R version 4.0.2 (2020-06-22) -- "Taking Off Again"

Copyright (C) 2020 The R Foundation for Statistical Computing

Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.

You are welcome to redistribute it under certain conditions.

Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.

Type 'contributors()' for more information and

'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or

'help.start()' for an HTML browser interface to help.

Type 'q()' to quit R.

> knitr::opts\_chunk$set(echo = TRUE)

> ## Set the working directory to the root of your DSC 520 directory

> setwd("C:/Users/binay/Documents/GitHub/dsc520/")

The working directory was changed to C:/Users/binay/Documents/GitHub/dsc520 inside a notebook chunk. The working directory will be reset when the chunk is finished running. Use the knitr root.dir option in the setup chunk to change the working directory for notebook chunks.>

> ## Load the `foreign` library

> library(foreign)

>

> ## Load the `data/ThoraricSurgery.arff` to

> thoracic\_surgery\_df <- read.arff('data/ThoraricSurgery.arff')

> head(thoracic\_surgery\_df)

>

>

> # Since Risk1Y = True means patient died and we want to predict whether patient survived.

> # We should set baseline category as True, (which means Not Survived)

> # otherwise default will be taken as False due to alphabetical order

> # This means T = 0 and F = 1

> thoracic\_surgery\_df$Risk1Yr<-relevel(thoracic\_surgery\_df$Risk1Yr, "T")

>

> # Could also split this df using split function to training set and test set.

> # However leaving it for this exercise, assuming all dataset is training dataset.

>

> # This model includes all other parameters as dependent

> lrmodel.1 <- glm(Risk1Yr ~ . , family ='binomial' , data = thoracic\_surgery\_df)

> summary(lrmodel.1)

Call:

glm(formula = Risk1Yr ~ ., family = "binomial", data = thoracic\_surgery\_df)

Deviance Residuals:

Min 1Q Median 3Q Max

-2.4929 0.2762 0.4199 0.5439 1.6084

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 1.655e+01 2.400e+03 0.007 0.99450

DGNDGN2 -1.474e+01 2.400e+03 -0.006 0.99510

DGNDGN3 -1.418e+01 2.400e+03 -0.006 0.99528

DGNDGN4 -1.461e+01 2.400e+03 -0.006 0.99514

DGNDGN5 -1.638e+01 2.400e+03 -0.007 0.99455

DGNDGN6 -4.089e-01 2.673e+03 0.000 0.99988

DGNDGN8 -1.803e+01 2.400e+03 -0.008 0.99400

PRE4 2.272e-01 1.849e-01 1.229 0.21909

PRE5 3.030e-02 1.786e-02 1.697 0.08971 .

PRE6PRZ1 4.427e-01 5.199e-01 0.852 0.39448

PRE6PRZ2 2.937e-01 7.907e-01 0.371 0.71030

PRE7T -7.153e-01 5.556e-01 -1.288 0.19788

PRE8T -1.743e-01 3.892e-01 -0.448 0.65419

PRE9T -1.368e+00 4.868e-01 -2.811 0.00494 \*\*

PRE10T -5.770e-01 4.826e-01 -1.196 0.23185

PRE11T -5.162e-01 3.965e-01 -1.302 0.19295

PRE14OC12 -4.394e-01 3.301e-01 -1.331 0.18318

PRE14OC13 -1.179e+00 6.165e-01 -1.913 0.05580 .

