10.2 Logistic Regression

Maxim Bilenkin

2025-02-11

1. Fit a Logistic Regression Model to Thoracic Surgery Binary Dataset

```r
# Clearing the environment to insure it has fresh start
rm(list = ls())

# Loading necessary libraries
library(foreign)
```

```r
# Downloading the ARFF file from the provided URL
url <-
"https://archive.ics.uci.edu/ml/machine-learning-databases/00277/ThoraricSurgery.arff"

download.file(url, destfile = "ThoraricSurgery.arff", method = "curl")

# Loading data from the downloaded ARFF file and saving into variable
data <- read.arff("ThoraricSurgery.arff")
```

Exploring the Dataset to learn and get insight of the data.

```r
str(data)
```

```
## 'data.frame':    470 obs. of  17 variables:
##  $ DGN    : Factor w/ 7 levels "DGN1","DGN2",..: 2 3 3 3 3 3 3 2 3 3 ...
##  $ 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 "PRZ0","PRZ1",..: 2 1 2 1 3 2 2 2 3 2 ...
##  $ PRE7   : Factor w/ 2 levels "F","T": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE8   : Factor w/ 2 levels "F","T": 1 1 1 1 2 1 1 1 1 1 ...
##  $ PRE9   : Factor w/ 2 levels "F","T": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE10  : Factor w/ 2 levels "F","T": 2 1 2 1 2 2 2 2 2 2 ...
##  $ PRE11  : Factor w/ 2 levels "F","T": 2 1 1 1 2 1 1 1 2 1 ...
##  $ PRE14  : Factor w/ 4 levels "OC11","OC12",..: 4 2 1 1 1 1 2 1 1 1 ...
##  $ PRE17  : Factor w/ 2 levels "F","T": 1 1 1 1 1 1 2 1 1 1 ...
##  $ PRE19  : Factor w/ 2 levels "F","T": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE25  : Factor w/ 2 levels "F","T": 1 1 1 1 1 1 1 2 1 1 ...
##  $ PRE30  : Factor w/ 2 levels "F","T": 2 2 2 1 2 1 2 2 2 2 ...
##  $ PRE32  : Factor w/ 2 levels "F","T": 1 1 1 1 1 1 1 1 1 1 ...
##  $ AGE    : num  60 51 59 54 73 51 59 66 68 54 ...
##  $ Risk1Yr: Factor w/ 2 levels "F","T": 1 1 1 1 2 1 2 2 1 1 ...
```

```r
summary(data)
```

```
##    DGN           PRE4            PRE5           PRE6     PRE7     PRE8     PRE9
##  DGN1:  1   Min.   :1.440   Min.   : 0.960   PRZ0:130   F:439   F:402   F:439
##  DGN2: 52   1st Qu.:2.600   1st Qu.: 1.960   PRZ1:313   T: 31   T: 68   T: 31
##  DGN3:349   Median :3.160   Median : 2.400   PRZ2: 27
```

```
##  DGN4: 47   Mean   :3.282   Mean   : 4.569
##  DGN5: 15   3rd Qu.:3.808   3rd Qu.: 3.080
##  DGN6:  4   Max.   :6.300   Max.   :86.300
##  DGN8:  2
##  PRE10   PRE11    PRE14     PRE17   PRE19   PRE25   PRE30   PRE32
##  F:147   F:392   OC11:177   F:435   F:468   F:462   F: 84   F:468
##  T:323   T: 78   OC12:257   T: 35   T:  2   T:  8   T:386   T:  2
##                  OC13: 19
##                  OC14: 17
##
##
##
##       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
##
```

Data Pre-processing.

The Risk1Yr variable has values as 1 and 2 instead of expected values 0 and 1.

Thus, will convert 1 to 0 and 2 to 1 to get expected binary values as 0 and 1.

```r
# Converting all factor columns to numeric
data[] <- lapply(data, function(x) if(is.factor(x)) as.numeric(as.factor(x)) else x)

# Splitting the dataset into training and testing sets
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```r
set.seed(123)
index <- createDataPartition(data$Risk1Yr, p = 0.8, list = FALSE)
train_data <- data[index, ]
test_data <- data[-index, ]

# Converting Risk1Yr variable values to binary 0 and 1
train_data$Risk1Yr <- ifelse(train_data$Risk1Yr == 2, 1, 0)
test_data$Risk1Yr <- ifelse(test_data$Risk1Yr == 2, 1, 0)

# Checking to make sure the conversion worked
unique(train_data$Risk1Yr)
```

```
## [1] 0 1
```

i. Fit a Logistic Regression Model to Thoracic Surgery Binary Dataset

```r
# Fitting the logistic regression model
model_glm <- glm(Risk1Yr ~ ., data = train_data, family = binomial)

# Summary of the logistic regression model
summary(model_glm)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ ., family = binomial, data = train_data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.878e+01  1.385e+03   0.014  0.98919
## DGN          2.980e-01  2.221e-01   1.342  0.17974
## PRE4        -2.499e-01  1.979e-01  -1.263  0.20667
## PRE5        -1.615e-02  1.772e-02  -0.912  0.36203
## PRE6         1.477e-01  4.280e-01   0.345  0.73010
## PRE7        -3.748e-01  6.698e-01  -0.560  0.57577
## PRE8         3.464e-01  4.111e-01   0.843  0.39949
## PRE9         1.367e+00  5.305e-01   2.577  0.00995 **
## PRE10       -1.131e-01  4.916e-01  -0.230  0.81803
## PRE11        6.089e-01  4.286e-01   1.421  0.15539
## PRE14        8.139e-01  2.081e-01   3.912 9.16e-05 ***
## PRE17        7.188e-01  4.965e-01   1.448  0.14766
## PRE19       -1.385e+01  9.720e+02  -0.014  0.98863
## PRE25        5.529e-01  1.030e+00   0.537  0.59121
## PRE30        7.449e-01  4.993e-01   1.492  0.13576
## PRE32       -1.331e+01  9.869e+02  -0.013  0.98924
## AGE          1.829e-03  1.887e-02   0.097  0.92278
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 323.37  on 375  degrees of freedom
## Residual deviance: 285.07  on 359  degrees of freedom
## AIC: 319.07
##
## Number of Fisher Scoring iterations: 14
```

