Data Analytics

Assign-3

Name: Sai Deepak Reddy

Roll: 17BCS011

Logistic Regression Model

Using the Diabetic test data, predicting the person having diabetes or not.

Getting Started:

Loading the data

data<-read.csv(file="C:/Users/Name IT/Desktop/DALR/diabetes2.csv", head = TRUE, sep = ",")

head(data)

```
> data<-read.csv(file.choose())
> data<-read.csv(file="C:/Users/Name IT/Desktop/DALR/diabetes2.csv", head = TRUE, sep =
",")
> head(data)
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
1           6     148            72            35       0 33.6
2           1      85            66            29       0 26.6
3           8     183            64             0       0 23.3
4           1      89            66            23      94 28.1
5           0     137            40            35     168 43.1
6           5     116            74             0       0 25.6
  DiabetesPedigreeFunction Age Outcome
1                    0.627  50       1
2                    0.351  31       0
3                    0.672  32       1
4                    0.167  21       0
5                    2.288  33       1
6                    0.201  30       0
> 
```

library(caTools)

caTools for splitting the data

split<-sample.split(data,SplitRatio = 0.75)

Split

Splits the data in a ratio of 75% train data and 25% test data.

train<-subset(data,split=="TRUE")

test<-subset(data,split=="FALSE")

Test and train are stored in test and train respectively.

model<-glm(Outcome~.,train,family = "binomial")

Using a general linear model and assuming it be binomial that is either true or false saying it as logistic model.

summary(model)   //gives the summary of the dataset

```
> model<-glm(Outcome~.,train,family = "binomial")
> summary(model)

Call:
glm(formula = Outcome ~ ., family = "binomial", data = train)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.4104  -0.7856  -0.4448   0.7946   2.6216  

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)    
(Intercept)              -8.160397   0.867611  -9.406  < 2e-16 ***
Pregnancies               0.100178   0.038451   2.605  0.00918 ** 
Glucose                   0.030687   0.004445   6.903 5.09e-12 ***
BloodPressure            -0.010640   0.006016  -1.769  0.07696 .  
SkinThickness             0.003171   0.008358   0.379  0.70436    
Insulin                  -0.001829   0.001140  -1.604  0.10866    
BMI                       0.101006   0.017836   5.663 1.49e-08 ***
DiabetesPedigreeFunction  0.568491   0.349761   1.625  0.10408    
Age                       0.018686   0.011048   1.691  0.09078 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 674.30  on 511  degrees of freedom
Residual deviance: 514.06  on 503  degrees of freedom
AIC: 532.06

Number of Fisher Scoring iterations: 5
```

The Data has attributes pregnancies, Glucode, Bloodpressure, Skinthickness, Insulin, BMI, Diabetes Pedigrationfunction, Age which indicates the factors to test whether a person is having diabetes or not.

Note:

***** indicated 99.9% Confidence Interval, ** indicates 99% CI, * indicates 95% CI and '.' indicates 90% CI, if nothing was indicated then it is independent of the function for testing.**

Null deviation says how much actually it is deviated when only intercept was considered.

Residual Deviation says how much actually it is deviated when intercept and all parameters was considered.

AIC should be as minimum as possible, it is the overall deviation from true values.

$$y = b_0 + b_1 x + b_2 x + \ldots.$$

Be the linear equation the the logistic expression for above linear equation is

$$Y = 1/1 + e^{-y}$$

The expected values above are actually the values of b_0 **,** b_1 **,...**

The intercept value is b_0

//Predicting the testing data set using the above obtained values.

```
res<-predict(model,test,type="response")
res
```

```
> res
          3           5           6          12          14          15          21 
0.712838545 0.844561388 0.164302489 0.879560876 0.498988415 0.574763222 0.412149315 
         23          24          30          32          33          39          41 
0.933421067 0.325930647 0.331362376 0.511231037 0.058825060 0.242390155 0.732668333 
         42          48          50          51          57          59          60 
0.705986375 0.053757114 0.026206901 0.038568777 0.863179490 0.797840404 0.259667407 
         66          68          69          75          77          78          84 
0.157005222 0.551902693 0.037734953 0.077960938 0.108587360 0.292584638 0.067030400 
         86          87          93          95          96         102         104 
0.208410623 0.607973341 0.471487827 0.228811116 0.519659900 0.285821401 0.044762195 
        105         111         113         114         120         122         123 
0.266945547 0.599742198 0.082623461 0.123088263 0.062182845 0.340814936 0.182668233 
        129         131         132         138         140         141         147 
0.243430347 0.589830131 0.623075057 0.081550439 0.203407173 0.189922188 0.108150152 
        149         150         156         158         159         165         167 
0.682221320 0.064633642 0.910770658 0.114783516 0.068296882 0.296386112 0.474568100 
        168         174         176         177         183         185         186 
0.315919118 0.265037056 0.838580542 0.164177212 0.004210578 0.366735715 0.935545659 
        192         194         195         201         203         204         210 
0.510471934 0.969019269 0.125026365 0.186524779 0.154013770 0.047520866 0.875644341 
        212         213         219         221         222         228         230 
0.646151013 0.856945315 0.175704576 0.585940259 0.692881588 0.778199120 0.412100152 
        231         237         239         240         246         248         249 
0.711951092 0.861659081 0.755483907 0.043785800 0.897170056 0.681685941 0.387531689 
        255         257         258         264         266         267         273 
0.248243297 0.267322903 0.154839564 0.564042973 0.309541518 0.676658274 0.150462930 
        275         276         282         284         285         291         293 
0.554462914 0.346897635 0.591724567 0.647725859 0.160980526 0.089639969 0.639024245 
        294         300         302         303         309         311         312 
0.508148379 0.322341691 0.400920780 0.147148165 0.291872708 0.107069230 0.255207190 
        318         320         321         327         329         330         336 
0.680599245 0.752779123 0.237335830 0.335839377 0.320269225 0.225632502 0.774795951 
```

