Informatics Applied to Quantitative Data: Part-II (MPH Course: Health Informatics and Decision Making)

Zoie Shui-Yee Wong; Tshewang Gyeltshen

Contents

1. Loading Data and Exploring the Data

Machine Learning Models for Binary Classification

DatasetLASA.csv :
The dataset *LASA* contains free text tokens generated by NLP program (originally free text format and now structured as shown).
The last column Y - "LASA" is the response variable and other columns are predictor variables, except column 1 (document ID).

LASA stands for look-alike and sound-alike medications.
The response variable has been coded by two classes - 0: for Not LASA, and 1 for LASA.

Using 3 different machine learning models (including logistic regression, LDA and ANN) we will develop and evaluate classifiers to identify the classes using the structured (free-text) data.

```
rm(list = ls())
setwd("C:/Users/zoies/Desktop/StLuke_MicrosoftPC/Teaching/HI-2023/W8Handson")
dat <- read.csv("DatasetLASA.csv")
```

Exploring the dataset

```
#Explore the dataset, rename column name, set class names
#head(dat)
class(dat)
```

[1] "data.frame"

```
colnames(dat)[1] <- 'DocID'
dat$LASAcases <- factor(dat$LASAcases, levels = c(0, 1), labels = c("NO", "YES"))
# Change the class of LASA variable to factor for classification purpose.
dat$LASAcases[1:5]
```

[1] NO YES NO YES NO Levels: NO YES

2. Developing Machine Learning Models

2.1 Logistic Regression

```
#Developing Logistic Regression model with binary response
glm_model <- glm(LASAcases ~ . -DocID, data = dat,family="binomial")

#Predict the LASA cases based on LR
prob.error <- predict(glm_model,type='response')
lr.class<- rep("NO",227)      #227 (Assumed all to be 0, NO LASA cases)
lr.class[prob.error>.5] <- "YES"   #Change the ones greater than 0.5 to 1, i.e YES LASA cases
lr.class[1:5]
```

[1] "NO" "NO" "YES" "NO" "NO"

2.2 Liner Discriminant Analysis - LDA

Linear discriminant analysis `lda()` usage is basically similar to `lm()`, `glm()`. The MASS package needs to be loaded in advance.

```
library(MASS)
# Fit the LDA model with all the predictor variables on the response variable LASA
lda_model = lda(LASAcases ~ .-DocID,data=dat)

#Predict the LASA cases based on LDA
lda.pred <- predict(lda_model, dat)
lda.class <- lda.pred$class
lda.class[1:5]
```

[1] NO YES NO YES NO Levels: NO YES

2.3 Artifical Neural Network - ANN

```
#install.packages(c('neuralnet'),dependencies = T)
library("neuralnet")
ann_model <- neuralnet(LASAcases ~. -DocID, data = dat, hidden = 1)
#You can control the hidden layers with 'hidden=' and simple ANN contains 1 hidden layer.

#Predict the LASA cases based on ANN
ann_pred <- predict(ann_model, dat)
labels <- c("NO", "YES")
#The labels object is created as a character vector containing the labels for each category in the resp

ann.class <- data.frame(max.col(ann_pred)) %>%
mutate(ann_pred=labels[max.col(ann_pred)]) %>%
dplyr::select(2) %>%
unlist()
ann.class[1:5]
```

ann_pred1 ann_pred2 ann_pred3 ann_pred4 ann_pred5 "NO" "YES" "NO" "YES" "NO"

3. Evaluating Model Performance

We will evaluate our model performances based on 2 by 2 confusion matrix and model evaluating metrics such as Precision, Recall, F-score, Accuracy, ROC and AUC.

3.1 Recall, Precision, F-1 score, and Accuracy

```
library(caret)
# Logistic regression model - evaluation
logit_cfnmtrx <- confusionMatrix(table(lr.class, dat$LASAcases))
logit_cfnmtrx$table
```

lr.class NO YES NO 110 17 YES 83 17

```
logit_cfnmtrx$byClass["Recall"]
```

Recall 0.57

```
logit_cfnmtrx$byClass["Precision"]
```

Precision 0.87

```
logit_cfnmtrx$byClass["F1"]
```

F1 0.69

```
logit_cfnmtrx$overall["Accuracy"]
```

Accuracy 0.56

```r
# LDA - evaluation
lda_cfnmtrx <- confusionMatrix(table(lda.class, dat$LASAcases))
lda_cfnmtrx$table
```

lda.class NO YES NO 193 4 YES 0 30

```r
lda_cfnmtrx$byClass["Recall"]
```

Recall 1

```r
lda_cfnmtrx$byClass["Precision"]
```

Precision 0.98

```r
lda_cfnmtrx$byClass["F1"]
```

F1 0.99

```r
lda_cfnmtrx$overall["Accuracy"]
```

Accuracy 0.98

```r
# ANN - evaluation
ann_cfnmtrx <- confusionMatrix(table(ann.class, dat$LASAcases))
ann_cfnmtrx$table
```

ann.class NO YES NO 193 4 YES 0 30

```r
ann_cfnmtrx$byClass["Recall"]
```

Recall 1

```r
ann_cfnmtrx$byClass["Precision"]
```

Precision 0.98

```r
ann_cfnmtrx$byClass["F1"]
```

F1 0.99

```r
ann_cfnmtrx$overall["Accuracy"]
```

Accuracy 0.98

3.2 ROC Curves and AUC Values

The `ROCR` and `pROC` packages are used for ROC curves.
In order to draw the ROC curve, it is necessary to convert all outputs into probability values.

```r
library(ROCR)
library(pROC)

# Extract the predicted probabilities for the positive class
logit0.pred <- predict(glm_model, dat, type = "response" )  # glm
lda0.pred <- predict(lda_model, dat)$posterior[,2]  # lda
ann.pred <- compute(ann_model, dat)$net.result
ann.pred <- ann.pred[, 1]

roc_logit <- roc(dat$LASAcases, logit0.pred)
roc_lda <- roc(dat$LASAcases, lda0.pred)
roc_ann <- roc(dat$LASAcases, ann.pred)

# Plot the ROC curve
par ( mfrow = c (2, 2 ) )
plot(roc_logit, main = "Logit") # GLM
plot(roc_lda, main = "LDA") # LDA
plot(roc_ann, main = "ANN") # ANN

#AUC - Area under the Curves
auc(roc_logit)
```

Area under the curve: 0.54

```r
auc(roc_lda)
```

Area under the curve: 0.99

```r
auc(roc_ann)
```

Area under the curve: 0.99