Chapter 18: Logistic Regression

Exercise 1: Diabetes

Cho dữ liệu diabetes.csv

Thông tin các cột dữ liệu:

- Pregnancies: số lần mang thai
- Glucose: Nồng độ glucose huyết tương 2 giờ trong thử nghiệm dung nạp glucose đường uống
- BloodPressure: Huyết áp tâm trương (mm Hg)
- SkinThickness: độ dày da gấp Triceps skin fold thickness (mm)
- Insulin: 2-Hour serum insulin (mu U/ml). insulin huyết thanh 2-giờ
- BMI: (weight in kg/(height in m)^2)
- DiabetesPedigreeFunction: Diabetes pedigree function
- Age: Age (years)
- Outcome: Class variable (0 or 1)

Chú ý: Tất cả các biến trên liên tục, mục đích là dự đoán ai đó có bị tiểu đường hay không (Outcome=1) dựa trên các biến khác. Các mẫu lấy từ phụ nữ trên 21 years old.

Yêu cầu: Áp dụng LogisticRegression để thực hiện việc dự đoán khả năng dương tính với bệnh tiểu đường (positive diabete - outputs) dựa trên các biến lâm sàng khác (clinical variables - inputs)

- Hãy áp dụng LogisticRegression để dự đoán khả năng dương tính với bệnh tiểu đường
- Đọc dữ liệu và gán cho biến data.
- Xem thông tin data: head(), số dòng, số cột, str, summary
- Vẽ biểu đồ quan sát mối liên hệ giữa các biến (corrplot)
- Tạo train:test từ dữ liệu data với tỉ lệ 70:30
- Áp dụng thuật toán LogisticRegression
- Tìm kết quả
- Trực quan hóa kết quả
- Hãy cho biết với những người có pregnant, glucose, pressure, triceps, insulin, mass, pedigree, age lần lượt như sau thì ai có khả năng dương tính với bệnh tiểu đường, ai không?
  - 8, 176, 90, 34, 300, 33.7, 0.467, 58
  - 1, 100, 66, 15, 56, 23.6, 0.666, 26
  - 12, 88, 74, 40, 54, 35.3, 0.378, 48

In [1]:
```
library(corrplot)
mydata <- read.csv("diabetes.csv")
```

In [2]:
```
## view the first few rows of the data
print(head(mydata))
```

```
  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
```

In [3]:
```
# print(tail(mydata))
```

In [4]:
```
print(summary(mydata))
```

```
  Pregnancies        Glucose      BloodPressure    SkinThickness  
 Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00  
 1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00  
 Median : 3.000   Median :117.0   Median : 72.00   Median :23.00  
 Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54  
 3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00  
 Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00  
    Insulin           BMI        DiabetesPedigreeFunction      Age        
 Min.   :  0.0   Min.   : 0.00   Min.   :0.0780           Min.   :21.00  
 1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00  
 Median : 30.5   Median :32.00   Median :0.3725           Median :29.00  
 Mean   : 79.8   Mean   :31.99   Mean   :0.4719           Mean   :33.24  
 3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00  
 Max.   :846.0   Max.   :67.10   Max.   :2.4200           Max.   :81.00  
    Outcome     
 Min.   :0.000  
 1st Qu.:0.000  
 Median :0.000  
 Mean   :0.349  
 3rd Qu.:1.000  
 Max.   :1.000  
```

In [5]:
```
print(str(mydata))
```

```
'data.frame':   768 obs. of  9 variables:
 $ Pregnancies              : int  6 1 8 1 0 5 3 10 2 8 ...
 $ Glucose                  : int  148 85 183 89 137 116 78 115 197 125 ...
 $ BloodPressure            : int  72 66 64 66 40 74 50 0 70 96 ...
 $ SkinThickness            : int  35 29 0 23 35 0 32 0 45 0 ...
 $ Insulin                  : int  0 0 0 94 168 0 88 0 543 0 ...
 $ BMI                      : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0
...
 $ DiabetesPedigreeFunction: num  0.627 0.351 0.672 0.167 2.288 ...
 $ Age                      : int  50 31 32 21 33 30 26 29 53 54 ...
 $ Outcome                  : int  1 0 1 0 1 0 1 0 1 1 ...
NULL
```

