Assignment: ASSIGNMENT 10.2

Name: Shekhar, Manish

Date: 2021-05-15

Logistic Regression Thoraric Surgery Binary dataset

1. Read the Thoraric surgery binary dataset

Downloaded data from web and copied in th current working directory

Reading using foreign library

Installing foreign library and loading it

```
# loading "foreign" library to be able to read .arff file
library("foreign")

# reading data into data frame
thoraric_df <- read.arff("ThoraricSurgery.arff")
str(thoraric_df)
```

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

```
# Attribute info
# 1. DGN: Diagnosis - specific combination of ICD-10 codes for primary and secondary as well multiple t
# 2. PRE4: Forced vital capacity - FVC (numeric)
# 3. PRE5: Volume that has been exhaled at the end of the first second of forced expiration - FEV1 (num
# 4. PRE6: Performance status - Zubrod scale (PRZ2,PRZ1,PRZ0)
# 5. PRE7: Pain before surgery (T,F)
# 6. PRE8: Haemoptysis before surgery (T,F)
# 7. PRE9: Dyspnoea before surgery (T,F)
# 8. PRE10: Cough before surgery (T,F)
# 9. PRE11: Weakness before surgery (T,F)
# 10. PRE14: T in clinical TNM - size of the original tumor, from OC11 (smallest) to OC14 (largest) (OC
# 11. PRE17: Type 2 DM - diabetes mellitus (T,F)
# 12. PRE19: MI up to 6 months (T,F)
```

```
# 13. PRE25: PAD - peripheral arterial diseases (T,F)
# 14. PRE30: Smoking (T,F)
# 15. PRE32: Asthma (T,F)
# 16. AGE: Age at surgery (numeric)
# 17. Risk1Y: 1 year survival period - (T)rue value if died (T,F)

head(thoraric_df)
```

```
##    DGN PRE4 PRE5 PRE6 PRE7 PRE8 PRE9 PRE10 PRE11 PRE14 PRE17 PRE19 PRE25 PRE30
## 1 DGN2 2.88 2.16 PRZ1    F    F    F     T     T  OC14     F     F     F     T
## 2 DGN3 3.40 1.88 PRZ0    F    F    F     F     F  OC12     F     F     F     T
## 3 DGN3 2.76 2.08 PRZ1    F    F    F     T     F  OC11     F     F     F     T
## 4 DGN3 3.68 3.04 PRZ0    F    F    F     F     F  OC11     F     F     F     F
## 5 DGN3 2.44 0.96 PRZ2    F    T    F     T     T  OC11     F     F     F     T
## 6 DGN3 2.48 1.88 PRZ1    F    F    F     T     F  OC11     F     F     F     F
##   PRE32 AGE Risk1Yr
## 1     F  60       F
## 2     F  51       F
## 3     F  59       F
## 4     F  54       F
## 5     F  73       T
## 6     F  51       F
```

