PracticaSpam

DavidVelasco

2023-01-25

En la detección de Spam se utilizan con frecuencia técnicas de machine learning para mejorar los índices de detección de correos no deseados. En el dataset adjunto, se han seleccionado para cada mensaje una serie de términos clave que suelen aparecer con frecuencia en los mensajes spam. Posteriormente, se ha realizado una codificación vectorial de los correos electrónicos considerando esos términos clave. Para cada correo disponemos de la clasificación por parte de los expertos humanos. Se pide realizar las siguientes tareas:

- Apartado 1) Sustituir un 2% de valores de la matriz de datos por NAs, de manera aleatoria. Imputar dichos valores faltantes y justificar la elección del método utilizado.(1 punto)
- Apartado 2) Realizar un análisis exploratorio de la matriz de datos. Comentar los resultados y utilizar visualizaciones cuando sea necesario. (2 puntos)
- Apartado 3) Eliminar aquellas palabras que tengan una correlación elevada con otras. Calcular el número de documentos en que aparece cada palabra y eliminar aquellas de menor frecuencia. Dibujar un histograma de dichas frecuencias calculadas. Nota: Con la codificación “bag of words” utilizada, el número de documentos en que aparece una palabra se obtiene sumando para cada variable todas las filas. (2 puntos)
- Apartado 4) Proyectar los datos sobre un subespacio de dimensión menor utilizando PCA. ¿ Cuántas componentes principales se deben utilizar para poder visualizar la estructura semántica de los documentos ? Obtener un plot utilizando dichas compoentes principales y comentar si hay estructura de grupo.(2 puntos)
- Apartado 5) Realizar un clustering de los mensajes de spam atendiendo a su contenido semántico. Discutir y comparar el resultado para dos algoritmos diferentes.(2 puntos)
- Apartado 6) Aplicar los mapas autoorganizativos para visualizar la estructura semántica de la colección de documentos utilizada. ¿ Qué ventajas tienen los mapas autoorganizativos con respecto a los algoritmos de clustering utilizados anteriormente ? (1 punto)

```
#Instalación de las librerias necesarias

 libs <-
c("plyr","readr","dplyr","corrplot","psych","ade4","imputeTS","cluster","
```

```
tidyverse","NbClust","factoextra")

  for (i in libs){
    #print(i)
    if(!require(i, character.only = TRUE))
    { install.packages(i, dependencies=TRUE); library(i) }
  }

#Comprobación de que está el *.csv en el directorio

currentDir <- getwd()
list.files(path="../datos")

##  [1] "anemonefish.xls"          "beer2.csv"
##  [3] "DatasetLimpio.xlsx"       "EXAMPLE_DataToClean.xlsx"
##  [5] "nombre_variables.txt"     "output"
##  [7] "religions.csv"            "spam.csv"
##  [9] "spam.xls"                 "student-mat.csv"
## [11] "student-por.csv"          "student.zip"

if (!file.exists("../datos"))
{stop(paste0("Se necesita que el directorio datos esté en:
",currentDir))}


ComprobarInputs <- function(path, dir,file)
{if (!file.exists(paste0(path,"/",dir)))
{stop(paste0("Se necesita que el directorio ", dir, " esté en: ",path))}
  else if (!file.exists(paste0(path, "/",dir,"/", file)))
    {stop(paste0("Se necesita que ", file," esté en: ", path, "/",
dir))}}

parentPath <- dirname(currentDir)

try(ComprobarInputs(parentPath,"datos", "spam.xls"), FALSE)

##Unificacion de los dos ficheros en un dataframe

#Cargo el xls y separo mediante un espacio los registros
dtDatos=read.table("../datos/spam.xls",sep=" ",header=FALSE)

#Cargo los valores del fichero nombre_variables que serán el nombre de
las columnas
dtCabeceraFilas=read_fwf("../datos/nombre_variables.txt",show_col_types =
FALSE)

#Superpongo las datos de las filas por columnas
dtCabeceraColumnas <- as.data.frame(t(dtCabeceraFilas))

#Unifico las dos tablas copiando la fila 1 en las cabecera del dataset
```

```r
final
names(dtDatos)<- dtCabeceraColumnas[1,]

  #Apartado 1)

  #Nrow= 4601 y ncol=58
  #nValores a sustituir=4601*58*0,02

  constante = 0.02
  filas<-nrow(dtDatos)
  columnas<-ncol(dtDatos)-1 #Borro 1 para no poner NA en la ultima
columna
  nvalores <- ceiling( filas* (columnas+1)  * constante)
  #Hay 5338 NA en la matriz

  #Duplico el dataframe para hacer este apartado
  dtDatosNA<-dtDatos

  #Pasos y justificación
  #---------------------

  #1. Recorro mediante un while hasta completar el 2% de los valores del
dataframe
  #2. Obtengo un valor aleatorio entre 1 y el numero de filas y entre 1 y
el numero de columnas
  #3. Compruebo si en esa celda hay un NA.
  #3.1 Si es un NA reiniciamos la iteración del bucle
  #3.2 Si no hay NA, seteamos el NA en esa celda e iteramos el bucle

  #Nota: La funcion sample(x:y,1) obtiene un(1) nº aleatorio entre 'x' e
'y'
  i=1
  while(i<=nvalores){
    valorfila <-sample(1:filas,1)
    valorcol<-sample(1:columnas,1)

    if(is.na(dtDatosNA[valorfila,valorcol])){
    }else{
      dtDatosNA[valorfila,valorcol]=NA
      i<-i+1
    }
  }
  #Compruebo que tengo los 5338 valores en dtDatosNa
  valoresNA <-sum(is.na(dtDatosNA))
  cat(paste("Total valores NA en el dataframe dtDatosNA: ",
valoresNA,sep=" "))
```

```
## Total valores NA en el dataframe dtDatosNA:  5338
```

```r
  #Comprobación por columnas
  data.frame(lapply( lapply( dtDatosNA,is.na),sum))
```

```
##   make address all X3d our over remove internet order mail receive
will people
## 1   88      90  95  86 105   93     98       91    94   82     121
102     86
##   report addresses free business email you credit your font X000 money
hp hpl
## 1     95        91  106      105    82 103     98   89   96   93    88
91 102
##   george X650 lab labs telnet X857 data X415 X85 technology X1999
parts pm
## 1     89   95  86   83     96   94   93   92 111         92    94
68 96
##   direct cs meeting original project  re edu table conference X. X..1
X..2 X..3
## 1    106 94      79       86      95 103  87    94         97 84   92
99  105
##   X..4 X..5 cap_run_length_average cap_run_length_longest
cap_run_length_total
## 1   96   88                    84                     82
108
##   clase
## 1     0
```

```r
  #Ultimo paso: Donde hay NA sustituir valores y justificar el metodo
  #El metodo elegido es na_interpolation de biblioteca imputeTS
  #Es utili cuando los valores NA estan distribuidos de forma aleatoria y
se desea
  #mantener la tendencia general de los datos

  library(imputeTS)
  dtDatosFilled<-na_interpolation(dtDatosNA)

  #Compruebo que no tengo ningun valor NA
  valoresNAfil <-sum(is.na(dtDatosFilled))
  cat(paste("Total valores NA en el dataframe dtDatosFilled: ",
valoresNAfil,sep=" "))
```

```
## Total valores NA en el dataframe dtDatosFilled:  0
```

```r
  #Apartado 2)

  #Para confirmar el spam.info compruebo el tipo de variable para cada
columna
  library(dplyr)
  glimpse(dtDatosFilled)
```

```
## Rows: 4,601
## Columns: 58
```

```
  #Apartado 2)

  summary(dtDatosFilled)   #Resumen del dataset
```

```
##       make           address             all               3d          
##  Min.   :0.0000   Min.   : 0.0000   Min.   :0.0000   Min.   : 0.00000  
##  1st Qu.:0.0000   1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.: 0.00000  
##  Median :0.0000   Median : 0.0000   Median :0.0000   Median : 0.00000  
##  Mean   :0.1048   Mean   : 0.2142   Mean   :0.2803   Mean   : 0.06747  
##  3rd Qu.:0.0000   3rd Qu.: 0.0000   3rd Qu.:0.4200   3rd Qu.: 0.00000  
##  Max.   :4.5400   Max.   :14.2800   Max.   :5.1000   Max.   :42.81000  
##       our              over             remove          internet      
##  Min.   : 0.0000   Min.   :0.00000   Min.   :0.0000   Min.   : 0.0000  
##  1st Qu.: 0.0000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.: 0.0000  
##  Median : 0.0000   Median :0.00000   Median :0.0000   Median : 0.0000  
##  Mean   : 0.3114   Mean   :0.09496   Mean   :0.1128   Mean   : 0.1059  
##  3rd Qu.: 0.3900   3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.: 0.0000  
##  Max.   :10.0000   Max.   :5.88000   Max.   :7.2700   Max.   :11.1100  
##      order             mail            receive            will        
##  Min.   :0.0000   Min.   : 0.000   Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.: 0.000   1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :0.0000   Median : 0.000   Median :0.0000   Median :0.1400  
##  Mean   :0.0911   Mean   : 0.239   Mean   :0.0596   Mean   :0.5396  
##  3rd Qu.:0.0000   3rd Qu.: 0.160   3rd Qu.:0.0000   3rd Qu.:0.8000  
##  Max.   :5.2600   Max.   :18.180   Max.   :2.6100   Max.   :9.6700  
##      people            report          addresses            free        
##  Min.   :0.00000   Min.   : 0.0000   Min.   :0.00000   Min.   : 0.0000  
##  1st Qu.:0.00000   1st Qu.: 0.0000   1st Qu.:0.00000   1st Qu.: 0.0000  
##  Median :0.00000   Median : 0.0000   Median :0.00000   Median : 0.0000  
##  Mean   :0.09438   Mean   : 0.0592   Mean   :0.04927   Mean   : 0.2481  
##  3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd Qu.:0.00000   3rd Qu.: 0.1100  
##  Max.   :5.55000   Max.   :10.0000   Max.   :4.41000   Max.   :20.0000  
##     business          email             you              credit       
##  Min.   :0.000   Min.   :0.0000   Min.   : 0.000   Min.   : 0.00000  
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.: 0.000   1st Qu.: 0.00000  
##  Median :0.000   Median :0.0000   Median : 1.310   Median : 0.00000  
##  Mean   :0.144   Mean   :0.1841   Mean   : 1.666   Mean   : 0.08571  
##  3rd Qu.:0.000   3rd Qu.:0.0000   3rd Qu.: 2.650   3rd Qu.: 0.00000  
##  Max.   :7.140   Max.   :9.0900   Max.   :18.750   Max.   :18.18000  
##       your              font              000             money        
##  Min.   : 0.0000   Min.   : 0.0000   Min.   :0.0000   Min.   : 0.00000  
##  1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.: 0.00000  
##  Median : 0.2300   Median : 0.0000   Median :0.0000   Median : 0.00000  
##  Mean   : 0.8111   Mean   : 0.1201   Mean   :0.1029   Mean   : 0.09474  
##  3rd Qu.: 1.2800   3rd Qu.: 0.0000   3rd Qu.:0.0000   3rd Qu.: 0.00000  
##  Max.   :11.1100   Max.   :17.1000   Max.   :5.4500   Max.   :12.50000  
##        hp                hpl              george            650         
##  Min.   : 0.0000   Min.   : 0.0000   Min.   : 0.000   Min.   :0.000  
##  1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.000   1st Qu.:0.000  
##  Median : 0.0000   Median : 0.0000   Median : 0.000   Median :0.000
```

