Sardar Patel Institute of Technology,Mumbai
Department of Electronics and Telecommunication Engineering
T.E. Sem-V (2018-2019)
ETL54-Statistical Computational Laboratory
Lab-4: Regression Analysis: Logistic & Multinomial Logistic Regression

Name: Shubham Parulekar **Roll No.2017120044**

Objective: To carry out logistic regression and multinomial regression and build a regression model.
System Requirements: Ubuntu OS with R and RStudio installed

What is Logistic Regression?

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.
The binary logistic regression model has extensions to more than two levels of the dependent variable: categorical outputs with more than two values are modeled by multinomial logistic regression, and if the multiple categories are ordered , by ordinal logistic regression.

Procedure:

Part 1 :

Train model mydata dataset :

Download the binary.csv from the classroom.
Open R studio
Type in the console:

```
> setwd("D:/R College Lab")
> getwd()
[1] "D:/R College Lab"
> mydata <- read.csv("binary.csv",header = T)
> str(mydata)
'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 ...
```

```
> mydata$admit = as.factor(mydata$admit)
> mydata$rank = as.factor(mydata$rank)
> str(mydata)
'data.frame':    400 obs. of 4 variables:
 $ admit: Factor w/ 2 levels "0","1": 1 2 2 2 1 2 2 1 2 1 ...
 $ 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 : Factor w/ 4 levels "1","2","3","4": 3 3 1 4 4 2 1 2 3 2 ...
> xtabs(~admit+rank,mydata)
     rank
admit 1 2 3  4
    0 28 97 93 55
    1 33 54 28 12
> set.seed(1234)
> ind <- sample(2,nrow(mydata),replace = T,prob = c(0.8,0.2))
> train_model <- mydata[ind==1,]
> test_model <- mydata[ind==2,]
> model <- glm(admit~gre+rank+gpa,family = binomial,data = train_model)
> summary(model)

Call:
glm(formula = admit ~ gre + rank + gpa, family = binomial, data = train_model)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5873 -0.8679 -0.6181 1.1301  2.1178

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.009514 1.316514 -3.805 0.000142 ***
gre          0.001631 0.001217 1.340 0.180180
rank2       -0.570976   0.358273  -1.594 0.111005
rank3       -1.125341   0.383372  -2.935 0.003331 **
rank4       -1.532942   0.477377  -3.211 0.001322 **
gpa          1.166408   0.388899   2.999 0.002706 **
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 404.39 on 324 degrees of freedom
Residual deviance: 369.99  on 319  degrees of freedom
```

```
AIC: 381.99

Number of Fisher Scoring iterations: 4
> pre <- predict(model,test_model,type = "response")
> head(pre)
        5        14        16        26        28        29
0.0930786 0.3002615 0.2076099 0.6375090 0.2088584 0.3653418
> P1 <- ifelse(pre>0.5,1,0)
> head(P1)
 5 14 16 26 28 29
 0 0 0 1 0  0
> table(P1)
P1
 0  1
69  6
> table(test_model$admit)

 0  1
50 25
> tab <- table(P1,test_model$admit)
> tab

P1   0  1
  0 48 21
  1 2  4

> tab <- table(predicted = p1, actual = testdata$admit)
> tab
```

Result :

	actual		
predicted	**0**	**1**	
0	48**(TP)**	21**(FN)**	*69*
1	2 **(FP)**	4 **(TN)**	*6*
	50	*25*	***75***

TN - true negative
TP - true positive
FP - false positive
FN - false negative

- **Accuracy:** Overall, how often is the classifier correct?
    - (TP+TN)/total = (4+48)/75 = 0.6933
- **Misclassification Rate:** Overall, how often is it wrong?
    - (FP+FN)/total = (2+21)/75 = 0.3066667
    - 1 - Accuracy = 1 - 0.6933 = 0.3066667
    - equivalent to 1 minus Accuracy
    - also known as "**Error Rate**"
- **True Positive Rate:** When it's actually yes, how often does it predict yes?
    - TP/actual yes = 48/50 = 0.96
    - also known as "**Sensitivity**" or "**Recall**"
- **False Positive Rate:** When it's actually no, how often does it predict yes?
    - FP/actual no = 2/50 = 0.04
- **True Negative Rate:** When it's actually no, how often does it predict no?
    - TN/actual no = 4/25 = 0.16
    - equivalent to 1 minus False Negative Rate = 1 - 0.84 = 0.16
    - also known as "**Specificity**"
- **Precision:** When it predicts yes, how often is it correct?
    - TP/predicted yes = 48/69 = 0.6957
- **Prevalence:** How often does the yes condition actually occur in our sample?
    - actual yes/total = 50/75 = 0.6667