2. Data Munging - Cleanse and standardize the data for modelling

```
### Changing DGN to factor with numerical values
thoraric_df$DGN <- factor(thoraric_df$DGN,
                          levels = c("DGN2","DGN3","DGN4","DGN8","DGN5","DGN6","DGN1"),
                          labels = c(2,3,4,8,5,6,1))

### Changing PRE6 to factor with numerical values
thoraric_df$PRE6 <- factor(thoraric_df$PRE6,
                           levels = c("PRZ0","PRZ1","PRZ2"),
                           labels = c(0,1,2))

### Changing PRE14 to factor with numerical values
thoraric_df$PRE14 <- factor(thoraric_df$PRE14,
                           levels = c("OC14","OC12","OC11","OC13"),
                           labels = c(14,12,11,13))

### Changing other PRE variables to factor with numerical values
thoraric_df$PRE7 <- factor(thoraric_df$PRE7,
                           levels = c("F","T"),
                           labels = c(0,1))

thoraric_df$PRE8 <- factor(thoraric_df$PRE8,
                           levels = c("F","T"),
                           labels = c(0,1))

thoraric_df$PRE9 <- factor(thoraric_df$PRE9,
                           levels = c("F","T"),
                           labels = c(0,1))
```

```r
thoraric_df$PRE10 <- factor(thoraric_df$PRE10,
                            levels = c("F","T"),
                            labels = c(0,1))

thoraric_df$PRE11 <- factor(thoraric_df$PRE11,
                            levels = c("F","T"),
                            labels = c(0,1))

thoraric_df$PRE17 <- factor(thoraric_df$PRE17,
                            levels = c("F","T"),
                            labels = c(0,1))

thoraric_df$PRE19 <- factor(thoraric_df$PRE19,
                            levels = c("F","T"),
                            labels = c(0,1))

thoraric_df$PRE25 <- factor(thoraric_df$PRE25,
                            levels = c("F","T"),
                            labels = c(0,1))

thoraric_df$PRE30 <- factor(thoraric_df$PRE30,
                            levels = c("F","T"),
                            labels = c(0,1))

thoraric_df$PRE32 <- factor(thoraric_df$PRE32,
                            levels = c("F","T"),
                            labels = c(0,1))

### Changing dependent variable Risk1Yr to factor with numerical values
thoraric_df$Risk1Yr <- factor(thoraric_df$Risk1Yr,
                            levels = c("F","T"),
                            labels = c(0,1))

# check the data
str(thoraric_df)
```

```
## 'data.frame':    470 obs. of  17 variables:
##  $ DGN    : Factor w/ 7 levels "2","3","4","8",..: 1 2 2 2 2 2 2 1 2 2 ...
##  $ 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 "0","1","2": 2 1 2 1 3 2 2 2 3 2 ...
##  $ PRE7   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE8   : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 1 ...
##  $ PRE9   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE10  : Factor w/ 2 levels "0","1": 2 1 2 1 2 2 2 2 2 2 ...
##  $ PRE11  : Factor w/ 2 levels "0","1": 2 1 1 1 2 1 1 1 2 1 ...
##  $ PRE14  : Factor w/ 4 levels "14","12","11",..: 1 2 3 3 3 3 2 3 3 3 ...
##  $ PRE17  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 1 ...
##  $ PRE19  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE25  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ PRE30  : Factor w/ 2 levels "0","1": 2 2 2 1 2 1 2 2 2 2 ...
##  $ PRE32  : Factor w/ 2 levels "0","1": 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 "0","1": 1 1 1 1 2 1 2 2 1 1 ...
```

```
head(thoraric_df)
```

```
##   DGN PRE4 PRE5 PRE6 PRE7 PRE8 PRE9 PRE10 PRE11 PRE14 PRE17 PRE19 PRE25 PRE30
## 1   2 2.88 2.16    1    0    0    0     1     1    14     0     0     0     1
## 2   3 3.40 1.88    0    0    0    0     0     0    12     0     0     0     1
## 3   3 2.76 2.08    1    0    0    0     1     0    11     0     0     0     1
## 4   3 3.68 3.04    0    0    0    0     0     0    11     0     0     0     0
## 5   3 2.44 0.96    2    0    1    0     1     1    11     0     0     0     1
## 6   3 2.48 1.88    1    0    0    0     1     0    11     0     0     0     0
##   PRE32 AGE Risk1Yr
## 1     0  60       0
## 2     0  51       0
## 3     0  59       0
## 4     0  54       0
## 5     0  73       1
## 6     0  51       0
```

3. Training and Test dataset creation

```
# Split the data into training and test data set
# using caTools library
library(caTools)
# installing some other required packages
library("car")
```

```
## Loading required package: carData
```

```
library("mlogit")
```

```
## Loading required package: dfidx
```

```
##
## Attaching package: 'dfidx'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
set.seed(123)
# using sample.split() to split the data on dependent variable
# choosing split ratio of 80% i.e. 80% training data and 20% test data
split = sample.split(thoraric_df$Risk1Yr, SplitRatio = 0.8)
training_set = subset(thoraric_df, split == TRUE)
test_set = subset(thoraric_df, split == FALSE)
```

4. Feature Scaling

```
# Scale the feature to bring the values to same scale without changing the variation within
# scaling numeric predictor variables in each training set and test set
training_set[,c(2,3,16)] = scale(training_set[,c(2,3,16)])
test_set[,c(2,3,16)] = scale(test_set[,c(2,3,16)])
str(training_set)
```

```
## 'data.frame':    376 obs. of  17 variables:
##  $ DGN    : Factor w/ 7 levels "2","3","4","8",..: 1 2 2 2 2 2 2 1 2 1 ...
##  $ PRE4   : num  -0.441 -0.576 1.221 -0.126 -1.07 ...
```

```
##  $ PRE5   : num  -0.202 -0.209 -0.108 -0.162 -0.202 ...
##  $ PRE6   : Factor w/ 3 levels "0","1","2": 2 2 2 3 2 2 2 3 2 1 ...
##  $ PRE7   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE8   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ PRE9   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE10  : Factor w/ 2 levels "0","1": 2 2 2 2 2 1 2 2 2 1 ...
##  $ PRE11  : Factor w/ 2 levels "0","1": 2 1 1 2 1 1 2 2 1 1 ...
##  $ PRE14  : Factor w/ 4 levels "14","12","11",..: 1 3 2 3 3 2 3 1 2 2 ...
##  $ PRE17  : Factor w/ 2 levels "0","1": 1 1 2 1 1 1 1 1 1 1 ...
##  $ PRE19  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE25  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PRE30  : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ PRE32  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ AGE    : num  -0.263 -0.376 -0.376 0.636 -0.938 ...
##  $ Risk1Yr: Factor w/ 2 levels "0","1": 1 1 2 1 1 1 1 2 1 1 ...
```

