PREDICTION OF HEART ATTACK ANALYSIS

Emre AK

8/4/23

```
install.packages("tidyverse")
install.packages("caret")
install.packages("dplyr")
install.packages("DALEX")
library(DALEX)
library(dplyr)
library(tidyverse)
library(caret)
```

```
library(readr)
heart <- read_csv("heart.csv")
```

```
Rows: 303 Columns: 14
-- Column specification --------------------------------------------------------
Delimiter: ","
dbl (14): age, sex, cp, trtbps, chol, fbs, restecg, thalachh, exng, oldpeak,...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

In this project we are going to use Heart attack prediction dataset. With 14 independent value such as (age, sex, cp, trtbps, etc.)and the all values are numeric. We try to predict Heart Attack.As you can see below there is libraries which are used for our project and codes. Our dataset has 303 row and 14 column.

```r
set.seed(123)
index <- sample(1 : nrow(heart), round(nrow(heart) * 0.80))
train <- heart[index, ]
test  <- heart[-index, ]
```

```r
model <- glm(output ~ ., data = train, family = "binomial")
model
```

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

Coefficients:
(Intercept)          age          sex           cp       trtbps         chol
   4.555347    -0.013283    -1.540661     0.769378    -0.017916    -0.003182
        fbs      restecg     thalachh         exng      oldpeak          slp
   0.132748     0.658516     0.019761    -1.077694    -0.678520     0.263729
        caa        thall
  -0.646435    -1.015510

Degrees of Freedom: 241 Total (i.e. Null);  228 Residual
Null Deviance:      333.5
Residual Deviance: 173.1    AIC: 201.1
```

Null Deviance and Residual Deviance: These values measure the classification performance of the model. The lower these values are, the better the model's performance.Null deviance represents the deviation obtained by the model without using any independent variables.

```r
summary(model)
```

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

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6127  -0.4311   0.1795   0.5835   2.4205

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
```

```
(Intercept)  4.555347   2.794683   1.630  0.10310
age         -0.013283   0.025289  -0.525  0.59941
sex         -1.540661   0.509861  -3.022  0.00251 **
cp           0.769378   0.195309   3.939 8.17e-05 ***
trtbps      -0.017916   0.011003  -1.628  0.10348
chol        -0.003182   0.004177  -0.762  0.44617
fbs          0.132748   0.566248   0.234  0.81465
restecg      0.658516   0.389009   1.693  0.09049 .
thalachh     0.019761   0.011439   1.727  0.08408 .
exng        -1.077694   0.448609  -2.402  0.01629 *
oldpeak     -0.678520   0.240961  -2.816  0.00486 **
slp          0.263729   0.411859   0.640  0.52195
caa         -0.646435   0.201346  -3.211  0.00132 **
thall       -1.015510   0.330194  -3.075  0.00210 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 333.48  on 241  degrees of freedom
Residual deviance: 173.07  on 228  degrees of freedom
AIC: 201.07

Number of Fisher Scoring iterations: 6
```

The results include the coefficient estimates, standard error, z-value, and p-value for each variable. The p-value is important in assessing the significance of each variable.If they have smal p-value, they are more significant and important in the model's performance. The model's AIC value (201.07) is used to evaluate the model fit. The lower AIC indicates better model fit.

```r
predicted_probs <- predict(model, test[,-14], type = "response")
head(predicted_probs)
```

```
        1         2         3         4         5         6
0.7079596 0.9397678 0.9823383 0.9748455 0.6193038 0.9260086
```

These percentages are giving knowladge about the effect of variables for target variables.

```r
predicted_classes <- ifelse(predicted_probs > 0.5, 1, 0)
head(predicted_classes)
```

```
1 2 3 4 5 6
1 1 1 1 1 1
```

These values are splitting for the over and under the 0.5 value and get the 0 or 1 values.

