Predict Heart Disease in Patients Code

Basil Ghauri

6/16/2020

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.0     v purrr   0.3.4
## v tibble  3.0.0     v stringr 1.4.0
## v tidyr   1.0.2     v forcats 0.5.0
## v readr   1.3.1

## -- Conflicts ------------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(ggplot2)
library(caTools)
library(class)
library(gmodels)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(fastDummies)
```

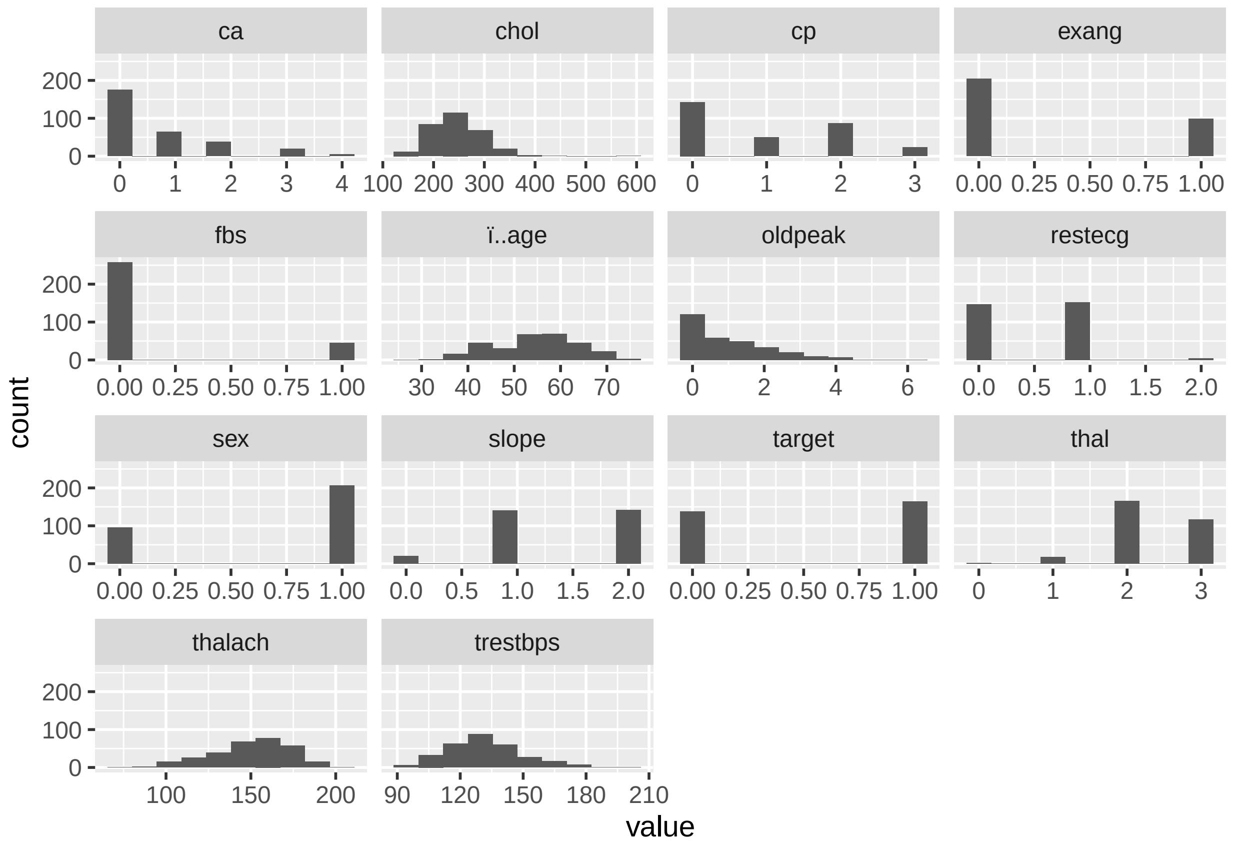
```r
# load data into R
data=read.csv("heart.csv")
head(data)
```

```
##   ï..age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal
## 1     63   1  3      145  233   1       0     150     0     2.3     0  0    1
## 2     37   1  2      130  250   0       1     187     0     3.5     0  0    2
## 3     41   0  1      130  204   0       0     172     0     1.4     2  0    2
## 4     56   1  1      120  236   0       1     178     0     0.8     2  0    2
## 5     57   0  0      120  354   0       1     163     1     0.6     2  0    2
## 6     57   1  0      140  192   0       1     148     0     0.4     1  0    1
##   target
## 1      1
## 2      1
## 3      1
## 4      1
## 5      1
## 6      1
```