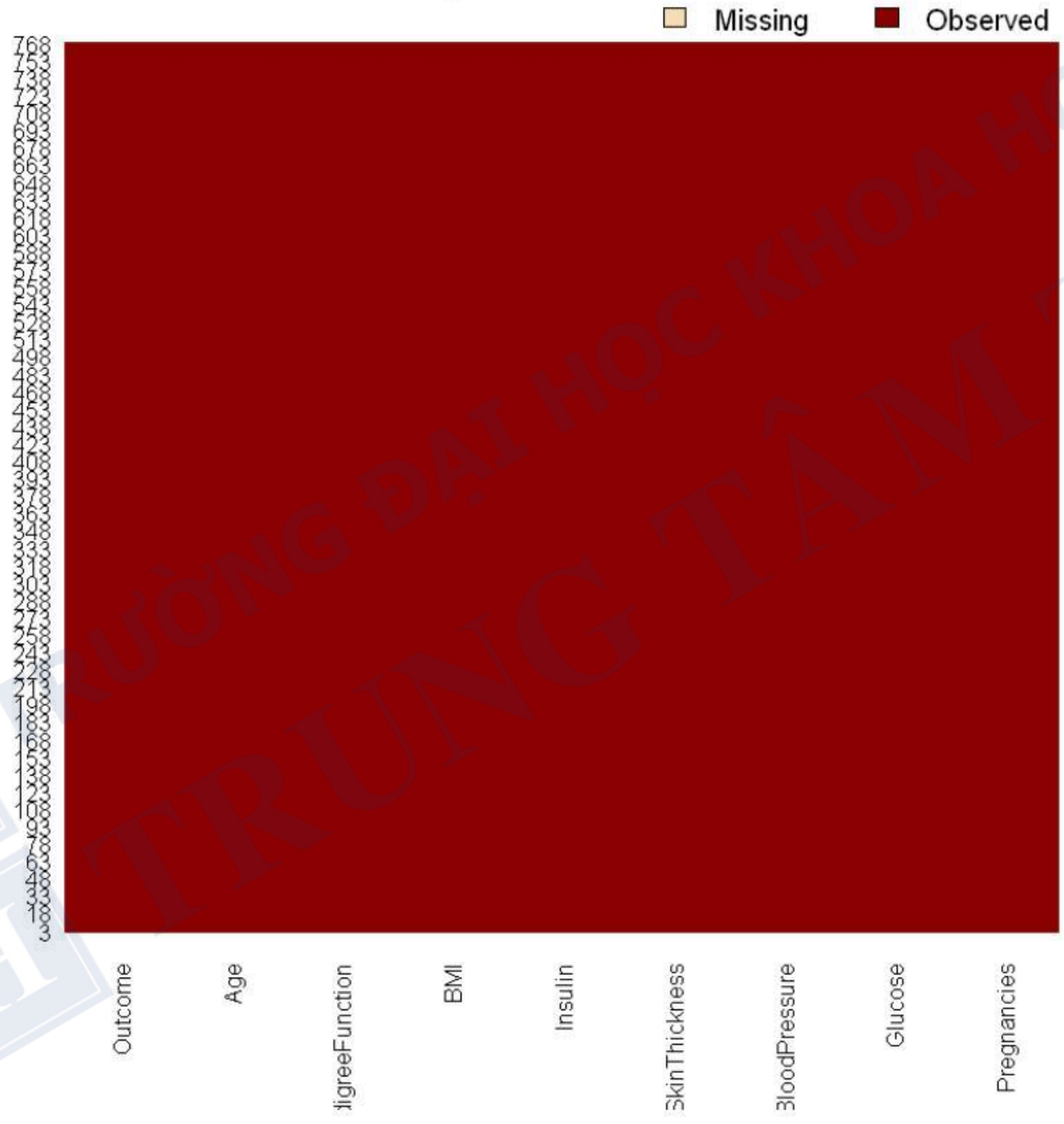
In [6]:
```
print(paste("cols:",ncol(mydata)))
print(paste("rows:",nrow(mydata)))
```

```
[1] "cols: 9"
[1] "rows: 768"
```

