **
## DGNDGN4        0.19245    0.52577   0.366  0.71434
## DGNDGN6      -14.62463 1197.99148  -0.012  0.99026
## DGNDGN5        2.00348    0.73779   2.715  0.00662 **
## DGNDGN8      -14.28428 2399.54474  -0.006  0.99525
## DGNDGN1      -14.50229 2399.54473  -0.006  0.99518
## AGE            0.01677    0.01835   0.914  0.36077
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 287.70  on 351  degrees of freedom
## Residual deviance: 272.71  on 344  degrees of freedom
## AIC: 288.71
##
## Number of Fisher Scoring iterations: 15
```

```
model.3 <- glm(Risk1Yr ~ DGN + PRE4 + PRE5 , data = surgery, family = binomial())
summary(model.3)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ DGN + PRE4 + PRE5, family = binomial(),
##     data = surgery)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3411  -0.5243  -0.4991  -0.4565   2.1575
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.60896    0.64125  -2.509  0.01210 *
## DGNDGN2        1.12284    0.40651   2.762  0.00574 **
## DGNDGN4        0.27457    0.52186   0.526  0.59880
## DGNDGN6      -14.42723 1198.46411  -0.012  0.99040
## DGNDGN5        2.37082    0.80182   2.957  0.00311 **
## DGNDGN8      -14.27102 2399.54476  -0.006  0.99525
## DGNDGN1      -14.45970 2399.54473  -0.006  0.99519
## PRE4          -0.11601    0.19470  -0.596  0.55129
## PRE5          -0.02021    0.01677  -1.205  0.22816
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 287.7  on 351  degrees of freedom
## Residual deviance: 271.1  on 343  degrees of freedom
## AIC: 289.1
##
## Number of Fisher Scoring iterations: 15
```

We tried to guess what variables would make senese to predict survival, such as age, or maybe DGN, whatever that it, or maybe PRE4 and PRE5, but the binomial regression showed no meaningful correlation to age or PRE4 or PRE5, so let's just test all the variables at one shot!

```
model.all <- glm(Risk1Yr ~ DGN + PRE4 + PRE5 + PRE6 + PRE7 + PRE8 + PRE9 + PRE10 + PRE11 + PRE14 +
                   PRE17 + PRE19 + PRE25 + PRE30 + PRE32 + AGE, data=surgery, family=binomial())
summary(model.all)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ DGN + PRE4 + PRE5 + PRE6 + PRE7 + PRE8 +
##     PRE9 + PRE10 + PRE11 + PRE14 + PRE17 + PRE19 + PRE25 + PRE30 +
##     PRE32 + AGE, family = binomial(), data = surgery)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4694  -0.4857  -0.3689  -0.2464   2.5993
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.621e+01  2.299e+03  -0.011  0.99091
## DGNDGN2      9.333e-01  4.810e-01   1.940  0.05235 .
## DGNDGN4      4.185e-01  5.704e-01   0.734  0.46318
```

```
## DGNDGN6     -1.350e+01  1.184e+03  -0.011  0.99090
## DGNDGN5      2.476e+00  8.889e-01   2.786  0.00534 **
## DGNDGN8     -1.307e+01  2.400e+03  -0.005  0.99565
## DGNDGN1     -1.407e+01  2.400e+03  -0.006  0.99532
## PRE4        -2.022e-01  2.336e-01  -0.866  0.38667
## PRE5        -2.675e-02  1.893e-02  -1.413  0.15775
## PRE6PRZ1    -9.223e-02  7.128e-01  -0.129  0.89705
## PRE6PRZ0    -3.038e-01  1.004e+00  -0.303  0.76221
## PRE7F       -1.023e+00  6.404e-01  -1.597  0.11026
## PRE8F        3.458e-01  5.080e-01   0.681  0.49602
## PRE9F       -1.405e+00  5.806e-01  -2.421  0.01549 *
## PRE10F       2.112e-01  5.533e-01   0.382  0.70269
## PRE11F      -6.877e-01  4.668e-01  -1.473  0.14069
## PRE14OC14    2.137e+00  6.877e-01   3.107  0.00189 **
## PRE14OC12    8.285e-01  4.471e-01   1.853  0.06392 .
## PRE14OC13    2.091e+00  7.075e-01   2.955  0.00313 **
## PRE17F      -1.388e+00  4.975e-01  -2.789  0.00528 **
## PRE19F       1.438e+01  1.616e+03   0.009  0.99290
## PRE25F      -1.572e-01  1.106e+00  -0.142  0.88700
## PRE30F      -6.905e-01  5.444e-01  -1.268  0.20468
## PRE32F       1.379e+01  1.636e+03   0.008  0.99327
## AGE         -5.430e-04  2.180e-02  -0.025  0.98013
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 287.70  on 351  degrees of freedom
## Residual deviance: 235.13  on 327  degrees of freedom
## AIC: 285.13
##
## Number of Fisher Scoring iterations: 15
```