```r
# number of unique values of each variable.
rapply(data,function(x)length(unique(x)))
```

```
##   ï..age      sex       cp trestbps     chol      fbs  restecg  thalach
##       41        2        4       49      152        2        3       91
##    exang  oldpeak    slope       ca     thal   target
##        2       40        3        5        4        2
```

```r
ggplot(gather(data),aes(value))+
  geom_histogram(bins=10)+
  facet_wrap(~key,scales = "free_x")
```



```r
# making dummy variables of variables with factors.
data1=dummy_cols(data,select_columns = c("ca","cp","exang","fbs","restecg","sex","slope","thal"))

#removing original columns
data1=data1%>%
  select(-ca,-cp,-exang,-fbs,-restecg,-sex,-slope,-thal)

# data after dummy variables.
head(data1)
```

```
##   ï..age trestbps chol thalach oldpeak target ca_0 ca_1 ca_2 ca_3 ca_4 cp_0
## 1     63      145  233     150     2.3      1    1    0    0    0    0    0
## 2     37      130  250     187     3.5      1    1    0    0    0    0    0
## 3     41      130  204     172     1.4      1    1    0    0    0    0    0
## 4     56      120  236     178     0.8      1    1    0    0    0    0    0
## 5     57      120  354     163     0.6      1    1    0    0    0    0    1
## 6     57      140  192     148     0.4      1    1    0    0    0    0    1
##   cp_1 cp_2 cp_3 exang_0 exang_1 fbs_0 fbs_1 restecg_0 restecg_1 restecg_2
## 1    0    0    1       1       0     0     1         1         0         0
## 2    0    1    0       1       0     1     0         0         1         0
## 3    1    0    0       1       0     1     0         1         0         0
## 4    1    0    0       1       0     1     0         0         1         0
## 5    0    0    0       0       1     1     0         0         1         0
## 6    0    0    0       1       0     1     0         0         1         0
##   sex_0 sex_1 slope_0 slope_1 slope_2 thal_0 thal_1 thal_2 thal_3
## 1     0     1       1       0       0      0      1      0      0
## 2     0     1       1       0       0      0      0      1      0
## 3     1     0       0       0       1      0      0      1      0
## 4     0     1       0       0       1      0      0      1      0
## 5     1     0       0       0       1      0      0      1      0
## 6     0     1       0       1       0      0      1      0      0
```

```
# normalize the remaining variables
data1_norm=data1%>%
  mutate_at(1:5,funs((.-min(.))/max(.-min(.))))
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.
```

```
head(data1_norm)
```

```
##      ï..age  trestbps      chol   thalach    oldpeak target ca_0 ca_1 ca_2 ca_3
## 1 0.7083333 0.4811321 0.2442922 0.6030534 0.37096774      1    1    0    0    0
## 2 0.1666667 0.3396226 0.2831050 0.8854962 0.56451613      1    1    0    0    0
## 3 0.2500000 0.3396226 0.1780822 0.7709924 0.22580645      1    1    0    0    0
## 4 0.5625000 0.2452830 0.2511416 0.8167939 0.12903226      1    1    0    0    0
## 5 0.5833333 0.2452830 0.5205479 0.7022901 0.09677419      1    1    0    0    0
## 6 0.5833333 0.4339623 0.1506849 0.5877863 0.06451613      1    1    0    0    0
##   ca_4 cp_0 cp_1 cp_2 cp_3 exang_0 exang_1 fbs_0 fbs_1 restecg_0 restecg_1
## 1    0    0    0    0    1       1       0     0     1         1         0
## 2    0    0    0    1    0       1       0     1     0         0         1
## 3    0    0    1    0    0       1       0     1     0         1         0
## 4    0    0    1    0    0       1       0     1     0         0         1
## 5    0    1    0    0    0       0       1     1     0         0         1
## 6    0    1    0    0    0       1       0     1     0         0         1
##   restecg_2 sex_0 sex_1 slope_0 slope_1 slope_2 thal_0 thal_1 thal_2 thal_3
## 1         0     0     1       1       0       0      0      1      0      0
## 2         0     0     1       1       0       0      0      0      1      0
## 3         0     1     0       0       0       1      0      0      1      0
## 4         0     0     1       0       0       1      0      0      1      0
## 5         0     1     0       0       0       1      0      0      1      0
## 6         0     0     1       0       1       0      0      1      0      0
```

KNN Model

