Código Vowel

Luis Balderas Ruiz

```r
############################################################
# INTRODUCCIÓN A LA CIENCIA DE DATOS
# Autor: Luis Balderas Ruiz
# EDA+Clasificación
# Dataset: vowel
############################################################

# Cálculo de binwidth óptimo para un histograma
binwd = function(data){
  size = length(data)
  dt = sd(data)
  cr = size^(1/3)
  return(1/(cr)*dt*3.49)
}

# Función para crear la matriz de correlación más bonita
flattenCorrMatrix <- function(cormat, pmat) {
  ut <- upper.tri(cormat)
  data.frame(
    row = rownames(cormat)[row(cormat)[ut]],
    column = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut],
    p = pmat[ut]
  )
}

# Lectura del fichero de datos para proceder al EDA
vowel = read.csv("./data/vowel/vowel.dat",header=FALSE, comment.char="@")
colnames(vowel) = c("TT","SpeakerNumber","Sex",
                    "F0","F1","F2","F3","F4","F5","F6","F7","F8","F9","Class")

# Estudiemos la estructura del conjunto
str(vowel)

# Borramos la variable TT porque no considero necesario dividir entre training y test para el EDA
vowel$TT = NULL

# Convierto SpeakerNumber y Sex a factores
vowel$Sex = factor(vowel$Sex, levels = c(0,1),
                   labels = c("Masculino","Femenino"))
vowel$SpeakerNumber = factor(vowel$SpeakerNumber)

# Resumen estadístico de cada variable
summary(vowel)
sapply(vowel[,3:12],sd)

# Búsqueda visual de correlaciones entre las variables en el dataset completo m
plot(vowel[,3:12])
```

```r
# Separación por sexos
library(tidyverse)
hombres = vowel %>% filter(vowel$Sex == "Masculino")
mujeres = vowel %>% filter(vowel$Sex == "Femenino")

# Búsqueda visual de correlaciones por sexos
plot(hombres[,3:12])
plot(mujeres[,3:12])

#####################################################################333
# Distribución de las variables numéricas. Skewness
library(ggplot2)
library(e1071)
dist_sexo = ggplot(vowel,aes(x=Sex, fill=SpeakerNumber))
+ geom_bar(alpha=1/3) +theme(legend.position = "top") + labs(title="Distribución por sexos")
dist_sexo

distf0 = ggplot(vowel,aes(x=F0, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F0)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F0")
distf0
# positiva
skewness(vowel$F0)
kurtosis(vowel$F0)

distf1 = ggplot(vowel,aes(x=F1, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F1)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F1")
distf1
#negativa
skewness(vowel$F1)
kurtosis(vowel$F1)

distf2 = ggplot(vowel,aes(x=F2, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F2)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F2")
distf2
# positiva
skewness(vowel$F2)
kurtosis(vowel$F2)

distf3 = ggplot(vowel,aes(x=F3, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F3)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F3")
distf3
# positiva
skewness(vowel$F3)
kurtosis(vowel$F3)

distf4 = ggplot(vowel,aes(x=F4, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F4)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F4")
distf4
```

```r
# positiva
skewness(vowel$F4)
kurtosis(vowel$F4)

distf5 = ggplot(vowel,aes(x=F5, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F5)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F5")
distf5
# positiva
skewness(vowel$F5)
kurtosis(vowel$F5)

distf6 = ggplot(vowel,aes(x=F6, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F6)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F6")
distf6
# negativa
skewness(vowel$F6)
kurtosis(vowel$F6)

distf7 = ggplot(vowel,aes(x=F7, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F7)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F7")
distf7
# positiva
skewness(vowel$F7)
kurtosis(vowel$F7)

distf8 = ggplot(vowel,aes(x=F8, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F8)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F8")
distf8
# positiva
skewness(vowel$F8)
kurtosis(vowel$F8)

distf9 = ggplot(vowel,aes(x=F9, fill=Sex))
+ geom_histogram(binwidth = binwd(vowel$F9)) + theme(legend.position = "right")
+ labs(title="Distribución de la variable F9")
distf9
# positiva
skewness(vowel$F9)
kurtosis(vowel$F9)
#########################################################################################################

#########################################################################################################
