Exercise 4

Konstantinos Vakalopoulos 12223236

2022-11-24

Premilinary work

First, the data (hcvdat1.csv) were loaded

```
d <- read.csv("C:/Users/vaka1/Desktop/hcvdat1.csv") #must be modified
```

and missing values were omitted.

```
d <- na.omit(d)
```

For addition information about the data, the following commands were used

```
summary(d)
```

```
##    Category              Age            Sex                 ALB
##  Length:570         Min.   :23.00   Length:570         Min.   :23.00
##  Class :character   1st Qu.:39.00   Class :character   1st Qu.:38.92
##  Mode  :character   Median :47.00   Mode  :character   Median :41.90
##                     Mean   :47.25                      Mean   :41.81
##                     3rd Qu.:54.00                      3rd Qu.:45.17
##                     Max.   :77.00                      Max.   :82.20
##       ALP              ALT              AST              BIL
##  Min.   : 11.30   Min.   :  0.90   Min.   : 12.00   Min.   :  1.80
##  1st Qu.: 53.60   1st Qu.: 16.52   1st Qu.: 21.30   1st Qu.:  5.20
##  Median : 66.35   Median : 22.70   Median : 25.50   Median :  7.10
##  Mean   : 68.28   Mean   : 25.80   Mean   : 32.17   Mean   : 11.03
##  3rd Qu.: 79.83   3rd Qu.: 31.88   3rd Qu.: 31.10   3rd Qu.: 10.80
##  Max.   :416.60   Max.   :118.10   Max.   :324.00   Max.   :209.00
##       CHE              CHOL             CREA             GGT
##  Min.   : 1.420   Min.   :1.430   Min.   :   8.00   Min.   :  4.50
##  1st Qu.: 6.940   1st Qu.:4.643   1st Qu.:  68.00   1st Qu.: 15.40
##  Median : 8.260   Median :5.330   Median :  77.00   Median : 22.35
##  Mean   : 8.208   Mean   :5.415   Mean   :  82.16   Mean   : 36.17
##  3rd Qu.: 9.592   3rd Qu.:6.120   3rd Qu.:  89.00   3rd Qu.: 35.88
##  Max.   :16.410   Max.   :9.670   Max.   :1079.10   Max.   :650.90
##       PROT
##  Min.   :51.00
##  1st Qu.:69.30
##  Median :72.05
##  Mean   :72.05
##  3rd Qu.:75.20
##  Max.   :86.50
```

```
str(d)
```

```
## 'data.frame':    570 obs. of  13 variables:
##  $ Category: chr  "BloodDonor" "BloodDonor" "BloodDonor" "BloodDonor" ...
##  $ Age     : int  32 32 32 32 32 32 32 32 32 32 ...
##  $ Sex     : chr  "m" "m" "m" "m" ...
##  $ ALB     : num  38.5 38.5 46.9 43.2 39.2 41.6 46.3 42.2 50.9 42.4 ...
##  $ ALP     : num  52.5 70.3 74.7 52 74.1 43.3 41.3 41.9 65.5 86.3 ...
##  $ ALT     : num  7.7 18 36.2 30.6 32.6 18.5 17.5 35.8 23.2 20.3 ...
##  $ AST     : num  22.1 24.7 52.6 22.6 24.8 19.7 17.8 31.1 21.2 20 ...
##  $ BIL     : num  7.5 3.9 6.1 18.9 9.6 12.3 8.5 16.1 6.9 35.2 ...
##  $ CHE     : num  6.93 11.17 8.84 7.33 9.15 ...
##  $ CHOL    : num  3.23 4.8 5.2 4.74 4.32 6.05 4.79 4.6 4.1 4.45 ...
##  $ CREA    : num  106 74 86 80 76 111 70 109 83 81 ...
##  $ GGT     : num  12.1 15.6 33.2 33.8 29.9 91 16.9 21.5 13.7 15.9 ...
##  $ PROT    : num  69 76.5 79.3 75.7 68.7 74 74.5 67.1 71.3 69.9 ...
##  - attr(*, "na.action")= 'omit' Named int [1:17] 122 320 330 414 425 434 499 534 535 539 ...
##   ..- attr(*, "names")= chr [1:17] "122" "320" "330" "414" ...
```