In [7]:
```
# check missing value
library(Amelia)
missmap(mydata, main = "Missing values vs observed")
# => no missing values
```

```
Loading required package: Rcpp

##
## Amelia II: Multiple Imputation
## (Version 1.7.4, built: 2015-12-05)
## Copyright (C) 2005-2020 James Honaker, Gary King and Matthew Blackwell
## Refer to http://gking.harvard.edu/amelia/ (http://gking.harvard.edu/amelia/)
for more information
##
```



In [8]:
```
# Check Class bias
print(table(mydata$Outcome))
```

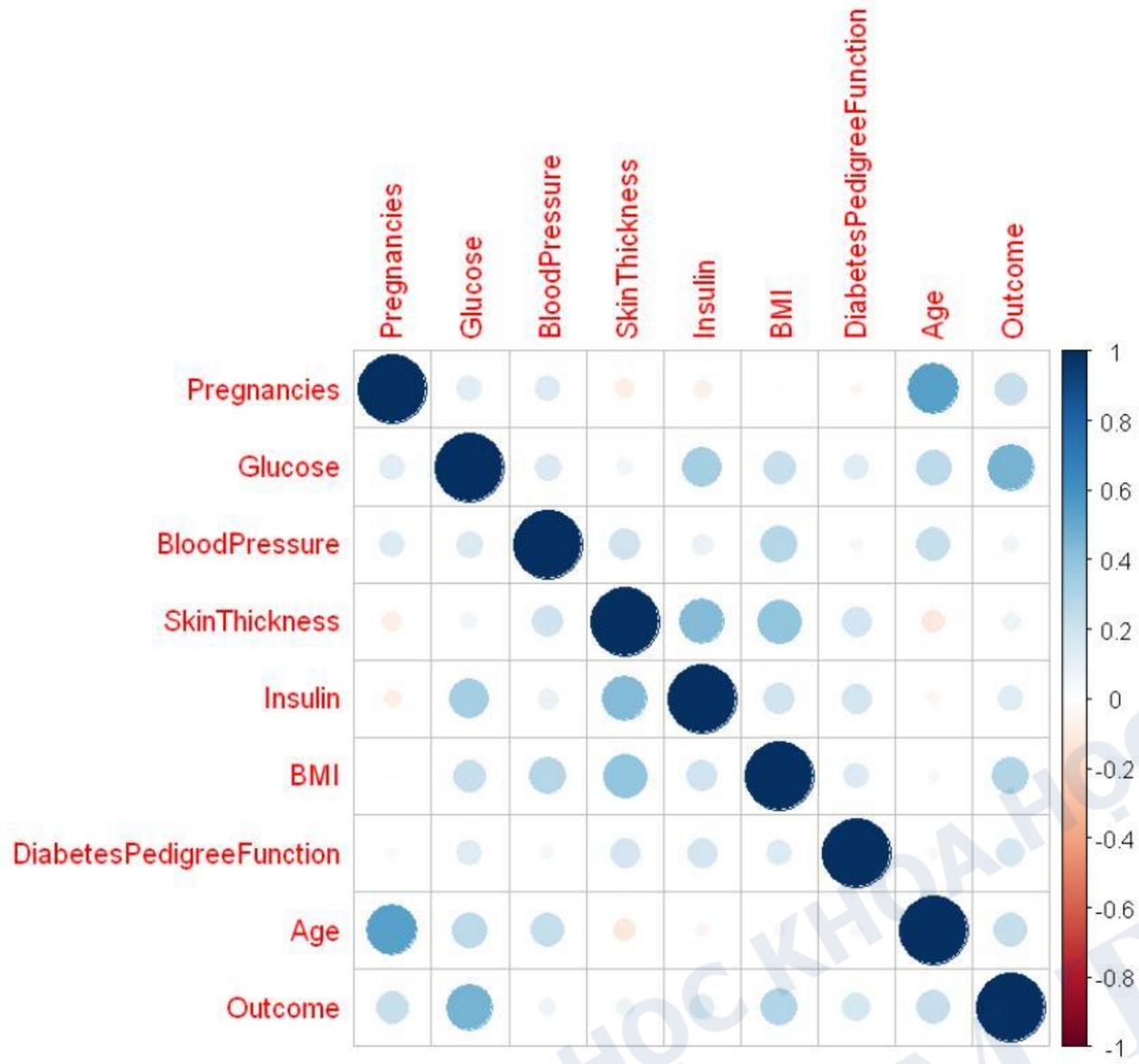
```
  0   1
500 268
```

In [9]:
```
# BoxPlot to Check for outliers
# Drop rows having outliers

# calculating the correlation between each pair of numeric variables
correlations <- cor(mydata[,1:9])
print(correlations)
```

```
                          Pregnancies    Glucose BloodPressure SkinThickness
Pregnancies                1.00000000 0.12945867    0.14128198   -0.08167177
Glucose                    0.12945867 1.00000000    0.15258959    0.05732789
BloodPressure              0.14128198 0.15258959    1.00000000    0.20737054
SkinThickness             -0.08167177 0.05732789    0.20737054    1.00000000
Insulin                   -0.07353461 0.33135711    0.08893338    0.43678257
BMI                        0.01768309 0.22107107    0.28180529    0.39257320
DiabetesPedigreeFunction -0.03352267 0.13733730    0.04126495    0.18392757
Age                        0.54434123 0.26351432    0.23952795   -0.11397026
Outcome                    0.22189815 0.46658140    0.06506836    0.07475223
                             Insulin        BMI DiabetesPedigreeFunction
Pregnancies               -0.07353461 0.01768309               -0.03352267
Glucose                    0.33135711 0.22107107                0.13733730