# BOXPLOTS
# bps_i --> Boxplot separando por Sexo (y con colores cada interlocutor)
# bpsn_i ---> Boxplot separando por interlocutores (y con colores el sexo)
bps0 = ggplot(vowel, aes(x=Sex, y=F0, fill = SpeakerNumber))
              + geom_boxplot(outlier.colour="red",
               outlier.shape=8,outlier.size=4)
bps0
```

```
bpsn0 = ggplot(vowel, aes(x = SpeakerNumber, y = F0, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn0

bps1 = ggplot(vowel, aes(x=Sex, y=F1, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps1
bpsn1 = ggplot(vowel, aes(x = SpeakerNumber, y = F1, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn1

bps2 = ggplot(vowel, aes(x=Sex, y=F2, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps2
bpsn2 = ggplot(vowel, aes(x = SpeakerNumber, y = F2, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn2

bps3 = ggplot(vowel, aes(x=Sex, y=F3, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps3
bpsn3 = ggplot(vowel, aes(x = SpeakerNumber, y = F3, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn3

bps4 = ggplot(vowel, aes(x=Sex, y=F4, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps4
bpsn4 = ggplot(vowel, aes(x = SpeakerNumber, y = F4, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn4

bps5 = ggplot(vowel, aes(x=Sex, y=F5, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps5
bpsn5 = ggplot(vowel, aes(x = SpeakerNumber, y = F5, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn5

bps6 = ggplot(vowel, aes(x=Sex, y=F6, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps6
bpsn6 = ggplot(vowel, aes(x = SpeakerNumber, y = F6, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn6

bps7 = ggplot(vowel, aes(x=Sex, y=F7, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps7
bpsn7 = ggplot(vowel, aes(x = SpeakerNumber, y = F7, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn7
```

```r
bps8 = ggplot(vowel, aes(x=Sex, y=F8, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps8
bpsn8= ggplot(vowel, aes(x = SpeakerNumber, y = F8, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn8

bps9 = ggplot(vowel, aes(x=Sex, y=F9, fill = SpeakerNumber))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bps9
bpsn9 = ggplot(vowel, aes(x = SpeakerNumber, y = F9, fill = Sex))
+ geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
bpsn9

##############################################################################################################

#################################################################

# Test de Shapiro-Wilks vs Kolmogorov-Smirnov (Corrección de Lillie)
library(nortest)
library(car)
# Variable F0.
# Shapiro: El bajo p-valor (9.807e-5) nos hace rechazar la hipótesis de normalidad
shapiro.test(vowel$F0)
# Lillie p-valor 0.000719 Rechazamos la normalidad
lillie.test(vowel$F0)
qqPlot(vowel$F0)

# Variable F1.
# Shapiro:pvalue (0.0312) rechaza normalidad
shapiro.test(vowel$F1)
# Lillie: pvalue 0.2628. No podemos rechazar la hipótesis de normalidad.
lillie.test(vowel$F1)
# Comprobamos que es bastante próxima a la normal con un qqPlot
qqPlot(vowel$F1)
ggplot(vowel, aes(x=F1)) + geom_histogram(aes(y=..density..),binwidth = binwd(vowel$F1))
+ stat_function(fun=dnorm, args=list(mean=mean(vowel$F1),sd=sd(vowel$F1)))

# Variable F2.
# Shapiro:El bajo p-valor (4.245e-5) nos hace rechazar la hipótesis de normalidad
shapiro.test(vowel$F2)
#Lillie: p-value = 0.001264 rechazamos la normalidad
lillie.test(vowel$F2)

# Variable F3.
# Shapiro:El bajo p-valor (4.324e-9) nos hace rechazar la hipótesis de normalidad
shapiro.test(vowel$F3)
qqPlot(vowel$F3)
# Lillie: p-value = 5.169e-07. Rechazamos la normalidad
lillie.test(vowel$F3)

# Variable F4.
# Shapiro: p-valor (0.07073) mayor que 0.05, por lo que no podemos rechazar la hipótesis de normalidad
```

```r
shapiro.test(vowel$F4)
qqPlot(vowel$F4)
# Lillie: p-value = 0.08258. No podemos rechazar la hipótesis de normalidad
lillie.test(vowel$F4)

# Variable F5.
#Shapiro: El bajo p-valor (5.435e-08) nos hace rechazar la hipótesis de normalidad
shapiro.test(vowel$F5)
# Lillie:p-value = 1.921e-08. Rechazamos la hipótesis de normalidad
lillie.test(vowel$F5)

# Variable F6.
# Shapiro:pvalue (0.0225)<0.05, rechazamos hipótesis de normalidad
shapiro.test(vowel$F6)
# Lillie: pvalue  0.09071, no podemos rechazar la hipótesis de normalidad
lillie.test(vowel$F6)
qqPlot(vowel$F6)


# Variable F7.
# Shapiro: pvalue (0.0005086)<0.05, rechazamos hipótesis de normalidad
shapiro.test(vowel$F7)
# Lillie:p-value = 0.0002122. Rechazamos la hipótesis de normalidad
lillie.test(vowel$F7)

