Proyecto - Kaggle Titanic

José Manuel Alvarez García

Noviembre 06, 2016

Para instalar el paquete knit se debe ejecutar lo siguiente:

```
knitr::opts_chunk$set(echo = TRUE)
install = function(pkg)
{
  # Si ya está instalado, no lo instala.
  if (!require(pkg, character.only = TRUE)) {
    install.packages(pkg)
    if (!require(pkg, character.only = TRUE)) stop(paste("load failure:", pkg))
  }
}
```

Inicialmente se debe estar posicionado en el directorio de la tarea:

```
setwd("C:/Users/José Manuel/Documents/ICD/proyecto-kaggle-titanic-josemalvarezg1")
```

Análisis exploratorio.

Se trabajará con un conjunto de datasets que contiene la base de datos de las personas que abordaron al Titanic, y se tiene como clase principal o variable predictora si una persona sobrevivió a la tragedia o no.

En el mismo también se tiene el siguiente conjunto de atributos (columnas):

1. survival: Representa si una persona sobrevivió a la tragedia o no. Toma como valores 0 si la persona no sobrevivió, y 1 en caso contrario.
2. pclass: Representa la clase pasajera de una persona en el Titanic. Puede tomar los siguientes valores: 1 como 1st (primera clase), 2 como 2nd (segunda clase) y 3 como 3rd (tercera clase).
3. name: Representa el nombre de una persona. Puede tomar cualquier valor del tipo String.
4. sex: Representa el género de una persona. Puede tomar los siguientes valores: female, male.
5. age: Representa la edad de una persona. Puede tomar cualquier valor numérico. En caso de que se esté considerando una edad estimada, está será mostrada como xx.5.
6. sibsp: Representa el número de hermanos o cónyugues de una persona. Puede tomar cualquier valor entero.
7. parch: Representa el número de padres o niños (hijos) de una persona. Puede tomar cualquier valor entero.
8. ticket: Representa el número de ticket de una persona. Puede tomar cualquier valor del tipo String.
9. fare: Representa la tarifa pagada (en dólares) por una persona. Puede tomar cualquier valor del tipo flotante.
10. cabin: Representa la cabina en donde estará alojada una persona. Puede tomar cualquier valor del tipo String.

11. embarked: Representa el puerto de embarcación de una persona al abordar el Titanic. Puede tomar los siguientes valores: C si abordó en Cherbourg, Q si abordó en Queenstown y S si abordó en Southampton.

Posteriormente, estas columnas serán renombradas en la tarea de pre-procesamiento.

Pre-Procesamiento.

Inicialmente se debe leer el dataset de training:

```
train = read.csv(file = "../data/train.csv", header = T)
```

Luego se debe leer el dataset testing:

```
test = read.csv(file = "../data/test.csv", header = T)
```

Se identificarán las columnas de ambos datasets de la siguiente manera:

```
colnames(train) <- c("ID", "Sobrevivio", "Clase", "Nombre", "Sexo", "Edad",
                     "Hermanos/Cónyuges", "Padres/Niños", "Ticket", "Tarifa",
                     "Cabina", "Embarcación")
colnames(test) <- c("ID", "Clase", "Nombre", "Sexo", "Edad",
                    "Hermanos/Cónyuges", "Padres/Niños", "Ticket", "Tarifa",
                    "Cabina", "Embarcación")
```

Se elimina la primera columna (ID) de los datasets. Es considerada innecesaria ya que el ID de cada pasajero es su mismo número de registro (fila) en el dataset.

```
train = train[,-1]
test = test[,-1]
```

Algunos elementos de la columna “Edad” son desconocidos y contienen valores N/A. Por lo que se calcula la edad promedio y es colocada a los N/A.

```
train[,5][is.na(train[, 5])] <- 0
train[,5][train[, 5] == 0] <- ceiling(mean(train[["Edad"]]))
```

Los valores de la columna “Sexo” son cambiados a numéricos. Para que así se identifiquen como Male = 0 y Female = 1.

```
train$Sexo = (train$Sexo=="female")*1
```

Si se desea realizar un análisis exploratorio para estudiar más a fondo el dataset se debe realizar lo siguiente:

```
library(FactoMineR)
```

Se muestran valores de interés del dataset de entrenamiento:

```
head(train)
```

```
##   Sobrevivio Clase                                                Nombre
## 1          0     3                             Braund, Mr. Owen Harris
## 2          1     1 Cumings, Mrs. John Bradley (Florence Briggs Thayer)
## 3          1     3                              Heikkinen, Miss. Laina
## 4          1     1        Futrelle, Mrs. Jacques Heath (Lily May Peel)
## 5          0     3                            Allen, Mr. William Henry
## 6          0     3                                    Moran, Mr. James
##   Sexo Edad Hermanos/Cónyuges Padres/Niños           Ticket  Tarifa Cabina
## 1    0   22                1            0        A/5 21171  7.2500
## 2    1   38                1            0         PC 17599 71.2833    C85
## 3    1   26                0            0 STON/O2. 3101282  7.9250
## 4    1   35                1            0           113803 53.1000   C123
## 5    0   35                0            0           373450  8.0500
## 6    0   24                0            0           330877  8.4583
##   Embarcación
## 1           S
## 2           C
## 3           S
## 4           S
## 5           S
## 6           Q
```

```
dim(train)
```

```
## [1] 891  11
```

```
names(train)
```

```
##  [1] "Sobrevivio"        "Clase"             "Nombre"
##  [4] "Sexo"              "Edad"              "Hermanos/Cónyuges"
##  [7] "Padres/Niños"      "Ticket"            "Tarifa"
## [10] "Cabina"            "Embarcación"
```

```
summary(train)
```

```
##    Sobrevivio         Clase
##  Min.   :0.0000   Min.   :1.000
##  1st Qu.:0.0000   1st Qu.:2.000
##  Median :0.0000   Median :3.000
##  Mean   :0.3838   Mean   :2.309
##  3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :1.0000   Max.   :3.000
##
##                                    Nombre         Sexo
##  Abbing, Mr. Anthony                  :  1   Min.   :0.0000
##  Abbott, Mr. Rossmore Edward          :  1   1st Qu.:0.0000
##  Abbott, Mrs. Stanton (Rosa Hunt)     :  1   Median :0.0000
```

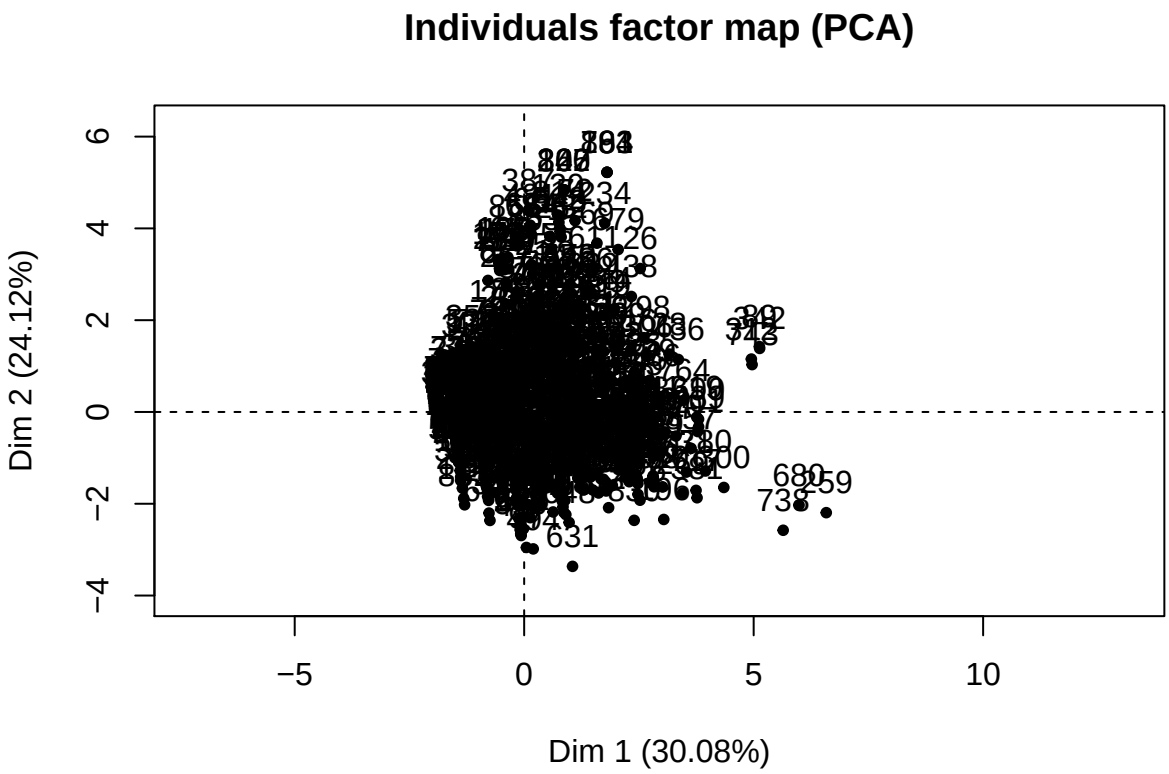
```
##  Abelson, Mr. Samuel                 :  1   Mean   :0.3524
##  Abelson, Mrs. Samuel (Hannah Wizosky):  1   3rd Qu.:1.0000
##  Adahl, Mr. Mauritz Nils Martin      :  1   Max.   :1.0000
##  (Other)                             :885
##       Edad       Hermanos/Cónyuges  Padres/Niños         Ticket
##  Min.   : 0.42   Min.   :0.000     Min.   :0.0000   1601    :  7
##  1st Qu.:22.00   1st Qu.:0.000     1st Qu.:0.0000   347082  :  7
##  Median :24.00   Median :0.000     Median :0.0000   CA. 2343:  7
##  Mean   :28.57   Mean   :0.523     Mean   :0.3816   3101295 :  6
##  3rd Qu.:35.00   3rd Qu.:1.000     3rd Qu.:0.0000   347088  :  6
##  Max.   :80.00   Max.   :8.000     Max.   :6.0000   CA 2144 :  6
##                                                     (Other) :852
##      Tarifa               Cabina    Embarcación
##  Min.   :  0.00              :687    :  2
##  1st Qu.:  7.91   B96 B98    :  4   C:168
##  Median : 14.45   C23 C25 C27:  4   Q: 77
##  Mean   : 32.20   G6         :  4   S:644
##  3rd Qu.: 31.00   C22 C26    :  3
##  Max.   :512.33   D          :  3
##                   (Other)    :186
```