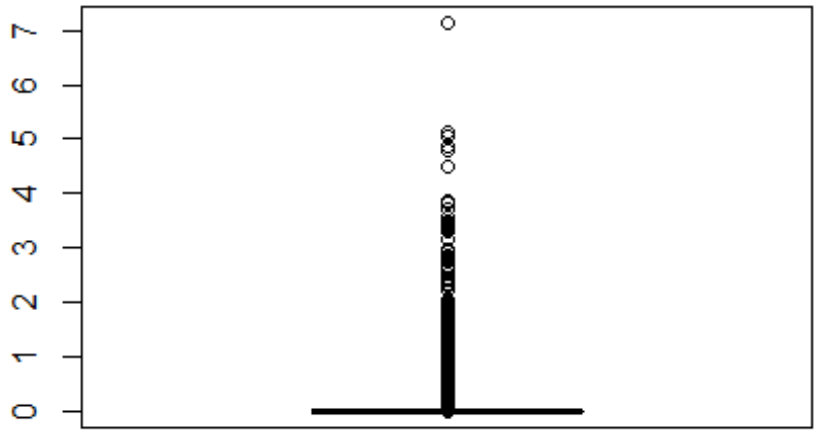
```
##  Mean   : 0.5504   Mean   : 0.2643   Mean   : 0.769   Mean   :0.125
##  3rd Qu.: 0.0000   3rd Qu.: 0.0000   3rd Qu.: 0.000   3rd Qu.:0.000
##  Max.   :20.8300   Max.   :16.6600   Max.   :33.330   Max.   :9.090
##       lab               labs           telnet             857
##  Min.   : 0.00000   Min.   :0.0000   Min.   : 0.00000   Min.
:0.00000
##  1st Qu.: 0.00000   1st Qu.:0.0000   1st Qu.: 0.00000   1st
Qu.:0.00000
##  Median : 0.00000   Median :0.0000   Median : 0.00000   Median
:0.00000
##  Mean   : 0.09782   Mean   :0.1026   Mean   : 0.06531   Mean
:0.04733
##  3rd Qu.: 0.00000   3rd Qu.:0.0000   3rd Qu.: 0.00000   3rd
Qu.:0.00000
##  Max.   :14.28000   Max.   :4.7600   Max.   :12.50000   Max.
:4.76000
##       data              415               85            technology
##  Min.   : 0.00000   Min.   :0.00000   Min.   : 0.0000   Min.
:0.00000
##  1st Qu.: 0.00000   1st Qu.:0.00000   1st Qu.: 0.0000   1st
Qu.:0.00000
##  Median : 0.00000   Median :0.00000   Median : 0.0000   Median
:0.00000
##  Mean   : 0.09759   Mean   :0.04816   Mean   : 0.1045   Mean
:0.09801
##  3rd Qu.: 0.00000   3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd
Qu.:0.00000
##  Max.   :18.18000   Max.   :4.76000   Max.   :20.0000   Max.
:7.69000
##       1999            parts               pm               direct
##  Min.   :0.0000   Min.   :0.00000   Min.   : 0.00000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.: 0.00000   1st Qu.:0.00000
##  Median :0.0000   Median :0.00000   Median : 0.00000   Median :0.00000
##  Mean   :0.1375   Mean   :0.01321   Mean   : 0.07607   Mean   :0.06558
##  3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.: 0.00000   3rd Qu.:0.00000
##  Max.   :6.8900   Max.   :8.33000   Max.   :11.11000   Max.   :4.76000
##        cs              meeting          original         project
##  Min.   :0.00000   Min.   : 0.0000   Min.   :0.0000   Min.   : 0.00000
##  1st Qu.:0.00000   1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.: 0.00000
##  Median :0.00000   Median : 0.0000   Median :0.0000   Median : 0.00000
##  Mean   :0.04408   Mean   : 0.1329   Mean   :0.0468   Mean   : 0.07879
##  3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd Qu.:0.0000   3rd Qu.: 0.00000
##  Max.   :7.14000   Max.   :14.2800   Max.   :3.5700   Max.   :20.00000
##        re               edu              table            conference
##  Min.   : 0.0000   Min.   : 0.0000   Min.   :0.000000   Min.   :
0.00000
##  1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.:0.000000   1st Qu.:
0.00000
##  Median : 0.0000   Median : 0.0000   Median :0.000000   Median :
0.00000
```

```
##  Mean   : 0.3062   Mean   : 0.1813   Mean   :0.005523   Mean   :
0.03254
##  3rd Qu.: 0.1300   3rd Qu.: 0.0000   3rd Qu.:0.000000   3rd Qu.:
0.00000
##  Max.   :21.4200   Max.   :22.0500   Max.   :2.170000   Max.
:10.00000
##        ;               (               [                !
##  Min.   :0.000   Min.   :0.0000   Min.   :0.00000   Min.   : 0.0000
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.: 0.0000
##  Median :0.000   Median :0.0660   Median :0.00000   Median : 0.0000
##  Mean   :0.039   Mean   :0.1396   Mean   :0.01695   Mean   : 0.2713
##  3rd Qu.:0.000   3rd Qu.:0.1880   3rd Qu.:0.00000   3rd Qu.: 0.3180
##  Max.   :4.385   Max.   :9.7520   Max.   :4.08100   Max.   :32.4780
##        $                #            cap_run_length_average
##  Min.   :0.00000   Min.   : 0.00000   Min.   :   1.000
##  1st Qu.:0.00000   1st Qu.: 0.00000   1st Qu.:   1.592
##  Median :0.00000   Median : 0.00000   Median :   2.272
##  Mean   :0.07545   Mean   : 0.04396   Mean   :   5.194
##  3rd Qu.:0.05300   3rd Qu.: 0.00000   3rd Qu.:   3.707
##  Max.   :6.00300   Max.   :19.82900   Max.   :1102.500
##  cap_run_length_longest cap_run_length_total     clase
##  Min.   :   1.00        Min.   :    1.0      Min.   :0.000
##  1st Qu.:   6.00        1st Qu.:   35.0      1st Qu.:0.000
##  Median :  15.00        Median :   95.0      Median :0.000
##  Mean   :  52.06        Mean   :  282.6      Mean   :0.394
##  3rd Qu.:  43.00        3rd Qu.:  267.0      3rd Qu.:1.000
##  Max.   :9989.00        Max.   :15841.0      Max.   :1.000
```

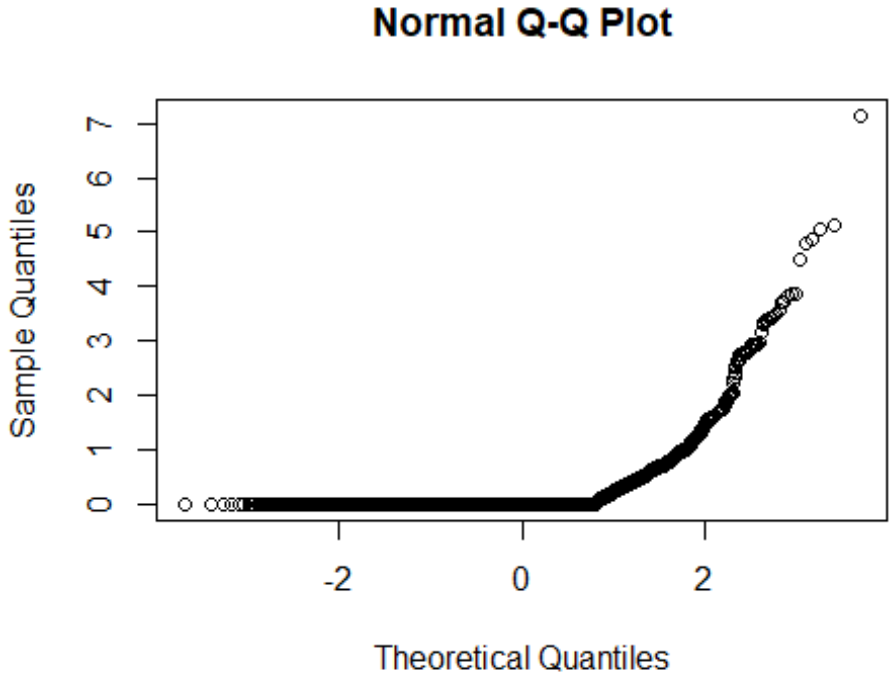
```
  #Destacar que la ultima columna es la que confirma si es o no spam el
correo
  dim(dtDatosFilled)  #Filas(emails) y columnas(variables)
```

```
## [1] 4601   58
```