# Variable F8.
# Shapiro:pvalue (0.001437)<0.05, rechazamos hipótesis de normalidad
shapiro.test(vowel$F8)
# Lillie: p-value = 0.1505. No podemos rechazar la hipótesis de normalidad
lillie.test(vowel$F8)
qqPlot(vowel$F8)

# Variable F9.
# Shapiro: pvalue (2.208e-12)<0.05, rechazamos hipótesis de normalidad
shapiro.test(vowel$F9)
# Lillie: p-value = 7.729e-14. Rechazamos la hipótesis de normalidad
lillie.test(vowel$F9)

## Por sexos

# Hombres

# pvalue (6.965e-05) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F0)
lillie.test(hombres$F0)
# pvalue (0.0002435) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F1)
lillie.test(hombres$F1)
# pvalue (0.0003589) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F2)
lillie.test(hombres$F2)
# pvalue (1.919e-07) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F3)
```

```r
lillie.test(hombres$F3)
# pvalue (0.07804) > 0.05. Acepto hipótesis de normalidad
shapiro.test(hombres$F4)
lillie.test(hombres$F4)
qqPlot(hombres$F4)
# pvalue (7.718e-09) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F5)
lillie.test(hombres$F5)
# pvalue (3.306e-07) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F6)
lillie.test(hombres$F6) # p value 0.05132. No puedo rechazar la hipótesis según el test de Lillie
qqPlot(hombres$F6)
# pvalue (0.009972) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F7)
lillie.test(hombres$F7)
# pvalue (0.00119) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F8)
lillie.test(hombres$F8)
# pvalue (1.466e-15) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(hombres$F9)
lillie.test(hombres$F9)

# Mujeres
# pvalue (0.005136) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(mujeres$F0)
lillie.test(mujeres$F0)
# pvalue (0.02526) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(mujeres$F1)
lillie.test(mujeres$F1)
# pvalue (1.284e-05) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(mujeres$F2)
lillie.test(mujeres$F2)
# pvalue (3.099e-07) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(mujeres$F3)
lillie.test(mujeres$F3)
# pvalue (0.1163) > 0.05. No puedo rechazar la hipótesis de normalidad
shapiro.test(mujeres$F4)
lillie.test(mujeres$F4) #pvalue 0.4692. No puedo rechazar la hipótesis de normalidad
qqPlot(mujeres$F4)
# pvalue (0.04365) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(mujeres$F5)
lillie.test(mujeres$F5) # pvalue 0.08174 > 0.05. No podemos rechazar la hipótesis de normalidad
qqPlot(mujeres$F5)
# pvalue (0.09697) > 0.05. No podemos rechazar la hipótesis de normalidad
shapiro.test(mujeres$F6)
lillie.test(mujeres$F6) # pvalue 0.26. No puedo rechazar la hipótesis de normalidad
qqPlot(mujeres$F6)
# pvalue (8.115e-07) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(mujeres$F7)
lillie.test(mujeres$F7)
# pvalue (0.003301) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(mujeres$F8)
lillie.test(mujeres$F8) # pvalue 0.1125. No puedo rechazar la hipótesis de normalidad
```

```r
qqPlot(mujeres$F8)
# pvalue (9.116e-08) < 0.05. Rechazo hipótesis de normalidad
shapiro.test(mujeres$F9)
lillie.test(mujeres$F9)
######################################################################

######################################################################
# Correlaciones entre variables
plot(vowel[,3:12])
plot(hombres[,3:12])
plot(mujeres[,3:12])
install.packages("corrplot")
library(corrplot)
corrplot(vowel[,3:12], type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)

install.packages("PerformanceAnalytics")
library("PerformanceAnalytics")
chart.Correlation(vowel[,3:12], histogram=TRUE, pch=19)
chart.Correlation(hombres[,3:12],histrogram=TRUE, pch=19)
chart.Correlation(mujeres[,3:12],histrogram=TRUE, pch=19)

library(Hmisc)