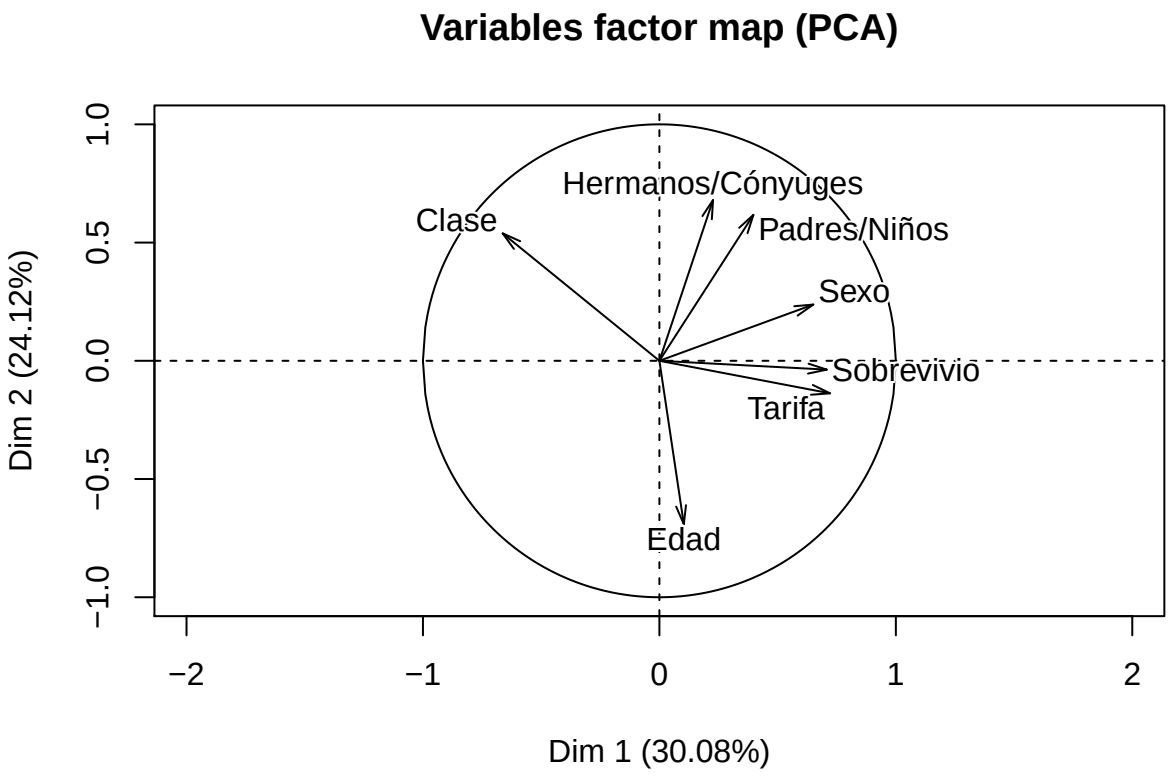
Se trabajará sólo con las columnas “Sobrevivió”, “Clase”, “Sexo”, “Edad”, “Hermanos/Cónyugues”, “Padres/Niños” y “Tarifa” para luego mostrar el PCA:

```
titanicPCA <- subset(train, select = c(1,2,4,5,6,7,9))
```

Se grafica el PCA de la siguiente manera:

```
pca <- PCA(titanicPCA)
```


Individuals factor map (PCA)
Dim 2 (24.12%)
Dim 1 (30.08%)
6
4
2
0
−2
−4
−5
0
5
10
631
680
259
738



K-Medias.

Para aplicar la técnica de clusterización K-Medias, se trabajará sólo con la columna "Edad".

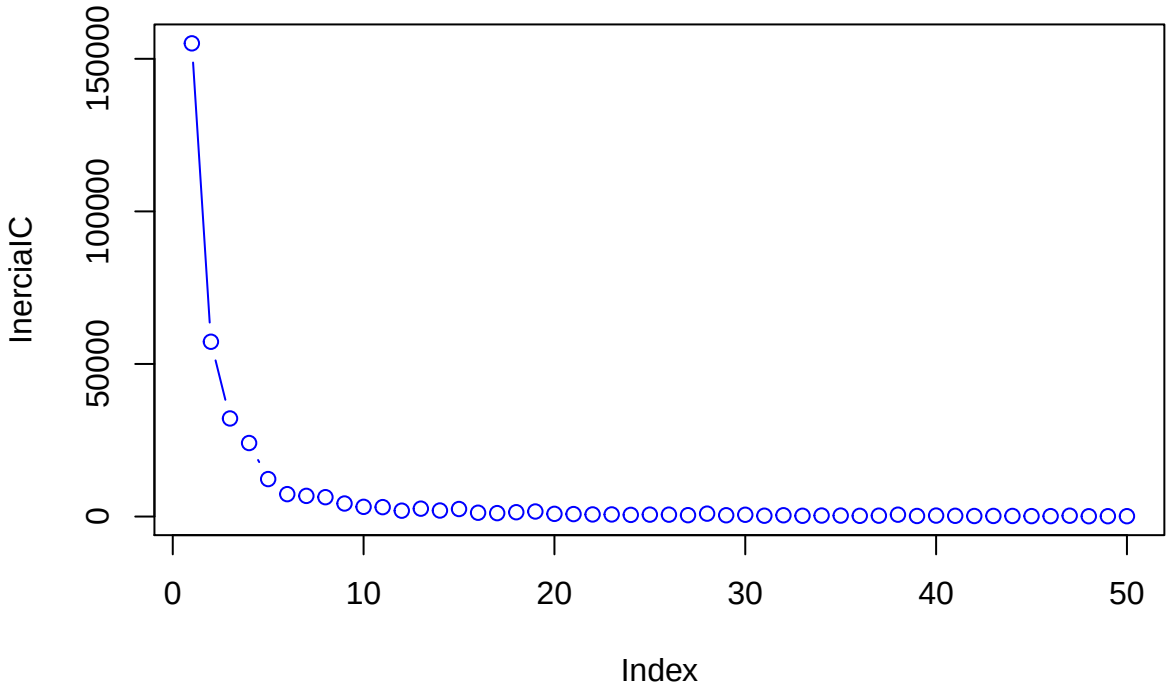
```
trainK = train[,c(5)]
```

Ahora bien, se buscará el K más adecuado para aplicar la técnica. Esto se realizará mediante el Codo de Jambu:

```
InerciaIC = rep(0,50)
for (k in 1:50) {
  grupos = kmeans(trainK, k)
  InerciaIC[k] = grupos$tot.withinss
}
```

