LOGISTIC REGRESSION; DIABETES PREDICTION

LANGAT ERICK

2023-08-06

IMPORT DATA

```
rm( list = ls() )     # Clear Environment objects
```

```
#data
suppressPackageStartupMessages(require(bookdown))
suppressPackageStartupMessages(require(tidyverse))
data <- read.csv("C:/Users/langa/OneDrive/Desktop/MACHINE LEARNING/DATASETS/diabetes.csv")
```

```
library(rsample)
suppressPackageStartupMessages(require(caTools))
set.seed(123)
split <- initial_split(data, prop = .80, strata = Outcome)
training <- training(split)
testing=testing(split)
colnames(data)
```

```
## [1] "Pregnancies"              "Glucose"
## [3] "BloodPressure"            "SkinThickness"
## [5] "Insulin"                  "BMI"
## [7] "DiabetesPedigreeFunction" "Age"
## [9] "Outcome"
```

```
split
```

```
## <Training/Testing/Total>
## <614/154/768>
```

```
#
# split <- sample.split(data,SplitRatio = .8 )
# training <- subset(data, split==TRUE)
# testing <- subset(data, split==FALSE)
```

```
#model
model <- glm(Outcome~., data = training,
             family=binomial)
summary(model)
```

```
## 
## Call:
## glm(formula = Outcome ~ ., family = binomial, data = training)
## 
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)    
## (Intercept)              -7.8057701  0.7691652 -10.148  < 2e-16 ***
## Pregnancies               0.0989413  0.0358231   2.762  0.00575 ** 
## Glucose                   0.0340970  0.0040511   8.417  < 2e-16 ***
## BloodPressure            -0.0150652  0.0055636  -2.708  0.00677 ** 
## SkinThickness            -0.0003797  0.0076295  -0.050  0.96031    
## Insulin                  -0.0008258  0.0009630  -0.858  0.39114    
## BMI                       0.0807636  0.0165430   4.882 1.05e-06 ***
## DiabetesPedigreeFunction  0.9489161  0.3391775   2.798  0.00515 ** 
## Age                       0.0160062  0.0102861   1.556  0.11968    
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 793.94  on 613  degrees of freedom
## Residual deviance: 597.24  on 605  degrees of freedom
## AIC: 615.24
## 
## Number of Fisher Scoring iterations: 5
```

```r
#optimize our model by removing less significant variables 1by1
model <- glm(Outcome~.-SkinThickness,data=training,
             family=binomial)
summary(model)
```

```
## 
## Call:
## glm(formula = Outcome ~ . - SkinThickness, family = binomial, 
##     data = training)
## 
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)    
## (Intercept)              -7.8053085  0.7690410 -10.149  < 2e-16 ***
## Pregnancies               0.0989713  0.0358226   2.763  0.00573 ** 
## Glucose                   0.0341273  0.0040058   8.519  < 2e-16 ***
## BloodPressure            -0.0151176  0.0054626  -2.767  0.00565 ** 
## Insulin                  -0.0008456  0.0008774  -0.964  0.33516    
## BMI                       0.0805137  0.0157585   5.109 3.23e-07 ***
## DiabetesPedigreeFunction  0.9472246  0.3374204   2.807  0.00500 ** 
## Age                       0.0160667  0.0102158   1.573  0.11578    
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 793.94  on 613  degrees of freedom
## Residual deviance: 597.25  on 606  degrees of freedom
## AIC: 613.25
```

```
##
## Number of Fisher Scoring iterations: 5
```

```
#predict
res <- predict(model, testing, type = "response")
# res <- ifelse(res>0.5,1,0) %>% as.data.frame()
res
```

```