Overall Statistics:

```
> result <- confusionMatrix(P1,ex)
> result
Confusion Matrix and Statistics

        Reference
Prediction 0  1
       0 48 21
       1 2  4
```

Accuracy : 0.6933
95% CI : (0.5762, 0.7947)
No Information Rate : 0.6667
P-Value [Acc > NIR] : 0.3612408

Kappa : 0.1481

Mcnemar's Test P-Value : 0.0001746

Sensitivity : 0.9600
Specificity : 0.1600
Pos Pred Value : 0.6957
Neg Pred Value : 0.6667
Prevalence : 0.6667
Detection Rate : 0.6400
Detection Prevalence : 0.9200
Balanced Accuracy : 0.5600

'Positive' Class : 0

Part 2:
Train model on iris dataset :

```
> library(nnet)
> library(caret)
> irisdata <- iris
> str(irisdata)
'data.frame':	150 obs. of 5 variables:
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

> set.seed(1234)
> int <- sample(2,nrow(irisdata),replace = T,prob = c(0.8,0.2))
> training <- irisdata[int==1,]
> testing <- irisdata[int==2,]
> model <- multinom(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,family
= multinomial,data = training)
# weights: 18 (10 variable)
initial  value 135.129312
iter  10 value 21.193841
iter  20 value 6.761291
iter  30 value 6.052019
iter  40 value 5.868539
iter  50 value 5.857401
iter  60 value 5.850943
iter  70 value 5.849947
iter  80 value 5.848203
final value 5.848035
converged
> summary(model)
Call:
multinom(formula = Species ~ Sepal.Length + Sepal.Width + Petal.Length +
    Petal.Width, data = training, family = multinomial)

Coefficients:
           (Intercept) Sepal.Length Sepal.Width Petal.Length Petal.Width
versicolor    18.11431    -5.144006  -7.306845     11.20527    0.558515
virginica    -21.71788    -7.503475 -13.553867     19.95351   18.016648

Std. Errors:
```

```
         (Intercept) Sepal.Length Sepal.Width Petal.Length Petal.Width
versicolor    121.5949     74.49895    119.7132     106.9020    266.4279
virginica     122.0041     74.52606    119.8172     107.0956    266.5057

Residual Deviance: 11.69607
AIC: 31.69607


> head(pred)
          setosa   versicolor      virginica
12 0.9999991 9.050984e-07 1.401492e-30
13 0.9999811 1.890666e-05 1.027185e-29
15 1.0000000 1.221982e-12 1.020006e-40
22 1.0000000 2.654515e-09 3.626318e-33
25 0.9999791 2.094711e-05 4.918632e-28
34 1.0000000 1.592562e-12 5.086200e-40
> pred <- predict(model,testing,type = "class")
> pred
 [1] setosa     setosa     setosa     setosa     setosa     setosa     setosa     setosa     versicolor
versicolor versicolor
[12] versicolor versicolor versicolor versicolor virginica virginica virginica virginica virginica
virginica  virginica
[23] virginica virginica virginica virginica virginica
Levels: setosa versicolor virginica
> table(pred)
pred
    setosa versicolor virginica
         8          7         12
> t <- table(pred,testing$Species)
> t

pred         setosa versicolor virginica
  setosa          8          0         0
  versicolor      0          7         0
  virginica       0          0        12
```

Result:

	Predicted			
Actual	setosa	versicolor	virginica	
setosa	8(TP)	0	0	8
versicolor	0	7(TP)	0	7
virginica	0	0	12(TP)	12
	8	7	12	27