res2<-rcorr(as.matrix(vowel[,3:12]))
corrmat = flattenCorrMatrix(res2$r, res2$P)
# Insignificant correlation are crossed
corrplot(res2$r, type="upper", order="hclust",
         p.mat = res2$P, sig.level = 0.01, insig = "blank")

res2h = rcorr(as.matrix(hombres[,3:12]))
corrmath = flattenCorrMatrix(res2h$r, res2h$P)
# Insignificant correlation are crossed
corrplot(res2h$r, type="upper", order="hclust",
         p.mat = res2h$P, sig.level = 0.01, insig = "blank")

res2m =  rcorr(as.matrix(mujeres[,3:12]))
corrmath = flattenCorrMatrix(res2m$r, res2m$P)
# Insignificant correlation are crossed
corrplot(res2m$r, type="upper", order="hclust",
         p.mat = res2m$P, sig.level = 0.01, insig = "blank")

######################################################################

######################################################################
# BOXPLOT POR CLASES
# vowel
boxplot(vowel$F0~vowel$Class,data=vowel)
boxplot(vowel$F1~vowel$Class,data=vowel)
boxplot(vowel$F2~vowel$Class,data=vowel)
boxplot(vowel$F3~vowel$Class,data=vowel)
boxplot(vowel$F4~vowel$Class,data=vowel)
boxplot(vowel$F5~vowel$Class,data=vowel)
```

```r
boxplot(vowel$F6~vowel$Class,data=vowel)
boxplot(vowel$F7~vowel$Class,data=vowel)
boxplot(vowel$F8~vowel$Class,data=vowel)
boxplot(vowel$F9~vowel$Class,data=vowel)
# hombres
boxplot(hombres$F0~hombres$Class,data=hombres)
boxplot(hombres$F1~hombres$Class,data=hombres)
boxplot(hombres$F2~hombres$Class,data=hombres)
boxplot(hombres$F3~hombres$Class,data=hombres)
boxplot(hombres$F4~hombres$Class,data=hombres)
boxplot(hombres$F5~hombres$Class,data=hombres)
boxplot(hombres$F6~hombres$Class,data=hombres)
boxplot(hombres$F7~hombres$Class,data=hombres)
boxplot(hombres$F8~hombres$Class,data=hombres)
boxplot(hombres$F9~hombres$Class,data=hombres)
# mujeres
boxplot(mujeres$F0~mujeres$Class,data=mujeres)
boxplot(mujeres$F1~mujeres$Class,data=mujeres)
boxplot(mujeres$F2~mujeres$Class,data=mujeres)
boxplot(mujeres$F3~mujeres$Class,data=mujeres)
boxplot(mujeres$F4~mujeres$Class,data=mujeres)
boxplot(mujeres$F5~mujeres$Class,data=mujeres)
boxplot(mujeres$F6~mujeres$Class,data=mujeres)
boxplot(mujeres$F7~mujeres$Class,data=mujeres)
boxplot(mujeres$F8~mujeres$Class,data=mujeres)
boxplot(mujeres$F9~mujeres$Class,data=mujeres)

######################################################################

######################################################################
# TRANSFORMACIONES
install.packages("scales")
library("scales")

vowel$logF2 = log1p(rescale(vowel$F2))
# pvalue 0.4792. No rechazamos la hipótesis de normalidad
lillie.test(vowel$logF2)
qqPlot(vowel$logF2)
ggplot(vowel, aes(x=logF2)) + geom_histogram(aes(y=..density..),
binwidth = binwd(vowel$logF2))
+ stat_function(fun=dnorm, args=list(mean=mean(vowel$logF2),sd=sd(vowel$logF2)))




######################################################################
######################################################################
# CLASIFICACIÓN

# C.1 kNN
library(tidyverse)
library(caret)
library(scales)
```

```r
library(class)
library(plyr)
library(ggplot2)
vowel = read.csv("./data/vowel/vowel.dat",header=FALSE, comment.char="@")
colnames(vowel) = c("TT","SpeakerNumber","Sex",
                    "F0","F1","F2","F3","F4","F5","F6","F7","F8","F9","Class")
for (i in 4:13){
  vowel[,i] = rescale(vowel[,i])
}
accuracy_vowel = c()
for (i in seq(1,15,2)){
  train.index <- createDataPartition(vowel$Class, p = .7, list = FALSE)
  v.train <- vowel[ train.index,]
  v.test  <- vowel[-train.index,]
  pr <- knn(train=v.train,test=v.test,cl=v.train$Class,k=i)
  acc1 = sum(pr==v.test$Class)/nrow(v.test)
  accuracy_vowel = append(accuracy_vowel,acc1)
}

plot(x=seq(1,15,2),y=accuracy_vowel,xlab="Valores de k",