##           2           5          17          25          27          34
## 0.055668395 0.908126514 0.369322744 0.651554397 0.703040711 0.037963978
##          36          41          53          54          57          66
## 0.169924516 0.734916263 0.072160384 0.812472303 0.872021484 0.118237539
##          67          68          71          76          78          80
## 0.201223062 0.457016319 0.206118898 0.002580436 0.207974407 0.109718273
##          82          85          91          93          96          98
## 0.009642763 0.615585614 0.024410702 0.291949392 0.508805590 0.022503973
##         103         106         109         115         117         124
## 0.069554908 0.273978107 0.110434633 0.735513600 0.345026926 0.360661275
##         131         132         140         148         152         156
## 0.635926460 0.661257070 0.179619861 0.325202422 0.149996374 0.841282560
##         157         165         175         176         190         192
## 0.102207259 0.288471768 0.057871592 0.856819434 0.348963078 0.451977578
##         195         201         206         209         217         225
## 0.101924492 0.182854724 0.133867944 0.100924171 0.296085630 0.083937990
##         229         232         233         239         240         242
## 0.963922244 0.627979129 0.035563570 0.767373837 0.050284615 0.119671749
##         246         256         261         265         269         273
## 0.919200497 0.220893390 0.766628400 0.310838222 0.064268741 0.143201321
##         280         284         289         290         295         297
## 0.133657165 0.608479434 0.068705905 0.251483484 0.495374174 0.279311891
##         299         314         324         332         334         340
## 0.433986271 0.247876041 0.695217771 0.085694918 0.186527621 0.867271310
##         371         374         377         379         382         384
## 0.907242084 0.174282357 0.042865955 0.803609877 0.044674625 0.119369708
##         388         392         396         399         416         421
## 0.376063012 0.837682602 0.441497078 0.035598441 0.549128158 0.360307677
##         438         442         444         447         452         476
## 0.460276963 0.083321101 0.378587837 0.088578174 0.292606961 0.262121614
##         477         480         492         498         499         509
## 0.176218256 0.367896275 0.092623549 0.065833979 0.805408099 0.134833634
##         517         521         523         525         526         533
## 0.642318746 0.021687277 0.061560170 0.259911072 0.050536411 0.237877418
##         538         546         547         549         555         560
## 0.037424827 0.825804373 0.956426535 0.589389191 0.115327724 0.159154031
##         575         577         578         588         591         592
## 0.317051160 0.261909403 0.415352960 0.115490189 0.762939072 0.204394901
##         605         609         615         619         622         624
## 0.870082011 0.485824159 0.729082101 0.489056231 0.166602549 0.170176780
##         628         633         635         637         642         648
## 0.238419170 0.127594849 0.133355517 0.161787622 0.339589663 0.775744051
##         649         650         655         659         661         663
## 0.436473019 0.080197964 0.117152176 0.538546954 0.671148505 0.681877477
##         674         678         683         687         694         701
## 0.684608987 0.114497915 0.209414680 0.178198277 0.589694887 0.269690321
##         704         705         713         715         719         720 
## 0.698787758 0.113372517 0.730068140 0.105062058 0.191357344 0.251763723 
##         736         737         740         746         753         762 
## 0.181634136 0.132901209 0.229450138 0.279517744 0.113681455 0.915566942 
##         765         766         767         768 
## 0.313609478 0.176436019 0.309134828 0.077615292
```

```
test <- testing$Outcome %>% as.data.frame()
```

```
cbind(res, test)
```