```
allpred <- predict(model.all, surgery, type="response")

(checkmodel <- table(actual=surgery$Risk1Yr, prediction= allpred > .5))
```

```
##       prediction
## actual FALSE TRUE
##      F   296    6
##      T    43    7
```

```
# Accuracy for the training set, sort of
((checkmodel[1,1]+checkmodel[2,2])/length(surgery[,1]))
```

```
## [1] 0.8607955
```

```
allpred2 <- predict(model.all, surgery.test, type="response")

(checkmodel2 <- table(actual=surgery.test$Risk1Yr, prediction= allpred2 > .5))
```

```
##       prediction
## actual FALSE TRUE
##      F    90    8
##      T    19    1
```

```r
# Accuracy for the training set, sort of
((checkmodel2[1,1]+checkmodel2[2,2])/length(surgery.test[,1]))
```

```
## [1] 0.7711864
```

So it looks like we found a few predictors: the DGN, specifically DGN5, and PRE9, and PRE14, specifically OC14. To extract the specific values that matter from the DGN variable, we create an additional variable that only shows DGN5 T/F, and likewise for OC14 in PRE14.

Let's test it!

```r
## Let's zero in on the best predictors!

model.opt1 <- glm(Risk1Yr ~ DGN + PRE9 + PRE14, data = surgery, family=binomial())
summary(model.opt1)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ DGN + PRE9 + PRE14, family = binomial(),
##     data = surgery)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2295  -0.4751  -0.4751  -0.3551   2.3645
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.7789     0.5887  -3.022 0.002515 **
## DGNDGN2        0.6741     0.4437   1.519 0.128691
## DGNDGN4        0.4006     0.5342   0.750 0.453345
## DGNDGN6      -14.1679  1190.7620  -0.012 0.990507
## DGNDGN5        2.0145     0.7668   2.627 0.008606 **
## DGNDGN8      -14.4414  2399.5447  -0.006 0.995198
## DGNDGN1      -14.4414  2399.5447  -0.006 0.995198
## PRE9F         -0.9534     0.5098  -1.870 0.061465 .
## PRE14OC14      2.1798     0.6420   3.395 0.000685 ***
## PRE14OC12      0.6077     0.4112   1.478 0.139483
## PRE14OC13      1.8439     0.6603   2.793 0.005225 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 287.70  on 351  degrees of freedom
## Residual deviance: 255.29  on 341  degrees of freedom
## AIC: 277.29
##
## Number of Fisher Scoring iterations: 15
```

```r
surgery$PRE9 <- relevel(surgery$PRE9, "F")

## Only the DGN5 matters, so let's extract that one; also OC14 in the PRE14
surgery$dgn5 <- c(surgery$DGN=="DGN5")
surgery$oc14 <- c(surgery$PRE14=="OC14")

model.opt2 <- glm(Risk1Yr ~ dgn5 + PRE9 + oc14, data=surgery, family=binomial())
```