5. create a model using glm()

```r
# creating a binary classifier
classifier = glm(formula = Risk1Yr ~ .,
                 family = binomial,
                 data = training_set)

# check the summary of the model
summary(classifier)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ ., family = binomial, data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5376  -0.5623  -0.4264  -0.2572   2.7614
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -2.3496     0.9065  -2.592  0.00954 **
## DGN3          -0.5258     0.4539  -1.159  0.24665
## DGN4          -0.2259     0.6578  -0.343  0.73130
## DGN8           3.6275     1.6522   2.196  0.02812 *
## DGN5           1.5984     0.7928   2.016  0.04380 *
## DGN6         -14.4107  1361.5598  -0.011  0.99156
## DGN1         -14.3872  2399.5448  -0.006  0.99522
## PRE4          -0.2054     0.1817  -1.130  0.25828
## PRE5          -0.2010     0.2095  -0.959  0.33736
## PRE61         -0.1361     0.6083  -0.224  0.82296
## PRE62          0.1528     0.8979   0.170  0.86491
## PRE71          0.1637     0.6390   0.256  0.79785
## PRE81          0.4511     0.4290   1.051  0.29310
## PRE91          1.2056     0.5562   2.167  0.03020 *
## PRE101         0.8221     0.5754   1.429  0.15312
## PRE111         0.1607     0.4698   0.342  0.73228
## PRE1412       -0.8287     0.6038  -1.373  0.16989
## PRE1411       -1.4391     0.6539  -2.201  0.02775 *
```

```
## PRE1413        0.1049     0.7989   0.131  0.89553
## PRE171         0.9403     0.5108   1.841  0.06566 .
## PRE191       -15.0099  2399.5448  -0.006  0.99501
## PRE251       -14.4685  1156.0383  -0.013  0.99001
## PRE301         1.0841     0.5962   1.818  0.06900 .
## PRE321       -13.7943  1670.9124  -0.008  0.99341
## AGE           -0.1849     0.1805  -1.024  0.30576
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 316.49  on 375  degrees of freedom
## Residual deviance: 271.73  on 351  degrees of freedom
## AIC: 321.73
##
## Number of Fisher Scoring iterations: 15
```

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

I am chossing PRE9: Dyspnoea before surgery (T,F) as the predictor variable that has the highest effect on the outcome because z-value associated with coefficient of this variable is 2.167 with p-value = 0.03020. P-value < 0.05 shows that effect is significant.

We do have couple of other variables that seems to have significant effect -

1. PRE1411 - with z-value = -2.201 and p-value = 0.02775. Not choosing it has most significant because it is one out of three dummy variables of one variable.

2. DGN8 - with z-value = 2.196 and p-value = 0.02812. Not choosing it has most significant because it is one of six dummy variables of one variable.