Question 1

Regarding the question 1, Principal Components Analysis was applied using the function princomp() in order to compute the PCA scores and then to visualize the first 2 components with color information by Category. Category is our response variable that contains information about the diagnosis: BloodDonor (healthy), Cirrhosis, and Hepatitis. First the variable Sex was transformed to binary variable with 0s and 1s for the gender female and male, respectively. Then the response variable Category was excluded from the data and afterwards the data were scaled with mean 0 and standard deviation 1.

```
d$Sex <- ifelse(d$Sex=="m",1,0) #replace the males with 1 and females with 0
X <- d[,-1]
X <- data.frame(scale(X))
```

The princomp() function was implemented and from the results we took the first 2 components.

```
pca.data <- princomp(X, scores = TRUE)
scores.pca <- pca.data$scores[,1:2]
```

In the below figure, the 2 components are visualized with color information by the response variable category.

```
plot(scores.pca, pch=18, col=as.factor(d$Category), xlab="Component 1", ylab="Component 2")
legend("topleft",pch = 18, col = 1:3,
       legend = levels(factor(d$Category)))
```

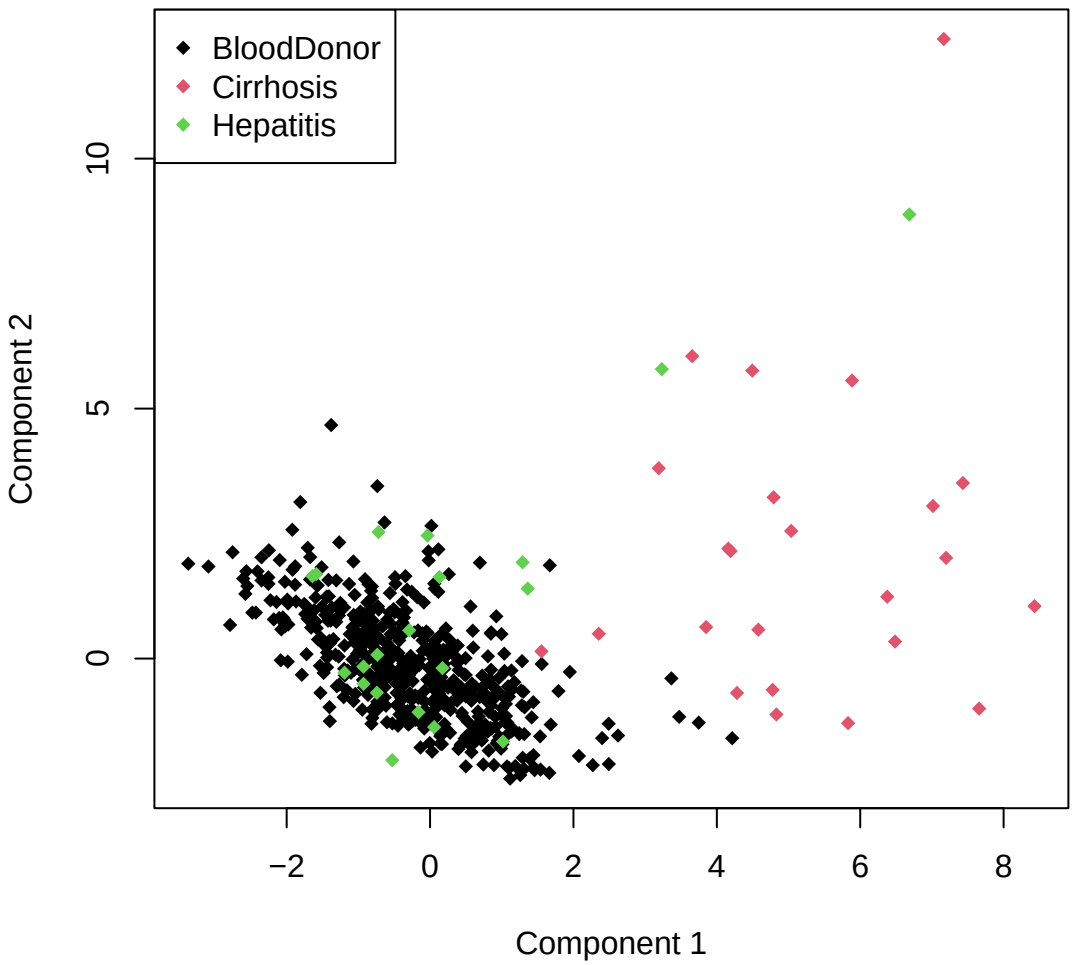

Figure 1: The first 2 Principal Components

According to the plot, almost all the observations are classified as BloodDonor. Despite that, most of the observations that belong to the class Hepatitis are in the cluster of the class BloodDonor. The observations in the class Cirrhosis can be easily observed because they are not involved in the other two classes.

In this classification task, we have a multiple class response variable. The values that this variable (named Category) gets are: BloodDonor, Cirrhosis and Hepatitis. Based on the plot it is observed that the group sizes are heavily imbalanced. More specifically, 24 observations are classified as Cirrhosis, 20 observations are classified as Hepatitis and the rest, which are 526 observations, are classified as BloodDonor. The main problem in the classification using imbalanced data is that the classification model will have poor predictive performance, specifically for the minority class. This is a problem because, in general, the minority class is more significant and, as a result, the issue is more susceptible to errors in classification for the minority class than the majority class. Thus, in order to deal with this imbalance problem, different sampling techniques and evaluation metrics will be needed.