```r
summary(model.opt2)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ dgn5 + PRE9 + oc14, family = binomial(),
##     data = surgery)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1241  -0.4772  -0.4772  -0.4772   2.1114
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.1151     0.1845 -11.464  < 2e-16 ***
## dgn5TRUE      1.9885     0.7377   2.696 0.007028 **
## PRE9T         1.0775     0.4890   2.203 0.027574 *
## oc14TRUE      1.8638     0.5367   3.473 0.000515 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 287.70  on 351  degrees of freedom
## Residual deviance: 268.04  on 348  degrees of freedom
## AIC: 276.04
##
## Number of Fisher Scoring iterations: 4
```

```r
surg.model <- data.frame(risk.predict=predict(model.opt2,
                                              newdata=data.frame(dgn5=surgery$dgn5,
                                                                 PRE9=surgery$PRE9,
                                                                 oc14=surgery$oc14 ), type="response"))
# Alternate version of the same command
surg.model <- predict(model.opt2, surgery, type="response")
(opt2.conf <- table(actual=surgery$Risk1Yr, predicted=surg.model > .5))
```

```
##       predicted
## actual FALSE TRUE
##      F   302    0
##      T    49    1
```

```r
((opt2.conf[1,1]+opt2.conf[2,2])/length(surgery[,1]))
```

```
## [1] 0.8607955
```

```r
## Now let's test this on the testing set:
surgery.test$PRE9 <- relevel(surgery.test$PRE9, "F")

## Only the DGN5 matters, so let's extract that one; also OC14 in the PRE14
surgery.test$dgn5 <- c(surgery.test$DGN=="DGN5")
surgery.test$oc14 <- c(surgery.test$PRE14=="OC14")

surg.model2 <- predict(model.opt2, surgery.test, type="response")

# Accuracy for the training set, sort of
(opt2.conf <- table(actual=surgery.test$Risk1Yr, predicted=surg.model2 > .3))
```

```
##       predicted
## actual FALSE TRUE
##      F    93    5
##      T    17    3
```

```
((opt2.conf[1,1]+opt2.conf[2,2])/length(surgery.test[,1]))
```

```
## [1] 0.8135593
```

Even though it is an improvement, it still is not great, because relatively few patients have those factors identified here.

Part II

Fit a Logistic Regression Model

Fit a logistic regression model to the binary-classifier-data.csv dataset

The dataset (found in binary-classifier-data.csv) contains three variables; label, x, and y. The label variable is either 0 or 1 and is the output we want to predict using the x and y variables.

```
setwd("C:/users/pahme/onedrive/documents/github/dsc520/data")
binclasdata <- read.csv("binary-classifier-data.csv")
str(binclasdata)
```

What is the accuracy of the logistic regression classifier?

```
## 'data.frame':    1498 obs. of  3 variables:
##  $ label: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ x    : num  70.9 75 73.8 66.4 69.1 ...
##  $ y    : num  83.2 87.9 92.2 81.1 84.5 ...
```

```
binclasdata$label <- as.factor(binclasdata$label)
str(binclasdata)
```

```
## 'data.frame':    1498 obs. of  3 variables:
##  $ label: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ x    : num  70.9 75 73.8 66.4 69.1 ...
##  $ y    : num  83.2 87.9 92.2 81.1 84.5 ...
```

```
bcd.model1 <- glm(label~ x+y, data=binclasdata, family=binomial())
summary(bcd.model1)
```

```
##
## Call:
## glm(formula = label ~ x + y, family = binomial(), data = binclasdata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3728  -1.1697  -0.9575   1.1646   1.3989
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.424809   0.117224   3.624  0.00029 ***
```

```
## x           -0.002571   0.001823  -1.411  0.15836
## y           -0.007956   0.001869  -4.257 2.07e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2075.8  on 1497  degrees of freedom
## Residual deviance: 2052.1  on 1495  degrees of freedom
## AIC: 2058.1
##
## Number of Fisher Scoring iterations: 4
```

```
bcd.model2 <- glm(label~y, data=binclasdata, family=binomial())
summary(bcd.model2)
```

```
##
## Call:
## glm(formula = label ~ y, family = binomial(), data = binclasdata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3335  -1.1350  -0.9886   1.1771   1.4287
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.332800   0.097188   3.424 0.000616 ***
## y           -0.008480   0.001831  -4.630 3.65e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2075.8  on 1497  degrees of freedom
## Residual deviance: 2054.1  on 1496  degrees of freedom
## AIC: 2058.1
##
## Number of Fisher Scoring iterations: 4
```

Now this is a poor model, having only improved the prediction over baseline by a little. The baseline is 49% TRUE.

