german

Shiva Sankar Modala

2023-02-10

# Loading the necessary libraries  
library(readr)  
library(data.table)  
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

#Load the German Credit Data sample dataset from the UCI Machine Learning Repository (german.data-numeric) into R using a dataframe in the table format  
creditGermData<-read.table("https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german/german.data-numeric",header = FALSE)  
set.seed(100)  
creditGermData$V25 = factor(creditGermData$V25)

# I used the caret package to perform a 80/20 test-train split using the createDataPartition()  
train\_Index = createDataPartition(y = creditGermData$V25 , p = 0.8, list = FALSE)

# Separating the Training data   
train\_Data = creditGermData[train\_Index,]

# Separating the Testing data  
testData = creditGermData[-train\_Index,]

# obtain a training fit for a logistic model via the glm()  
logisticModel = glm(V25~.,family=binomial,data=train\_Data)  
actualVals = train\_Data$V25

# 50% cut-off factor so that the probabilities > 0.5 are 2 and rest are 1  
fittedVals = ifelse(logisticModel$fitted.values > 0.5,2,1)  
fittedVals = factor(fittedVals)

# Gives the confusion matrix for the fitted and train data  
cm = confusionMatrix(fittedVals, train\_Data$V25)

# The training Precision/Recall and F1 results are:

cat("\n Training Precision: ", cm$byClass[5] \* 100, "%")

##   
## Training Precision: 82.16039 %

cat("\n Training Recall: ", cm$byClass[6] \* 100, "%")

##   
## Training Recall: 89.64286 %

cat("\n Training F1-Score: ", cm$byClass[7] \* 100, "%")

##   
## Training F1-Score: 85.73868 %

probs = predict(logisticModel, testData, type = "response")

fittedVals\_test = ifelse(probs > 0.5,2,1)  
fittedVals\_test = factor(fittedVals\_test)

cm\_test = confusionMatrix(fittedVals\_test, testData$V25)  
cm\_test

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2  
## 1 124 36  
## 2 16 24  
##   
## Accuracy : 0.74   
## 95% CI : (0.6734, 0.7993)  
## No Information Rate : 0.7   
## P-Value [Acc > NIR] : 0.122775   
##   
## Kappa : 0.3158   
##   
## Mcnemar's Test P-Value : 0.008418   
##   
## Sensitivity : 0.8857   
## Specificity : 0.4000   
## Pos Pred Value : 0.7750   
## Neg Pred Value : 0.6000   
## Prevalence : 0.7000   
## Detection Rate : 0.6200   
## Detection Prevalence : 0.8000   
## Balanced Accuracy : 0.6429   
##   
## 'Positive' Class : 1   
##

cat("\n Testing Precision: ", cm\_test$byClass[5] \* 100, "%")

##   
## Testing Precision: 77.5 %

cat("\n Testing Recall: ", cm\_test$byClass[6] \* 100, "%")

##   
## Testing Recall: 88.57143 %

cat("\n Testing F1-Score: ", cm\_test$byClass[7] \* 100, "%")

##   
## Testing F1-Score: 82.66667 %

# use the trainControl and train functions to perform a k=10 fold cross-validation fit of the same model,  
# Define training control  
train.control = trainControl(method = "cv", number = 10)

# Training the model  
logisticModel2 = train(V25~., data = train\_Data, method = "glm", family = "binomial", trControl =train.control)  
fittedVals\_cv = ifelse(logisticModel2$finalModel$fitted.values > 0.5,2,1)  
fittedVals\_cv = factor(fittedVals\_cv)

# Confusion matrix  
cm\_cv = confusionMatrix(fittedVals\_cv, train\_Data$V25)  
cm\_cv

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2  
## 1 502 109  
## 2 58 131  
##   
## Accuracy : 0.7912   
## 95% CI : (0.7614, 0.8189)  
## No Information Rate : 0.7   
## P-Value [Acc > NIR] : 3.653e-09   
##   
## Kappa : 0.4708   
##   
## Mcnemar's Test P-Value : 0.0001092   
##   
## Sensitivity : 0.8964   
## Specificity : 0.5458   
## Pos Pred Value : 0.8216   
## Neg Pred Value : 0.6931   
## Prevalence : 0.7000   
## Detection Rate : 0.6275   
## Detection Prevalence : 0.7638   
## Balanced Accuracy : 0.7211   
##   
## 'Positive' Class : 1   
##

cat("\n Training Precision with 10-fold CV: ", cm\_cv$byClass[5] \* 100, "%")

##   
## Training Precision with 10-fold CV: 82.16039 %

cat("\n Training Recall with 10-fold CV: ", cm\_cv$byClass[6] \* 100, "%")

##   
## Training Recall with 10-fold CV: 89.64286 %

cat("\n Training F1-Score with 10-fold CV: ", cm\_cv$byClass[7] \* 100, "%")

##   
## Training F1-Score with 10-fold CV: 85.73868 %

probs\_cv = predict(logisticModel2, testData, type = "prob")

# 50% cut-off factor so that the probabilities > 0.5 are 2 and rest are 1  
fittedVals\_cv\_test = ifelse(probs > 0.5,2,1)  
fittedVals\_cv\_test = factor(fittedVals\_test)  
cm\_cv\_test = confusionMatrix(fittedVals\_test, testData$V25)

# cross-validated training Precision/Recall and F1 values.

cat("\n Testing Precision: ", cm\_cv\_test$byClass[5] \* 100, "%")

##   
## Testing Precision: 77.5 %

cat("\n Testing Recall: ", cm\_cv\_test$byClass[6] \* 100, "%")

##   
## Testing Recall: 88.57143 %

cat("\n Testing F1-Score: ", cm\_cv\_test$byClass[7] \* 100, "%")

##   
## Testing F1-Score: 82.66667 %

cat("\n From the above observations, we can observe that both the cross validation and basic model have same result.")

##   
## From the above observations, we can observe that both the cross validation and basic model have same result.