```r
#checking how many people have heart disease and how many dont
table(data1_norm$target)
```

```
##
##   0   1
## 138 165
```

Splitting in Train and Test Datasets

```r
set.seed(123)
samp_size=floor(0.75*nrow(data1_norm))

samp_ind=sample(seq_len(nrow(data1_norm)),size = samp_size)
```

```r
# making train and test dataset by dividing data into 75% for train and 25% for test dataset.
data_train=data1_norm[samp_ind,-6]
data_test=data1_norm[-samp_ind,-6]
```

```r
#extracting the dependent variables and splitting it into train and test datasets.
data_train_labels=data1_norm[samp_ind,6]
data_test_labels=data1_norm[-samp_ind,6]
```

```r
# Knn Model
data_test_pred=knn(train = data_train,test = data_test,cl=data_train_labels,k=17)
```

```r
# checking accuracy
CrossTable(x=data_test_labels,y=data_test_pred,prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  76
##
##
##                  | data_test_pred
## data_test_labels |         0 |         1 | Row Total |
## -----------------|-----------|-----------|-----------|
##                0 |        27 |         8 |        35 |
##                  |     0.771 |     0.229 |     0.461 |
##                  |     0.871 |     0.178 |           |
##                  |     0.355 |     0.105 |           |
## -----------------|-----------|-----------|-----------|
##                1 |         4 |        37 |        41 |
##                  |     0.098 |     0.902 |     0.539 |
##                  |     0.129 |     0.822 |           |
##                  |     0.053 |     0.487 |           |
## -----------------|-----------|-----------|-----------|
##     Column Total |        31 |        45 |        76 |
##                  |     0.408 |     0.592 |           |
## -----------------|-----------|-----------|-----------|
##
##
```

```
confusionMatrix(table(data_test_labels,data_test_pred))
```

```
## Confusion Matrix and Statistics
##
##                 data_test_pred
## data_test_labels  0  1
##                0 27  8
##                1  4 37
##
##                Accuracy : 0.8421
##                  95% CI : (0.7404, 0.9157)
##     No Information Rate : 0.5921
##     P-Value [Acc > NIR] : 2.411e-06
##
##                   Kappa : 0.6796
##
##  Mcnemar's Test P-Value : 0.3865
##
##             Sensitivity : 0.8710
##             Specificity : 0.8222
##          Pos Pred Value : 0.7714
##          Neg Pred Value : 0.9024
##              Prevalence : 0.4079
##          Detection Rate : 0.3553
##    Detection Prevalence : 0.4605
##       Balanced Accuracy : 0.8466
##
##        'Positive' Class : 0
##
```

Multiple Logistic Regression Model

Setting up train and test dataset.

```
set.seed(123)
samp_size2=floor(0.75*nrow(data1))

samp_ind2=sample(seq_len(nrow(data1_norm)),size = samp_size2)
```

```r
data_train2=data1[samp_ind2,]
data_test2= data1[-samp_ind2,-6]

data_test2_label=data1[-samp_ind2,6]
```

```r
model_glm=glm(target~.,data = data_train2,family = binomial)
summary(model_glm)
```