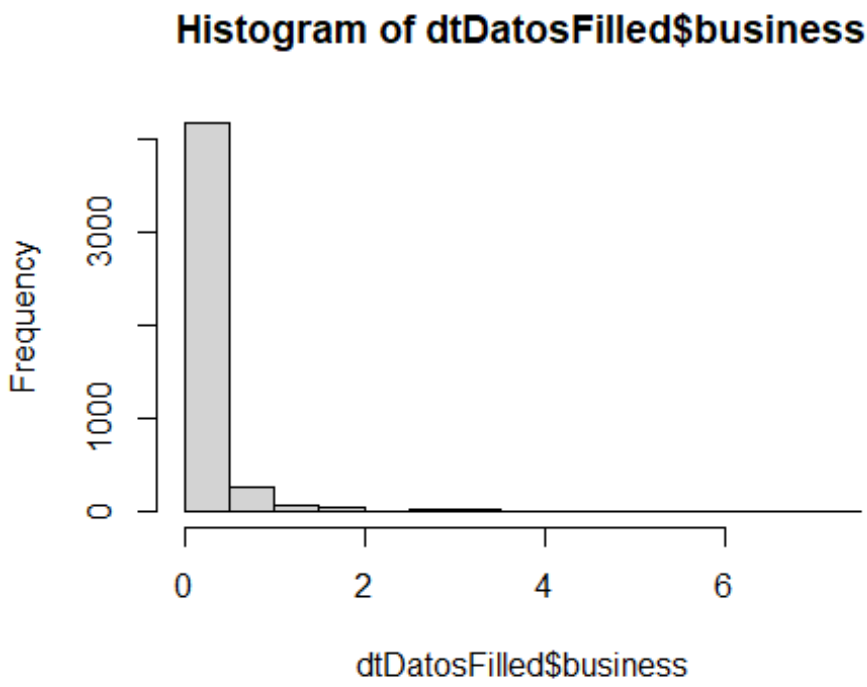
```
  #Visualizacion

  #Vamos a comprobar por ejemplo si business/free/you siguen una
distribucio normal
  boxplot(dtDatosFilled$business)
```



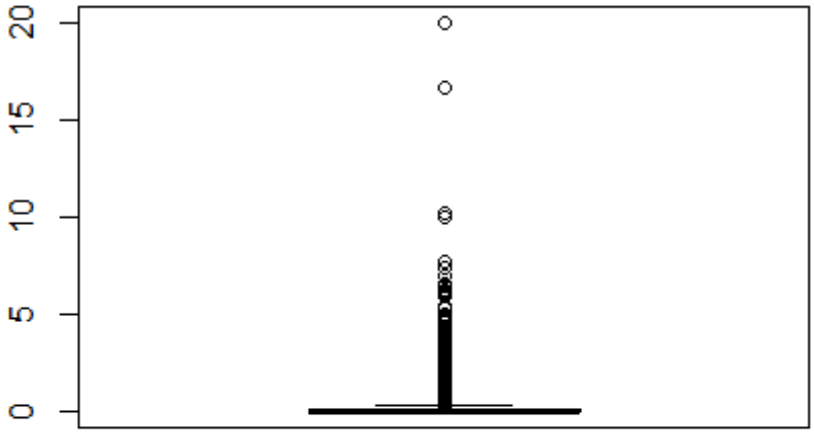

```
qqnorm(dtDatosFilled$business)
```



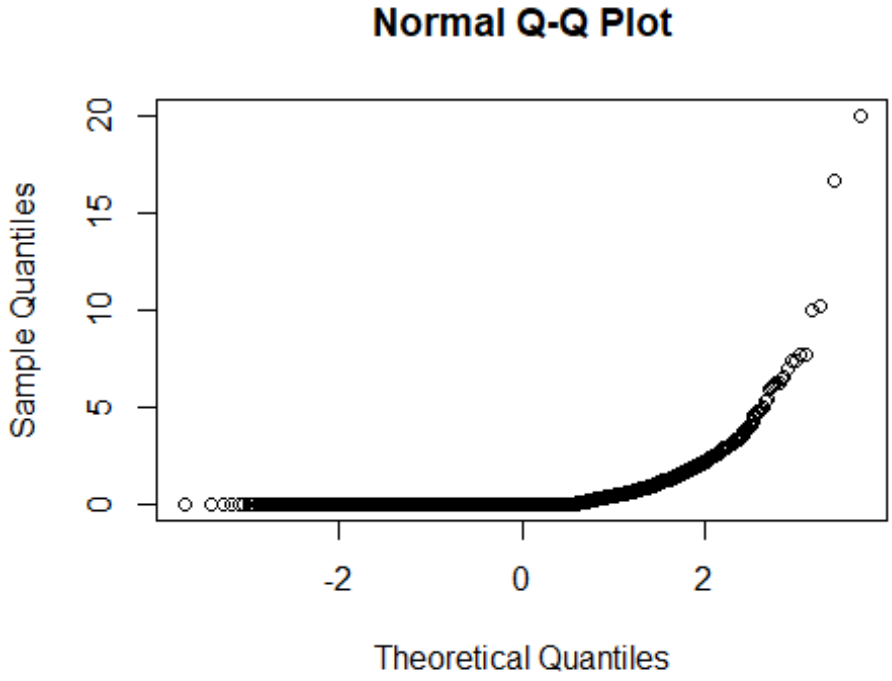

```
hist(dtDatosFilled$business)
```



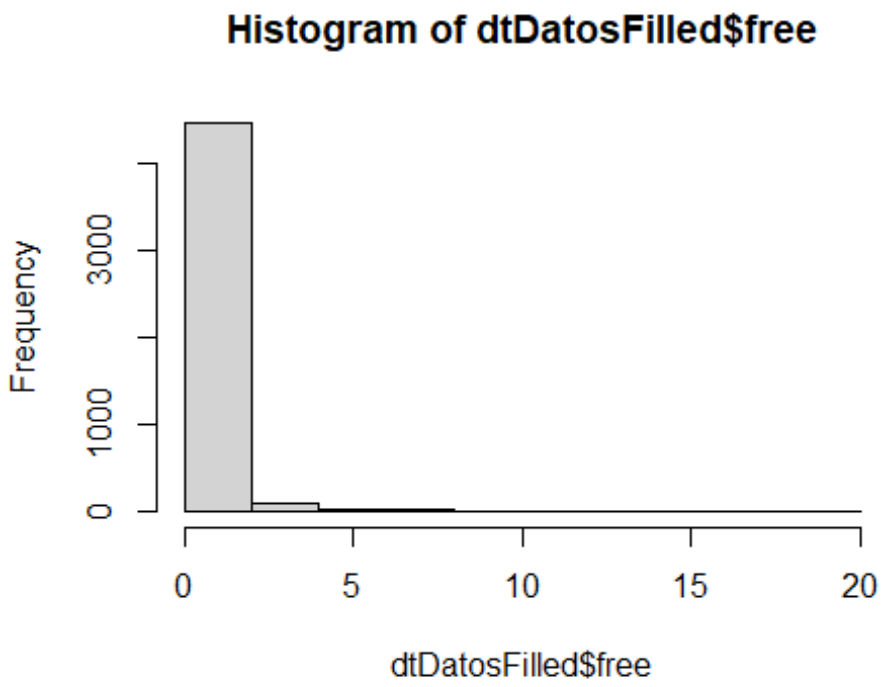

```
  shapiro.test(dtDatosFilled$business)

##
##  Shapiro-Wilk normality test
##
## data:  dtDatosFilled$business
## W = 0.36337, p-value < 2.2e-16

  boxplot(dtDatosFilled$free)
```