ylab="Accuracy sobre vowel", main="Resultado kNN",ylim=c(0,1),type='o',col='blue')

## REGIONES CON KNN
train.index <- createDataPartition(vowel$Class, p = .7, list = FALSE)
v.train <- vowel[ train.index,]
v.test  <- vowel[-train.index,]
pr <- knn(train=v.train,test=v.test,cl=v.train$Class,k=3)
acc1 = sum(pr==v.test$Class)/nrow(v.test)
accuracy_vowel = append(accuracy_vowel,acc1)
plot.df = data.frame(v.test, predicted = pr)
plot.df1 = data.frame(x = plot.df$F0,
                      y = plot.df$F1,
                      predicted = plot.df$predicted)

find_hull = function(df) df[chull(df$x, df$y), ]
boundary = ddply(plot.df1, .variables = "predicted", .fun = find_hull)

ggplot(plot.df, aes(F0, F1, color = predicted, fill = predicted)) +
  geom_point(size = 5) +
  geom_polygon(data = boundary, aes(x,y), alpha = 0.5)

ggplot(plot.df, aes(F2, F6, color = predicted, fill = predicted)) +
  geom_point(size = 5) +
  geom_polygon(data = boundary, aes(x,y), alpha = 0.5)
##############################
# Prueba por sexos

hombres = vowel %>% filter(Sex==0)
mujeres = vowel %>% filter(Sex==1)

acc_hombres = c()
for (i in c(1,3,5,7,9)){
  hombres.index = createDataPartition(hombres$Class, p = .7, list = FALSE)
```

```
  h.train <- hombres[hombres.index,]
  h.test = hombres[-hombres.index,]
  modelo_hombres = knn(train=h.train,test = h.test, cl=h.train$Class, k=3)
  acc_h = sum(modelo_hombres==h.test$Class)/nrow(h.test)
  acc_hombres = append(acc_hombres,acc_h)
}

plot(x=c(1,3,5,7,9),y=acc_hombres,xlab="Valores de k",
     ylab="Accuracy sobre hombres",ylim=c(0,1), main="Resultado kNN",type='o',col='blue')

acc_mujeres = c()
for(i in c(1,3,5,7,9)){
  mujeres.index = createDataPartition(mujeres$Class, p = .7, list = FALSE)
  m.train <- mujeres[mujeres.index,]
  m.test = mujeres[-mujeres.index,]
  modelo_mujeres = knn(train=m.train,test = m.test, cl=m.train$Class, k=3)
  acc_m = sum(modelo_mujeres==m.test$Class)/nrow(m.test)
  acc_mujeres = append(acc_mujeres,acc_m)
}

plot(x=c(1,3,5,7,9),y=acc_mujeres,xlab="Valores de k",
     ylab="Accuracy sobre mujeres",ylim=c(0,1), main="Resultado kNN",type='o',col='blue')

# Cross-validation

nombre <- "./data/vowel/vowel"
run_knn_fold <- function(i, x, tt = "test",k_par) {
  file <- paste(x, "-10-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-10-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else {
    test <- x_tst
  }
  pr <- knn(train=x_tra,test=test,cl=x_tra$Y,k=k_par)
  return(sum(pr==test$Y)/nrow(test))
}
kfolds_list_train = c()
kfolds_list_test = c()
for(i in seq(1,15,2)){
  acc_mean_train = mean(sapply(1:10,run_knn_fold,nombre,"train",i))
  acc_mean_test = mean(sapply(1:10,run_knn_fold,nombre,"test",i))
  kfolds_list_train = append(kfolds_list_train,acc_mean_train)
  kfolds_list_test = append(kfolds_list_test,acc_mean_test)
}
```

```r
plot(x=seq(1,15,2),y=kfolds_list_train,xlab="Valores de k",
     ylab="Accuracy sobre vowel en train",ylim=c(0,1), main="Resultado kNN 10-kfoldCross validation",
     type='o',col='blue')
plot(x=seq(1,15,2),y=kfolds_list_test,xlab="Valores de k",
     ylab="Accuracy sobre vowel en test",ylim=c(0,1), main="Resultado kNN 10-kfoldCross validation",
     type='o',col='blue')

resultados_knn3_tr = sapply(1:10,run_knn_fold,nombre,"train",3)
resultados_knn1_tr = sapply(1:10,run_knn_fold,nombre,"train",1)
resultados_knn3_test = sapply(1:10,run_knn_fold,nombre,"test",3)
resultados_knn1_test = sapply(1:10,run_knn_fold,nombre,"test",1)
acc_mean_test_knn = mean(resultados_knn3_test)



##############################################################################################
## C.2
# LDA
library(MASS)
# checks (ya hecho en EDA)
sapply(vowel[,3:13],shapiro.test)
sapply(vowel[,4:13],var)

