a.csv

PedroFlores

10 de abril de 2016

TAREA3 #: Aprendizaje No Supervisado

Bibliotecas a usar:

```
options(repos="https://cran.rstudio.com" )
install.packages("devtools")

library(devtools)

install.packages("rgl")

library(rgl)
```

Funciones a usar:

```
# metodo codo de jambu
codo_jambu= function(data){
  data_s <- data
  t <- (nrow(data_s)-1)*sum(apply(data_s,2,var))
  for (i in 2:15) t[i] <- sum(kmeans(data_s, centers=i)$withinss)
  plot(1:15, t, type="b", xlab="# Clusters", ylab="groups")
}



#matriz de confusion
confusion_matrix = function(cl, clust)
{
  return(table(True = cl, Predicted = clust))
}

#diagonal de matriz
sum_diagonal = function(matrix)
{
  suma = 0;

  for (i in 1:ncol(matrix))
  {
  suma = suma + matrix[i,i]
  }

  return(suma)
```

```
}


#evaluar matriz de confusion

evaluation = function(Matrix, Rows)
{
  aciertos = sum_diagonal(Matrix)
  Tasa_aciertos = (aciertos/Rows) * 100
  Tasa_fallos = 100 - Tasa_aciertos

 print( paste("Tasa de aciertos: ", Tasa_aciertos))
 print( paste("Tasa de fallos:  ",Tasa_fallos) )
}
```

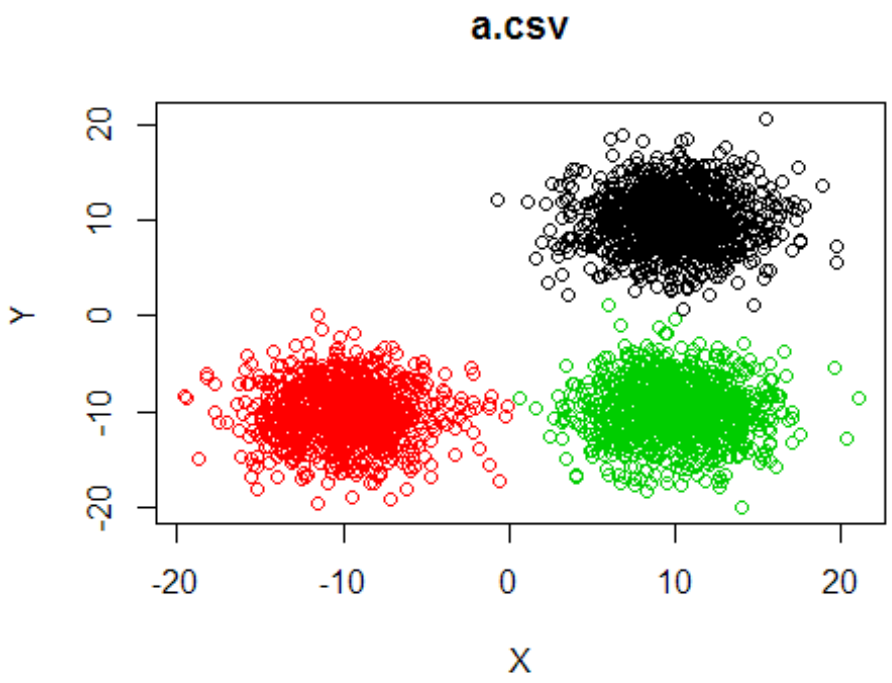
Estudio de cada data set

a.csv:

