**Mid Term Exam**

**Data Mining Applications in Management**

**Name – Souvik Samanta**

**Roll No - 19PGDM064**

**Question1.**

1. The dataset contains 12 predictor variables except “TIME (*follow up period)”* which can be used to predict the death of the patient. We are assuming all the levels of enzymes, creatinine, sodium are relevant to heart disease which may not be that relevant (needs further domain knowledge in medical field). So, we will take every variable except TIME for modelling and further analysis.
2. Using NB algorithm initially for building the model. NB can be useful in this case as the dependent variable is binary in nature and NB can handle continuous and factor input variables well.

**R-Code with output –**

# Clearing environment  
rm(list=ls())  
  
# Setting Working Directory  
setwd("C:/Users/souvi/Desktop/DMAM")  
  
# Importing Dataset  
data <- read.csv("Question1 end term.csv")  
  
#Structure of Data  
str(data)

## 'data.frame': 299 obs. of 13 variables:  
## $ age : num 75 55 65 50 65 90 75 60 65 80 ...  
## $ anaemia : int 0 0 0 1 1 1 1 1 0 1 ...  
## $ creatinine\_phosphokinase: int 582 7861 146 111 160 47 246 315 157 123 ...  
## $ diabetes : int 0 0 0 0 1 0 0 1 0 0 ...  
## $ ejection\_fraction : int 20 38 20 20 20 40 15 60 65 35 ...  
## $ high\_blood\_pressure : int 1 0 0 0 0 1 0 0 0 1 ...  
## $ platelets : num 265000 263358 162000 210000 327000 ...  
## $ serum\_creatinine : num 1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4 ...  
## $ serum\_sodium : int 130 136 129 137 116 132 137 131 138 133 ...  
## $ sex : int 1 1 1 1 0 1 1 1 0 1 ...  
## $ smoking : int 0 0 1 0 0 1 0 1 0 1 ...  
## $ time : int 4 6 7 7 8 8 10 10 10 10 ...  
## $ DEATH\_EVENT : int 1 1 1 1 1 1 1 1 1 1 ...

#Removing TIME column  
data$time <- NULL  
#Converting factor variables  
data$anaemia <- factor(data$anaemia)  
data$diabetes <- factor(data$diabetes)  
data$high\_blood\_pressure <- factor(data$high\_blood\_pressure)  
data$sex <- factor(data$sex)  
data$smoking <- factor(data$smoking)  
data$DEATH\_EVENT <- factor(data$DEATH\_EVENT)  
str(data)

## 'data.frame': 299 obs. of 12 variables:  
## $ age : num 75 55 65 50 65 90 75 60 65 80 ...  
## $ anaemia : Factor w/ 2 levels "0","1": 1 1 1 2 2 2 2 2 1 2 ...  
## $ creatinine\_phosphokinase: int 582 7861 146 111 160 47 246 315 157 123 ...  
## $ diabetes : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...  
## $ ejection\_fraction : int 20 38 20 20 20 40 15 60 65 35 ...  
## $ high\_blood\_pressure : Factor w/ 2 levels "0","1": 2 1 1 1 1 2 1 1 1 2 ...  
## $ platelets : num 265000 263358 162000 210000 327000 ...  
## $ serum\_creatinine : num 1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4 ...  
## $ serum\_sodium : int 130 136 129 137 116 132 137 131 138 133 ...  
## $ sex : Factor w/ 2 levels "0","1": 2 2 2 2 1 2 2 2 1 2 ...  
## $ smoking : Factor w/ 2 levels "0","1": 1 1 2 1 1 2 1 2 1 2 ...  
## $ DEATH\_EVENT : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...

head(data)

## age anaemia creatinine\_phosphokinase diabetes ejection\_fraction  
## 1 75 0 582 0 20  
## 2 55 0 7861 0 38  
## 3 65 0 146 0 20  
## 4 50 1 111 0 20  
## 5 65 1 160 1 20  
## 6 90 1 47 0 40  
## high\_blood\_pressure platelets serum\_creatinine serum\_sodium sex smoking  
## 1 1 265000 1.9 130 1 0  
## 2 0 263358 1.1 136 1 0  
## 3 0 162000 1.3 129 1 1  
## 4 0 210000 1.9 137 1 0  
## 5 0 327000 2.7 116 0 0  
## 6 1 204000 2.1 132 1 1  
## DEATH\_EVENT  
## 1 1  
## 2 1  
## 3 1  
## 4 1  
## 5 1  
## 6 1

summary(data)