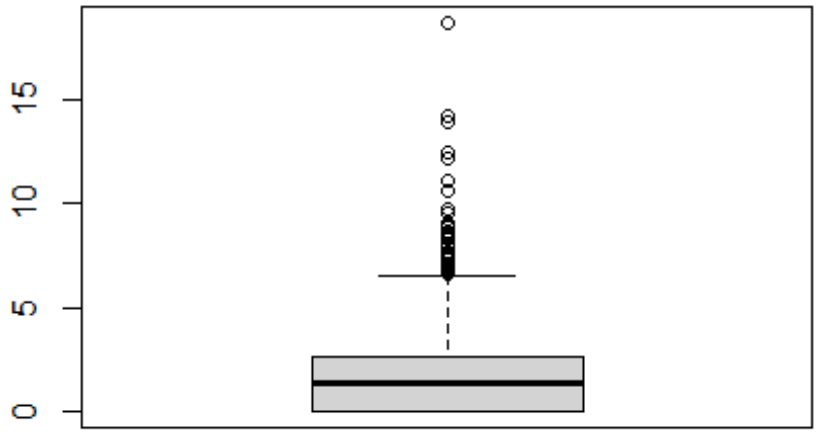

```
qqnorm(dtDatosFilled$free)
```



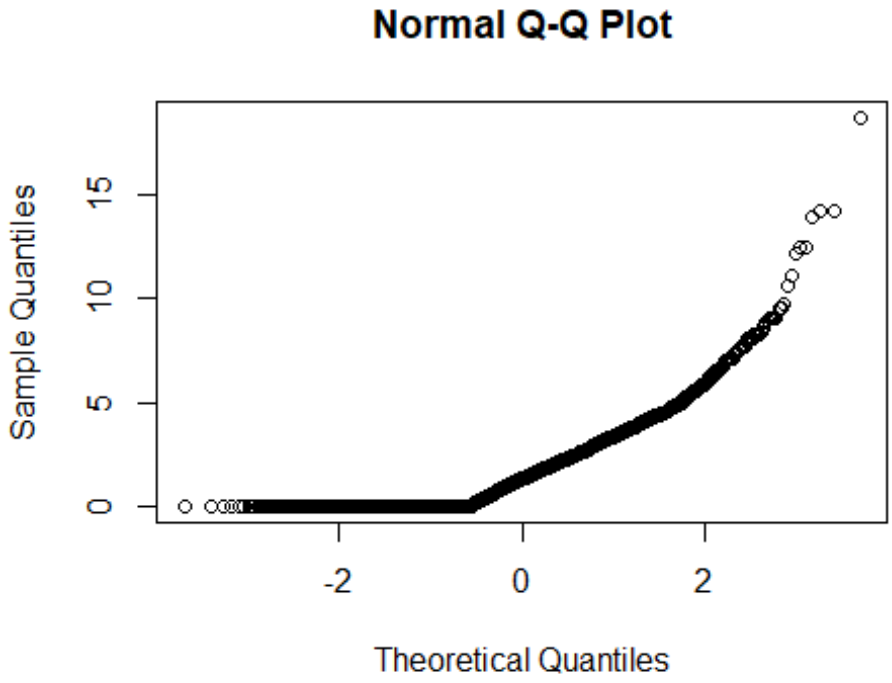

```
hist(dtDatosFilled$free)
```



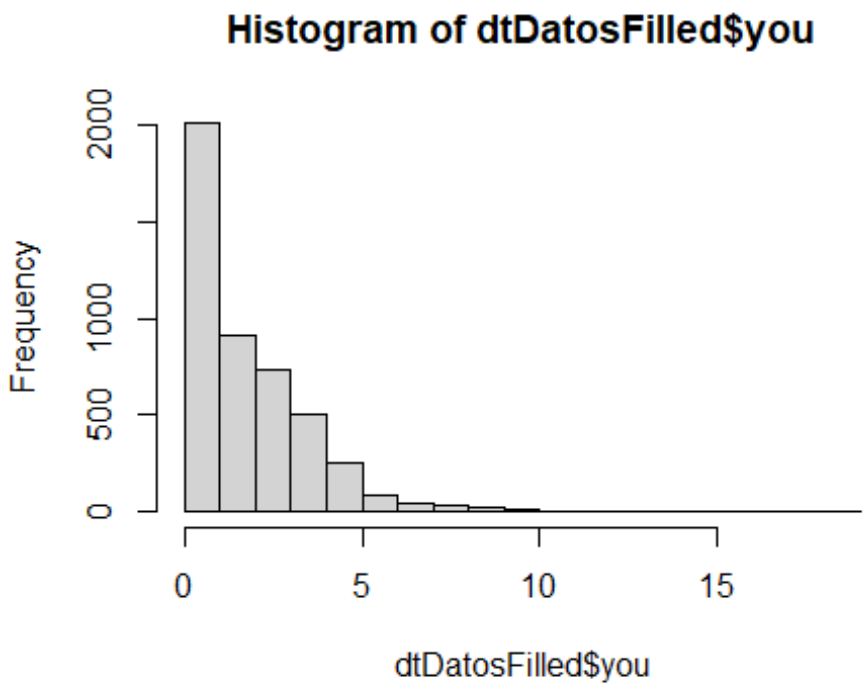

```
  shapiro.test(dtDatosFilled$free)

##
##  Shapiro-Wilk normality test
##
## data:  dtDatosFilled$free
## W = 0.33554, p-value < 2.2e-16

  boxplot(dtDatosFilled$you)
```



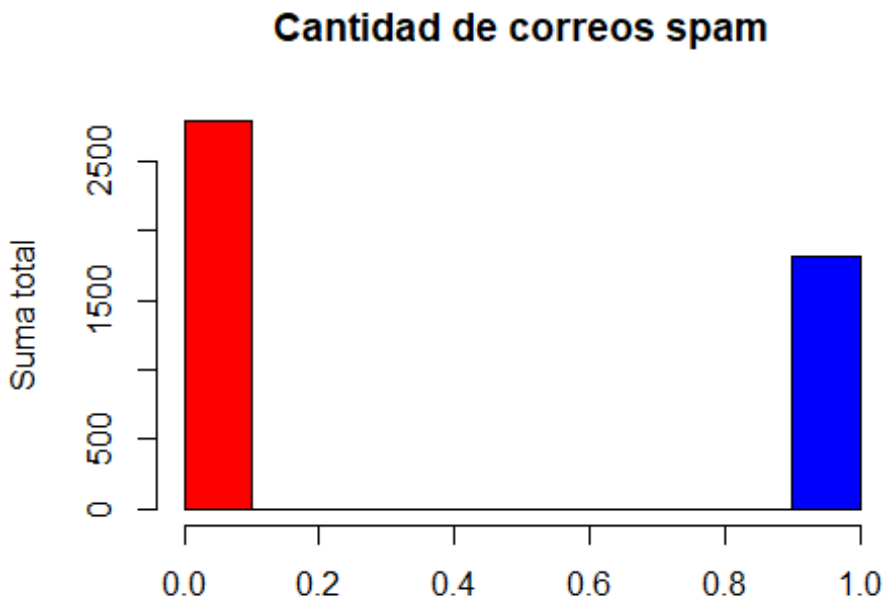

```
qqnorm(dtDatosFilled$you)
```



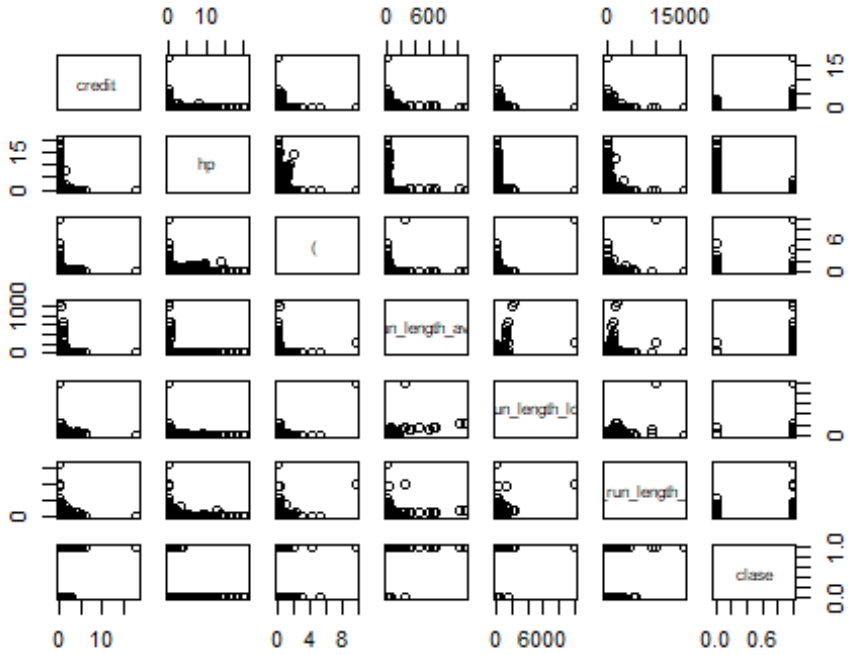

```
hist(dtDatosFilled$you)
```




```
  shapiro.test(dtDatosFilled$you)

##
##  Shapiro-Wilk normality test
##
## data:  dtDatosFilled$you
## W = 0.84836, p-value < 2.2e-16

  #Ninguna de ellas sigue una distribucion normal. La más cerca de
hacerlo es la variable 'you'

  #Correos spam o no spam
  hist(dtDatosFilled$clase,main="Cantidad de correos spam",
       col= c("red","blue"),
       ylab="Suma total",
       xlab="")
```




```
  p<-table(dtDatosFilled$clase)
  prop.table(p) #Porcentaje de correos si o no spam

##
##         0         1
## 0.6059552 0.3940448

  #Compruebo correlacion entre variables aleatoriamente mediante graficos
  plot(dtDatosFilled[,c(20,25,50,55,56,57,58)]) #No se observa ninguna
correlacion entre las variables seleccionadas
```




```
  #Resumen por variables
  require(psych)
  psych::describe(dtDatosFilled$cap_run_length_total)
```

```
##    vars    n  mean     sd median trimmed    mad min   max range skew
kurtosis
## X1    1 4601 282.6 603.97     95  156.86 114.16   1 15841 15840 8.79
148.04
##     se
## X1 8.9
```

```
  psych::describe(dtDatosFilled$free)
```

```
##    vars    n mean   sd median trimmed mad min max range skew kurtosis
se
## X1    1 4601 0.25 0.78      0    0.08   0   0  20    20 9.32   157.08
0.01
```

```
  psych::describe(dtDatosFilled$money)
```

```
##    vars    n mean   sd median trimmed mad min  max range  skew
kurtosis   se
## X1    1 4601 0.09 0.45      0    0.01   0   0 12.5  12.5 14.51
293.51 0.01
```

```
  #Apartado 3)
  #La correlacion entre dos variables va desde -1 a 1(correlacion
negativa o positiva perfecta)
```

```r
  #Cuanto más cerca este de 1 o -1 la correlación es más alta.
  #Atendiendo al apartado anterior donde se representa la correlacion en
el cruce de variables
  #Considerando a partir de 0.90(neg. o pos.) una correlacion alta:
  #Se observa correlacion alta para las variables 857 y 415

  M<-cor(Filter(is.numeric, dtDatosFilled))

  library(corrplot)
  cor_matrix <- cor(dtDatosFilled)

  #Considero a partir de 85% una correlacion alta
  listaPalabrasCorr <- apply(cor_matrix, 1, function(x) which(x >= 0.85 &
x < 1))
  columnasEliminar <- unlist(listaPalabrasCorr)
  cat(paste("Las columna a borrar son:"))
```

```
## Las columna a borrar son:
```

```r
  cat(paste(columnasEliminar,sep=" "))
```

```
## 34 32
```

