DSC520_Week10_Assignment_Guruprasad_VelikaduKrishnamoorthy

Guruprasad Velikadu Krishnamoorthy

2023-02-19

Week 10 Assignment

```
library(foreign)
library(caTools)
library(dplyr)
library(kableExtra)
```

1.b.i 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.

```
# Loading the arff file using Foreign package
thoraric_data = read.arff("ThoraricSurgery.arff")

kbl(head(thoraric_data), caption = "Thoraric DataFrame", booktabs = T) %>%
    kable_styling(latex_options = c("striped", "hold_position"))
```

Table 1: Thoraric DataFrame

DGN	PRE4	PRE5	PRE6	PRE7	PRE8	PRE9	PRE10	PRE11	PRE14	PRE17	PRE19	PRE25	P
DGN2	2.88	2.16	PRZ1	F	F	F	T	T	OC14	F	F	F	T
DGN3	3.40	1.88	PRZ0	F	F	F	F	F	OC12	F	F	F	T
DGN3	2.76	2.08	PRZ1	F	F	F	T	F	OC11	F	F	F	T
DGN3	3.68	3.04	PRZ0	F	F	F	F	F	OC11	F	F	F	F
DGN3	2.44	0.96	PRZ2	F	T	F	T	T	OC11	F	F	F	T
DGN3	2.48	1.88	PRZ1	F	F	F	T	F	OC11	F	F	F	F

```r
# Examining the structure
str(thoraric_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
# creating a split variable with 80:20 ratio for train and test datasets
set.seed(40)
split <- sample.split(thoraric_data, SplitRatio = 0.8)
split
```

```
##  [1]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
## [13]  TRUE  TRUE FALSE FALSE  TRUE
```

```r
train <- subset(thoraric_data, split == "TRUE")
test <- subset(thoraric_data, split == "FALSE")
# Creating Logistic Regression Model
td_model1 <- glm(Risk1Yr ~ ., data = train, family = binomial())
summary(td_model1)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ ., family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7762  -0.5497  -0.4312  -0.2686   2.3549
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -15.68634 2399.54536  -0.007  0.99478
## DGNDGN2       15.02505 2399.54477   0.006  0.99500
## DGNDGN3       14.32090 2399.54475   0.006  0.99524
## DGNDGN4       14.51034 2399.54479   0.006  0.99518
## DGNDGN5       16.48816 2399.54483   0.007  0.99452
## DGNDGN6        0.22533 2936.96158   0.000  0.99994
```

```
## DGNDGN8       17.92799 2399.54521   0.007  0.99404
## PRE4          -0.25673    0.21303  -1.205  0.22816
## PRE5          -0.03431    0.01817  -1.888  0.05903 .
## PRE6PRZ1      -0.53640    0.55999  -0.958  0.33813
## PRE6PRZ2      -0.16557    0.85070  -0.195  0.84568
## PRE7T          1.13739    0.58034   1.960  0.05001 .
## PRE8T          0.03009    0.43633   0.069  0.94503
## PRE9T          1.42902    0.56200   2.543  0.01100 *
## PRE10T         0.46628    0.50582   0.922  0.35661
## PRE11T         0.45313    0.44037   1.029  0.30350
## PRE14OC12      0.32802    0.35738   0.918  0.35870
## PRE14OC13      0.97382    0.68707   1.417  0.15638
## PRE14OC14      1.52965    0.67835   2.255  0.02414 *
## PRE17T         1.34764    0.48521   2.777  0.00548 **
## PRE19T       -14.64351 1664.49659  -0.009  0.99298
## PRE25T         0.03566    1.06110   0.034  0.97319
## PRE30T         1.09045    0.53228   2.049  0.04050 *
## PRE32T       -14.69129 2399.54476  -0.006  0.99511
## AGE           -0.01978    0.02019  -0.980  0.32723
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 321.52  on 360  degrees of freedom
## Residual deviance: 272.86  on 336  degrees of freedom
## AIC: 322.86
##
## Number of Fisher Scoring iterations: 15
```

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

```r
# Reading the p-values from our model
p_values <- coef(summary(td_model1))[, 4]
p_values[p_values <= 0.05]
```

```
##       PRE9T   PRE14OC14      PRE17T      PRE30T
## 0.010997948 0.024136566 0.005479208 0.040496926
```

Solution: # Results suggest based on the p values that the variables PRE9T, PRE14OC14, PRE17T, PRE30T are statistically significant as the p values were less than 0.05.