## age anaemia creatinine\_phosphokinase diabetes ejection\_fraction  
## Min. :40.00 0:170 Min. : 23.0 0:174 Min. :14.00   
## 1st Qu.:51.00 1:129 1st Qu.: 116.5 1:125 1st Qu.:30.00   
## Median :60.00 Median : 250.0 Median :38.00   
## Mean :60.83 Mean : 581.8 Mean :38.08   
## 3rd Qu.:70.00 3rd Qu.: 582.0 3rd Qu.:45.00   
## Max. :95.00 Max. :7861.0 Max. :80.00   
## high\_blood\_pressure platelets serum\_creatinine serum\_sodium sex   
## 0:194 Min. : 25100 Min. :0.500 Min. :113.0 0:105   
## 1:105 1st Qu.:212500 1st Qu.:0.900 1st Qu.:134.0 1:194   
## Median :262000 Median :1.100 Median :137.0   
## Mean :263358 Mean :1.394 Mean :136.6   
## 3rd Qu.:303500 3rd Qu.:1.400 3rd Qu.:140.0   
## Max. :850000 Max. :9.400 Max. :148.0   
## smoking DEATH\_EVENT  
## 0:203 0:203   
## 1: 96 1: 96   
##   
##   
##   
##

# Creating Training and validation dataset  
set.seed(1)  
train.index <- sample(c(1:dim(data)[1]), dim(data)[1]\*0.8)  
# Dataset is not that huge and for getting maximum training data, taking train-test split of 80:20  
valid.index <- setdiff(c(1:dim(data)[1]), train.index)  
train.df <- data[train.index, ]  
valid.df <- data[valid.index, ]  
  
dim(train.df)

## [1] 239 12

t<- table(train.df$DEATH\_EVENT); t

##   
## 0 1   
## 164 75

dim(valid.df)

## [1] 60 12

v<-table(valid.df$DEATH\_EVENT); v

##   
## 0 1   
## 39 21

###################################################################  
#Naive Bayes  
###################################################################  
library(e1071)  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

bayes.model <- naiveBayes(DEATH\_EVENT ~ ., data=train.df)  
bayes.model

##   
## Naive Bayes Classifier for Discrete Predictors  
##   
## Call:  
## naiveBayes.default(x = X, y = Y, laplace = laplace)  
##   
## A-priori probabilities:  
## Y  
## 0 1   
## 0.6861925 0.3138075   
##   
## Conditional probabilities:  
## age  
## Y [,1] [,2]  
## 0 58.13821 10.89999  
## 1 66.14223 13.53820  
##   
## anaemia  
## Y 0 1  
## 0 0.597561 0.402439  
## 1 0.560000 0.440000  
##   
## creatinine\_phosphokinase  
## Y [,1] [,2]  
## 0 564.0366 802.7577  
## 1 752.0667 1453.2998  
##   
## diabetes  
## Y 0 1  
## 0 0.6036585 0.3963415  
## 1 0.5600000 0.4400000  
##   
## ejection\_fraction  
## Y [,1] [,2]  
## 0 40.43902 11.12185  
## 1 34.06667 12.17589  
##   
## high\_blood\_pressure  
## Y 0 1  
## 0 0.6768293 0.3231707  
## 1 0.5733333 0.4266667  
##   
## platelets  
## Y [,1] [,2]  
## 0 271622.0 102287.9  
## 1 249261.1 97799.6  
##   
## serum\_creatinine  
## Y [,1] [,2]  
## 0 1.207012 0.7132335  
## 1 1.803200 1.3285149  
##   
## serum\_sodium  
## Y [,1] [,2]  
## 0 137.25 3.963600  
## 1 135.84 4.570883  
##   
## sex  
## Y 0 1  
## 0 0.3475610 0.6524390  
## 1 0.3466667 0.6533333  
##   
## smoking  
## Y 0 1  
## 0 0.6768293 0.3231707  
## 1 0.7466667 0.2533333

# generate confusion matrix for training data

pred.train1 <- predict(bayes.model,train.df,type = "class")  
confusionMatrix(pred.train1, as.factor(train.df$DEATH\_EVENT),positive="1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 150 52  
## 1 14 23  
##   
## Accuracy : 0.7238   
## 95% CI : (0.6625, 0.7795)  
## No Information Rate : 0.6862   
## P-Value [Acc > NIR] : 0.1173   
##   
## Kappa : 0.2566   
##   
## Mcnemar's Test P-Value : 5.254e-06   
##   
## Sensitivity : 0.30667   
## Specificity : 0.91463   
## Pos Pred Value : 0.62162   
## Neg Pred Value : 0.74257   
## Prevalence : 0.31381   
## Detection Rate : 0.09623   
## Detection Prevalence : 0.15481   
## Balanced Accuracy : 0.61065   
##   
## 'Positive' Class : 1   
##

