Predict the onset of diabetes based on diagnostic measures

R Markdown

```
#importing required packages for the analysis
pacman::p_load(caret, data.table, gains, leaps, MASS, tidyverse)
theme_set(theme_classic())
options(digits = 3)
```

```
#importing data
data.df <- fread("diabetes.csv")

#Checking the stucture of the data
str(data.df)
```

```
## Classes 'data.table' and '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 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
#spliting data into training and test data set
split <- round(nrow(data.df) * 0.7)
train.df <- data.df[1:split, ]
test.df <- data.df[(split+1):nrow(data.df), ]

print("Train Data")
```

```
## [1] "Train Data"
```

```
str(train.df)
```

```
## Classes 'data.table' and 'data.frame':   538 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 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
print("Test Data")
```

```
## [1] "Test Data"
```

```
str(test.df)
```

```
## Classes 'data.table' and 'data.frame':   230 obs. of  9 variables:
##  $ Pregnancies             : int  0 3 8 3 10 4 1 8 5 4 ...
##  $ Glucose                 : int  127 129 100 128 90 84 88 186 187 131 ...
##  $ BloodPressure           : int  80 92 74 72 85 90 78 90 76 68 ...
##  $ SkinThickness           : int  37 49 40 25 32 23 29 35 27 21 ...
##  $ Insulin                 : int  210 155 215 190 0 56 76 225 207 166 ...
##  $ BMI                     : num  36.3 36.4 39.4 32.4 34.9 39.5 32 34.5 43.6 33.1 ...
##  $ DiabetesPedigreeFunction: num  0.804 0.968 0.661 0.549 0.825 ...
##  $ Age                     : int  23 32 43 27 56 25 29 37 53 28 ...
##  $ Outcome                 : int  0 1 1 1 1 0 0 1 1 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

Logistic Regression

```
set.seed(42)

#I have used the logistic regression as it is classification

logit.reg <- glm(Outcome~ ., data = train.df, family = "binomial")

options(scipen=999)
summary(logit.reg)
```

```
##
## Call:
## glm(formula = Outcome ~ ., family = "binomial", data = train.df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.424  -0.777  -0.424   0.802   2.754
##
## Coefficients:
##                           Estimate Std. Error z value             Pr(>|z|)
## (Intercept)               -7.91882    0.84097   -9.42 < 0.0000000000000002 ***
## Pregnancies                0.12618    0.03751    3.36              0.00077 ***
## Glucose                    0.03168    0.00434    7.31     0.00000000000028 ***
## BloodPressure             -0.01073    0.00599   -1.79              0.07317 .
## SkinThickness              0.00129    0.00830    0.16              0.87679
## Insulin                   -0.00130    0.00107   -1.21              0.22490
## BMI                        0.09292    0.01744    5.33     0.00000009961519 ***
```

```
## DiabetesPedigreeFunction  0.91793    0.34585    2.65             0.00795 **
## Age                       0.00563    0.01096    0.51             0.60769
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 697.52  on 537  degrees of freedom
## Residual deviance: 521.10  on 529  degrees of freedom
## AIC: 539.1
##
## Number of Fisher Scoring iterations: 5
```

```
# Generate odds-ratios
print("odds-ratios")
```

```
## [1] "odds-ratios"
```

```
exp(coef(logit.reg))
```

```
##              (Intercept)              Pregnancies                  Glucose
##                 0.000364                 1.134485                 1.032182
##            BloodPressure            SkinThickness                  Insulin
##                 0.989332                 1.001287                 0.998704
##                      BMI DiabetesPedigreeFunction                      Age
##                 1.097372                 2.504095                 1.005644
```

Above model state that pregnancies, Glucose, BMI, and DiabetesPedigreeFunction are most important variables in predict whether or not the patients in the dataset have diabetes or not at p value of 0.01.

Model Selection

```
logitnew <- stepAIC(logit.reg, trace = 0)  # trace = 0 suppress intermediate steps
```

Performance Evaluation