7. Predicting using the model

```r
# predicting using test data
# keeping only the predictor variables in test_data when passing to the predict()
prob_predict <- predict(classifier, type = 'response', newdata = test_set[,c(1:16)])
# creating the vector converting probability vector to vector or 0 or 1
# assuming threshold probability is 50%
y_pred <- ifelse(prob_predict > 0.5, 1, 0)
# building the confusion matrix
# pass the vector of "actual values" of predicted variable from test set, and vector of predicted values
cm <- table(test_set[,17],y_pred)
cm
```

```
##    y_pred
##      0  1
##   0 79  1
##   1 14  0
```

```r
# calculating accuracy of the model
# accuracy = total correct prediction / total number of observation
accuracy = (79+0)/94
accuracy
```

```
## [1] 0.8404255
```

```r
# Thus accuracy of the model is 84%

# Creating function to get various R^square values for the logistic regression model
# Per Discovering Statistics Using R book
logisticPseudoR2s <- function(LogModel){
    dev <- LogModel$deviance
    nullDev <- LogModel$null.deviance
    modelN <- length(LogModel$fitted.values)
    R.l <- 1 - dev/nullDev
    R.cs <- 1 - exp(-(nullDev-dev)/modelN)
    R.n <- R.cs / (1 - (exp(-(nullDev/modelN))))
    cat("Pseudo R^2 for logistic regression\n")
    cat("Hosmer and Lemeshow R^2: ", round(R.l, 3), "\n")
    cat("Cox and Snell R^2: ", round(R.cs, 3), "\n")
    cat("Nagelkerke R^2: ", round(R.n, 3), "\n")
}

# Getting multiple different R^squared values for the model
logisticPseudoR2s(classifier)
```

```
## Pseudo R^2 for logistic regression
## Hosmer and Lemeshow R^2:  0.141
## Cox and Snell R^2:  0.112
## Nagelkerke R^2:  0.197
```

8. Recreating the model removing some insignificant predictors.

```r
### Taking out predictor variables with highest p-values (backward elimination)
# Taking out below variables
# PRE61         -0.1361     0.6083  -0.224  0.82296
# PRE62          0.1528     0.8979   0.170  0.86491
# PRE71          0.1637     0.6390   0.256  0.79785
# PRE111         0.1607     0.4698   0.342  0.73228
# PRE191       -15.0099  2399.5448  -0.006  0.99501
# PRE251       -14.4685  1156.0383  -0.013  0.99001
# PRE321       -13.7943  1670.9124  -0.008  0.99341

# Recreating data frame with only selected predictor variables
thoraric_df_2 <- thoraric_df[,c(-4,-5,-9,-12,-13,-15)]

# Recreating training and test set with only selected predictor variables
set.seed(123)
# using sample.split() to split the data on dependent variable
# choosing split ratio of 80% i.e. 80% training data and 20% test data
split = sample.split(thoraric_df_2$Risk1Yr, SplitRatio = 0.8)
training_set_2 = subset(thoraric_df_2, split == TRUE)
test_set_2 = subset(thoraric_df_2, split == FALSE)

# Scale the feature to bring the values to same scale without changing the variation within
# scaling numeric predictor variables in each training set and test set
training_set_2[,c(2,3,10)] = scale(training_set_2[,c(2,3,10)])
test_set_2[,c(2,3,10)] = scale(test_set_2[,c(2,3,10)])
```

```r
# creating a binary classifier
classifier_2 = glm(formula = Risk1Yr ~ .,
                   family = binomial,
                   data = training_set_2)

# check the summary of the model
summary(classifier_2)
```

```
##
## Call:
## glm(formula = Risk1Yr ~ ., family = binomial, data = training_set_2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5424  -0.5583  -0.4276  -0.2621   2.7802
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -2.32888    0.88874  -2.620  0.00878 **
## DGN3          -0.54297    0.44828  -1.211  0.22581
## DGN4          -0.22166    0.65324  -0.339  0.73437
## DGN8           3.68238    1.64436   2.239  0.02513 *
## DGN5           1.63992    0.78475   2.090  0.03664 *
## DGN6         -13.43190  827.06218  -0.016  0.98704
## DGN1         -13.32991 1455.39764  -0.009  0.99269
## PRE4          -0.18994    0.17886  -1.062  0.28826
## PRE5          -0.18617    0.20205  -0.921  0.35683
## PRE81          0.51461    0.41089   1.252  0.21041
## PRE91          1.16809    0.53951   2.165  0.03038 *
## PRE101         0.80400    0.43245   1.859  0.06300 .
## PRE1412       -0.90474    0.59703  -1.515  0.12967
## PRE1411       -1.48670    0.64627  -2.300  0.02142 *
## PRE1413        0.08577    0.79171   0.108  0.91373
## PRE171         0.94526    0.50893   1.857  0.06326 .
## PRE301         1.07672    0.59195   1.819  0.06892 .
## AGE           -0.14615    0.17304  -0.845  0.39835
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 316.49  on 375  degrees of freedom
## Residual deviance: 273.92  on 358  degrees of freedom
## AIC: 309.92
##
## Number of Fisher Scoring iterations: 14
```

```r
# predicting using test data
# keeping only the predictor variables in test_data when passing to the predict()
prob_predict_2 <- predict(classifier_2, type = 'response', newdata = test_set_2[,c(1:10)])
# creating the vector converting probability vector to vector or 0 or 1
# assuming threshold probability is 50%
y_pred_2 <- ifelse(prob_predict_2 > 0.5, 1, 0)
# building the confusion matrix
```

```r
# pass the vector of "actual values" of predicted variable from test set, and vector of predicted values
cm_2 <- table(test_set_2[,11],y_pred_2)
cm_2
```

```
##    y_pred_2
##      0  1
##   0 78  2
##   1 13  1
```

```r
# calculating accuracy of the model
# accuracy = total correct prediction / total number of observation
accuracy_2 = (78+1)/94
accuracy_2
```

```
## [1] 0.8404255
```

```r
# Thus accuracy of the model 2 did not change and is as model 1 = 84%
# Though we see gain in False positive and reduction in False negative

# Getting multiple different R^squared values for the model
logisticPseudoR2s(classifier_2)
```

```
## Pseudo R^2 for logistic regression
## Hosmer and Lemeshow R^2:  0.135
## Cox and Snell R^2:  0.107
## Nagelkerke R^2:  0.188
```

```r
# We can compare the models by finding the difference in the deviance statistics
modelchi <- classifier$deviance - classifier_2$deviance
chidf <- classifier$df.residual - classifier_2$df.residual
# get the significance
chisq.prob <- 1 - pchisq(modelchi, chidf)
```

```
## Warning in pchisq(modelchi, chidf): NaNs produced
```

```r
# print the results
modelchi
```

```
## [1] -2.187558
```

```r
chidf
```

```
## [1] -7
```

```r
chisq.prob
```

```
## [1] NaN
```