BloodPressure              0.08893338 0.28180529                0.04126495
SkinThickness              0.43678257 0.39257320                0.18392757
Insulin                    1.00000000 0.19785906                0.18507093
BMI                        0.19785906 1.00000000                0.14064695
DiabetesPedigreeFunction   0.18507093 0.14064695                1.00000000
Age                       -0.04216295 0.03624187                0.03356131
Outcome                    0.13054795 0.29269466                0.17384407
                                 Age    Outcome
Pregnancies                0.54434123 0.22189815
Glucose                    0.26351432 0.46658140
BloodPressure              0.23952795 0.06506836
SkinThickness             -0.11397026 0.07475223
Insulin                   -0.04216295 0.13054795
BMI                        0.03624187 0.29269466
DiabetesPedigreeFunction   0.03356131 0.17384407
Age                        1.00000000 0.23835598
Outcome                    0.23835598 1.00000000
```

In [10]:
```
corrplot(correlations, method="circle")
```



In [11]:
```
# divided into train and test: 70 - 30
n = nrow(mydata)
trainIndex = sample(1:n, size = round(0.7*n), replace=FALSE)
train = mydata[trainIndex ,]
test = mydata[-trainIndex ,]
print("Rows of training data and test data:")
print(nrow(train))
print(nrow(test))
```

```
[1] "Rows of training data and test data:"
[1] 538
[1] 230
```

In [12]:
```
# estimates a logistic regression model using the glm (generalized linear model)
mylogit <- glm(Outcome ~ ., data = train, family = "binomial")
print(summary(mylogit))
```

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

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5598  -0.7087  -0.3912   0.6966   3.0360