```
##
## Call:
## glm(formula = target ~ ., family = binomial, data = data_train2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6521  -0.2621   0.1034   0.4725   2.8936
##
## Coefficients: (8 not defined because of singularities)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.042e+00  3.167e+03   0.001  0.99949
## ï..age       1.755e-02  3.037e-02   0.578  0.56327
## trestbps    -3.066e-02  1.315e-02  -2.331  0.01974 *
## chol        -2.059e-03  5.092e-03  -0.404  0.68593
## thalach      3.165e-02  1.536e-02   2.061  0.03934 *
## oldpeak     -5.058e-01  2.810e-01  -1.800  0.07187 .
## ca_0        -1.589e+01  1.646e+03  -0.010  0.99230
## ca_1        -1.714e+01  1.646e+03  -0.010  0.99169
## ca_2        -1.935e+01  1.646e+03  -0.012  0.99062
## ca_3        -1.733e+01  1.646e+03  -0.011  0.99160
## ca_4               NA         NA      NA       NA
## cp_0        -2.223e+00  7.652e-01  -2.906  0.00367 **
## cp_1        -1.535e+00  8.933e-01  -1.719  0.08568 .
## cp_2        -6.829e-01  7.774e-01  -0.878  0.37971
## cp_3               NA         NA      NA       NA
## exang_0      1.004e+00  5.458e-01   1.840  0.06574 .
## exang_1            NA         NA      NA       NA
## fbs_0       -6.247e-01  6.839e-01  -0.913  0.36099
## fbs_1              NA         NA      NA       NA
## restecg_0    1.398e+01  2.705e+03   0.005  0.99588
## restecg_1    1.484e+01  2.705e+03   0.005  0.99562
## restecg_2          NA         NA      NA       NA
## sex_0        1.706e+00  6.739e-01   2.532  0.01135 *
## sex_1              NA         NA      NA       NA
## slope_0      3.344e-01  1.285e+00   0.260  0.79472
## slope_1     -9.487e-01  5.689e-01  -1.668  0.09536 .
## slope_2            NA         NA      NA       NA
## thal_0      -1.639e+01  3.956e+03  -0.004  0.99669
## thal_1       2.748e+00  1.078e+00   2.550  0.01077 *
## thal_2       1.654e+00  5.185e-01   3.190  0.00142 **
## thal_3             NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 312.74  on 226  degrees of freedom
## Residual deviance: 131.40  on 204  degrees of freedom
## AIC: 177.4
##
## Number of Fisher Scoring iterations: 16
```

```
# variable selection to improve model
model_glm_sel=step(model_glm)
```

```
## Start:  AIC=177.4
## target ~ ï..age + trestbps + chol + thalach + oldpeak + ca_0 +
##     ca_1 + ca_2 + ca_3 + ca_4 + cp_0 + cp_1 + cp_2 + cp_3 + exang_0 +
##     exang_1 + fbs_0 + fbs_1 + restecg_0 + restecg_1 + restecg_2 +
##     sex_0 + sex_1 + slope_0 + slope_1 + slope_2 + thal_0 + thal_1 +
##     thal_2 + thal_3

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=177.4
## target ~ ï..age + trestbps + chol + thalach + oldpeak + ca_0 +
##     ca_1 + ca_2 + ca_3 + ca_4 + cp_0 + cp_1 + cp_2 + cp_3 + exang_0 +
##     exang_1 + fbs_0 + fbs_1 + restecg_0 + restecg_1 + restecg_2 +
##     sex_0 + sex_1 + slope_0 + slope_1 + slope_2 + thal_0 + thal_1 +
##     thal_2

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=177.4
## target ~ ï..age + trestbps + chol + thalach + oldpeak + ca_0 +
##     ca_1 + ca_2 + ca_3 + ca_4 + cp_0 + cp_1 + cp_2 + cp_3 + exang_0 +
##     exang_1 + fbs_0 + fbs_1 + restecg_0 + restecg_1 + restecg_2 +
##     sex_0 + sex_1 + slope_0 + slope_1 + thal_0 + thal_1 + thal_2

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=177.4
## target ~ ï..age + trestbps + chol + thalach + oldpeak + ca_0 +
##     ca_1 + ca_2 + ca_3 + ca_4 + cp_0 + cp_1 + cp_2 + cp_3 + exang_0 +
##     exang_1 + fbs_0 + fbs_1 + restecg_0 + restecg_1 + restecg_2 +
##     sex_0 + slope_0 + slope_1 + thal_0 + thal_1 + thal_2

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=177.4
## target ~ ï..age + trestbps + chol + thalach + oldpeak + ca_0 +
##     ca_1 + ca_2 + ca_3 + ca_4 + cp_0 + cp_1 + cp_2 + cp_3 + exang_0 +
##     exang_1 + fbs_0 + fbs_1 + restecg_0 + restecg_1 + sex_0 +
##     slope_0 + slope_1 + thal_0 + thal_1 + thal_2

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: algorithm did not converge