PRE14OC14 -1.653e+00 6.094e-01 -2.713 0.00668 \*\*

PRE17T -9.266e-01 4.445e-01 -2.085 0.03709 \*

PRE19T 1.466e+01 1.654e+03 0.009 0.99293

PRE25T 9.789e-02 1.003e+00 0.098 0.92227

PRE30T -1.084e+00 4.990e-01 -2.172 0.02984 \*

PRE32T 1.398e+01 1.645e+03 0.008 0.99322

AGE 9.506e-03 1.810e-02 0.525 0.59944

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 395.61 on 469 degrees of freedom

Residual deviance: 341.19 on 445 degrees of freedom

AIC: 391.19

Number of Fisher Scoring iterations: 15

>

> # Trying out another model

> # This model includes only DGN parameters as dependent

> lrmodel.2 <- glm(Risk1Yr ~ DGN , family ='binomial' , data = thoracic\_surgery\_df)

>

> summary(lrmodel.2)

Call:

glm(formula = Risk1Yr ~ DGN, family = "binomial", data = thoracic\_surgery\_df)

Deviance Residuals:

Min 1Q Median 3Q Max

-2.0464 0.5128 0.5128 0.5128 1.1774

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 1.557e+01 1.455e+03 0.011 0.991

DGNDGN2 -1.436e+01 1.455e+03 -0.010 0.992

DGNDGN3 -1.360e+01 1.455e+03 -0.009 0.993

DGNDGN4 -1.382e+01 1.455e+03 -0.009 0.992

DGNDGN5 -1.543e+01 1.455e+03 -0.011 0.992

DGNDGN6 1.300e-08 1.627e+03 0.000 1.000

DGNDGN8 -1.557e+01 1.455e+03 -0.011 0.991

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 395.61 on 469 degrees of freedom

Residual deviance: 379.79 on 463 degrees of freedom

AIC: 393.79

Number of Fisher Scoring iterations: 14

>

>

> result <- predict(lrmodel.1,thoracic\_surgery\_df,type = "response")

> # result

> # validating - putting the actual value and counts of Predicted values in a matrix

> # Since T = 0 and F = 1, so if result < 0.5, it should be T

> confmatrix <- table(ActualValue=thoracic\_surgery\_df$Risk1Yr, PredictedValue = result < 0.5)

> confmatrix

PredictedValue

ActualValue FALSE TRUE

T 67 3

F 390 10

> # accuracy - Cases where we predicted correctly by Total Predictions

> # from matrix, we see when Actual Value is T, confmatrix needs to pick 1,2

> # and when F it should pick 2,1

> (confmatrix[1,2]+confmatrix[2,1])/sum(confmatrix)

[1] 0.8361702

>

Restarting R session...

> knitr::opts\_chunk$set(echo = TRUE)

> ## Set the working directory to the root of your DSC 520 directory

> setwd("C:/Users/binay/Documents/GitHub/dsc520/")

The working directory was changed to C:/Users/binay/Documents/GitHub/dsc520 inside a notebook chunk. The working directory will be reset when the chunk is finished running. Use the knitr root.dir option in the setup chunk to change the working directory for notebook chunks.>

> ## Load the `caTools` library

> library(caTools)

package �caTools� was built under R version 4.0.3

> #library(MASS)

>

> ## Load the `data/binary-classifier-data.csv` to

> binary\_classifier\_df <- read.csv("data/binary-classifier-data.csv")

> head(binary\_classifier\_df)

> summary(binary\_classifier\_df)

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

>

>

> # Since label is number converting to factors, so that it becomes categorical

> binary\_classifier\_df$label <- as.factor(binary\_classifier\_df$label)

>

> # Splitting the dataset for the model into train and test datasets.

> myData <- sample.split(binary\_classifier\_df$label, SplitRatio=0.8)

>

>

> train <- subset(binary\_classifier\_df, myData==TRUE)

> test <- subset(binary\_classifier\_df, myData==FALSE)

>

>

>

> # This model includes all other parameters as dependent

> # Using train dataset to generate the model

> lrmodel.1 <- glm(label ~ . , family ='binomial' , data = train)

> summary(lrmodel.1)

Call:

glm(formula = label ~ ., family = "binomial", data = train)

Deviance Residuals:

Min 1Q Median 3Q Max

-1.3498 -1.1666 -0.9719 1.1686 1.3789

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 0.381339 0.129407 2.947 0.003211 \*\*

x -0.002292 0.002025 -1.132 0.257527

y -0.007384 0.002078 -3.553 0.000381 \*\*\*

---

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: 1645.1 on 1196 degrees of freedom