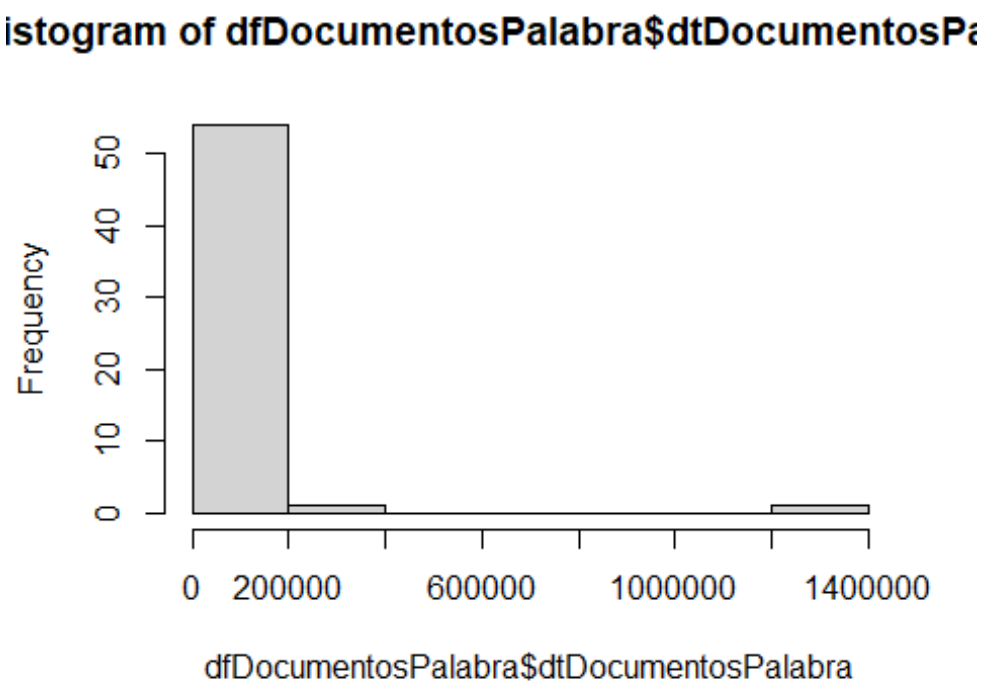
```r
  cor(dtDatosFilled$`857`, dtDatosFilled$`415`)
```

```
## [1] 0.9671843
```

```r
  #Se borran las columnas 34 y 32 que son las que tienen una correlacion
muy alta entre ellas
  dtDatosFinal<-dtDatosFilled
  dtDatosFinal$`857`<- NULL
  dtDatosFinal$`415`<- NULL

  M2<-cor(Filter(is.numeric, dtDatosFinal)) #La siguiente correlacion mas
alta está alrededor del 0.65

  #Calculo del nº de documentos en los que aparece una palabra = Suma de
cada variable en todas las filas
  dtDocumentosPalabra<-colSums(dtDatosFinal)
  class(dtDocumentosPalabra)
```

```
## [1] "numeric"
```

```r
  dfDocumentosPalabra<-data.frame(dtDocumentosPalabra)

  #Ploteo el histograma para ver que variables puedo suprimir
  hist(dfDocumentosPalabra$dtDocumentosPalabra)
```

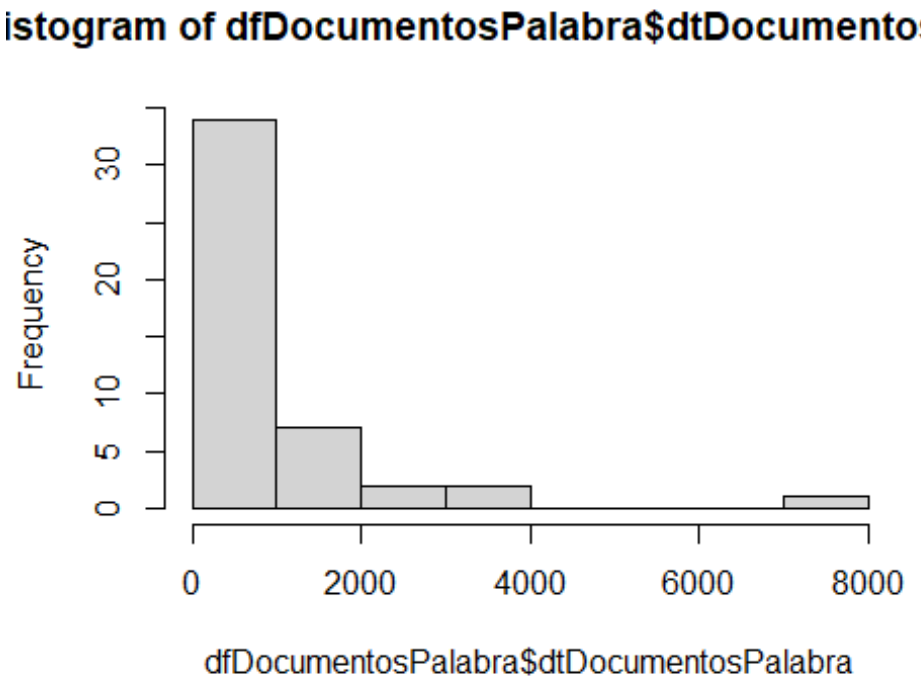



```
  #Ordeno de menor a mayor y decido que borro las columnas cuyo valor es
inferior a 200
  library(dplyr)
  dfDocumentosPalabra %>% arrange(dtDocumentosPalabra)

##                          dtDocumentosPalabra
## table                                25.4100
## parts                                60.7700
## [                                    77.9775
## conference                          149.7350
## ;                                   179.4535
## #                                   202.2545
## cs                                  202.8350
## original                            215.3350
## addresses                           226.6900
## report                              272.3900
## receive                             274.2300
## telnet                              300.5050
## direct                              301.7300
## 3d                                  310.4500
## $                                   347.1585
## pm                                  350.0150
## project                             362.5250
## credit                              394.3450
## order                               419.1700
## people                              434.2350
## money                               435.8850
```

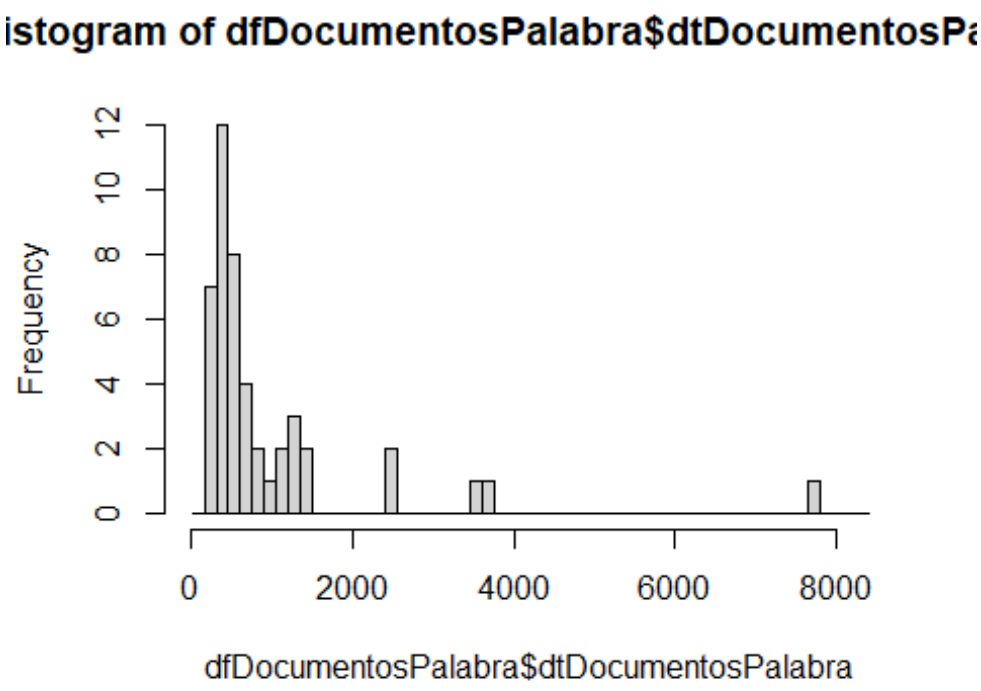
```
## over                                436.9250
## data                                449.0300
## lab                                 450.0750
## technology                          450.9350
## labs                                471.9850
## 000                                 473.6500
## 85                                  480.6750
## make                                482.3100
## internet                            487.1500
## remove                              519.1250
## font                                552.5400
## 650                                 575.0400
## meeting                             611.6700
## 1999                                632.8350
## (                                   642.2080
## business                            662.4300
## edu                                 834.2600
## email                               847.0600
## address                             985.5700
## mail                               1099.7350
## free                               1141.4350
## hpl                                1216.2550
## !                                  1248.4770
## all                                1289.7850
## re                                 1408.6500
## our                                1432.8750
## clase                              1813.0000
## will                               2482.5300
## hp                                 2532.4550
## george                             3537.9850
## your                               3731.7300
## you                                7667.5200
## cap_run_length_average            23898.8155
## cap_run_length_longest           239524.0000
## cap_run_length_total            1300228.5000
```

```
  #Borro las columnas con menos frecuencia, son 5: table -conference - ;
- parts - [

  dtDatosFinal$table<- NULL
  dtDatosFinal$conference<- NULL
  dtDatosFinal$`;`<- NULL
  dtDatosFinal$parts<- NULL
  dtDatosFinal$`[`<- NULL
  dtDatosFinal$cs<- NULL

  #Además, considero que es mejor eliminar las ultimas 4 columnas para
observar mejor el histograma
  #de la frecuencia de las palabras y que haya ruido
  #Lo guardo en otro dataframe porque voy a trabajar con dtDatosFinal en
```

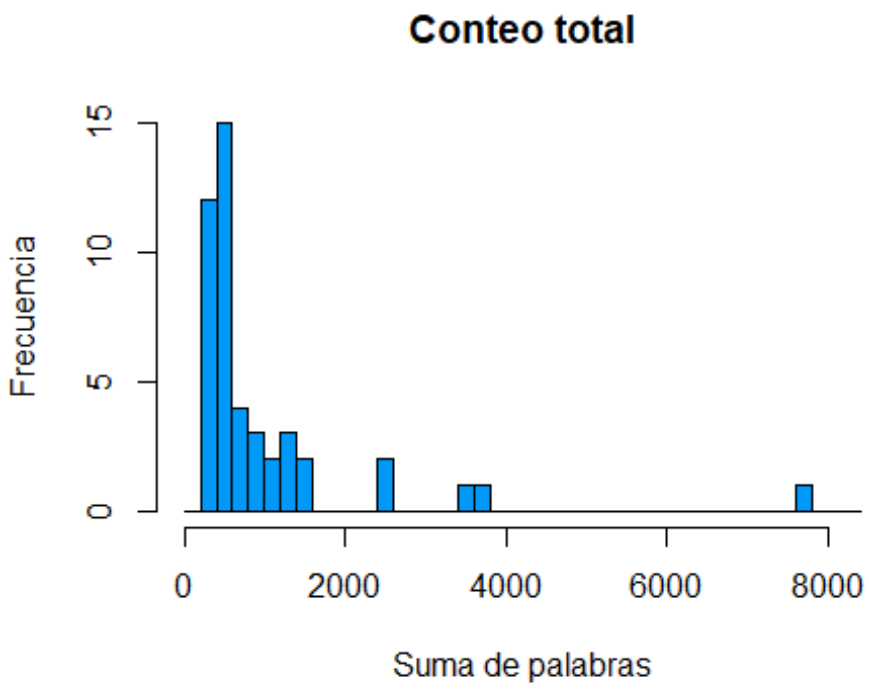
```
los apartados siguientes.
  dtDatosFinal2 <- dtDatosFinal

  dtDatosFinal2$cap_run_length_total<- NULL
  dtDatosFinal2$cap_run_length_longest <- NULL
  dtDatosFinal2$cap_run_length_average<- NULL
  dtDatosFinal2$clase<- NULL


  dtDocumentosPalabra<-colSums(dtDatosFinal2)
  dfDocumentosPalabra<-data.frame(dtDocumentosPalabra)

  #Ploteamos
  hist(dfDocumentosPalabra$dtDocumentosPalabra)
```