Question 2(a)

The data were split into train and test set.

```
#split the data into train and test
set.seed(12223236)
n <- nrow(d)
train <- sample(1:n,round(n*2/3))
test <- (1:n)[-train]
```

Linear Discriminant Analysis were implemented using the function lda from the library MASS. First the test set error and then the cross validation error was calculated using 5-fold cross validation.

```
library(MASS)
```

The test set error is:

```
#Test set error
model.lda <- lda(Category~., data=d[train,])
pred <- predict(model.lda,d[test,])
TAB <- table(d[test,]$Category,pred$class)
TAB
```

```
##
##              BloodDonor Cirrhosis Hepatitis
##   BloodDonor        179         0         1
##   Cirrhosis           1         6         2
##   Hepatitis           1         0         0
```

```
error.test.lda <- 1-sum(diag(TAB))/sum(TAB)
error.test.lda
```

```
## [1] 0.02631579
```

and the cross validation error is:

```
#Cross validation Error
data <- d
mypredict.lda <- function(object, newdata) {predict(object, newdata = newdata)$class}
library(ipred)
control <- control.errorest(k = 5,random = FALSE)
data$Category <- factor(data$Category)
model.ldacv <- errorest(Category~., data=data[train,],predict= mypredict.lda,
                    model=lda,est.para=control)
model.ldacv$error
```

```
## [1] 0.03157895
```

Both errors are quite low mainly because most of the observations are belong to the class BloodDonor and thus the model classify most of the observations correctly.

Question 2(b)

Quadratic Discriminant Analysis were implemented using the function qda from the library MASS. First the test set error and then the cross validation error was calculated using Leave One Out cross validation. The Leave One Out cross validation was used because of the imbalanced observations. Basically, using k fold cross validation, where k = 5 or k = 10, there are more variables than observations regarding the observations that belong to the minority classes and thus the different covariance matrices can not be calculated.