train.index <- createDataPartition(vowel$Class, p = .7, list = FALSE)
v.train <- vowel[ train.index,]
v.test  <- vowel[-train.index,]

v.train$TT= NULL
v.train$SpeakerNumber = NULL
v.test$TT = NULL
v.test$SpeakerNumber = NULL
v.train$Class = as.factor(v.train$Class)
v.test$Class = as.factor(v.test$Class)
vowel.lda.predict <- train(Class ~ ., method = "lda", data = v.train)
confusionMatrix(v.test$Class, predict(vowel.lda.predict, v.test))

##############################
# Prueba por sexos

hombres = vowel %>% filter(Sex==0)
mujeres = vowel %>% filter(Sex==1)

# para hombres
train.index <- createDataPartition(hombres$Class, p = .7, list = FALSE)
v.train <- hombres[ train.index,]
v.test  <- hombres[-train.index,]

v.train$TT= NULL
v.train$SpeakerNumber = NULL
v.train$Sex = NULL
v.test$TT = NULL
v.test$SpeakerNumber = NULL
v.test$Sex = NULL
```

```
v.train$Class = as.factor(v.train$Class)
v.test$Class = as.factor(v.test$Class)
hombres.lda.predict <- train(Class ~ ., method = "lda", data = v.train)
confusionMatrix(v.test$Class, predict(hombres.lda.predict, v.test))

# para mujeres
train.index <- createDataPartition(mujeres$Class, p = .7, list = FALSE)
v.train <- mujeres[ train.index,]
v.test  <- mujeres[-train.index,]

v.train$TT= NULL
v.train$SpeakerNumber = NULL
v.train$Sex = NULL
v.test$TT = NULL
v.test$SpeakerNumber = NULL
v.test$Sex = NULL
v.train$Class = as.factor(v.train$Class)
v.test$Class = as.factor(v.test$Class)
mujeres.lda.predict <- train(Class ~ ., method = "lda", data = v.train)
confusionMatrix(v.test$Class, predict(mujeres.lda.predict, v.test))

install.packages("klaR")
library(klaR)
X11(width=15, height=15)
partimat(Class ~F0+F1+F2+F3, data = v.test, method = "lda")

# Cross-validation

nombre <- "./data/vowel/vowel"
run_lda_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-10-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-10-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
  }
  else {
    test <- x_tst
  }
  x_tra$X1 = NULL
  x_tra$X2 = NULL
  test$X1 = NULL
  test$X2 =  NULL
  x_tra$Y = as.factor(x_tra$Y)
  test$Y = as.factor(test$Y)
  modelo <- train(Y ~ ., method = "lda", data = x_tra)
  pr = predict(modelo,test)
```

```
  return(sum(pr==test$Y)/nrow(test))
}
acc_mean_train_lda = mean(sapply(1:10,run_lda_fold,nombre,"train"))
resultados_lda_tr = sapply(1:10,run_lda_fold,nombre,"train")
resultados_lda_test = sapply(1:10,run_lda_fold,nombre,"test")
acc_mean_test_lda = mean(resultados_lda_test)


##############################################################################################
# C.3
# QDA

# CHECKS: Misma varianza entre elementos de la misma clase

for(i in 0:10){
  aux = vowel %>% filter(Class == i)
  print(paste("Clase ",i))
  print(sapply(aux[,4:13],var))
}


train.index <- createDataPartition(vowel$Class, p = .7, list = FALSE)
v.train <- vowel[ train.index,]
v.test  <- vowel[-train.index,]

v.train$TT= NULL
v.train$SpeakerNumber = NULL
v.test$TT = NULL
v.test$SpeakerNumber = NULL
v.train$Class = as.factor(v.train$Class)
v.test$Class = as.factor(v.test$Class)
vowel.lda.predict <- train(Class ~ ., method = "qda", data = v.train)
confusionMatrix(v.test$Class, predict(vowel.lda.predict, v.test))

library(klaR)
X11(width=15, height=15)
partimat(Class ~F0+F1+F2+F3, data = v.test, method = "qda")

# Cross-validation

nombre <- "./data/vowel/vowel"
run_qda_fold <- function(i, x, tt = "test") {
  file <- paste(x, "-10-", i, "tra.dat", sep="")
  x_tra <- read.csv(file, comment.char="@", header=FALSE)
  file <- paste(x, "-10-", i, "tst.dat", sep="")
  x_tst <- read.csv(file, comment.char="@", header=FALSE)
  In <- length(names(x_tra)) - 1
  names(x_tra)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tra)[In+1] <- "Y"