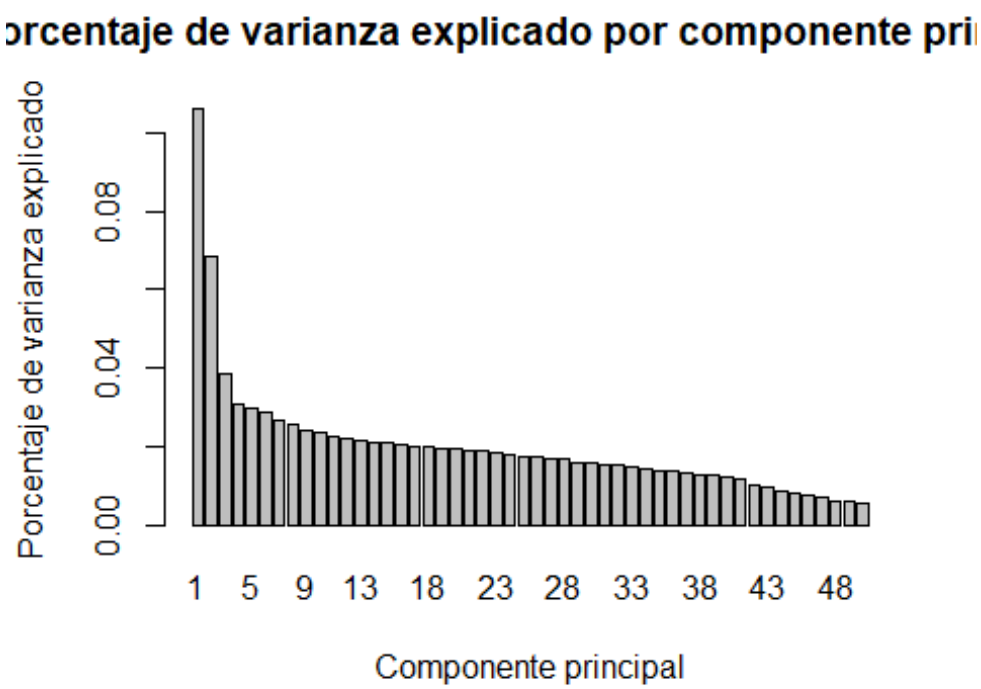

```
  #Ploteamos poniendo limites en el eje de las x
  hist(dfDocumentosPalabra$dtDocumentosPalabra,
       breaks =seq(5,8500,150))
```



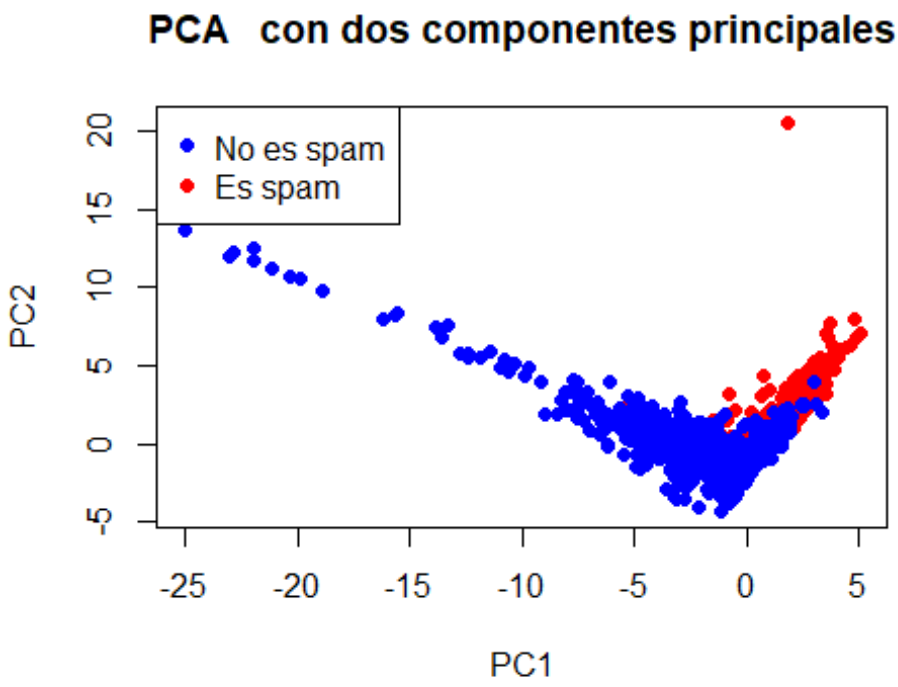

```
#Ploteamos de nuevo
hist(dfDocumentosPalabra$dtDocumentosPalabra,
     breaks =seq(0,8500,200),
     col="#0099F8",
     border="#000000",
     ylab="Frecuencia",
     xlab="Suma de palabras",
     main='Conteo total')
```



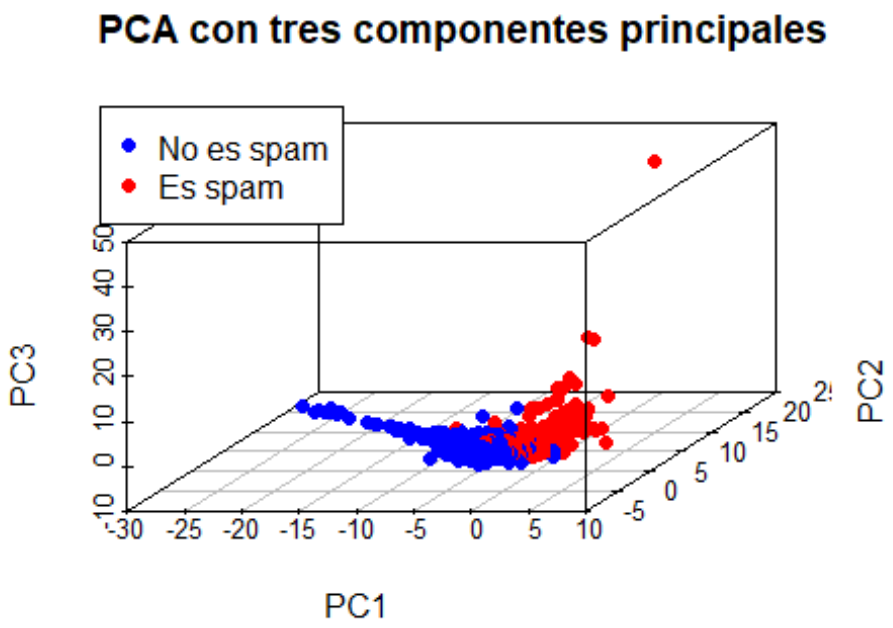

```
  #Observamos que hay una palabra(you) que aparece con diferencia más
veces que el resto.
  #La mayoria de palabra están entre las 0 y las 170 apariciones

  #Apartado 4)

  #Lo primero antes de realizar un analisis de componentes es normalizar
la matriz.
  dtDatosFinalNorm <- scale(dtDatosFinal)


  dtDatosFinal_PCA <- prcomp(dtDatosFinalNorm[,1:50], center = TRUE,
scale. = TRUE)

  barplot(dtDatosFinal_PCA$sdev^2 / sum(dtDatosFinal_PCA$sdev^2),
        names.arg = 1:length(dtDatosFinal_PCA$sdev),
        xlab = "Componente principal",
        ylab = "Porcentaje de varianza explicado",
        main = "Porcentaje de varianza explicado por componente
principal")
```




```
  #Para ver la legenda de los graficos tienen que ejecutarse las dos
sentencias a la vez
  plot(dtDatosFinal_PCA$x[,1:2], col = ifelse(dtDatosFinal[,50] == 1,
"red", "blue"), pch = 16, cex = 1, main = "PCA   con dos componentes
principales")
  legend("topleft", legend = c("No es spam", "Es spam"), col = c("blue",
"red"), pch = 16)
```




```
  library(scatterplot3d)
  scatterplot3d(x = dtDatosFinal_PCA$x[,1:3], y = NULL, z = NULL, color =
ifelse(dtDatosFinal[,50] == 1, "red", "blue"), pch = 16, cex.symbols = 1,
main = "PCA con tres componentes principales")
  legend("topleft", legend = c("No es spam", "Es spam"), col = c("blue",
"red"), pch = 16)
```