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)               -8.617615   0.868548  -9.922  < 2e-16 ***
Pregnancies                0.138184   0.040714   3.394 0.000689 ***
Glucose                    0.038237   0.004636   8.247  < 2e-16 ***
BloodPressure             -0.017082   0.006894  -2.478 0.013212 *
SkinThickness              0.008086   0.008332   0.971 0.331773
Insulin                   -0.001323   0.001086  -1.219 0.222855
BMI                        0.089461   0.018447   4.850 1.24e-06 ***
DiabetesPedigreeFunction   0.837717   0.371315   2.256 0.024065 *
Age                        0.012923   0.011409   1.133 0.257341
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 692.48  on 537  degrees of freedom
Residual deviance: 490.74  on 529  degrees of freedom
AIC: 508.74

Number of Fisher Scoring iterations: 5
```

In [13]:
```
pred = predict(mylogit,
               newdata = test,
               type = "response")

pred_value <- ifelse(pred > 0.5, 1, 0)
print("Testdata admit vs predict (10 rows:)")
result <- data.frame(testAdmit = test$Outcome[30:40], pred_value[30:40])
print(result)
```

```
[1] "Testdata admit vs predict (10 rows:)"
    testAdmit pred_value.30.40.
120         0                  0
121         1                  1
123         0                  0
127         0                  0
129         1                  0
132         1                  1
134         0                  0
135         0                  0
136         0                  0
140         0                  0
143         0                  0
```

In [14]:
```
print(pred_value[30:40])
print(test$Outcome[30:40])
```

```
120 121 123 127 129 132 134 135 136 140 143
  0   1   0   0   0   1   0   0   0   0   0
 [1] 0 1 0 0 1 1 0 0 0 0 0
```

In [15]:
```
# SOLUTION 1
accuracy <- table(pred_value, test[,"Outcome"])
accuracy = sum(diag(accuracy))/sum(accuracy)
print(paste("Accuracy s1:", accuracy))
```

```
[1] "Accuracy s1: 0.752173913043478"
```

In [16]:
```
# SOLUTION 2
misClasificError <- mean(pred_value != test$Outcome)
print(paste('Accuracy s2: ',1-misClasificError))
```

```
[1] "Accuracy s2:  0.752173913043478"
```

In [17]:
```
summary(mylogit)
```

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

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5598  -0.7087  -0.3912   0.6966   3.0360

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)             -8.617615   0.868548  -9.922  < 2e-16 ***
Pregnancies              0.138184   0.040714   3.394 0.000689 ***
Glucose                  0.038237   0.004636   8.247  < 2e-16 ***
BloodPressure           -0.017082   0.006894  -2.478 0.013212 *
SkinThickness            0.008086   0.008332   0.971 0.331773
Insulin                 -0.001323   0.001086  -1.219 0.222855
BMI                      0.089461   0.018447   4.850 1.24e-06 ***
DiabetesPedigreeFunction 0.837717   0.371315   2.256 0.024065 *
Age                      0.012923   0.011409   1.133 0.257341
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 692.48  on 537  degrees of freedom
Residual deviance: 490.74  on 529  degrees of freedom
AIC: 508.74

Number of Fisher Scoring iterations: 5
```

In [18]:
```
# make new prediction
#.  8, 176, 90, 34, 300, 33.7, 0.467, 58
#.  1, 100, 66, 15, 56, 23.6, 0.666, 26
#.  12, 88, 74, 40, 54, 35.3, 0.378, 48

print(colnames(test))

y1 <- predict(mylogit,
              newdata = data.frame(Pregnancies = c(8, 1, 12),
                                   Glucose = c(176, 100, 88),
                                   BloodPressure = c(90, 66, 74),
                                   SkinThickness = c(34, 15, 40),
                                   Insulin = c(300, 56, 54),
                                   BMI = c(33.7, 23.6, 35.3),
                                   DiabetesPedigreeFunction = c(0.467, 0.666, 0.
                                   Age = c(58, 26, 48)),
              type='response')
y1 <- ifelse(y1 > 0.5, 1, 0)
print("results:")
print(y1)
```

```
[1] "Pregnancies"              "Glucose"
[3] "BloodPressure"            "SkinThickness"
[5] "Insulin"                  "BMI"
[7] "DiabetesPedigreeFunction" "Age"
[9] "Outcome"
[1] "results:"
1 2 3
1 0 0
```

In [ ]: