Logistic Regression

Id

24/12/2019

“Logistic regression will be applied on datasets where classifcation output is needed.” “The source of the data is from UCLA which has 4 variable called admit, GRE score, GPA and rank of their undergrad school. Our aim is to build a model so that predict the probability of that student getting admit if we are given his profile.”

df <- read.csv("https://stats.idre.ucla.edu/stat/data/binary.csv")  
str(df)

## 'data.frame': 400 obs. of 4 variables:  
## $ admit: int 0 1 1 1 0 1 1 0 1 0 ...  
## $ gre : int 380 660 800 640 520 760 560 400 540 700 ...  
## $ gpa : num 3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...  
## $ rank : int 3 3 1 4 4 2 1 2 3 2 ...

#We see that variable are either integer or number.

summary(df)

## admit gre gpa rank   
## Min. :0.0000 Min. :220.0 Min. :2.260 Min. :1.000   
## 1st Qu.:0.0000 1st Qu.:520.0 1st Qu.:3.130 1st Qu.:2.000   
## Median :0.0000 Median :580.0 Median :3.395 Median :2.000   
## Mean :0.3175 Mean :587.7 Mean :3.390 Mean :2.485   
## 3rd Qu.:1.0000 3rd Qu.:660.0 3rd Qu.:3.670 3rd Qu.:3.000   
## Max. :1.0000 Max. :800.0 Max. :4.000 Max. :4.000

#We can notice that there are a greater number of rejects than there are acceptance #since the mean of variable admit is less than “0.5”. “We do this to check if the admits are distributed well enough in each category of rank. If let’s say one rank has only 5 admit or reject information, then it will not be necessary to include that rank in analysis.”

xtabs(~ admit +rank ,data=df)

## rank  
## admit 1 2 3 4  
## 0 28 97 93 55  
## 1 33 54 28 12

#getting a sense of data of what we are predicting. #if the data set has balanced stuff.

table(df$admit)

##   
## 0 1   
## 273 127

set.seed(167)

####split dataset

n=nrow(df)   
indexes = sample(n,n\*(80/100))   
trainset = df[indexes,]   
testset = df[-indexes,]

#Fitting the reg Model with family binomial #determining admit with all the other data

full.model = glm(trainset$admit~., data= trainset, family='binomial')   
#summary of model  
summary(full.model)

##   
## Call:  
## glm(formula = trainset$admit ~ ., family = "binomial", data = trainset)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.5750 -0.8919 -0.6359 1.2033 2.1620   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.339556 1.230837 -2.713 0.006663 \*\*   
## gre 0.003092 0.001227 2.520 0.011741 \*   
## gpa 0.568491 0.352207 1.614 0.106510   
## rank -0.492289 0.139208 -3.536 0.000406 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 402.10 on 319 degrees of freedom  
## Residual deviance: 371.03 on 316 degrees of freedom  
## AIC: 379.03  
##   
## Number of Fisher Scoring iterations: 3

#getting the predicted value

phat\_i=predict(full.model,testset, type="response") # phat\_i   
  
predictedvalues=rep(0,length(phat\_i))   
#  
predictedvalues[phat\_i>0.5]=1   
actual=testset$admit   
  
dfPredicted=data.frame(actual,predictedvalues)

#getting the confuction matrix and accuracy “You can manually calculate the accuracy from confusion matrix values.”

confusion\_matrix=table( predictedvalues, actualvalues=actual) #confusion matrix  
confusion\_matrix

## actualvalues  
## predictedvalues 0 1  
## 0 53 17  
## 1 3 7

accuracy=mean(predictedvalues == actual) # accuary

accuracy

## [1] 0.75

“Another example of simple logistic regression using PimaIndiansDiabetics2 dataset”

#install.packages(“mlbench”)

“tidyverse for easy data manipulation and visualization caret for easy machine learning workflow”

library(tidyverse)

## -- Attaching packages ---------------------------------------------- tidyverse 1.3.0 --

## <U+2713> ggplot2 3.2.1 <U+2713> purrr 0.3.3  
## <U+2713> tibble 2.1.3 <U+2713> dplyr 0.8.3  
## <U+2713> tidyr 1.0.0 <U+2713> stringr 1.4.0  
## <U+2713> readr 1.3.1 <U+2713> forcats 0.4.0

## -- Conflicts ------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

data("PimaIndiansDiabetes2", package = "mlbench")

“Data cleanup should include the following- Remove potential outliers Make sure that the predictor variables are normally distributed. If not, you can use log, root, Box-Cox transformation. Remove highly correlated predictors to minimize overfitting. The presence of highly correlated predictors might lead to an unstable model solution.” #This just omits the empty values. Just 1 part of data analysis/creation

PimaIndiansDiabetes2 <- na.omit(PimaIndiansDiabetes2)

# Inspect the data

sample\_n(PimaIndiansDiabetes2, 3)

## pregnant glucose pressure triceps insulin mass pedigree age diabetes  
## 1 2 105 58 40 94 34.9 0.225 25 neg  
## 2 3 158 76 36 245 31.6 0.851 28 pos  
## 3 0 180 90 26 90 36.5 0.314 35 pos

# Split the data into training and test set

set.seed(123)  
training.samples <- PimaIndiansDiabetes2$diabetes %>%   
 createDataPartition(p = 0.8, list = FALSE)  
train.data <- PimaIndiansDiabetes2[training.samples, ]  
test.data <- PimaIndiansDiabetes2[-training.samples, ]

# Fit the model  
model <- glm( diabetes ~., data = train.data, family = binomial)  
# Summarize the model  
summary(model)

##   
## Call:  
## glm(formula = diabetes ~ ., family = binomial, data = train.data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5832 -0.6544 -0.3292 0.6248 2.5968   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.053e+01 1.440e+00 -7.317 2.54e-13 \*\*\*  
## pregnant 1.005e-01 6.127e-02 1.640 0.10092   
## glucose 3.710e-02 6.486e-03 5.719 1.07e-08 \*\*\*  
## pressure -3.876e-04 1.383e-02 -0.028 0.97764   
## triceps 1.418e-02 1.998e-02 0.710 0.47800   
## insulin 5.940e-04 1.508e-03 0.394 0.69371   
## mass 7.997e-02 3.180e-02 2.515 0.01190 \*   
## pedigree 1.329e+00 4.823e-01 2.756 0.00585 \*\*   
## age 2.718e-02 2.020e-02 1.346 0.17840   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 398.80 on 313 degrees of freedom  
## Residual deviance: 267.18 on 305 degrees of freedom  
## AIC: 285.18  
##   
## Number of Fisher Scoring iterations: 5

"Estimate: the intercept (b0) and the beta coefficient estimates associated to each predictor variable

Std.Error: the standard error of the coefficient estimates. This represents the accuracy of the coefficients. The larger the standard error, the less confident we are about the estimate.

z value: the z-statistic, which is the coefficient estimate (column 2) divided by the standard error of the estimate (column 3) Pr(>|z|): The p-value corresponding to the z-statistic.

"

# Make predictions  
probabilities <- model %>% predict(test.data, type = "response")  
predicted.classes <- ifelse(probabilities > 0.5, "pos", "neg")  
# Model accuracy  
mean(predicted.classes == test.data$diabetes)

## [1] 0.7564103

“Doing the same thing with 1 variable, diabetics ~ glucose”

modelSingle <- glm( diabetes ~ glucose, data = train.data, family = binomial)  
summary(modelSingle)

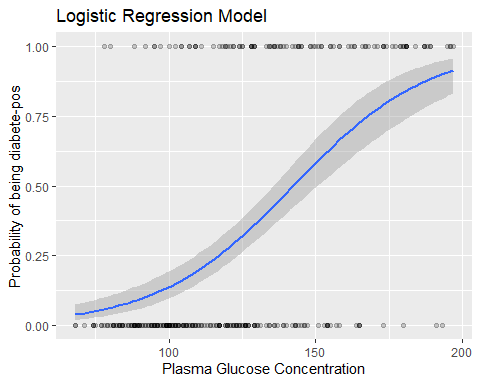
##   
## Call:  
## glm(formula = diabetes ~ glucose, family = binomial, data = train.data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1441 -0.7417 -0.4729 0.6859 2.3848   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -6.158820 0.700097 -8.797 < 2e-16 \*\*\*  
## glucose 0.043272 0.005341 8.102 5.42e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 398.8 on 313 degrees of freedom  
## Residual deviance: 305.7 on 312 degrees of freedom  
## AIC: 309.7  
##   
## Number of Fisher Scoring iterations: 4

#using a random sample data, not the test set.  
newdata <- data.frame(glucose = c(20, 180))  
probabilitiesgl <- modelSingle %>% predict(newdata, type = "response")  
predicted.classesgl <- ifelse(probabilitiesgl > 0.5, "pos", "neg")  
predicted.classesgl

## 1 2   
## "neg" "pos"

#change train data diab to 1,0 to plot, just to show corealtion

mutatedStuff <- mutate(train.data, prob = ifelse(diabetes == "pos", 1, 0))   
ggplot(mutatedStuff,aes(glucose, prob)) +  
 geom\_point(alpha = 0.2) +  
 geom\_smooth(method = "glm", method.args = list(family = "binomial")) +  
 labs(  
 title = "Logistic Regression Model",   
 x = "Plasma Glucose Concentration",  
 y = "Probability of being diabete-pos"  
 )