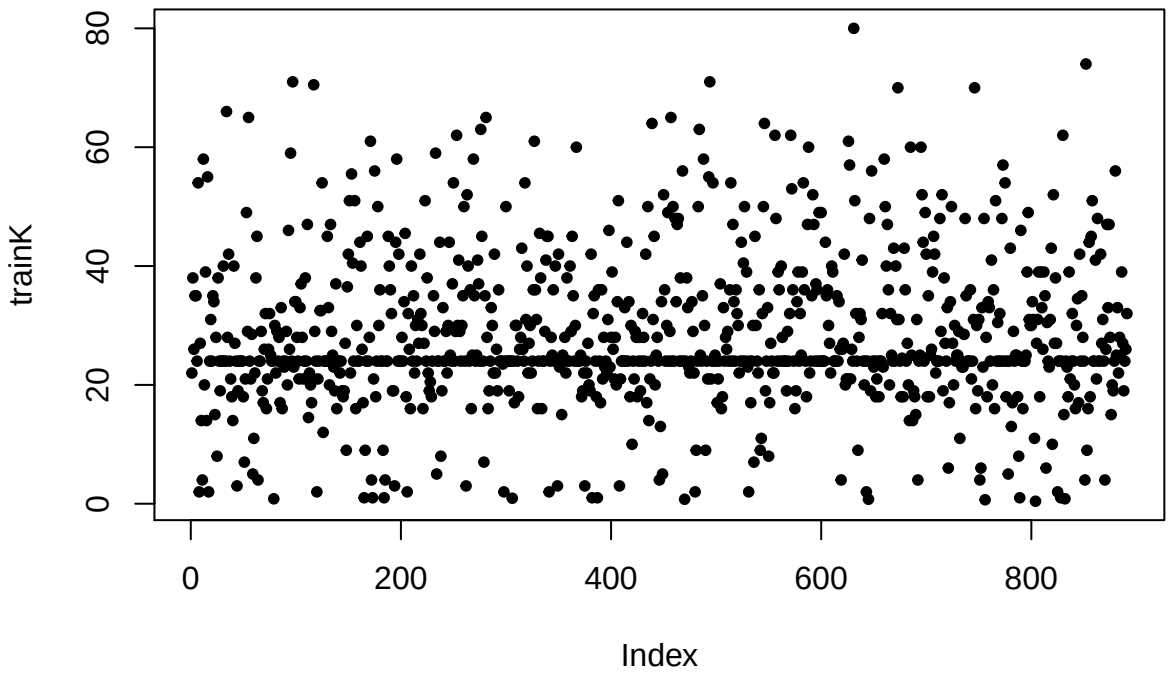
Si se grafica la Inercia Inter-Clases se puede observar que cambia muy poco a partir de K=4 y K=5.

```
plot(InerciaIC, col = "blue", type = "b")
```

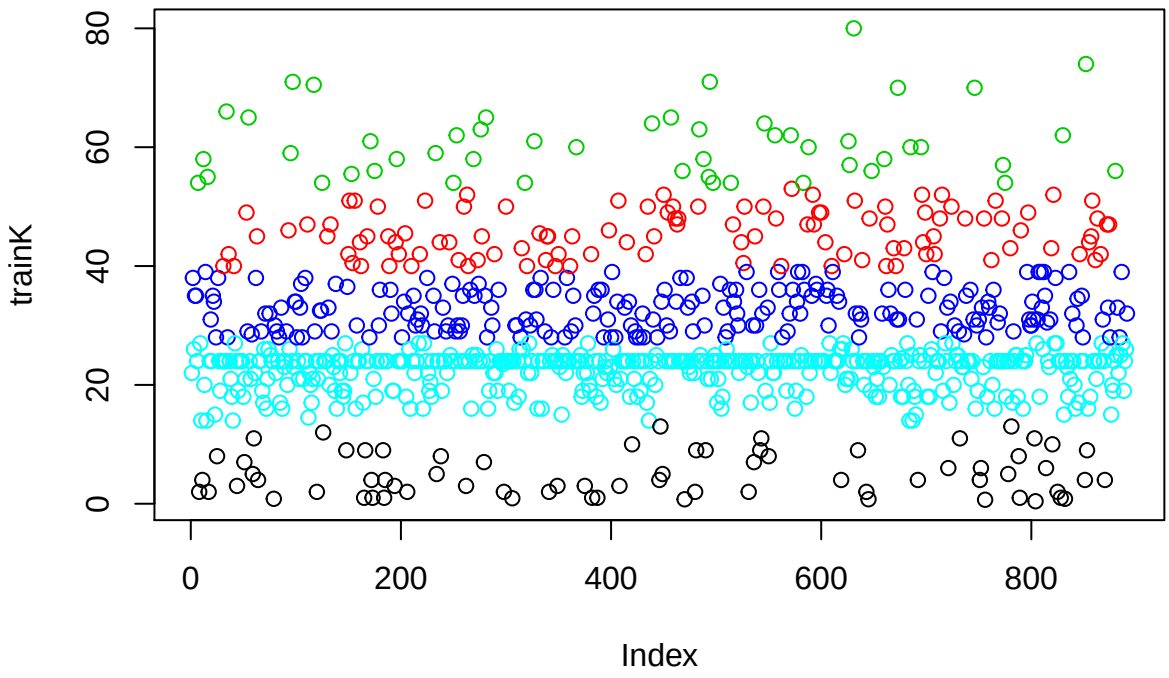


Por lo que calcula K-Medias con K=5 y 100 iteraciones, y se grafica:

```
clusters <- kmeans(trainK, 5, iter.max = 100)
plot(trainK, pch = 20)
```




```
plot(trainK, col = clusters$cluster)
```



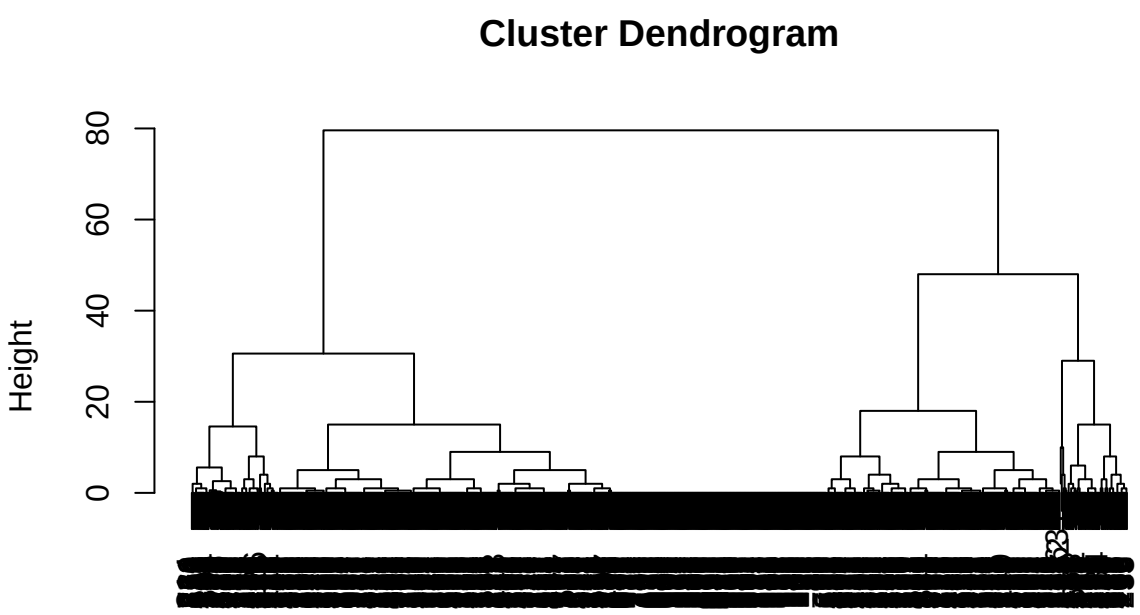
Clasificación Jerárquica.