The test set error is:

```
#Test set error
model.qda <- qda(Category~., data=d[train,])
pred <- predict(model.qda,d[test,])
TAB <- table(d[test,]$Category,pred$class)
error.test.qda <- 1-sum(diag(TAB))/sum(TAB)
error.test.qda
```

```
## [1] 0.01578947
```

and the cross validation error is:

```
#Cross validation Error
mypredict.qda <- function(object, newdata) {predict(object, newdata = newdata)$class}

control <- control.errorest(k = nrow(data[train,]))
data$Category <- factor(data$Category)
model.qdacv <- errorest(Category~., data=data[train,],predict= mypredict.qda,
                        model=qda,est.para=control)
model.qdacv$error
```

```
## [1] 0.04210526
```

Both errors are quite low mainly because most of the observations are belong to the class BloodDonor and thus the model classify most of the observations correctly.

Question 3(a)

As mentioned above, the groups sizes are very different. For that reason, different strategy, regarding the selection of the training and test data, must be applied. Thus, one effective strategy is to split the data using the stratification technique. With this technique the data set split into train and test set maintaining the same proportions of samples in each class as shown in the original data set. In R, in order to stratify the data, the function createDataPartition() from the caret package was used. The data were split into 2/3 training data and 1/3 test data.

```
set.seed(12223236)
train_str <- caret::createDataPartition(d$Category,
                                        times = 1,p=2/3, list=F) #stratified separation
test_str <- (1:n)[-train_str]
```

Then the test error was calculated for the Linear Discrimant Analysis model.

```
model.lda.str <- lda(Category~., data=d[train_str,])
pred.str <- predict(model.lda.str,d[test_str,])
TAB <- table(d[test_str,]$Category,pred.str$class)
error.test.lda.str <- 1-sum(diag(TAB))/sum(TAB)
error.test.lda.str
```

```
## [1] 0.03174603
```

Now, let's compare the error from the question 2(a). The error from the question 2(a) was:

```
pred <- predict(model.lda,d[test,])
error.test.lda
```

```
## [1] 0.02631579
```

it is observed that the error is slightly higher using the stratification technique. However, if the confusion matrices will be compared, other metrics, including the confusion matrix, are different.

The results from the stratification technique are presented below, using the function confusionMatrix() from the caret package.

```
caret::confusionMatrix(pred.str$class,factor(d[test_str,]$Category),mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   BloodDonor Cirrhosis Hepatitis
##   BloodDonor        174         0         3
##   Cirrhosis           0         6         0
##   Hepatitis           1         2         3
##
## Overall Statistics
##
##                Accuracy : 0.9683
##                  95% CI : (0.9322, 0.9883)
##     No Information Rate : 0.9259
##     P-Value [Acc > NIR] : 0.01174
##
##                   Kappa : 0.7568
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: BloodDonor Class: Cirrhosis Class: Hepatitis
## Sensitivity                     0.9943          0.75000          0.50000
## Specificity                     0.7857          1.00000          0.98361
## Pos Pred Value                  0.9831          1.00000          0.50000
## Neg Pred Value                  0.9167          0.98907          0.98361
## Precision                       0.9831          1.00000          0.50000
## Recall                          0.9943          0.75000          0.50000
## F1                              0.9886          0.85714          0.50000
```

```
## Prevalence                     0.9259          0.04233          0.03175
## Detection Rate                 0.9206          0.03175          0.01587
## Detection Prevalence           0.9365          0.03175          0.03175
## Balanced Accuracy              0.8900          0.87500          0.74180
```

The results from the random sampling technique are presented below.

```
caret::confusionMatrix(factor(pred$class),factor(d[test,]$Category),mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   BloodDonor Cirrhosis Hepatitis
##   BloodDonor        179         1         1
##   Cirrhosis           0         6         0
##   Hepatitis           1         2         0
##
## Overall Statistics
##
##                Accuracy : 0.9737
##                  95% CI : (0.9397, 0.9914)
##     No Information Rate : 0.9474
##     P-Value [Acc > NIR] : 0.06209
##
##                   Kappa : 0.7257
##
##  Mcnemar's Test P-Value : 0.39163
##
## Statistics by Class:
##
##                      Class: BloodDonor Class: Cirrhosis Class: Hepatitis
## Sensitivity                     0.9944          0.66667         0.000000
## Specificity                     0.8000          1.00000         0.984127
## Pos Pred Value                  0.9890          1.00000         0.000000
## Neg Pred Value                  0.8889          0.98370         0.994652
## Precision                       0.9890          1.00000         0.000000
## Recall                          0.9944          0.66667         0.000000
## F1                              0.9917          0.80000              NaN
## Prevalence                      0.9474          0.04737         0.005263
## Detection Rate                  0.9421          0.03158         0.000000
## Detection Prevalence            0.9526          0.03158         0.015789
## Balanced Accuracy               0.8972          0.83333         0.492063
```