AIC: 1651.1

Number of Fisher Scoring iterations: 4

>

> # Using test dataset to see if the model is good

> result <- predict(lrmodel.1,test,type = "response")

>

> # result

> # validating - putting the actual value and counts of Predicted values in a matrix

> # Setting to T if result > 0.5

> confmatrix <- table(ActualValue=test$label, PredictedValue = result > 0.5)

> confmatrix

PredictedValue

ActualValue FALSE TRUE

0 96 57

1 66 80

> # accuracy - Cases where we predicted correctly by Total Predictions

> # from matrix, we see when Actual Value is T, confmatrix needs to pick 1,2

> # and when F it should pick 2,1

> (confmatrix[1,1]+confmatrix[2,2])/sum(confmatrix)

[1] 0.5886288

>

> library(class)

>

> # Generating knn model with k=1

> knnmodel.1 <- knn(train[2:3],test[2:3],k=1,cl=train$label)

> summary(knnmodel.1)

0 1

157 142

>

> ##create confusion matrix

> tab <- table(knnmodel.1,test$label)

>

> ##this function divides the correct predictions by total number of predictions that tell us how accurate the model is.

> accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) \* 100}

> accuracy(tab)

[1] 95.31773

>

> # Running for multiple K Values

> for(i in 1:20){

+ ##print(paste("Model with K=", i))

+ knnmodel.i <- knn(train[2:3],test[2:3],k=i,cl=train$label)

+ table.i <- table(knnmodel.i,test$label)

+ print(paste("Accuracy for Model with K=", i ," is ", accuracy(table.i)))

+ }

[1] "Accuracy for Model with K= 1 is 95.3177257525084"

[1] "Accuracy for Model with K= 2 is 94.9832775919732"

[1] "Accuracy for Model with K= 3 is 96.3210702341137"

[1] "Accuracy for Model with K= 4 is 97.6588628762542"

[1] "Accuracy for Model with K= 5 is 96.989966555184"

[1] "Accuracy for Model with K= 6 is 96.989966555184"

[1] "Accuracy for Model with K= 7 is 97.6588628762542"

[1] "Accuracy for Model with K= 8 is 97.3244147157191"

[1] "Accuracy for Model with K= 9 is 97.3244147157191"

[1] "Accuracy for Model with K= 10 is 97.6588628762542"

[1] "Accuracy for Model with K= 11 is 97.6588628762542"

[1] "Accuracy for Model with K= 12 is 97.6588628762542"

[1] "Accuracy for Model with K= 13 is 97.6588628762542"

[1] "Accuracy for Model with K= 14 is 97.6588628762542"

[1] "Accuracy for Model with K= 15 is 97.6588628762542"

[1] "Accuracy for Model with K= 16 is 97.3244147157191"

[1] "Accuracy for Model with K= 17 is 96.989966555184"

[1] "Accuracy for Model with K= 18 is 97.3244147157191"

[1] "Accuracy for Model with K= 19 is 96.989966555184"

[1] "Accuracy for Model with K= 20 is 96.6555183946488"

>

>

>

>

>

> # creating df for plotting the comparison against actuals

> df <- binary\_classifier\_df

> df$predict <- predict(lrmodel.1, df,type = "response")

>

> # Adding details to test df for plotting

> test$knnpredict <- knn(train[2:3],test[2:3],k=1,cl=train$label)

>

> library(ggplot2)

> ggplot(data = binary\_classifier\_df, aes(y = y, x = x, color = label)) +

+ geom\_point() + ggtitle("Actual Data")

>

> ggplot(data = df, aes(y = y, x = x, color = predict>0.5)) +

+ geom\_point() + ggtitle("Prediction-Logistic Regression")

>

> ggplot(data = test, aes(y = y, x = x, color = knnpredict == label)) +

+ geom\_point() + ggtitle("Prediction-KNN-K=1")

>

>