# We got an accuracy of 72.4%  
# we got sensitivity of 30.67%  
  
# generate confusion matrix for validation data  
pred.valid1 <- predict(bayes.model,valid.df,type = "class")  
confusionMatrix(pred.valid1, as.factor(valid.df$DEATH\_EVENT),positive="1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 39 16  
## 1 0 5  
##   
## Accuracy : 0.7333   
## 95% CI : (0.6034, 0.8393)  
## No Information Rate : 0.65   
## P-Value [Acc > NIR] : 0.1100955   
##   
## Kappa : 0.2889   
##   
## Mcnemar's Test P-Value : 0.0001768   
##   
## Sensitivity : 0.23810   
## Specificity : 1.00000   
## Pos Pred Value : 1.00000   
## Neg Pred Value : 0.70909   
## Prevalence : 0.35000   
## Detection Rate : 0.08333   
## Detection Prevalence : 0.08333   
## Balanced Accuracy : 0.61905   
##   
## 'Positive' Class : 1   
##

# We got an accuracy of 73.3%  
# we got sensitivity of 23.81%

1. In this case our model should be able to predict whether a patient may die or now accurately. So, the true positives should be very high as compared to true negatives. So, we should look for good sensitivity in our model. In this model followings are the efficiency measures-

|  |  |  |
| --- | --- | --- |
| **NB MODEL** | Accuracy | Sensitivity |
| Training Data | 72.4% | 30.67% |
| Validation Data | 73.3% | 23.81% |

So, for both training and testing we got poor results of sensitivity. We can use other models to increase this score.

1. Yes, we can improve the results by using some advanced algorithms. We can use other algorithms of classification which may increase the efficiency of the results. Let us use ensemble models like Adaboost to increase the efficiency.

**R-Code with output –**

############################################  
# ADABOOST  
############################################  
library(ada)

## Warning: package 'ada' was built under R version 4.0.3

## Loading required package: rpart

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

cvcontrol <- trainControl(method="repeatedcv", number = 10,repeats = 10)  
train.ada <- train(DEATH\_EVENT ~ ., data=train.df, method="ada",trControl=cvcontrol)  
  
#computing confusion matrix for training data  
pred.train2 <- predict(train.ada,train.df,type = "raw")  
confusionMatrix(pred.train2, as.factor(train.df$DEATH\_EVENT),positive="1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 152 24  
## 1 12 51  
##   
## Accuracy : 0.8494   
## 95% CI : (0.7976, 0.8922)  
## No Information Rate : 0.6862   
## P-Value [Acc > NIR] : 5.378e-09   
##   
## Kappa : 0.6344   
##   
## Mcnemar's Test P-Value : 0.06675   
##   
## Sensitivity : 0.6800   
## Specificity : 0.9268   
## Pos Pred Value : 0.8095   
## Neg Pred Value : 0.8636   
## Prevalence : 0.3138   
## Detection Rate : 0.2134   
## Detection Prevalence : 0.2636   
## Balanced Accuracy : 0.8034   
##   
## 'Positive' Class : 1   
##

# We got an accuracy of 84.9%  
# we got sensitivity of 68%  
  
#computing confusion matrix for validation data  
pred.valid2 <- predict(train.ada,valid.df,type = "raw")  
confusionMatrix(pred.valid2, as.factor(valid.df$DEATH\_EVENT),positive="1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 36 12  
## 1 3 9  
##   
## Accuracy : 0.75   
## 95% CI : (0.6214, 0.8528)  
## No Information Rate : 0.65   
## P-Value [Acc > NIR] : 0.06561   
##   
## Kappa : 0.3902   
##   
## Mcnemar's Test P-Value : 0.03887   
##   
## Sensitivity : 0.4286   
## Specificity : 0.9231   
## Pos Pred Value : 0.7500   
## Neg Pred Value : 0.7500   
## Prevalence : 0.3500   
## Detection Rate : 0.1500   
## Detection Prevalence : 0.2000   
## Balanced Accuracy : 0.6758   
##   
## 'Positive' Class : 1   
##

# We got an accuracy of 75%  
# we got sensitivity of 42.9%

1. By using Adaboost, both accuracy and sensitivity scores have been increased largely. Now our model can predict around 70% of cases accurately for a test data set as well as on a new data point which is far better than the results, we got using NB model.

As Adaboost uses boosting technique which gives more weightage for the next iteration on the data points which caused misclassifications in previous models. Thus, in a way, this algorithm improves itself throughout iterations. The efficiency scores are as follows where we can compare with the previous model how the scores have improved-

|  |  |  |
| --- | --- | --- |
| **NB MODEL** | Accuracy | Sensitivity |
| Training Data | 72.4% | 30.67% |
| Validation Data | 73.3% | 23.81% |

|  |  |  |
| --- | --- | --- |
| **AdaBOOST MODEL** | Accuracy | Sensitivity |
| Training Data | 84.9% | 68% |
| Validation Data | 75% | 42.9% |