Se trabajará el dataset pre-procesado anteriormente como una matriz y luego se calculará la matriz de distancia:

```
datos = as.matrix(trainK)
distancia = dist(datos)
```

Se aplicarán y se graficarán los métodos de clasificación jerárquica. Para el método Complete:

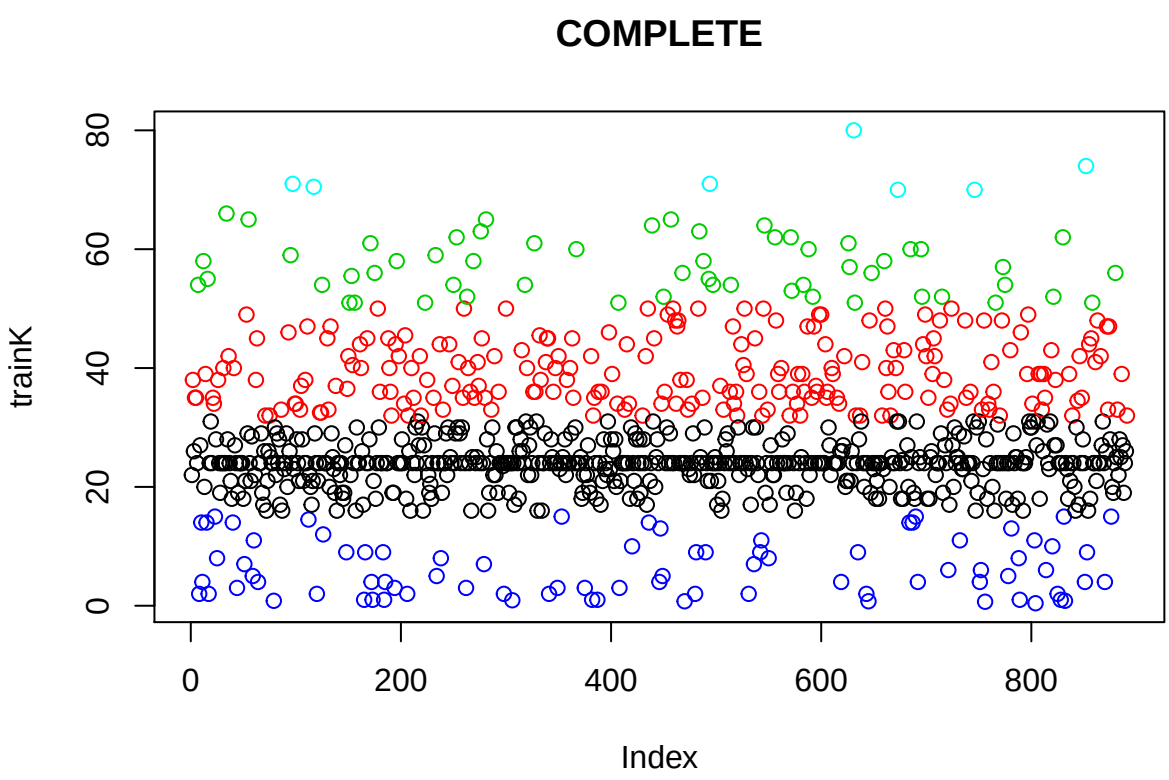
```
cluster = hclust(distancia, method = "complete")
plot(cluster)
```




```
#Se determina la altura requerida con k clusters, cortando el dendograma con k clases:
corteD = cutree(cluster, k = 5)
#Observamos la cantidad de clusters.
unique(corteD)
```

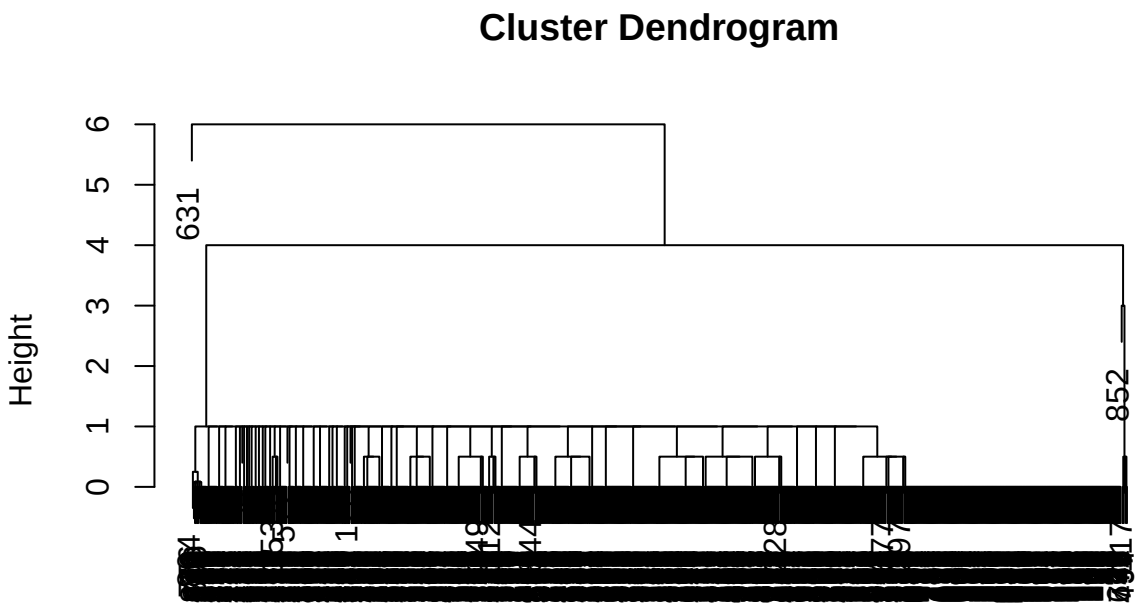
```
## [1] 1 2 3 4 5
```

```
#Graficamos los clusters.
plot(trainK, col = corteD, main = "COMPLETE")
```



Para el método Single:

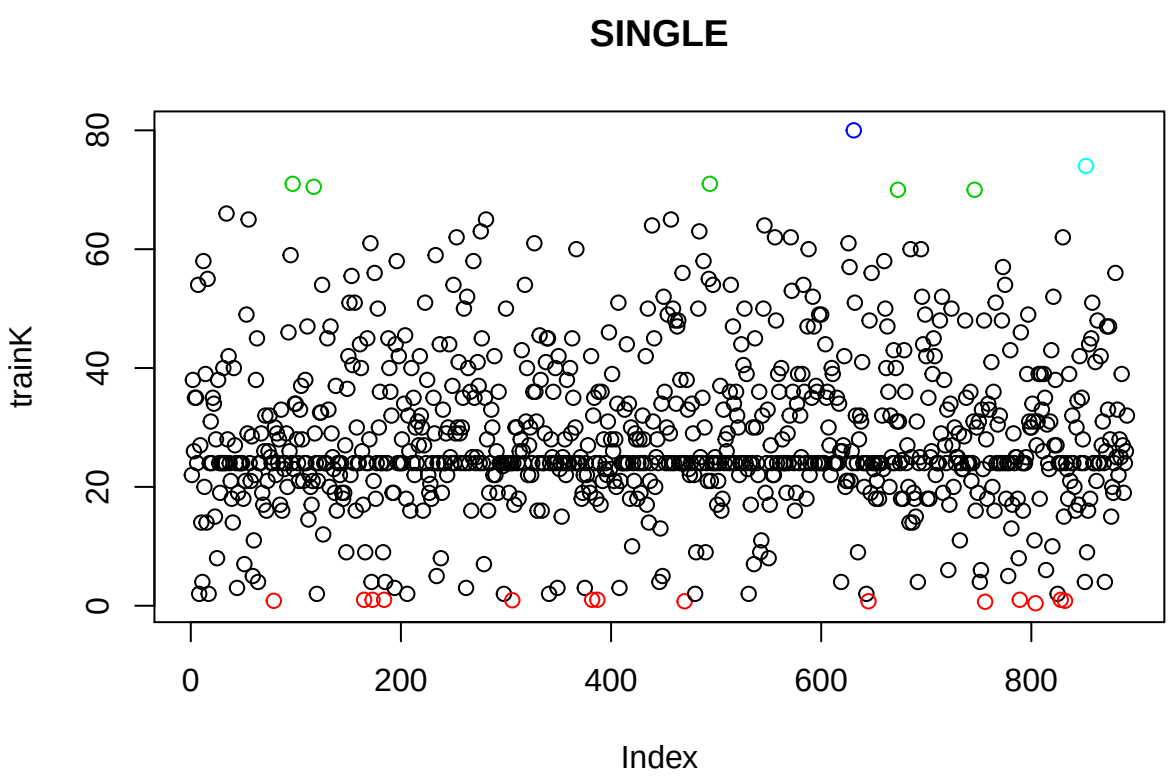
```
cluster = hclust(distancia, method = "single")
plot(cluster)
```




```
#Se determina la altura requerida con k clusters, cortando el dendograma con k clases:
corteD = cutree(cluster, k = 5)
#Observamos la cantidad de clusters
unique(corteD)
```