```
##             res .
## 2   0.055668395 0
## 5   0.908126514 1
## 17  0.369322744 1
## 25  0.651554397 1
## 27  0.703040711 1
## 34  0.037963978 0
## 36  0.169924516 0
## 41  0.734916263 0
## 53  0.072160384 0
## 54  0.812472303 1
## 57  0.872021484 1
## 66  0.118237539 0
## 67  0.201223062 1
## 68  0.457016319 0
## 71  0.206118898 1
## 76  0.002580436 0
## 78  0.207974407 0
## 80  0.109718273 0
## 82  0.009642763 0
## 85  0.615585614 1
## 91  0.024410702 0
## 93  0.291949392 0
## 96  0.508805590 0
## 98  0.022503973 0
## 103 0.069554908 0
## 106 0.273978107 0
## 109 0.110434633 0
## 115 0.735513600 1
## 117 0.345026926 1
## 124 0.360661275 0
## 131 0.635926460 1
## 132 0.661257070 1
## 140 0.179619861 0
## 148 0.325202422 0
## 152 0.149996374 0
## 156 0.841282560 1
## 157 0.102207259 0
## 165 0.288471768 1
## 175 0.057871592 0
## 176 0.856819434 1
## 190 0.348963078 1
## 192 0.451977578 0
## 195 0.101924492 0
## 201 0.182854724 0
## 206 0.133867944 0
## 209 0.100924171 0
## 217 0.296085630 1
## 225 0.083937990 0
## 229 0.963922244 0
## 232 0.627979129 1
## 233 0.035563570 0
## 239 0.767373837 1
## 240 0.050284615 0
## 242 0.119671749 0
## 246 0.919200497 1
## 256 0.220893390 1
## 261 0.766628400 0
## 265 0.310838222 1
## 269 0.064268741 0
## 273 0.143201321 0
## 280 0.133657165 0
## 284 0.608479434 1
## 289 0.068705905 0
## 290 0.251483484 0
## 295 0.495374174 0
## 297 0.279311891 1
## 299 0.433986271 1
## 314 0.247876041 0
## 324 0.695217771 1
## 332 0.085694918 0
## 334 0.186527621 0
## 340 0.867271310 1
## 371 0.907242084 1
## 374 0.174282357 0
## 377 0.042865955 0
## 379 0.803609877 1
## 382 0.044674625 0
## 384 0.119369708 0
## 388 0.376063012 1
## 392 0.837682602 1
## 396 0.441497078 0
## 399 0.035598441 0
## 416 0.549128158 1
## 421 0.360307677 0
## 438 0.460276963 0
## 442 0.083321101 0
## 444 0.378587837 1
## 447 0.088578174 0
## 452 0.292606961 1
## 476 0.262121614 0
## 477 0.176218256 1
## 480 0.367896275 0
## 492 0.092623549 0
## 498 0.065833979 0
## 499 0.805408099 1
## 509 0.134833634 0
## 517 0.642318746 1
## 521 0.021687277 0