Based on the two results, we can focus our interest on the metric Balanced Accuracy. Balanced Accuracy is the arithmetic mean of sensitivity and specificity (Sensitivity= TP / (TP + FN), Specificity =TN / (TN + FP)) and can be used when dealing with imbalanced data. For each class, the balanced accuracy was calculated. It is observed that the balanced accuracy for the classes BloodDonor and Cirrhosis is approximately the same on both strategies. However, the balanced accuracy for the Hepatitis class was increased using the stratification technique and thus it would be more optimal to use the stratified data.

Question 3(b)

Quadratic Discrimant Analysis is unstable for very small sample sizes (compared to the number of variables) and this was observed when the CV-error was calculated. Thus, Principal Component Analysis was

implemented in order to reduce the number of variables. The first 2 components from question 1 were used and then the test error and the CV-error were calculated from the Quadratic Discrimant Analysis. Also, regarding the split of the data, the stratification technique was used.

The test error is:

```
scores <- data.frame(pca.data$scores[,1:2])
pca.qda <- cbind(scores, Category=d$Category)

model.qda <- qda(Category~., data=pca.qda[train_str,])
pred <- predict(model.qda,pca.qda[test_str,])
TAB <- table(pca.qda[test_str,]$Category,pred$class)
error.test.qda <- 1-sum(diag(TAB))/sum(TAB)
error.test.qda
```

```
## [1] 0.05291005
```

and the Leave One Out CV-error is:

```
#Cross validation Error
mypredict.qda <- function(object, newdata) {predict(object, newdata = newdata)$class}

control <- control.errorest(k = nrow(pca.qda[train,]))
pca.qda$Category <- factor(pca.qda$Category)
model.qdacv <- errorest(Category~., data=pca.qda[train,],predict= mypredict.qda,
                        model=qda,est.para=control)
model.qdacv$error
```

```
## [1] 0.06315789
```

Question 3(c)

In a classification task, there are different methods for feature importance. The methods applied for feature selection/importance are three: Filtering, Wrapper and Embedded methods. The filtering methods are applied before the model is created. More specifically, from the initial data set the relation of each attribute with the target class is compared, evaluated and depending on the final result, more efficient ones are selected (examples: Correlation, ANOVA, Information Gain). Regarding the Wrapper methods, The specific methods select the best attributes based on the performance of the model. It is obvious that more computing power is required compared to filtering methods, because in order for feature selection to take place, the model must first be created (examples: Forward Selection). Finally, in Embedded methods the feature selection takes place while the model is being trained. An example of such a method is decision trees and random forest for classification tasks and Lasso and Ridge regression for regression tasks. Embedded methods are a combination of the above methods in terms of computing power and efficiency.

Below, it is presented the feature importance based on the information gain.

```
#Information gain

#Sys.setenv(JAVA_HOME='C:\\Program Files\\Java\\jre1.8.0_351')
library(rJava)
library(FSelector)
data.ig <- information.gain(Category~., d, unit ="log2")
data.ig
```

```
##      attr_importance
## Age       0.02601655
## Sex       0.00000000
## ALB       0.12725995
## ALP       0.13775743
## ALT       0.18760579
## AST       0.22299705
## BIL       0.10439055
## CHE       0.17334517
## CHOL      0.07958711
## CREA      0.04312773
## GGT       0.11240701
## PROT      0.02933078
```

According to the table, the features AST, ALT and CHE are important, based on, of course, the information gain.