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

Answer:

Looking at the statistical summary we can see that only two variables PRE9 and PRE14 had the greatest effect on the survival rate. Both variables are statistically significant. For example, variable PRE9 has z-value of 2.577, p-value of 0.00995 and statistically significant at the 1% level. Variable PRE14 has z-value of 3.912, p-value of 9.16e-05 and statistically significant at the 0.1% level. Thus, these are the two factors that help us understand the most important factors that help for a patient to survive one year after the thoracic surgery was done. Additionally, medical personal can improve treatment for patients by concentrating more on these two PRE9 and PRE14 variables and prolong patients lives.

iii. 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?

Evaluating the Model

```r
# Predicting on the test data
predictions_on_the_testdata <- predict(model_glm, newdata = test_data, type = "response")
predicted_classes <- ifelse(predictions_on_the_testdata > 0.5, 1, 0)

# Creating a confusion matrix
confusionMatrix(as.factor(predicted_classes), as.factor(test_data$Risk1Yr))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 78 12
##          1  4  0
##
##                Accuracy : 0.8298
##                  95% CI : (0.7384, 0.8995)
##     No Information Rate : 0.8723
##     P-Value [Acc > NIR] : 0.91350
##
##                   Kappa : -0.0682
##
##  Mcnemar's Test P-Value : 0.08012
##
##             Sensitivity : 0.9512
##             Specificity : 0.0000
##          Pos Pred Value : 0.8667
##          Neg Pred Value : 0.0000
##              Prevalence : 0.8723
##          Detection Rate : 0.8298
##    Detection Prevalence : 0.9574
##       Balanced Accuracy : 0.4756
##
##        'Positive' Class : 0
##
```

Answer:

According to the calculated outcome, the accuracy of the model based on the dataset is 0.8298. We can state that 82.98% of the predictions were correct using the model with this dataset.

2. Fit a Logistic Regression Model

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

```r
# Loading necessary libraries
library(caret)

# Loading the data file
data <-
read.csv(
"C:/Users/maxim/OneDrive/Desktop/Bellevue University/DSC 520/binary-classifier-data.csv")

# Exploring the dataset to get insight and understanding of the data
str(data)
```

```
## '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 ...
```

```r
summary(data)
```

```
##      label              x                y
##  Min.   :0.000   Min.   : -5.20   Min.   : -4.019
##  1st Qu.:0.000   1st Qu.: 19.77   1st Qu.: 21.207
```

```
##  Median :0.000   Median : 41.76   Median : 44.632
##  Mean   :0.488   Mean   : 45.07   Mean   : 45.011
##  3rd Qu.:1.000   3rd Qu.: 66.39   3rd Qu.: 68.698
##  Max.   :1.000   Max.   :104.58   Max.   :106.896
```

```r
head(data)
```

```
##   label        x        y
## 1     0 70.88469 83.17702
## 2     0 74.97176 87.92922
## 3     0 73.78333 92.20325
## 4     0 66.40747 81.10617
## 5     0 69.07399 84.53739
## 6     0 72.23616 86.38403
```

```r
# Fitting the Logistic Regression Model
# Using 'label' as the target variable
model_glm <- glm(label ~ x + y, data = data, family = binomial)

# Summarizing the model
summary(model_glm)
```

```
##
## Call:
## glm(formula = label ~ x + y, family = binomial, data = data)
##
## 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
```

```r
# Evaluating the Model
# Splitting dataset into two sets if not already split
set.seed(123)
index <- createDataPartition(data$label, p = 0.8, list = FALSE)
train_data <- data[index, ]
test_data <- data[-index, ]

# Fitting the model on the training data
model_glm <- glm(label ~ x + y, data = train_data, family = binomial)

# Predicting on the test data
predictions <- predict(model_glm, newdata = test_data, type = "response")
predicted_classes <- ifelse(predictions > 0.5, 1, 0)
```

```r
# Creating a confusion matrix to evaluate the model
confusionMatrix(as.factor(predicted_classes), as.factor(test_data$label))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 77 58
##          1 81 83
##
##                Accuracy : 0.5351
##                  95% CI : (0.4768, 0.5927)
##     No Information Rate : 0.5284
##     P-Value [Acc > NIR] : 0.43146
##
##                   Kappa : 0.0753
##
##  Mcnemar's Test P-Value : 0.06204
##
##             Sensitivity : 0.4873
##             Specificity : 0.5887
##          Pos Pred Value : 0.5704
##          Neg Pred Value : 0.5061
##              Prevalence : 0.5284
##          Detection Rate : 0.2575
##    Detection Prevalence : 0.4515
##       Balanced Accuracy : 0.5380
##
##        'Positive' Class : 0
##
```

```r
# Calculating the Accuracy
accuracy <- sum(predicted_classes == test_data$label) / nrow(test_data)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.535117056856187"
```

b. i. What is the accuracy of the logistic regression classifier?

Answer:

Looking at the outcome of the model, the accuracy of the logistic regression classifier for this dataset is 0.5351. This means that the model correctly predicted 53.51% of the outcomes. This is a slightly better outcome than randomly guessing, where chance the chance for prediction is 50/50. However, this model doesn't have impressive predictive power because it is almost on par with the random guessing.