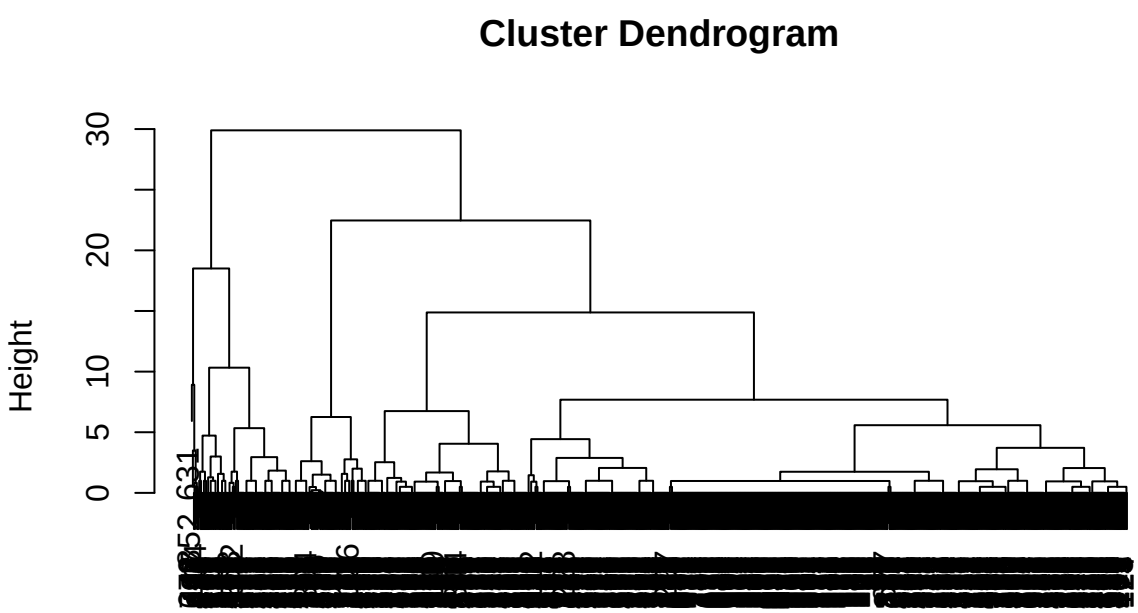
```
## [1] 1 2 3 4 5
```

```
#Graficamos los clusters.
plot(trainK, col = corteD, main = "SINGLE")
```



Para el método Average:

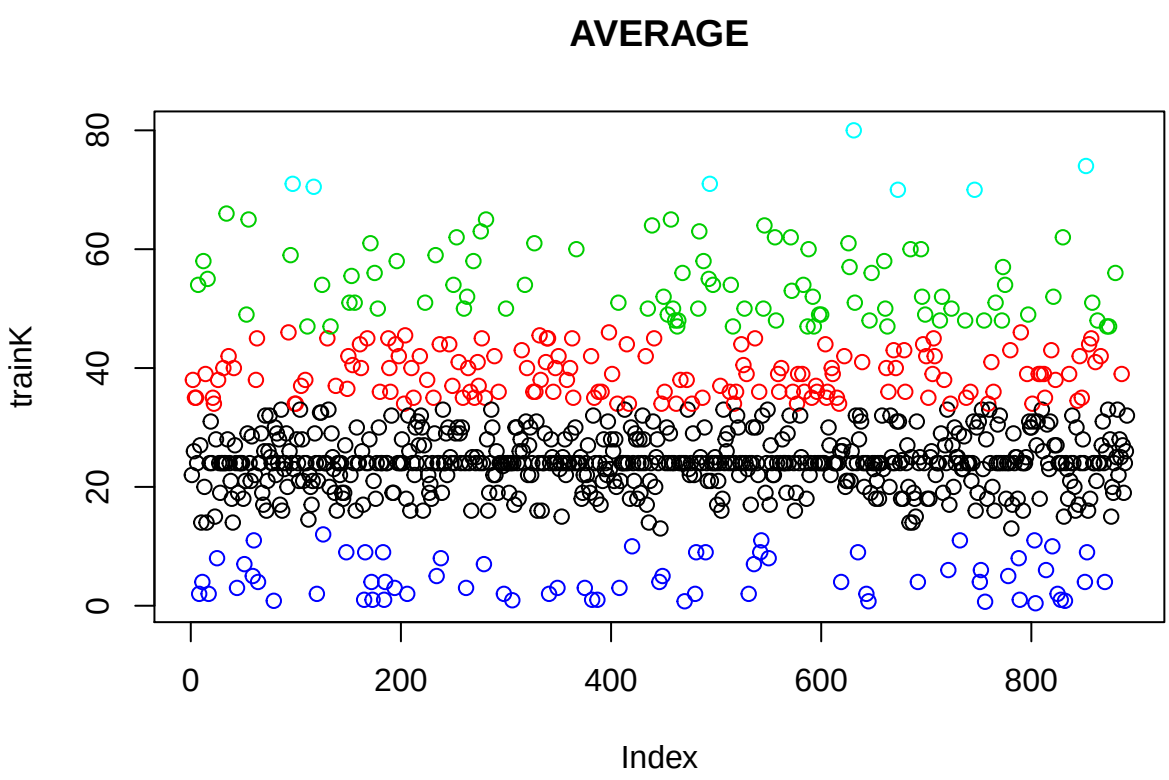
```
cluster = hclust(distancia, method = "average")
plot(cluster)
```




```
#Se determina la altura requerida con k clusters, cortando el dendograma con k clases:
corteD = cutree(cluster, k = 5)
#Observamos la cantidad de clusters.
unique(corteD)
```

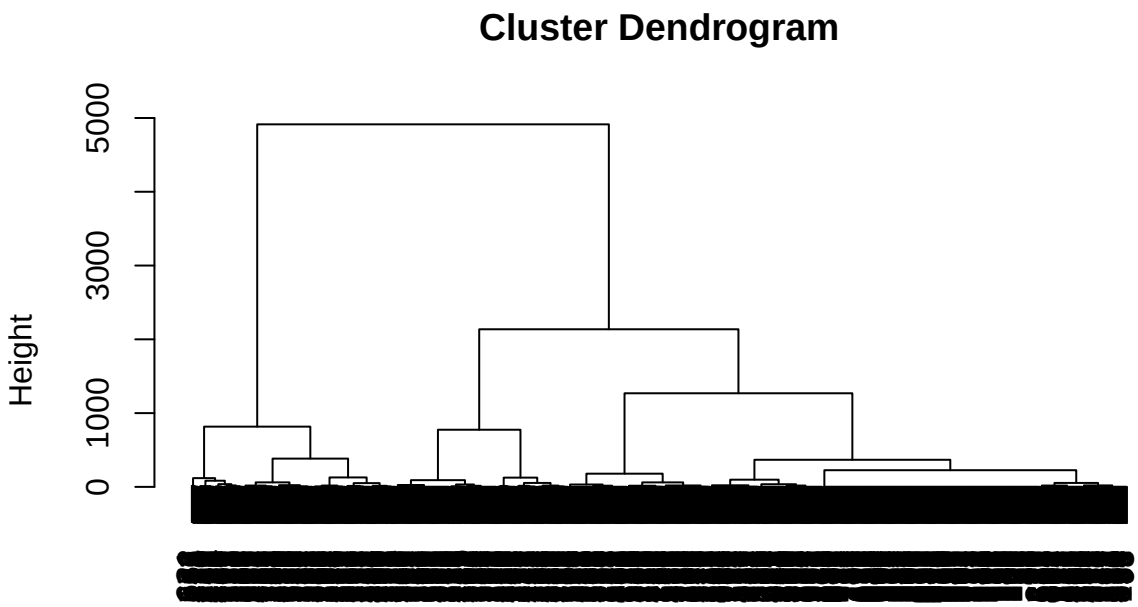
```
## [1] 1 2 3 4 5
```

```
#Graficamos los clusters.
plot(trainK, col = corteD, main = "AVERAGE")
```



Para el método Ward:

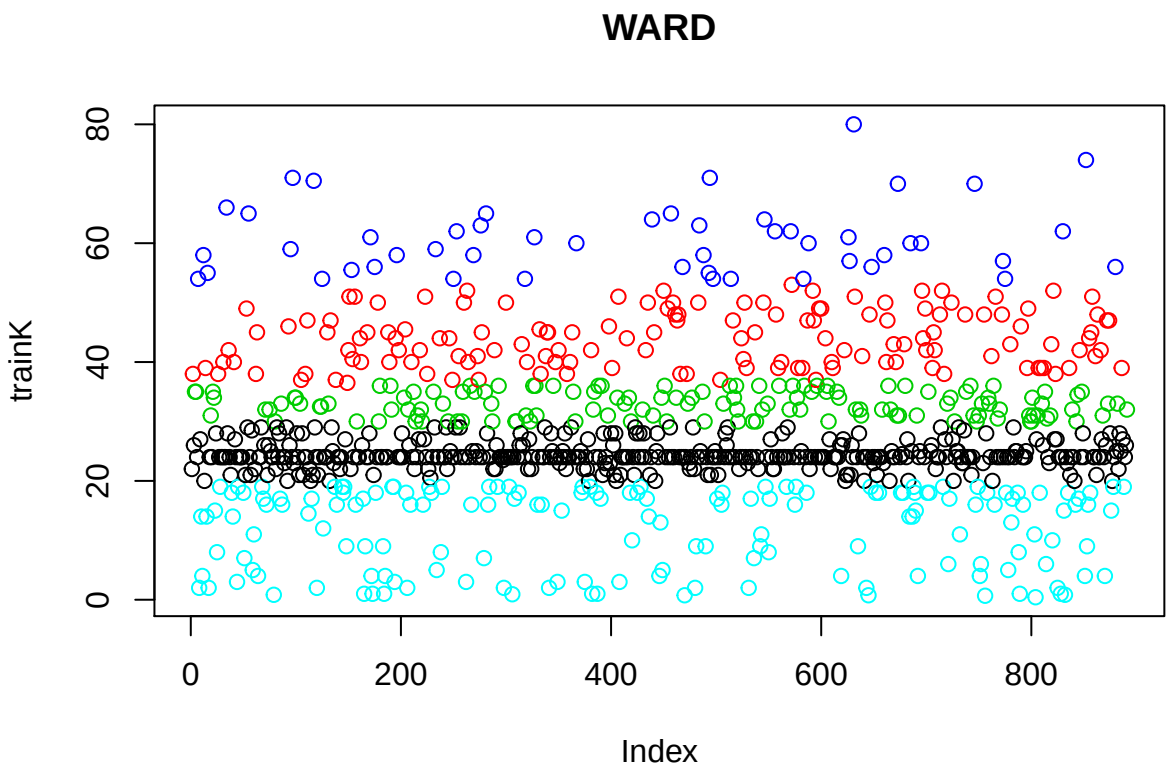
```
cluster = hclust(distancia, method = "ward.D")
plot(cluster)
```




```
#Se determina la altura requerida con k clusters, cortando el dendograma con k clases:
corteD = cutree(cluster, k = 5)
#Observamos la cantidad de clusters.
unique(corteD)
```

```
## [1] 1 2 3 4 5
```

```
#Graficamos los clusters.
plot(trainK, col = corteD, main = "WARD")
```



Se puede observar que la clasificación por el método Ward es la que más se adapta al clustering del dataset utilizando K-Medias con K=5.

Reglas de Asociación.

Se deben incluir las bibliotecas “arules” y “arulesViz” de la siguiente manera:

```
library(arules)
library(arulesViz)
```

Para leer el dataset Titanic Raw se debe ejecutar lo siguiente:

```
load("../data/titanic.raw.Rdata")
str(titanic.raw)
```

```
## 'data.frame':    2201 obs. of  4 variables:
##  $ Class   : Factor w/ 4 levels "1st","2nd","3rd",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Sex     : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Age     : Factor w/ 2 levels "Adult","Child": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Survived: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
#Se transforma dataframe en transaccional.
trans <- as(titanic.raw, "transactions")
```

Seguidamente, ya teniendo el dataset como valores transaccionales se generan las reglas de asociación y se muestra un resumen de las mismas:

```
reglas <- apriori(trans)
```

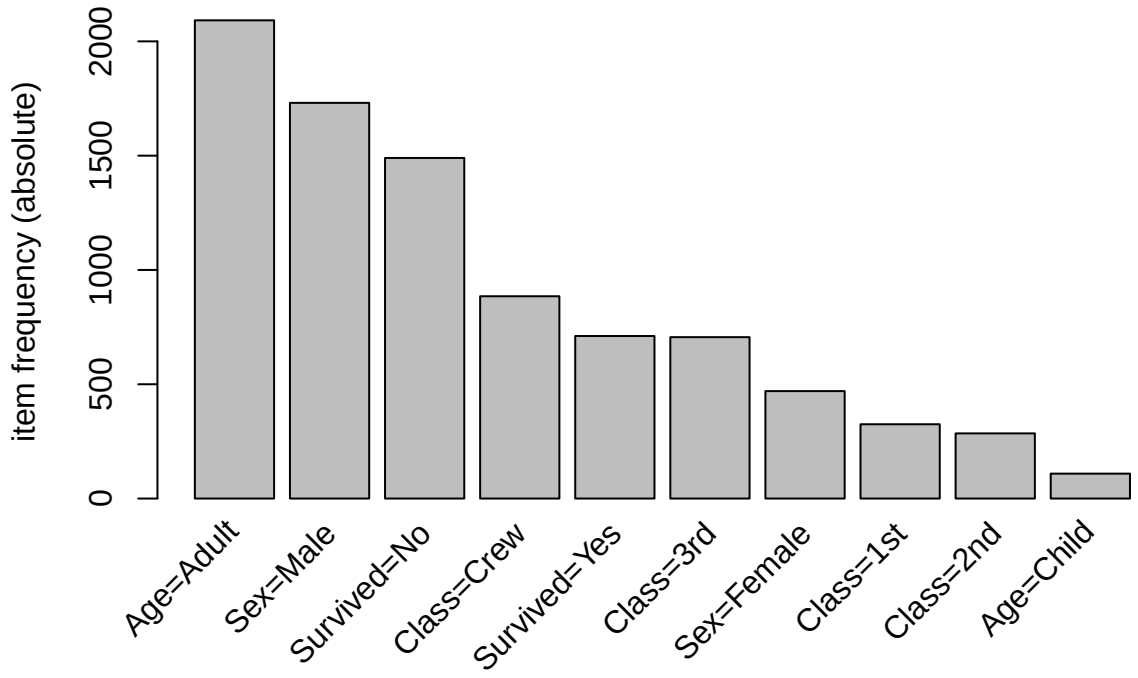
```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 220
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[10 item(s), 2201 transaction(s)] done [0.00s].
## sorting and recoding items ... [9 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [27 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
summary(reglas)
```

```
## set of 27 rules
##
## rule length distribution (lhs + rhs):sizes
##  1  2  3  4
##  1 10 11  5
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   2.741   3.000   4.000
##
## summary of quality measures:
##     support         confidence          lift
##  Min.   :0.1186   Min.   :0.8130   Min.   :0.9344
##  1st Qu.:0.1838   1st Qu.:0.9107   1st Qu.:0.9671
##  Median :0.3044   Median :0.9242   Median :1.0327
##  Mean   :0.3506   Mean   :0.9364   Mean   :1.0650
##  3rd Qu.:0.3969   3rd Qu.:0.9779   3rd Qu.:1.1696
##  Max.   :0.9505   Max.   :1.0000   Max.   :1.2659
##
## mining info:
##   data ntransactions support confidence
##  trans          2201     0.1        0.8
```

Para conocer las 10 transacciones con mayor número de apariciones en el dataset se realiza lo siguiente:

```
itemFrequencyPlot(trans,topN=10,type="absolute")
```



Para ordenar las reglas se tiene lo siguiente:

```
#Se ordenan las reglas por confianza
confianzaAlta <-sort(reglas, by="confidence", decreasing=TRUE)
inspect(head(confianzaAlta))
```

```
##     lhs                                    rhs          support   confidence
## [1] {Class=Crew}                        => {Age=Adult} 0.4020900 1.0000000
## [2] {Class=Crew,Survived=No}            => {Age=Adult} 0.3057701 1.0000000
## [3] {Class=Crew,Sex=Male}               => {Age=Adult} 0.3916402 1.0000000
## [4] {Class=Crew,Sex=Male,Survived=No}   => {Age=Adult} 0.3044071 1.0000000
## [5] {Class=Crew,Survived=No}            => {Sex=Male}  0.3044071 0.9955423
## [6] {Class=Crew,Age=Adult,Survived=No}  => {Sex=Male}  0.3044071 0.9955423
##     lift
## [1] 1.052103
## [2] 1.052103
## [3] 1.052103
## [4] 1.052103
## [5] 1.265851
## [6] 1.265851
```