##
## Step:  AIC=184.7
## target ~ ï..age + trestbps + chol + thalach + oldpeak + ca_0 +
##     ca_1 + ca_2 + ca_3 + ca_4 + cp_0 + cp_1 + cp_2 + cp_3 + exang_0 +
##     exang_1 + fbs_0 + fbs_1 + restecg_0 + restecg_1 + sex_0 +
##     slope_0 + slope_1 + thal_0 + thal_2
```

```r
summary(model_glm_sel)
```

```
##
## Call:
## glm(formula = target ~ ï..age + trestbps + chol + thalach + oldpeak +
##     ca_0 + ca_1 + ca_2 + ca_3 + ca_4 + cp_0 + cp_1 + cp_2 + cp_3 +
##     exang_0 + exang_1 + fbs_0 + fbs_1 + restecg_0 + restecg_1 +
##     sex_0 + slope_0 + slope_1 + thal_0 + thal_2, family = binomial,
##     data = data_train2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5495  -0.3048   0.1226   0.4827   2.7128
##
## Coefficients: (3 not defined because of singularities)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  7.359e+13  2.610e+14   0.282  0.77796
## ï..age       1.631e-02  2.930e-02   0.557  0.57768
## trestbps    -2.872e-02  1.333e-02  -2.155  0.03116 *
## chol        -3.467e-03  4.910e-03  -0.706  0.48018
## thalach      2.536e-02  1.424e-02   1.780  0.07501 .
## oldpeak     -4.933e-01  2.760e-01  -1.788  0.07383 .
## ca_0        -7.359e+13  2.610e+14  -0.282  0.77796
## ca_1        -7.359e+13  2.610e+14  -0.282  0.77796
## ca_2        -7.359e+13  2.610e+14  -0.282  0.77796
## ca_3        -7.359e+13  2.610e+14  -0.282  0.77796
## ca_4        -7.359e+13  2.610e+14  -0.282  0.77796
## cp_0        -1.898e+00  7.220e-01  -2.628  0.00858 **
## cp_1        -1.177e+00  8.708e-01  -1.352  0.17632
## cp_2        -5.942e-01  7.599e-01  -0.782  0.43425
## cp_3                NA         NA      NA       NA
## exang_0      1.073e+00  5.317e-01   2.017  0.04365 *
## exang_1             NA         NA      NA       NA
## fbs_0       -5.955e-01  6.585e-01  -0.904  0.36576
## fbs_1               NA         NA      NA       NA
## restecg_0    2.299e+01  2.160e+05   0.000  0.99992
## restecg_1    2.388e+01  2.160e+05   0.000  0.99991
## sex_0        1.514e+00  6.512e-01   2.325  0.02009 *
## slope_0      7.489e-01  1.200e+00   0.624  0.53253
## slope_1     -8.692e-01  5.603e-01  -1.551  0.12087
## thal_0      -2.560e+01  3.098e+05   0.000  0.99993
## thal_2       1.369e+00  5.013e-01   2.732  0.00630 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 312.74  on 226  degrees of freedom
## Residual deviance: 138.70  on 204  degrees of freedom
## AIC: 184.7
##
## Number of Fisher Scoring iterations: 25
```

```
AIC(model_glm,model_glm_sel)
```

```
##               df      AIC
## model_glm     23 177.3955
## model_glm_sel 23 184.7007
```

```
data_test2$prediction=predict(model_glm,newdata = data_test2,type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
data_test2$prediction=ifelse(data_test2$prediction>0.5,1,0)
```

```
confusionMatrix(table(data_test2_label,data_test2$prediction))
```

```
## Confusion Matrix and Statistics
##
##
## data_test2_label  0  1
##                0 25 10
##                1  5 36
##
##                Accuracy : 0.8026
##                  95% CI : (0.6954, 0.8851)
##     No Information Rate : 0.6053
##     P-Value [Acc > NIR] : 0.0001938
##
##                   Kappa : 0.5986
##
##  Mcnemar's Test P-Value : 0.3016996
##
##             Sensitivity : 0.8333
##             Specificity : 0.7826
##          Pos Pred Value : 0.7143
##          Neg Pred Value : 0.8780
##              Prevalence : 0.3947
##          Detection Rate : 0.3289
##    Detection Prevalence : 0.4605
##       Balanced Accuracy : 0.8080
##
##        'Positive' Class : 0
##
```

Conclusion

As shown above from the two Machine Learning Models applied the KNN (Nearest Neighbour) Model was better then Logisitic Regression because its accuracy was approximately 4% more.