Today, I am going to cover how to perform logistic regression in R.

To run the logistic regression in R, we can call glm() function in R. glm stands for generalized linear modle. The syntax is very similar to the lm() (linear model) function.

Let’s load the dataset

library(readxl)

StudentsPerformance <- read\_excel("C:/Users/yliu3/OneDrive - Maryville University/Online DSCI502 R Programming/DataSets/StudentsPerformance.xlsx")

Then convert the categorical variables to factor.

#convert categorical variables to factors

StudentsPerformance$Gender <- as.factor(StudentsPerformance$Gender)

StudentsPerformance$Race <- as.factor(StudentsPerformance$Race)

Let’s run the glm model.

glm.result <- glm(Gender ~ MathScore+ ReadingScore + WritingScore + Race,family=binomial,data=StudentsPerformance )

glm() function has three parameters:

* R **formula** specifies the target and predictors.
* **family** specifies the probability distribution. We use binomial distribution for binary classification.
* **data** specifies the data source in data frame format.

Let’s look at the model results by using the summary() function as we did for lm()

summary(glm.result)

### Interpretation the Coefficients in glm model.

For a typical linear model like

If the ReadingScore increases 1 unit, the corresponding MathScore (Left side of equation) increases (coefficient of ReadingScore). On the other hand, if the ReadingScore (Left side of equation) decreases 1 unit, the corresponding MathScore decreases (coefficient of ReadingScore).

But for a logistic regression, it is a **different story** since we model the log of odds ratios instead of probability.

If the MathSocre increases 1 unit, then the **log of odds ratio (not the probability**), (Left side of equation) increases units and that is the coefficient of MathScore.

If the MathSocre decreases 1 unit, then the **log of odds ratio** (Left side of equation) decreases units and that is the coefficient of MathScore.

### Model Selection Based on McFadden/Pseudo R squared

There is no exact R squared formula for logistic regression. There is a McFadden/pseudo R squared to check the in sample fit. The higher of McFadden/pseudo R squared, the better the sample fit. We prefer to **choose logistic models with higher McFadden R squared**.

To get McFadden/pseudo R squared, we need install a package **pscl** in RStudio.

1. Click **Tools** Menu
2. Click **Install Packages…**
3. Type **pscl** under Packages
4. Click **Install** button

After we install it, we can get the McFadden/pseudo R squared by running the following R codes

#load the pscl package  
library(pscl)

#get the McFadden/pseudo R squared  
pR2(glm.result)

We can see that the McFadden/pseudo R squared is 60.11%

### Forecast the Probability

The glm models log of odds ratios directly. To predict the probability, we can use the following R codes.

#predicted probabilities  
predicted.prob <- predict(glm.result, StudentsPerformance, type = "response" )  
  
#look at the first five probabilities  
predicted.prob[1:5]

## 1 2 3 4 5   
## 0.56653085 0.99921000 0.79244067 0.08965245 0.28550828

#set up threshold  
threshold <- 0.5  
#assign female and male respectively based on the levels, which are female, male in the model

#save the forecasted probabilities in a column  
StudentsPerformance$predicted.gender <- ifelse(predicted.prob >= threshold,"male","female")  
  
#compute the accuracy of this model  
my.accuracy <- mean(StudentsPerformance$predicted.gender == StudentsPerformance$Gender)  
  
message("The accuracy is ", my.accuracy)

## The accuracy is 0.884

we need to specify three parameters in predict function:

* glm model results
* data frame containing the **same** variables/columns
* set type = “**response**” and R engine will solve the probabilities for us.

After we forecast the probabilities, we need to map it to “female” and “male” by choosing a threshold, for example, 0.5. If it is larger than this threshold, it is “male” which is the 2nd level of the gender in R memory, otherwise, it is female which is the first level of gender in R memory.

Finally, we compute the accuracy of this model by checking how many predicted values match the actual values using the logical operator “==”. If they match, it returns 1/TRUE, otherwise, it returns 0/FALSE. The average of these results is the accuracy of this model.

Let’s run the R codes. The accuracy is 88.4%.