The above picture depicts the predicted values of the test data set, showing the probabilities of the person being diabetic or not. Say person 3 shown above is diabetic or not? The model says the person has diabetes with a probability 0.7128.

And person 5 has diabetes with prob of 0.8445 and person 6 has diabetes with prob of 0.164

Cool, the person 3 and 5are actually having diabetes, and person 6 doesn't.

```
    DiabetesPedigreeFunction Age Outcome
3                      0.672  32       1
5                      2.288  33       1
6                      0.201  30       0
12                     0.537  34       1
14                     0.398  59       1
15                     0.587  51       1
21                     0.704  27       0
23                     0.451  41       1
24                     0.263  29       1
```

But how accurate is the model?

If I assume the case of probability above 0.5 is having diabetes and not having otherwise.

Defining confusion matrix

confmatrix <- table(Actualvalue=test$Outcome,Predictedvalue= res > 0.5)

confmatrix

```
> confmatrix <- table(Actualvalue=test$Outcome,Predictedvalue= res > 0.5)
> confmatrix
           Predictedvalue
Actualvalue FALSE TRUE
          0   156   21
          1    26   53
```

The above picture says, the actual person who is having diabetes and the prediction was true are 53 persons and the actual person who is having diabetes and the prediction was false are 26 persons(This is a dangerous error).

The actual person who is not having diabetes and the prediction was false are 156 persons and the actual person who is not having diabetes and the prediction was true are 21 persons.

So accuracy of the model was (156+53)/(156+53+26+21) which is 79%

How to put the threshold value so that the error who's having diabetes and predicting it wrong would be reduced?

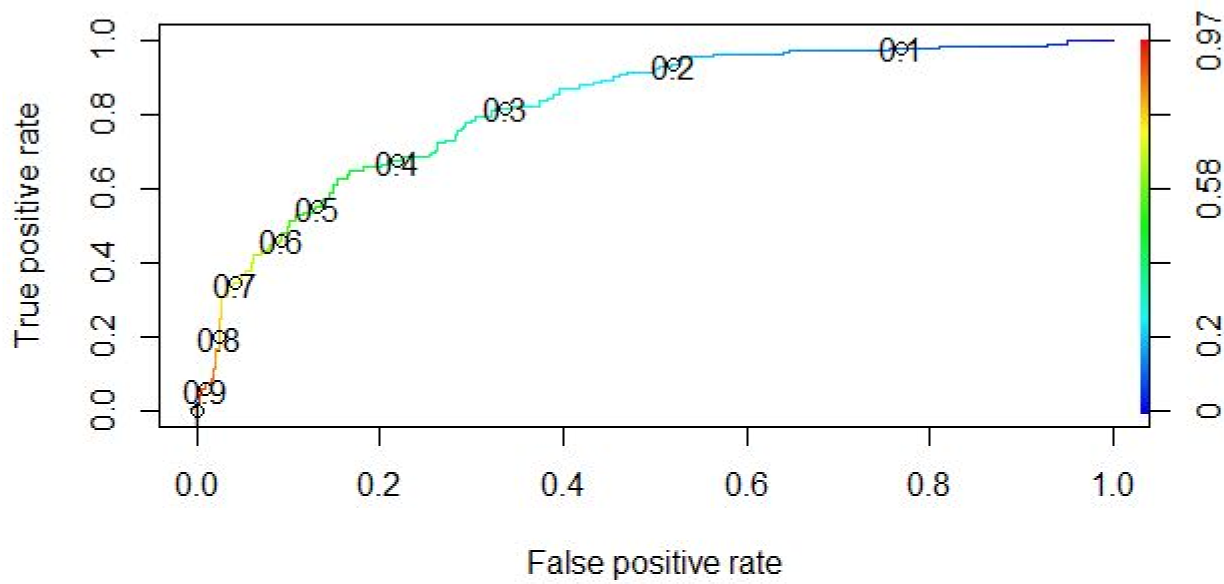
library(ROCR)

ROCRPred = prediction(res,train$Outcome)

ROCRPred<-performance(ROCRPred,"tpr","fpr")

plot(ROCRPred,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))

Says how much the true value is deviated from predicted value and vice versa.



The above picture shows

If the value used to classify the test is 0.5 the actual true value would be greater than 50% and false positive rate would be less than 20%

If the value used to classify the test is 0.4 the actual true value would be greater than 60% and false positive rate would be greater than 20%

If the value used to classify the test is 0.6 the actual true value would be less than 50% and false positive rate would be less than 20%

Most accurate threshold value is 0.5

Files can be downloaded here: https://github.com/saideepakreddi/Logistic-Regression-In-R

Email ID: saideepakreddyemani@gmail.com