## 523 0.061560170 0
## 525 0.259911072 0
## 526 0.050536411 0
## 533 0.237877418 0
## 538 0.037424827 0
## 546 0.825804373 1
## 547 0.956426535 1
## 549 0.589389191 0
## 555 0.115327724 0
## 560 0.159154031 0
## 575 0.317051160 0
## 577 0.261909403 0
## 578 0.415352960 1
## 588 0.115490189 0
## 591 0.762939072 1
## 592 0.204394901 0
## 605 0.870082011 1
## 609 0.485824159 0
## 615 0.729082101 1
## 619 0.489056231 1
## 622 0.166602549 0
## 624 0.170176780 0
## 628 0.238419170 0
## 633 0.127594849 0
## 635 0.133355517 0
## 637 0.161787622 0
## 642 0.339589663 0
## 648 0.775744051 1
## 649 0.436473019 1
## 650 0.080197964 0
## 655 0.117152176 0
## 659 0.538546954 0
## 661 0.671148505 0
## 663 0.681877477 1
## 674 0.684608987 0
## 678 0.114497915 0
## 683 0.209414680 0
## 687 0.178198277 0
## 694 0.589694887 1
## 701 0.269690321 0
## 704 0.698787758 0
## 705 0.113372517 0
## 713 0.730068140 1
## 715 0.105062058 0
## 719 0.191357344 0
## 720 0.251763723 1
## 736 0.181634136 0
## 737 0.132901209 0
## 740 0.229450138 1
## 746 0.279517744 0
## 753 0.113681455 0
## 762 0.915566942 1
## 765 0.313609478 0
## 766 0.176436019 0
## 767 0.309134828 1
## 768 0.077615292 0
```

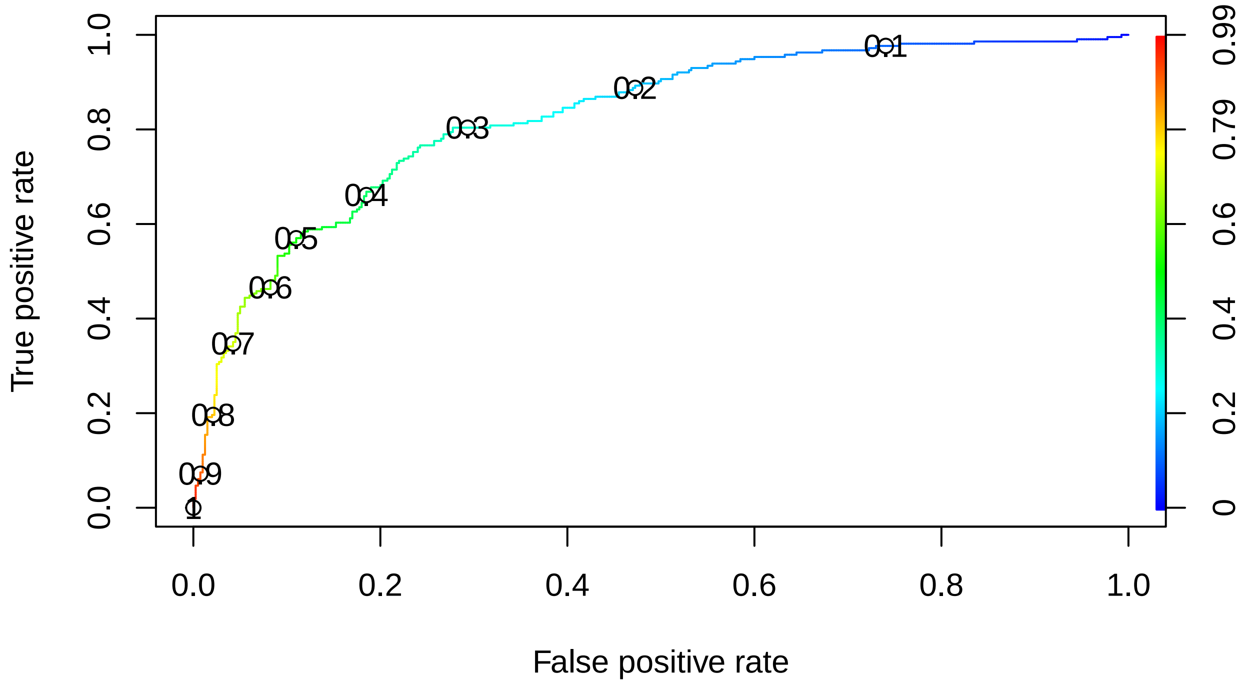
MODEL ACCURACY

```
#confusionmatrix
suppressPackageStartupMessages(require(caret))
(table <- table(testing$Outcome, res>0.5))
```

```
##
##     FALSE TRUE
##   0    91    9
##   1    21   33
```

HOW TO SET THE TRESHOLD

```
library(ROCR)
res1 <- predict(model, training, type = 'response')
ROCRpred <- prediction(res1,training$Outcome)
ROCperf <- performance(ROCRpred, 'tpr','fpr')
plot(ROCperf, colorize=TRUE, print.cutoffs.at=seq(0.1,by=0.1))
```



```r
res <- predict(model, testing, type = 'response')
(table(testing$Outcome, res>0.5))#
```

```
##
##     FALSE TRUE
##   0    91    9
##   1    21   33
```

```r
(table(testing$Outcome, res>0.3))#
```

```
##
##     FALSE TRUE
##   0    78   22
##   1    10   44
```

```r
#accuracy

sum(91,33)/sum(91,9,21,33)
```

```
## [1] 0.8051948
```

```r
sum(78,44)/sum(78,22,10,44)
```

```
## [1] 0.7922078
```