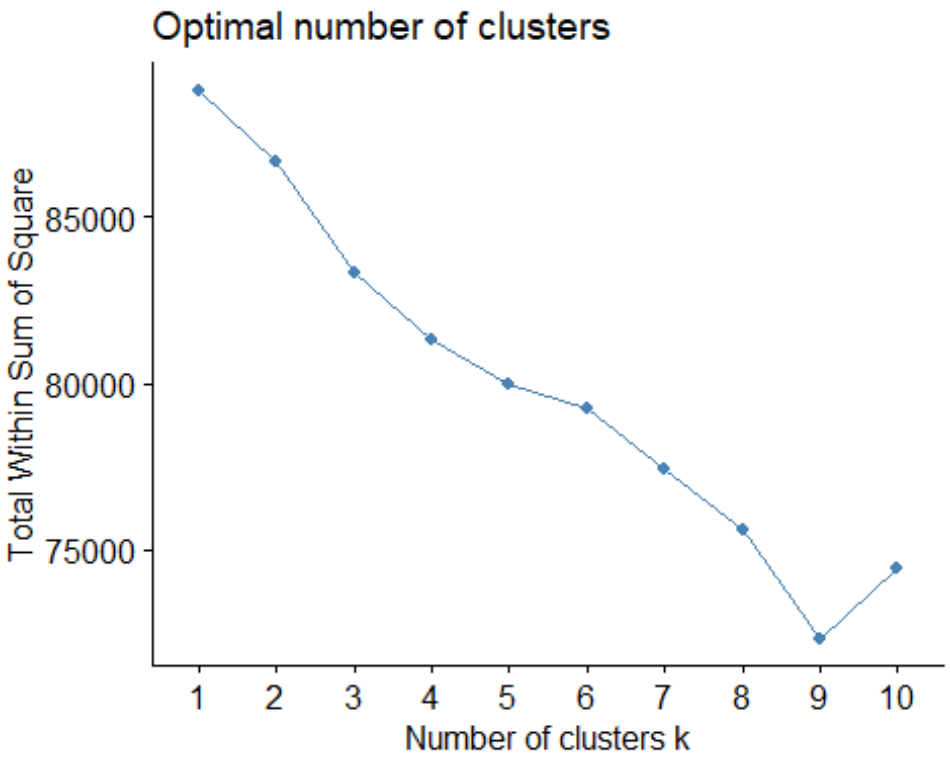

```
  #Observando el barplot se observa que las dos primeras componentes
principales explican el 18% y junto con la        #tercera explican el
22%. Bajo mi punto de vista con 2 componentes principales ya se puede
visualizar la
  #estructura semántica de los doscumentos.

  #Ploteando el grafico con 2 y 3 CP se puede ver que si hay estructura
de grupo.

  #Apartado 5)
  #Clustering para dos metodos diferentes

  library(cluster)
  library(factoextra)
  library(magrittr)
```

```
##
## Attaching package: 'magrittr'
```

```
## The following object is masked from 'package:purrr':
##
##     set_names
```

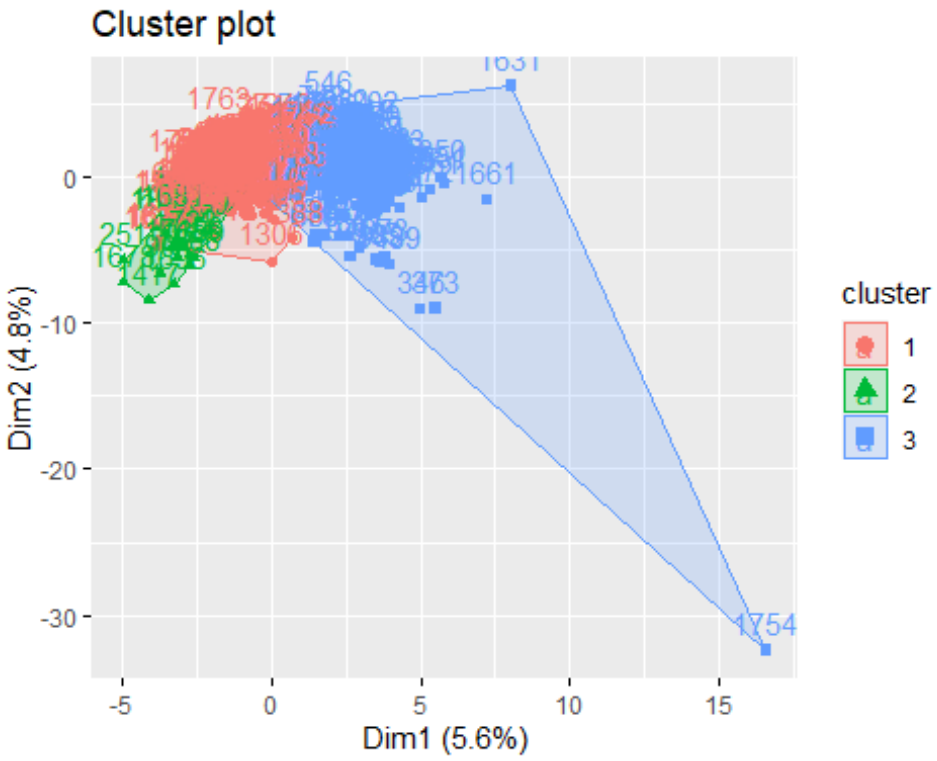
```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
#Me quedo solo con los correos spam
dtDatosFinal_spam <- subset(dtDatosFinal, clase == 1)

#Borro la ultima columna ya que no es necesaria
dtDatosFinal_spam$clase<- NULL

#Escalo para que todas las variables tengan mismo peso
dtDatosFinal_spam_scaled <- scale(dtDatosFinal_spam)

#Uso el método del codo para encontrar el nº óptimo de clusters
if(!require('factoextra')) {
  install.packages('factoextra')
  library('factoextra')
}
fviz_nbclust(dtDatosFinal_spam_scaled, kmeans, method="wss")
```

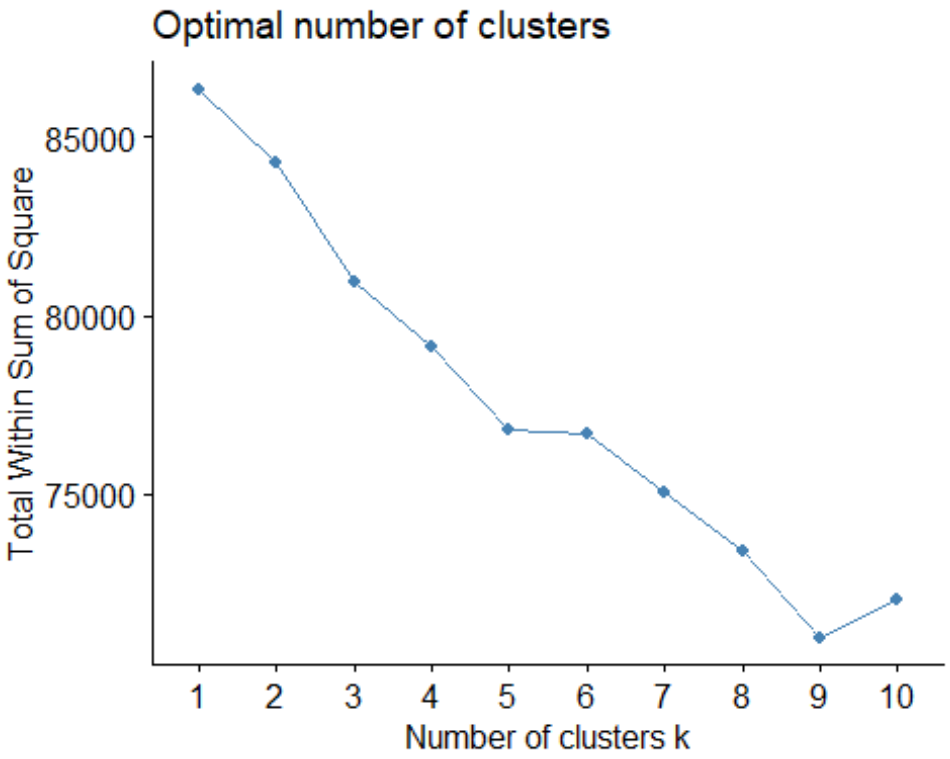



```
#Según el método del codo he decidido que lo mejor es hacer 3 grupos.

#Como no hay mucha correspondencia entre todas las variables se podrian
#llegar a hacer 9 grupos pero sabemos que no lo mejor por toda la
varianza que está sin explicar



library(stats)
library(ggplot2)
```

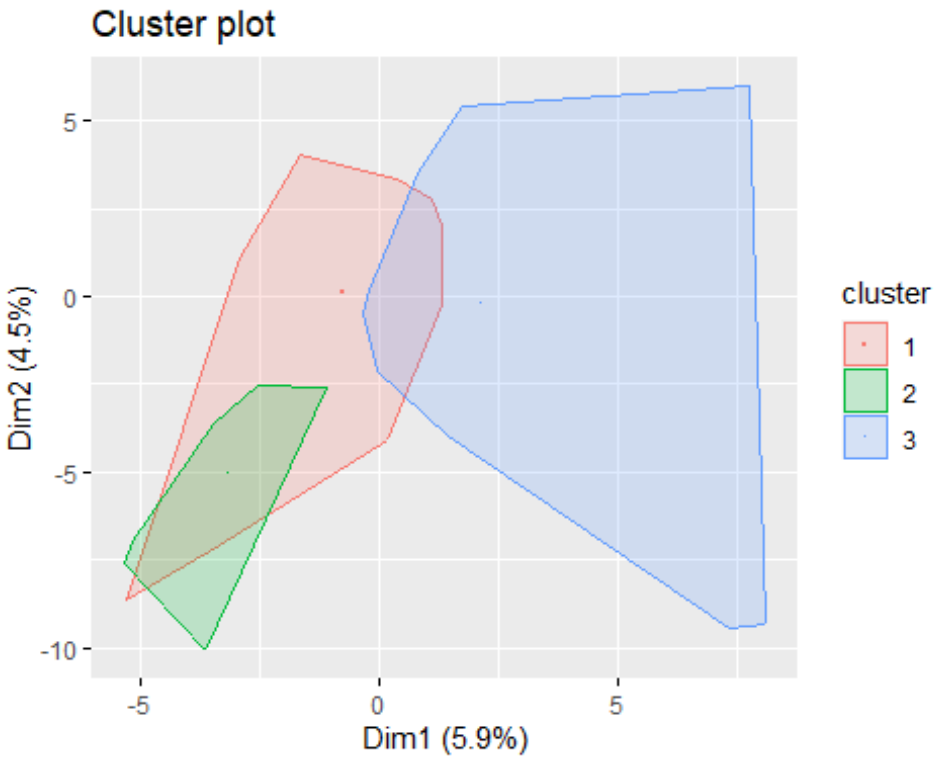
```
  if(!require('ggplot2')) {
    install.packages('ggplot2')
    library('ggplot2')
  }

  #1-er algoritmo: K-means

  fviz_cluster(kmeans(dtDatosFinal_spam_scaled,centers=3, iter.max =
1500, nstart=25), data=dtDatosFinal_spam_scaled)
```