Se observa que por las primeras seis reglas, las cuales tienen una confianza de 1 o muy cercana a 1, siempre será verídico considerar que una persona perteneciente a la tripulación del Titanic del género masculino que

no haya sobrevivido sea un adulto (100%); o que una persona adulta perteneciente a la tripulación del Titanic y que no haya sobrevivido sea del género masculino (99%). El porcentaje es medido en confianza.

```
#Se ordenan las reglas por soporte
supportAlto <-sort(reglas, by="support", decreasing=TRUE)
inspect(head(supportAlto))
```

```
##     lhs                         rhs          support   confidence lift
## [1] {}                       => {Age=Adult} 0.9504771 0.9504771  1.000000
## [2] {Sex=Male}               => {Age=Adult} 0.7573830 0.9630272  1.013204
## [3] {Survived=No}            => {Age=Adult} 0.6533394 0.9651007  1.015386
## [4] {Survived=No}            => {Sex=Male}  0.6197183 0.9154362  1.163995
## [5] {Sex=Male,Survived=No}   => {Age=Adult} 0.6038164 0.9743402  1.025106
## [6] {Age=Adult,Survived=No}  => {Sex=Male}  0.6038164 0.9242003  1.175139
```

Se observa que las primeras seis reglas son las que más frecuentan en el dataset; en estas se puede notar que si una persona no sobrevivió fue porque era un adulto (65%) o era del género masculino (61%); o que si una persona del género masculino que no sobrevivió fue porque era un adulto (60%); o si una persona adulta que no sobrevivió fue porque era del género masculino (60%). El porcentaje es medido en soporte.

```
#Se ordenan las reglas por lift
liftAlto <-sort(reglas, by="lift", decreasing=TRUE)
inspect(head(liftAlto))
```

```
##     lhs                                    rhs           support
## [1] {Class=Crew,Survived=No}            => {Sex=Male}    0.3044071
## [2] {Class=Crew,Age=Adult,Survived=No}  => {Sex=Male}    0.3044071
## [3] {Class=Crew}                        => {Sex=Male}    0.3916402
## [4] {Class=Crew,Age=Adult}              => {Sex=Male}    0.3916402
## [5] {Class=3rd,Sex=Male,Age=Adult}      => {Survived=No} 0.1758292
## [6] {Class=3rd,Sex=Male}                => {Survived=No} 0.1917310
##     confidence lift
## [1] 0.9955423  1.265851
## [2] 0.9955423  1.265851
## [3] 0.9740113  1.238474
## [4] 0.9740113  1.238474
## [5] 0.8376623  1.237379
## [6] 0.8274510  1.222295
```

Se observa que las primeras seis reglas son las más probables en ocurrir; en estas se puede notar que si una persona no sobrevivió y pertenecía a la tripulación fue porque o era del género masculino; o si era una persona adulta del género masculino perteneciente a la 3ra clase pasajera, entonces no sobrevivió.

Árboles de Decisión

Se cargan las bibliotecas para la utilización del método de clasificación de árboles de decisión:

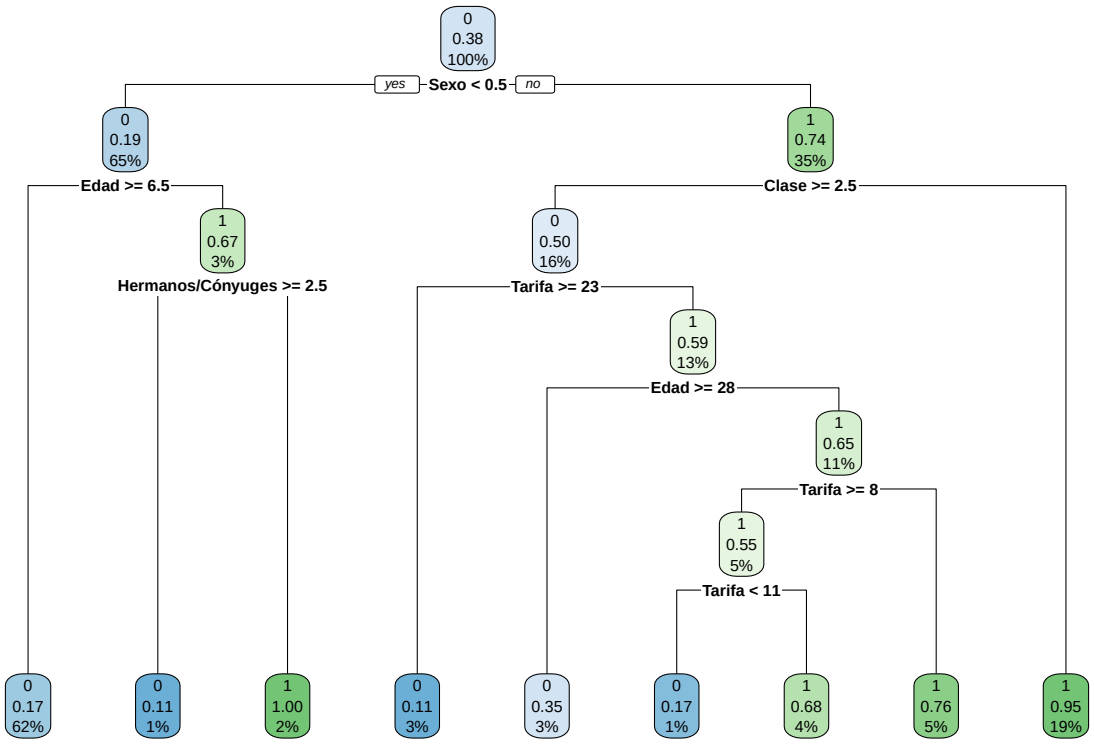
```
library("rpart")
library("rpart.plot")
```

Para esta técnica se trabajará sólo con las columnas “Sobrevivio”, “Clase”, “Sexo”, “Edad”, “Hermanos/Cónyugues”, “Padres/Niños” y “Tarifa”.

```
trainT <- subset(train, select = c(1,2,4,5,6,7,9))
```

Se obtiene el árbol de decisión en base a la columna Clase del dataset y se grafica:

```
tree <- rpart(Sobrevivio ~ ., trainT, method = "class")
rpart.plot(tree)
```



Se muestran las predicciones en base al árbol anteriormente generado:

```
p <- predict(tree, trainT, type="class")
table(trainT[,1], p)
```

```
##    p
##       0   1
##   0 517  32
##   1 107 235
```

Máquinas de Soporte Vectorial

Se debe incluir la biblioteca "e1071" de la siguiente manera:

```
library("e1071")
```

Para esta técnica se trabajará sólo con las columnas "Sobrevivio", "Clase", "Sexo", "Edad", "Hermanos/Cónyugues", "Padres/Niños" y "Tarifa" para el conjunto de Training; y las mismas para uno segundo sin incluir la columna "Sobrevivio".

```
trainSVM <- subset(train, select = c(1,2,4,5,6,7,9))
trainSVM2 <- subset(train, select = c(2,4,5,6,7,9))
```

Ahora bien, se debe entrenar el modelo probando diferentes kernels. Se entrena el modelo con el kernel "sigmoid".

```
svm_model <- svm(Sobrevivio ~ .,kernel="sigmoid",data = trainSVM)
pred <- predict(svm_model,trainSVM2)
pred <- replace(pred, pred < 0.5, 0)
pred <- replace(pred, pred >= 0.5, 1)
```

Se muestra la matriz de confusión.

```
table(pred, trainSVM$Sobrevivio)
```

```
##
## pred   0   1
##    0 344 179
##    1 205 163
```

Se entrena el modelo con el kernel "radial".

```
svm_model <- svm(Sobrevivio ~ .,kernel="radial",data = trainSVM)
pred <- predict(svm_model,trainSVM2)
pred <- replace(pred, pred < 0.5, 0)
pred <- replace(pred, pred >= 0.5, 1)
```

Se muestra la matriz de confusión.

```
table(pred, trainSVM$Sobrevivio)
```

```
##
## pred   0   1
##    0 493  91
##    1  56 251
```

Se entrena el modelo con el kernel "polynomial".

```
svm_model <- svm(Sobrevivio ~ .,kernel="polynomial",data = trainSVM)
pred <- predict(svm_model,trainSVM2)
pred <- replace(pred, pred < 0.5, 0)
pred <- replace(pred, pred >= 0.5, 1)
```

Se muestra la matriz de confusión.

```
#Se muestra la matriz de confusión.
table(pred, trainSVM$Sobrevivio)
```

```
##
## pred   0   1
##    0 486 106
##    1  63 236
```

Se entrena el modelo con el kernel "linear".

```
svm_model <- svm(Sobrevivio ~ .,kernel="linear",data = trainSVM)
pred <- predict(svm_model,trainSVM2)
pred <- replace(pred, pred < 0.5, 0)
pred <- replace(pred, pred >= 0.5, 1)
```