```
logit.reg.pred <- predict(logit.reg, test.df[, -9], type = "response")
# response will create probability
t(t(head(logit.reg.pred, 10)))
```

```
##      [,1]
## 1  0.3234
## 2  0.4681
## 3  0.4371
## 4  0.3088
## 5  0.4106
## 6  0.1413
```

```
## 7  0.0808
## 8  0.8281
## 9  0.9471
## 10 0.3096
```

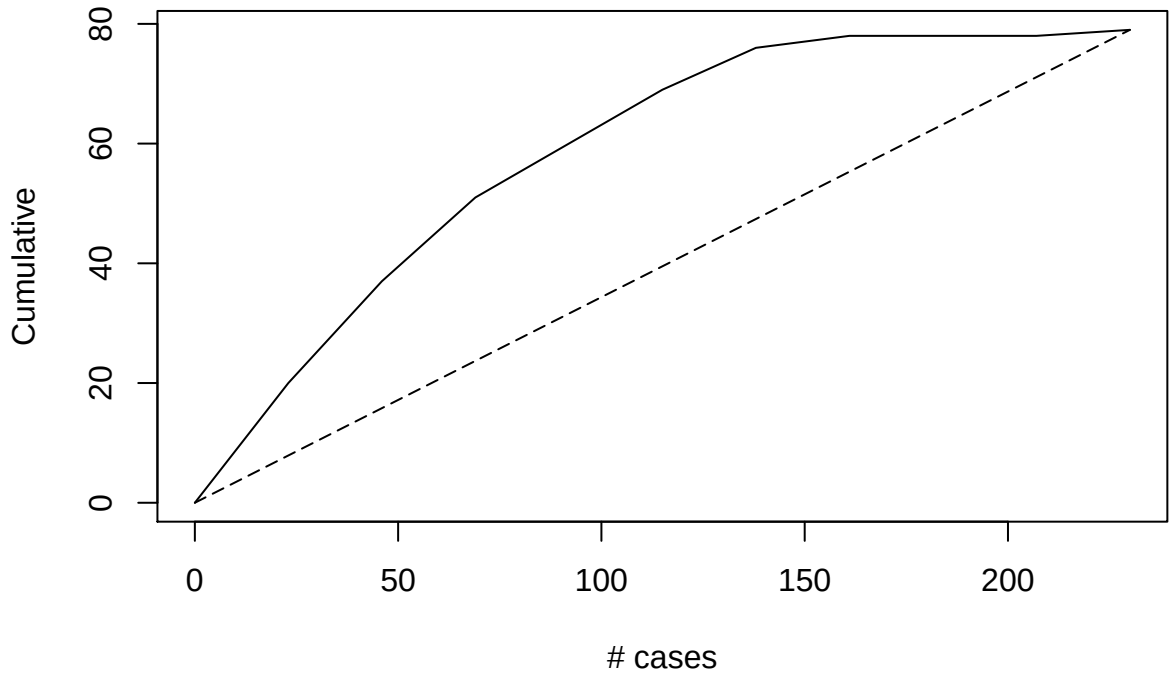
```
# generate confusion matrix
table(test.df$Outcome, logit.reg.pred > 0.5)
```

```
##
##     FALSE TRUE
##   0   139   12
##   1    36   43
```

#The prediction model gives an accuracy of 79.19%

```
gain <- gains(test.df$Outcome, logit.reg.pred, groups = 10)

  ### Plot Lift Chart
plot(c(0,gain$cume.pct.of.total*sum(test.df$Outcome))~c(0,gain$cume.obs),
     xlab = "# cases", ylab = "Cumulative", main = "", type = "l")
lines(c(0,sum(test.df$Outcome))~c(0, dim(test.df)[1]), lty = 5)
```




```
 ### Plot decile-wise chart
heights <- gain$mean.resp/mean(test.df$Outcome)
```

```r
midpoints <- barplot(heights, names.arg = gain$depth,  ylim = c(0,9), col = "gold3",
                     xlab = "Percentile", ylab = "Mean Response",
                     main = "Decile-wise lift chart")
```