  names(x_tst)[1:In] <- paste ("X", 1:In, sep="")
  names(x_tst)[In+1] <- "Y"
  if (tt == "train") {
    test <- x_tra
```

```
  }
  else {
    test <- x_tst
  }
  x_tra$X1 = NULL
  x_tra$X2 = NULL
  test$X1 = NULL
  test$X2 =  NULL
  x_tra$Y = as.factor(x_tra$Y)
  test$Y = as.factor(test$Y)
  modelo <- train(Y ~ ., method = "qda", data = x_tra)
  pr = predict(modelo,test)
  return(sum(pr==test$Y)/nrow(test))
}

acc_mean_train_qda = mean(sapply(1:10,run_qda_fold,nombre,"train"))
resultados_qda_tr = sapply(1:10,run_qda_fold,nombre,"train")
resultados_qda_test = sapply(1:10,run_qda_fold,nombre,"test")
acc_mean_test_qda = mean(resultados_qda_test)

##############################################################################
# C.4
# Comparación de algoritmos
# TEST
tabla = cbind(resultados_knn1_test,resultados_knn3_test
              ,resultados_lda_test,resultados_qda_test)
tabla_resultados = as.data.frame(tabla,col.names=c("1NN", "3NN","LDA","QDA"))

# COMPARACIONES CON PARES: TEST DE WILCOXON
## 1) 1NN -  LDA


wilc_1_3 = cbind(tabla_resultados[,1],tabla_resultados[,3])
colnames(wilc_1_3) <- c(colnames(tabla_resultados)[1], colnames(tabla_resultados)[3])
head(wilc_1_3)

K1NNvsLDAtst = wilcox.test(wilc_1_3[,1],wilc_1_3[,2]
                           ,alternative = "two.sided",paired=TRUE)
Rmas = K1NNvsLDAtst$statistic
pvalue = K1NNvsLDAtst$p.value
K1NNvsLDAtst = wilcox.test(wilc_1_3[,2],wilc_1_3[,1]
                           ,alternative = "two.sided",paired=TRUE)
Rmenos = K1NNvsLDAtst$statistic
Rmas
Rmenos
pvalue

## 2) 1NN - QDA


wilc_1_4 = cbind(tabla_resultados[,1],tabla_resultados[,4])
colnames(wilc_1_4) <- c(colnames(tabla_resultados)[1], colnames(tabla_resultados)[4])
head(wilc_1_4)
```

```r
K1NNvsQDAtst = wilcox.test(wilc_1_4[,1],wilc_1_4[,2],
                           alternative = "two.sided",paired=TRUE)
Rmas = K1NNvsQDAtst$statistic
pvalue = K1NNvsQDAtst$p.value
K1NNvsQDAtst = wilcox.test(wilc_1_4[,2],wilc_1_4[,1],
                           alternative = "two.sided",paired=TRUE)
Rmenos = K1NNvsQDAtst$statistic
Rmas
Rmenos
pvalue

## 3) 3NN- LDA

wilc_2_3 = cbind(tabla_resultados[,2],tabla_resultados[,3])
colnames(wilc_2_3) <- c(colnames(tabla_resultados)[2], colnames(tabla_resultados)[3])
head(wilc_2_3)

K3NNvsLDAtst = wilcox.test(wilc_2_3[,1],wilc_2_3[,2]
                           ,alternative = "two.sided",paired=TRUE)
Rmas = K3NNvsLDAtst$statistic
pvalue = K3NNvsLDAtst$p.value
K3NNvsLDAtst = wilcox.test(wilc_2_3[,2],wilc_2_3[,1]
                           ,alternative = "two.sided",paired=TRUE)
Rmenos = K3NNvsLDAtst$statistic
Rmas
Rmenos
pvalue

## 4) 3NN - QDA

wilc_2_4 = cbind(tabla_resultados[,2],tabla_resultados[,4])
colnames(wilc_2_4) <- c(colnames(tabla_resultados)[2], colnames(tabla_resultados)[4])
head(wilc_2_4)

K3NNvsQDAtst = wilcox.test(wilc_2_4[,1],wilc_2_4[,2]
                           ,alternative = "two.sided",paired=TRUE)
Rmas = K3NNvsQDAtst$statistic
pvalue = K3NNvsQDAtst$p.value
K3NNvsQDAtst = wilcox.test(wilc_2_4[,2],wilc_2_4[,1]
                           ,alternative = "two.sided",paired=TRUE)
Rmenos = K3NNvsQDAtst$statistic
Rmas
Rmenos
pvalue


## 5) LDA-QDA


wilc_3_4 = cbind(tabla_resultados[,3],tabla_resultados[,4])
colnames(wilc_3_4) <- c(colnames(tabla_resultados)[3]
                        , colnames(tabla_resultados)[4])
head(wilc_3_4)
```

```r
LDAvsQDAtst = wilcox.test(wilc_3_4[,1],wilc_3_4[,2]
                          ,alternative = "two.sided",paired=TRUE)
Rmas = LDAvsQDAtst$statistic
pvalue = LDAvsQDAtst$p.value
LDAvsQDAtst = wilcox.test(wilc_3_4[,2],wilc_3_4[,1]
                          ,alternative = "two.sided",paired=TRUE)
Rmenos = LDAvsQDAtst$statistic