Se muestra la matriz de confusión.

```
table(pred, trainSVM$Sobrevivio)
```

```
##
## pred   0   1
##    0 468 109
##    1  81 233
```

El mejor método fue "radial", por lo que se afina el modelo considerando este kernel.

```
svm_tune <- 0
x <- subset(trainSVM, select = -Sobrevivio)
y <- trainSVM$Sobrevivio
svm_tune <- tune(svm, train.x=x, train.y=y, kernel="radial", ranges=list(cost=10^(-1:2), gamma=c(.5,1,2)
print(svm_tune)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1   0.5
##
## - best performance: 0.1544322
```

Ahora, se evalúa el modelo luego de afinarlo y conseguir el cost y gamma óptimos.

```
svm_model_after_tune <- svm(Sobrevivio ~ ., kernel="radial", data = trainSVM, cost=1, gamma=0.5)
summary(svm_model_after_tune)
```

```
##
## Call:
## svm(formula = Sobrevivio ~ ., data = trainSVM, kernel = "radial",
##     cost = 1, gamma = 0.5)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.5
##     epsilon:  0.1
##
##
## Number of Support Vectors:  443
```

```
pred <- predict(svm_model_after_tune,trainSVM2)
pred <- replace(pred, pred < 0.5, 0)
pred <- replace(pred, pred >= 0.5, 1)
```

Se muestra la matriz de confusión.

```
table(pred,trainSVM$Sobrevivio)
```

```
##
## pred   0   1
##    0 504  92
##    1  45 250
```

Curvas ROC

Se debe incluir la biblioteca “pROC” de la siguiente manera:

```
library("pROC")
```

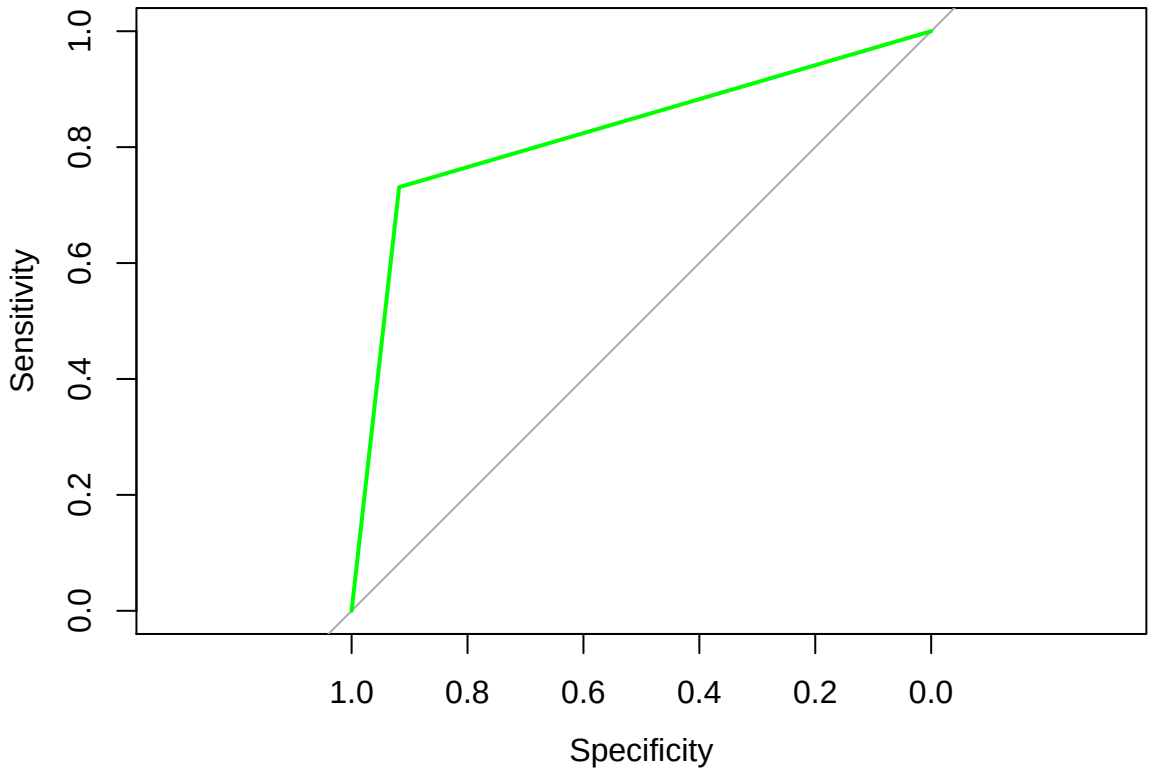
```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

Se evaluará y se graficará la curva ROC para árboles de decisión como sigue:

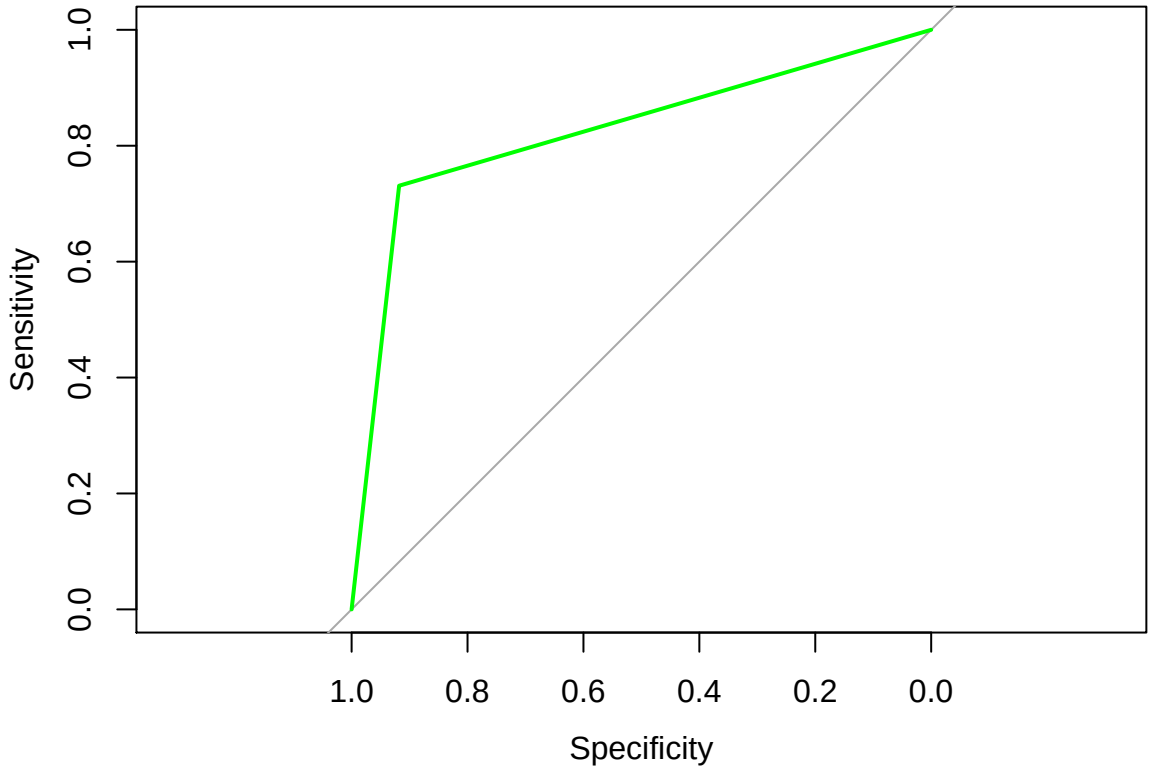
```
pTree <- as.numeric(p)
pTree = (pTree==2)*1
svmTree <- roc(train$Sobrevivio, pred)
plot(svmTree, type = "l", col = "green")
```




```
##
## Call:
## roc.default(response = train$Sobrevivio, predictor = pred)
##
## Data: pred in 549 controls (train$Sobrevivio 0) < 342 cases (train$Sobrevivio 1).
## Area under the curve: 0.8245
```

Ahora evaluará y se graficará la curva ROC para el modelo SVM:

```
pred <- as.numeric(pred)
svmROC <- roc(train$Sobrevivio, pred)
plot(svmROC, type = "l", col = "green")
```




```
##
## Call:
## roc.default(response = train$Sobrevivio, predictor = pred)
##
## Data: pred in 549 controls (train$Sobrevivio 0) < 342 cases (train$Sobrevivio 1).
## Area under the curve: 0.8245
```

Finalmente, se tiene que el área bajo la curva ROC es de 0.8245, por lo que para ambos modelos se predecirá con un 82% la supervivencia de una persona en la tragedia del Titanic.