1.b.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?

```
# Running the test through the model
result1 <- predict(td_model1, train, type = "response")
head(result1)
```

```
##          1          3          4          5          6          7
## 0.58451597 0.09176584 0.02980572 0.16873548 0.04126114 0.25568256
```

```
# Validate the model by creating ConfusionMatrix
confmatrix <- table(Actual_value = train$Risk1Yr, Predicted_Value = result1 > 0.5)
confmatrix
```

```
##             Predicted_Value
## Actual_value FALSE TRUE
##            F   291   11
##            T    52    7
```

```
# Calculating the Accuracy
accuracy_model1 <- round(((confmatrix[[1, 1]] + confmatrix[[2, 2]])/sum(confmatrix)), digits = 4) *
    100
accuracy_model1
```

```
## [1] 82.55
```

```
# Creating functions to calculate the Precision and Recall
model_precision <- function(input_matrix) {
    # Selecting only True positive
    tp <- input_matrix[2, 2]
    # Selecting only false positive
    fp <- input_matrix[1, 2]
    return(tp/(tp + fp))
}

model_recall <- function(input_matrix) {
    # Selecting onlytrue positive
    tp <- input_matrix[2, 2]
    # Selecting only false negative
    fn <- input_matrix[2, 1]
    return(tp/(tp + fn))
}
# Calculating Precision and Recall for the model
precision_model1 <- model_precision(confmatrix)
precision_model1
```

```
## [1] 0.3888889
```

```
recall_model1 <- model_recall(confmatrix)
recall_model1
```

```
## [1] 0.1186441
```

Solution: The accuracy of the model Thoraric model is 82.55 %, Precision is 0.3888889 and recall is 0.1186441

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

```
# Reading the Binary Classifier Dataset
bin_clas_df <- read.csv("binary-classifier-data.csv")

kbl(head(bin_clas_df), caption = "Binary Classifier DataFrame", booktabs = T) %>%
    kable_styling(latex_options = c("striped", "hold_position"))
```

Table 2: Binary Classifier DataFrame

label	x	y
0	70.88469	83.17702
0	74.97176	87.92922
0	73.78333	92.20325
0	66.40747	81.10617
0	69.07399	84.53739
0	72.23616	86.38403

```
# Changing the type of label as Factor
bin_clas_df$label <- as.factor(bin_clas_df$label)
# Creating a new split variable
split_2 <- sample.split(bin_clas_df, SplitRatio = 0.8)
# Creating seperate dataset for training and testing the model
train_2 <- subset(bin_clas_df, split == "TRUE")
test_2 <- subset(bin_clas_df, split == "FALSE")
# Creating new model for Binary Classifier dataset
bin_clas_model <- glm(label ~ x + y, data = train_2, family = binomial())
summary(bin_clas_model)
```

```
##
## Call:
## glm(formula = label ~ x + y, family = binomial(), data = train_2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3710  -1.1665  -0.9581   1.1643   1.4069
##
```

```
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.426966   0.133741   3.192 0.001411 **
## x           -0.002367   0.002087  -1.134 0.256781
## y           -0.008204   0.002153  -3.811 0.000138 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1586.7  on 1144  degrees of freedom
## Residual deviance: 1568.0  on 1142  degrees of freedom
## AIC: 1574
##
## Number of Fisher Scoring iterations: 4
```

```
# Results indicate y can be significant in predicting the label as the p value is less
# than 0.05
```

2.b. 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.

What is the accuracy of the logistic regression classifier?

```
# Running the test through the model
res_bin <- predict(bin_clas_model, train_2, type = "response")

# Validate the model by creating a confusion matrix
confmatrix_2 <- table(Actual_value = train_2$label, Predicted_Value = res_bin > 0.5)
confmatrix_2
```

```
##             Predicted_Value
## Actual_value FALSE TRUE
##            0   326  260
##            1   229  330
```

```
# Accuracy
bin_clas_model_accuracy <- round(((confmatrix_2[[1, 1]] + confmatrix_2[[2, 2]])/sum(confmatrix_2)),
    digits = 4) * 100
bin_clas_model_accuracy
```

```
## [1] 57.29
```

```
# # Calculating Precision and Recall for the model
precision_bin_clas_model <- model_precision(confmatrix_2)
precision_bin_clas_model
```

```
## [1] 0.559322
```

```
recall_bin_clas_model <- model_recall(confmatrix_2)
recall_bin_clas_model
```

```
## [1] 0.5903399
```

Solution: The accuracy of the model Binary Classifier data model is 57.29 %, Precision is 0.559322 and recall is 0.5903399