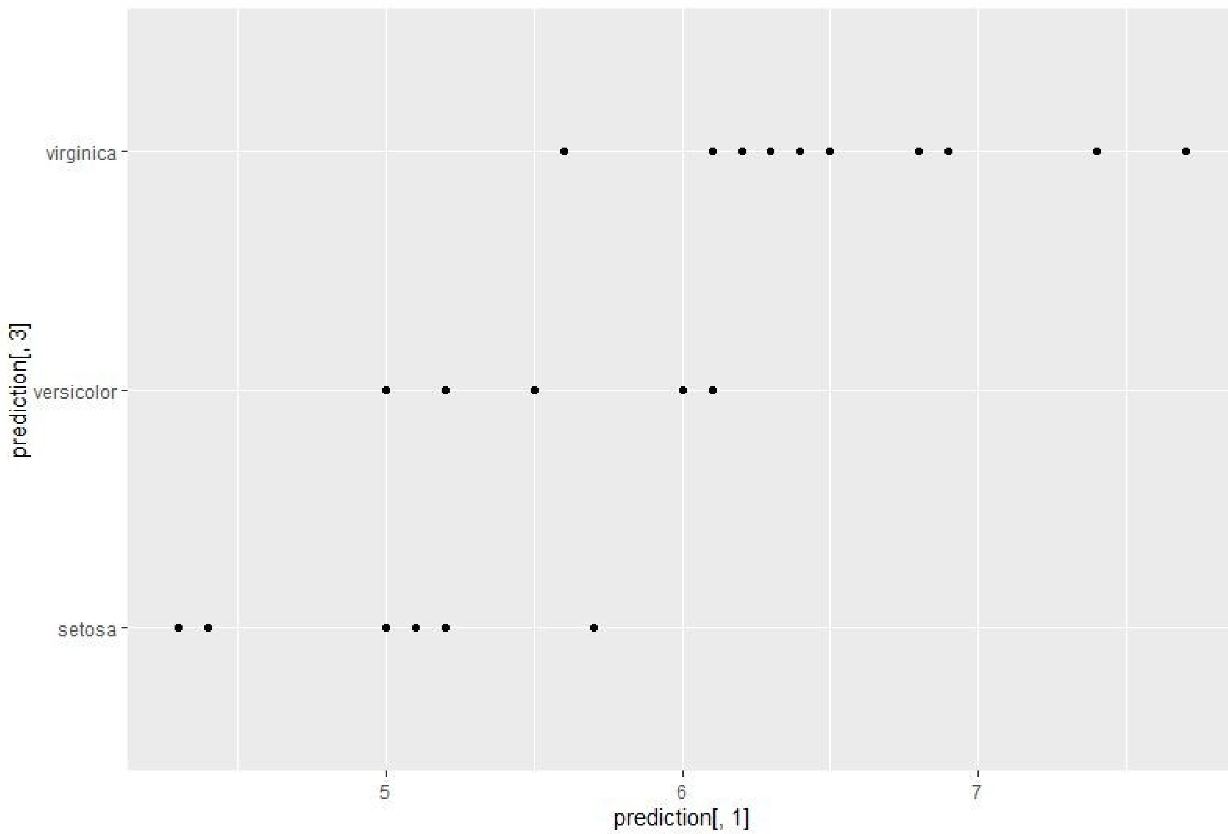
TN - true negative
TP - true positive
FP - false positive
FN - false negative

- **Accuracy:** Overall, how often is the classifier correct?
    - (TP+TN)/total = (27)/27 = 1=100%
- **Misclassification Rate:** Overall, how often is it wrong?
    - (FP+FN)/total = (0)/27 = 0=0%
    - 1 - Accuracy = 1 - 1= 0
    - equivalent to 1 minus Accuracy
    - also known as "**Error Rate**"
- **True Positive Rate:** When it's actually yes, how often does it predict yes?
    - TP/actual yes = 27/27 = 1=100%
    - also known as "**Sensitivity**" or "**Recall**"
- **False Positive Rate:** When it's actually no, how often does it predict yes?
    - FP/actual no = 0/27 = 0%
- **True Negative Rate:** When it's actually no, how often does it predict no?
    - TN/actual no = 27/27 = 1=100%
    - equivalent to 1 minus False Positive Rate = 1 - 0 = 1
    - also known as "**Specificity**"
- **Precision:** When it predicts yes, how often is it correct?
    - TP/predicted yes = 27/27 = 1=100%
- **Prevalence:** How often does the yes condition actually occur in our sample?

- actual yes/total = 27/27 = 1=100%

Overall Statistics:

```
> resultiris <- confusionMatrix(pred,testing$Species)
> resultiris
Confusion Matrix and Statistics

            Reference
Prediction setosa versicolor virginica
  setosa          8          0          0
  versicolor      0          7          0
  virginica       0          0         12

Overall Statistics

               Accuracy : 1
                 95% CI : (0.8723, 1)
    No Information Rate : 0.4444
    P-Value [Acc > NIR] : 3.098e-10

                  Kappa : 1

Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: setosa Class: versicolor Class: virginica
Sensitivity                 1.0000            1.0000           1.0000
Specificity                 1.0000            1.0000           1.0000
Pos Pred Value              1.0000            1.0000           1.0000
Neg Pred Value              1.0000            1.0000           1.0000
Prevalence                  0.2963            0.2593           0.4444
Detection Rate              0.2963            0.2593           0.4444
Detection Prevalence        0.2963            0.2593           0.4444
Balanced Accuracy           1.0000            1.0000           1.0000

> prediction <- data.frame(testing$Sepal.Length,testing$Species,pred)
> qplot(prediction[,1],prediction[,3])
```



Conclusion :

In this experiment we build a logistic regression model for mydata which is a csv dataset and iris dataset in R. We also learned on how to use multinom function in case of more than binary probabilities. We also learned to predict the result of test set from the model created using the training set and represent the output in form of table. We learned about the confusion matrix which is present in the caret library and learned to calculate the Accuracy,Precision, Prevalence and various rates to determine the efficiency and effectiveness of the logistic regression model.