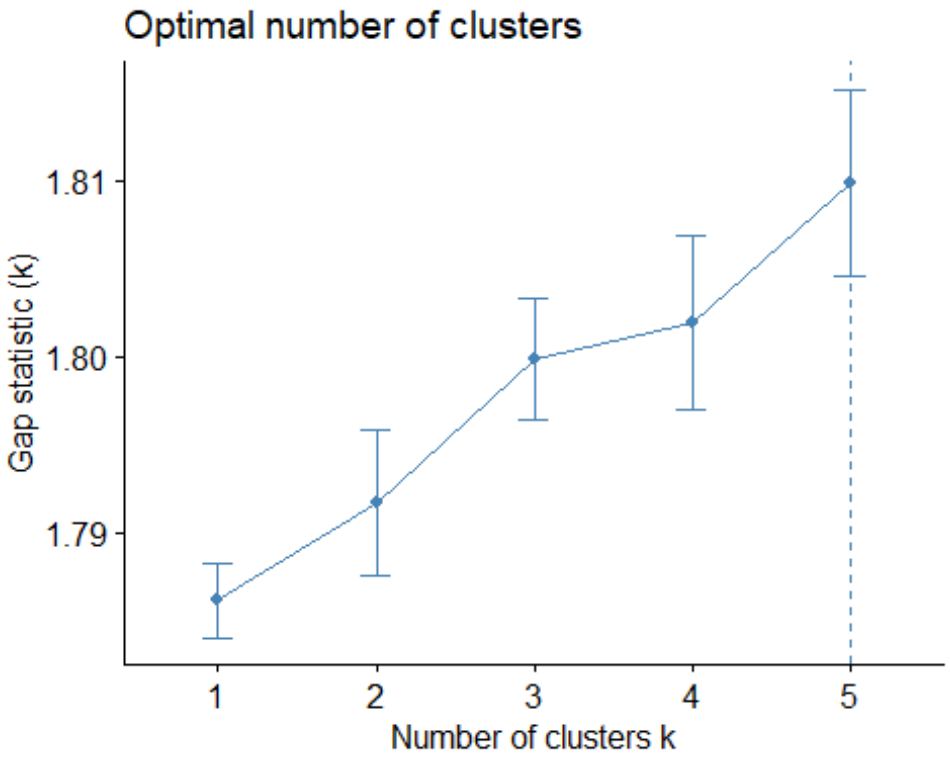

```
  #Haciendo pruebas con distintos parámetros tomamos la decision de
eliminar el outlier
  #porque no ayuda a explicar los grupos y está muy lejos de cualquier
grupo

  dtDatosFinal_spam_scaled_sin_outlier <- dtDatosFinal_spam_scaled[-
1754,]
  fviz_nbclust(dtDatosFinal_spam_scaled_sin_outlier, kmeans,
method="wss")
```



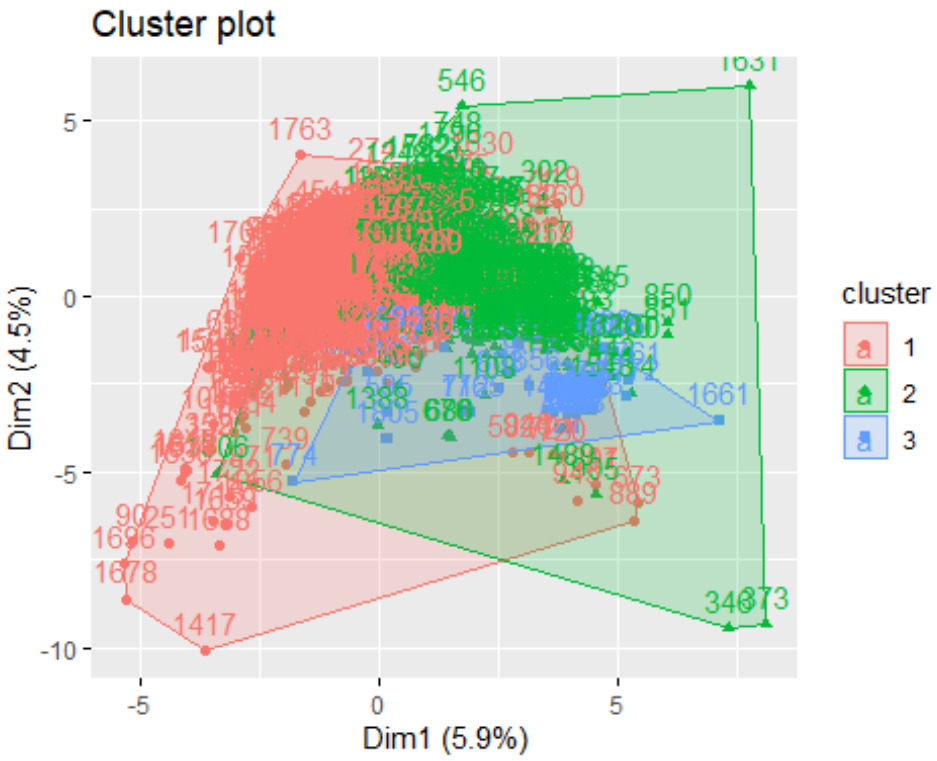

```
  #Se disminuye la cantidad de grupos y se mejora la calidad de los
grupos

  fviz_cluster(kmeans(dtDatosFinal_spam_scaled_sin_outlier,centers=3,
iter.max = 1500, nstart=25,), data=dtDatosFinal_spam_scaled_sin_outlier,
geom = "density",
             show.centroids = TRUE)
```




```
  #2º Algortimo K-Medioides = PAM

  if(!require('cluster')) {
    install.packages('cluster')
    library('cluster')
  }

  #Usando otra forma para comprobar el numero ideal de grupos decido que
el tamaño ideal es 3
  gap_stat <- clusGap(dtDatosFinal_spam_scaled_sin_outlier,
                      FUN = pam,
                      K.max = 5, #Nº Maximos de clusters
                      B = 5) #Nº de veces que se genera conjunto aleatorio

  fviz_gap_stat(gap_stat)
```




```r
#Guardar en kmed con 3 clusters
kmed <- pam(dtDatosFinal_spam_scaled_sin_outlier, k = 3)

fviz_cluster(kmed, data = dtDatosFinal_spam_scaled_sin_outlier)
```



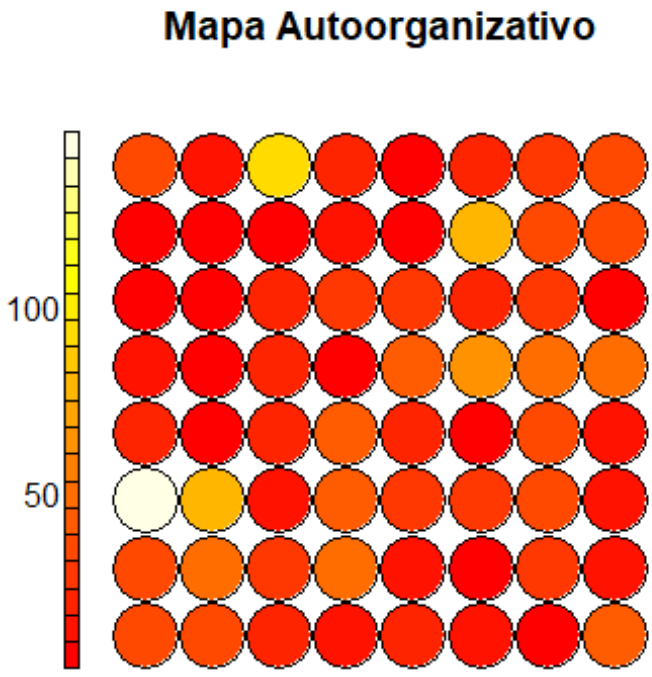

```
  #Partiendo de que las observaciones son muy dispares y muchas variables
para cada observacion
  #Haciendo el clustering solo para los correos que son spam, de los 3
grupos que se ven en el plot,
  #se puede concluir que hay un grupo que es considerado spam
consistentemente y los otros dos grupos tienen
  #tendencia; uno de ellos a 100% ser considerado correo malicioso y otro
de ellos que es considerado
  #spam pero tiene ciertos componentes que no lo deja claramente agrupado
en esta categoría

  #Apartado 6)

  if(!require('kohonen')) {
    install.packages('kohonen')
    library('kohonen')
  }
```

```
## Loading required package: kohonen
```

```
##
## Attaching package: 'kohonen'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

```
  # Cargamos la matriz con los datos con lo que realicé el apartado
anterior.(Matriz sin el outlier)
  data(dtDatosFinal_spam_scaled_sin_outlier)

## Warning in data(dtDatosFinal_spam_scaled_sin_outlier): data set
## 'dtDatosFinal_spam_scaled_sin_outlier' not found

  # Creamos el mapa autoorganizativo
  som_map <- som(dtDatosFinal_spam_scaled_sin_outlier[, -
ncol(dtDatosFinal_spam_scaled_sin_outlier)], grid = somgrid(8, 8,
"rectangular"))

  # Visualizamos el mapa
  plot(som_map, type = "count", main = "Mapa Autoorganizativo")
```




```
  #Las ventajas que tienen los mapas autoorganizativos frente a los
algoritmos de clustering son:

  # 1.Los mapas autoorganizativos preservan la topología de los datos de
entrada, lo que significa que los puntos      similares se agrupan juntos
en el mapa y los puntos diferentes se separan. Esto puede ser muy útil
para              identificar patrones y tendencias complejos en los
datos que podrían ser difíciles de detectar con otros
métodos.

  # 2.Los mapas autoorganizativos son una buena opción para la
exploración de datos y la                                identificación de
```

```
patrones sin conocimiento previo de la estructura de los datos.

  # 3.Fácil de implementar: Los mapas autoorganizativos son relativamente
fáciles de implementar y no requieren mucha   configuración previa. Esto
los hace accesibles y fáciles de usar para una amplia variedad de
usuarios, incluyendo    aquellos sin experiencia previa en aprendizaje
automático o análisis de datos.
```