“Multiple logistic regression The multiple logistic regression is used to predict the probability of class membership based on multiple predictor variables, as follow:”

modelmultiple <- glm( diabetes ~ glucose + mass + pregnant,   
 data = train.data, family = binomial)  
summary(modelmultiple)

##   
## Call:  
## glm(formula = diabetes ~ glucose + mass + pregnant, family = binomial,   
## data = train.data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.2066 -0.6448 -0.3576 0.6523 2.4407   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -9.323698 1.125997 -8.280 < 2e-16 \*\*\*  
## glucose 0.038862 0.005404 7.191 6.43e-13 \*\*\*  
## mass 0.094585 0.023530 4.020 5.83e-05 \*\*\*  
## pregnant 0.144667 0.045126 3.206 0.00135 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 398.80 on 313 degrees of freedom  
## Residual deviance: 279.88 on 310 degrees of freedom  
## AIC: 287.88  
##   
## Number of Fisher Scoring iterations: 5

“Or just doing it with all the values”

modelmultipleAll <- glm( diabetes ~ .,   
 data = train.data, family = binomial)  
summary(modelmultipleAll)

##   
## Call:  
## glm(formula = diabetes ~ ., family = binomial, data = train.data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5832 -0.6544 -0.3292 0.6248 2.5968   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.053e+01 1.440e+00 -7.317 2.54e-13 \*\*\*  
## pregnant 1.005e-01 6.127e-02 1.640 0.10092   
## glucose 3.710e-02 6.486e-03 5.719 1.07e-08 \*\*\*  
## pressure -3.876e-04 1.383e-02 -0.028 0.97764   
## triceps 1.418e-02 1.998e-02 0.710 0.47800   
## insulin 5.940e-04 1.508e-03 0.394 0.69371   
## mass 7.997e-02 3.180e-02 2.515 0.01190 \*   
## pedigree 1.329e+00 4.823e-01 2.756 0.00585 \*\*   
## age 2.718e-02 2.020e-02 1.346 0.17840   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 398.80 on 313 degrees of freedom  
## Residual deviance: 267.18 on 305 degrees of freedom  
## AIC: 285.18  
##   
## Number of Fisher Scoring iterations: 5

“Estimate: the intercept (b0) and the beta coefficient estimates associated to each predictor variable Std.Error: the standard error of the coefficient estimates. This represents the accuracy of the coefficients. The larger the standard error, the less confident we are about the estimate. z value: the z-statistic, which is the coefficient estimate (column 2) divided by the standard error of the estimate (column 3) Pr(>|z|): The p-value corresponding to the z-statistic. The smaller the p-value, the more significant the estimate is.”

“It can be seen that only 5 out of the 8 predictors are significantly associated to the outcome. These include: pregnant, glucose, pressure, mass and pedigree. The coefficient estimate of the variable glucose is b = 0.045, which is positive. This means that an increase in glucose is associated with increase in the probability of being diabetes-positive. However the coefficient for the variable pressure is b = -0.007, which is negative. This means that an increase in blood pressure will be associated with a decreased probability of being diabetes-positive.”

“The regression coefficient for glucose is 0.042. This indicate that one unit increase in the glucose concentration will increase the odds of being diabetes-positive by exp(0.042) 1.04 times.”

" it can be noticed that some variables - triceps, insulin and age - are not statistically significant. Keeping them in the model may contribute to overfitting. Therefore, they should be eliminated. This can be done automatically using statistical techniques, including stepwise regression and penalized regression methods."

modelReduced <- glm( diabetes ~ pregnant + glucose + pressure + mass + pedigree,   
 data = train.data, family = binomial)

"Making predictions We’ll make predictions using the test data in order to evaluate the performance of our logistic regression model.

The procedure is as follow:

Predict the class membership probabilities of observations based on predictor variables Assign the observations to the class with highest probability score (i.e above 0.5) The R function predict() can be used to predict the probability of being diabetes-positive, given the predictor values.

Predict the probabilities of being diabetes-positive:"

probabilities <- modelReduced %>% predict(test.data, type = "response")  
head(probabilities)

## 19 21 32 55 64 71   
## 0.1352603 0.5127526 0.6795376 0.7517408 0.2734867 0.1648174

predicted.classes <- ifelse(probabilities > 0.5, "pos", "neg")  
head(predicted.classes)

## 19 21 32 55 64 71   
## "neg" "pos" "pos" "pos" "neg" "neg"