Rmas
Rmenos
pvalue

# Comparativa general (Friedman)
test_friedman <- friedman.test(as.matrix(tabla_resultados))
test_friedman

# Post-hoc Holm
tam <- dim(tabla_resultados)
groups <- rep(1:tam[2], each=tam[1])
pairwise.wilcox.test(as.matrix(tabla_resultados),
                     groups, p.adjust = "holm", paired = TRUE)

# TRAIN
tabla = cbind(resultados_knn1_tr,resultados_knn3_tr,resultados_lda_tr,resultados_qda_tr)
tabla_resultados_tr = as.data.frame(tabla,col.names=c("1NN", "3NN","LDA","QDA"))

# COMPARACIONES CON PARES: TEST DE WILCOXON
## 1) 1NN -  LDA


wilc_1_3 = cbind(tabla_resultados_tr[,1],tabla_resultados_tr[,3])
colnames(wilc_1_3) <- c(colnames(tabla_resultados_tr)[1]
                        , colnames(tabla_resultados_tr)[3])
head(wilc_1_3)

K1NNvsLDAtst = wilcox.test(wilc_1_3[,1],wilc_1_3[,2],
                           alternative = "two.sided",paired=TRUE)
Rmas = K1NNvsLDAtst$statistic
pvalue = K1NNvsLDAtst$p.value
K1NNvsLDAtst = wilcox.test(wilc_1_3[,2],wilc_1_3[,1],
                           alternative = "two.sided",paired=TRUE)
Rmenos = K1NNvsLDAtst$statistic
Rmas
Rmenos
pvalue

## 2) 1NN - QDA


wilc_1_4 = cbind(tabla_resultados_tr[,1],tabla_resultados_tr[,4])
colnames(wilc_1_4) <- c(colnames(tabla_resultados_tr)[1],
                        colnames(tabla_resultados_tr)[4])
head(wilc_1_4)
```

```
K1NNvsQDAtst = wilcox.test(wilc_1_4[,1],wilc_1_4[,2],
                           alternative = "two.sided",paired=TRUE)
Rmas = K1NNvsQDAtst$statistic
pvalue = K1NNvsQDAtst$p.value
K1NNvsQDAtst = wilcox.test(wilc_1_4[,2],wilc_1_4[,1],
                           alternative = "two.sided",paired=TRUE)
Rmenos = K1NNvsQDAtst$statistic
Rmas
Rmenos
pvalue

## 3) 3NN- LDA

wilc_2_3 = cbind(tabla_resultados_tr[,2],tabla_resultados_tr[,3])
colnames(wilc_2_3) <- c(colnames(tabla_resultados_tr)[2],
                        colnames(tabla_resultados_tr)[3])
head(wilc_2_3)

K3NNvsLDAtst = wilcox.test(wilc_2_3[,1],wilc_2_3[,2],
                           alternative = "two.sided",paired=TRUE)
Rmas = K3NNvsLDAtst$statistic
pvalue = K3NNvsLDAtst$p.value
K3NNvsLDAtst = wilcox.test(wilc_2_3[,2],wilc_2_3[,1],
                           alternative = "two.sided",paired=TRUE)
Rmenos = K3NNvsLDAtst$statistic
Rmas
Rmenos
pvalue

## 4) 3NN - QDA

wilc_2_4 = cbind(tabla_resultados_tr[,2],tabla_resultados_tr[,4])
colnames(wilc_2_4) <- c(colnames(tabla_resultados_tr)[2],
                        colnames(tabla_resultados_tr)[4])
head(wilc_2_4)

K3NNvsQDAtst = wilcox.test(wilc_2_4[,1],wilc_2_4[,2],
                           alternative = "two.sided",paired=TRUE)
Rmas = K3NNvsQDAtst$statistic
pvalue = K3NNvsQDAtst$p.value
K3NNvsQDAtst = wilcox.test(wilc_2_4[,2],wilc_2_4[,1],
                           alternative = "two.sided",paired=TRUE)
Rmenos = K3NNvsQDAtst$statistic
Rmas
Rmenos
pvalue


## 5) LDA-QDA


wilc_3_4 = cbind(tabla_resultados_tr[,3],tabla_resultados_tr[,4])
colnames(wilc_3_4) <- c(colnames(tabla_resultados_tr)[3], colnames(tabla_resultados_tr)[4])
```

```r
head(wilc_3_4)

LDAvsQDAtst = wilcox.test(wilc_3_4[,1],wilc_3_4[,2],
                          alternative = "two.sided",paired=TRUE)
Rmas = LDAvsQDAtst$statistic
pvalue = LDAvsQDAtst$p.value
LDAvsQDAtst = wilcox.test(wilc_3_4[,2],wilc_3_4[,1],
                          alternative = "two.sided",paired=TRUE)
Rmenos = LDAvsQDAtst$statistic
Rmas
Rmenos
pvalue

# Comparativa general (Friedman)
test_friedman_tr <- friedman.test(as.matrix(tabla_resultados_tr))
test_friedman_tr

# Post-hoc Holm
tam <- dim(tabla_resultados_tr)
groups <- rep(1:tam[2], each=tam[1])
pairwise.wilcox.test(as.matrix(tabla_resultados_tr),
                     groups, p.adjust = "holm", paired = TRUE)
```