2. Fit a Logistic Regression Model on the binary classifier data

```r
# Read the binary classifier data file into data frame
# I copied it to working directory
bin_cls_data <- read.csv('binary-classifier-data.csv')
# check the structure of the data
str(bin_cls_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 ...
```

```
head(bin_cls_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
```

```
summary(bin_cls_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
```

```
# Creating the training and test data
set.seed(123)
# using sample.split() to split the data on dependent variable
# choosing split ratio of 80% i.e. 80% training data and 20% test data
split = sample.split(bin_cls_data$label, SplitRatio = 0.8)
training_set_3 = subset(bin_cls_data, split == TRUE)
test_set_3 = subset(bin_cls_data, split == FALSE)

# Scale the feature to bring the values to same scale without changing the variation within
# scaling numeric predictor variables in each training set and test set
training_set_3[,c(2,3)] = scale(training_set_3[,c(2,3)])
test_set_3[,c(2,3)] = scale(test_set_3[,c(2,3)])

# creating a binary classifier
classifier_3 = glm(formula = label ~ .,
                 family = binomial,
                 data = training_set_3)

# check the summary of the model
summary(classifier_3)
```

```
##
## Call:
## glm(formula = label ~ ., family = binomial, data = training_set_3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3971  -1.1687  -0.9331   1.1606   1.4137
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.04982    0.05836  -0.854    0.393
## x           -0.08259    0.05974  -1.383    0.167
```

```
## y           -0.25659    0.06005  -4.273 1.93e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1661.5  on 1198  degrees of freedom
## Residual deviance: 1637.6  on 1196  degrees of freedom
## AIC: 1643.6
##
## Number of Fisher Scoring iterations: 4
```

```
# predicting using test data
# keeping only the predictor variables in test_data when passing to the predict()
prob_predict_3 <- predict(classifier_3, type = 'response', newdata = test_set_3[,c(2,3)])
# creating the vector converting probability vector to vector or 0 or 1
# assuming threshold probability is 50%
y_pred_3 <- ifelse(prob_predict_3 > 0.5, 1, 0)

# building the confusion matrix
# pass the vector of "actual values" of predicted variable from test set, and vector of predicted value
cm_3 <- table(test_set_3[,1],y_pred_3)
cm_3
```

```
##    y_pred_3
##      0  1
##   0 81 72
##   1 62 84
```

```
# calculating accuracy of the model
# accuracy = total correct prediction / total number of observation
accuracy_3 = (86+83)/(86+83+67+63)
accuracy_3
```

```
## [1] 0.5652174
```

```
# Thus accuracy of the model is 56.5%

# Getting multiple different R^squared values for the model
logisticPseudoR2s(classifier_3)
```

```
## Pseudo R^2 for logistic regression
## Hosmer and Lemeshow R^2:  0.014
## Cox and Snell R^2:  0.02
## Nagelkerke R^2:  0.026
```