```
TP <- sum(predicted_classes[which(test$output == 1)] == 1)
FP <- sum(predicted_classes[which(test$output == 1)] == 0)
TN <- sum(predicted_classes[which(test$output == 0)] == 0)
FN <- sum(predicted_classes[which(test$output == 0)] == 1)
recall      <- TP / (TP + FN)
specificity <- TN / (TN + FP)
precision   <- TP / (TP + FP)
accuracy    <- (TN + TP) / (TP + FP + TN + FN)
recall
```

```
[1] 0.7948718
```

```
specificity
```

```
[1] 0.9090909
```

```
precision
```

```
[1] 0.9393939
```

```
accuracy
```

```
[1] 0.8360656
```

Recall, represents the proportion of true positive examples that were correctly identified among all positive examples. It shows how many of the true positives were recognized correctly. Here, recall is calculated as 0.7948. Specificity, represents the proportion of true negative examples that were correctly identified among all negative examples. It shows how many of the true negatives were recognized correctly. Here, specificity is calculated as 0.9090. Precision, represents the probability that the examples identified as positive are actually positive. It shows how many of the positively identified examples are actually positive. Here, precision is calculated as 0.9393. Accuracy, represents the proportion of examples that were correctly identified among all examples. It shows how many of all examples were identified correctly. Here, accuracy is calculated as 0.8360.

```r
table(train$output) / dim(train)[1]
```

```
        0         1 
0.4545455 0.5454545 
```

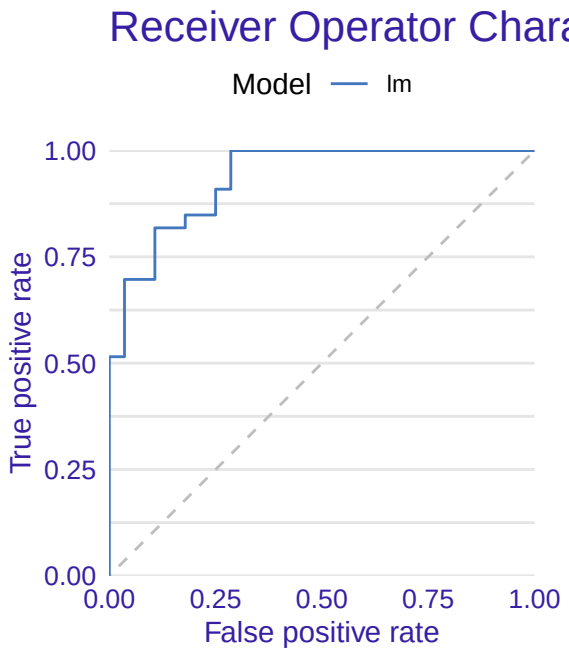
These percentages are represent that 0 and 1. It shows that these values have balanced.

```r
confusionMatrix(table(ifelse(test$output == "1", "1", "0"),
                      predicted_classes),
                positive = "1")
```

```
Confusion Matrix and Statistics

   predicted_classes
     0  1
  0 20  8
  1  2 31

               Accuracy : 0.8361
                 95% CI : (0.7191, 0.9185)
    No Information Rate : 0.6393
    P-Value [Acc > NIR] : 0.000614

                  Kappa : 0.6645

 Mcnemar's Test P-Value : 0.113846

            Sensitivity : 0.7949
            Specificity : 0.9091
         Pos Pred Value : 0.9394
         Neg Pred Value : 0.7143
             Prevalence : 0.6393
         Detection Rate : 0.5082
   Detection Prevalence : 0.5410
      Balanced Accuracy : 0.8520

       'Positive' Class : 1
```

```
explain_lr <- DALEX::explain(model   = model,
                        data    = test[, -14],
                        y       = test$output == "1",
                        type    = "classification",
                        verbose = FALSE)
```

```
performance_lr <- model_performance(explain_lr)
plot(performance_lr, geom = "roc")
```




```
performance_lr
```

```
Measures for:  classification
recall     : 0.9393939
precision  : 0.7948718
f1         : 0.8611111
accuracy   : 0.8360656
auc        : 0.9339827

Residuals:
         0%         10%         20%         30%         40%         50%
-0.92500426 -0.66440247 -0.23803316 -0.05420990 -0.00710291  0.01057154
```

```
       60%          70%          80%          90%         100%
0.02908870   0.05916045   0.19665257   0.30022165   0.59491469
```

The residuals indicate the deviation of the predicted values from the actual values. The values range from -0.92500426 to 0.59491469, with the majority of the residuals falling between -0.23803316 and 0.30022165. The AUC(Area of under the curve) is 0.93 that value is perfect. Because the max value can be 1.