```
# ok let's produce a training and testing set:
rnd <- sample.split(binclasdata[,1], SplitRatio = .75)
binclasdata.train <- subset(binclasdata, rnd==TRUE)
binclasdata.test <- subset(binclasdata, rnd==FALSE)

bcd.model2 <- glm(label~y, data=binclasdata.train, family=binomial())
summary(bcd.model2)
```

```
##
## Call:
## glm(formula = label ~ y, family = binomial(), data = binclasdata.train)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -1.3637  -1.1253  -0.9574   1.1724   1.4724
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.396152   0.111837   3.542 0.000397 ***
## y           -0.009985   0.002123  -4.703 2.57e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1556.2  on 1122  degrees of freedom
## Residual deviance: 1533.6  on 1121  degrees of freedom
## AIC: 1537.6
##
## Number of Fisher Scoring iterations: 4
```

```
bcdpred <- predict(bcd.model2, binclasdata.train, type="response")
(model2.conf <- table(actual=binclasdata.train$label, predicted=bcdpred >= .5))
```

```
##       predicted
## actual FALSE TRUE
##      0   328  247
##      1   263  285
```

```
# Check accuracy of the model on the training set
((model2.conf[1,1]+model2.conf[2,2])/sum(model2.conf))
```

```
## [1] 0.5458593
```

```
# it's pretty lousy. Let's check the testing set:
bcdpred <- predict(bcd.model2, binclasdata.test, type="response")
(model2.conf <- table(actual=binclasdata.test$label, predicted=bcdpred >= .5))
```

```
##       predicted
## actual FALSE TRUE
##      0   103   89
##      1   103   80
```

```
# Check accuracy on the testing set
((model2.conf[1,1]+model2.conf[2,2])/sum(model2.conf))
```

```
## [1] 0.488
```

So this is a pretty lousy model. Accuracy not much better than the overall ratio.

```
## What about the interaction between x and y?
bcd.model3 <- glm(label~y*x, data=binclasdata.train, family=binomial())
summary(bcd.model3)
```

```
##
## Call:
## glm(formula = label ~ y * x, family = binomial(), data = binclasdata.train)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -1.9233  -1.1469  -0.4607   1.1179   1.5484
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.0644041  0.2191266  -4.857 1.19e-06 ***
## y            0.0269672  0.0045323   5.950 2.68e-09 ***
## x            0.0326580  0.0045210   7.224 5.06e-13 ***
## y:x         -0.0007669  0.0000852  -9.001  < 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: 1556.2  on 1122  degrees of freedom
## Residual deviance: 1438.5  on 1119  degrees of freedom
## AIC: 1446.5
##
## Number of Fisher Scoring iterations: 4
```

```
bcdpred3 <- predict(bcd.model3, binclasdata.train, type="response")
(model3.conf <- table(actual=binclasdata.train$label, predicted=bcdpred3 >= .5))
```

```
##       predicted
## actual FALSE TRUE
##      0   335  240
##      1   134  414
```

```
# Check accuracy of the model on the training set
((model3.conf[1,1]+model3.conf[2,2])/sum(model3.conf))
```

```
## [1] 0.6669635
```

```
# it's pretty lousy. Let's check the testing set:
bcdpred3 <- predict(bcd.model3, binclasdata.test, type="response")
(model3.conf <- table(actual=binclasdata.test$label, predicted=bcdpred3 >= .5))
```

```
##       predicted
## actual FALSE TRUE
##      0   103   89
##      1    50  133
```

```
# Check accuracy on the testing set
((model3.conf[1,1]+model3.conf[2,2])/sum(model3.conf))
```

```
## [1] 0.6293333
```

That was much better- checking the interactive effects gave the best prediction .