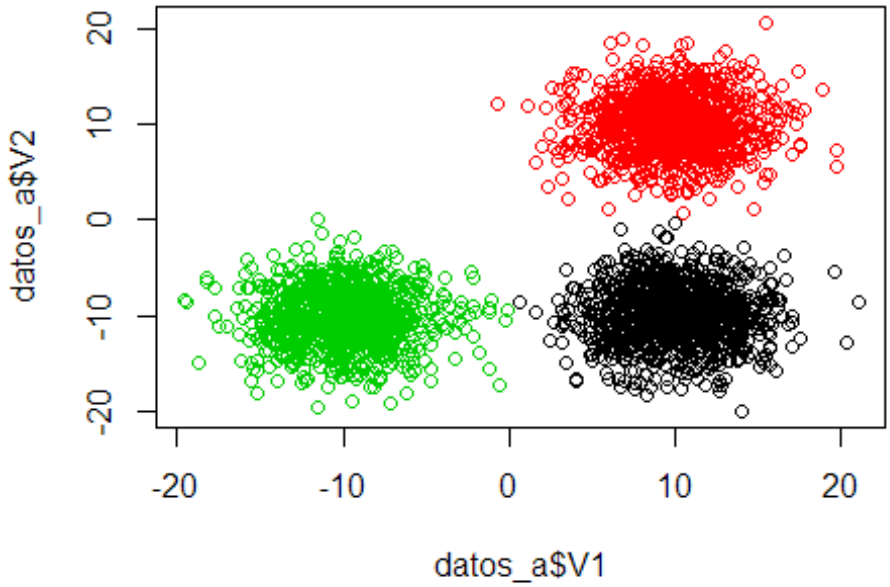
```
datos_a = read.csv("a.csv",header = F)

#analisis exploratorio
summary(datos_a)
```

```
##        V1                 V2                V3
##  Min.   :-19.612   Min.   :-20.006   Min.   :0
##  1st Qu.: -8.100   1st Qu.:-11.055   1st Qu.:0
##  Median :  7.869   Median : -8.121   Median :1
##  Mean   :  3.294   Mean   : -3.392   Mean   :1
##  3rd Qu.: 10.846   3rd Qu.:  7.943   3rd Qu.:2
##  Max.   : 21.015   Max.   : 20.680   Max.   :2
```

```
#redefinir las clases, para que esten en un rango de [1:3] , en vez de [0
:2]
datos_a$V3 = as.numeric(datos_a$V3)
datos_a$V3 = datos_a$V3 + 1


# visualizar datos:
plot(datos_a[,1:2], col = datos_a$V3,
     xlab = "X", ylab = "Y",
     main = "a.csv")
```




```r
# Podemos observar 3 cluster con forma de esfera


#Aplicamos K medias al data set a.csv


a_kmean = kmeans(x = datos_a[, c("V1", "V2")],
                 centers = 3)


#observamos cómo queda la gráfica luego de aplicar el modelo

plot(x = datos_a$V1,
     y = datos_a$V2,
     col = a_kmean$cluster)


points(x = a_kmean$centers[, c("V1", "V2")],
       col = 1:4, pch = 19, cex = 3)
```



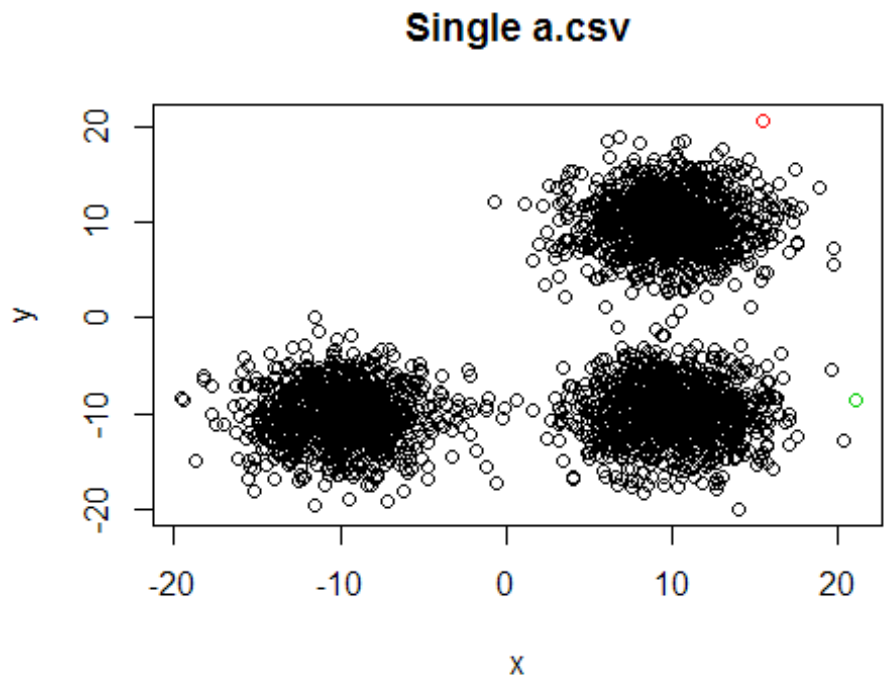

```
#matriz de confusion para k means y evaluación

matrix_kmeans_a = confusion_matrix(datos_a$V3, a_kmean$cluster)

evaluation(matrix_kmeans_a, nrow(datos_a))

## [1] "Tasa de aciertos:  0"
## [1] "Tasa de fallos:   100"
```

Observaciones:

Podemos apreciar que k medias funciona correctamente para este data set, en el cual hay una buena cantidad de aciertos en su matriz de confusión

```
###############HCLUST#########################

#matriz de distancias

a = datos_a
a$V3 = NULL
a = as.matrix(a)
distance_a = dist(a)
```

```r
#modelo single

modelo_single_a = hclust(distance_a, method="single")
corte_a1= cutree(modelo_single_a, k = 3)

plot(x = a[,1], y = a[,2], main = paste(c("Single", "a.csv") , collapse =
" "),
    xlab = "x", ylab = "y",
    col = corte_a1)
```




```r
#matriz de confusion para k single y evaluacion

matrix_single_a = confusion_matrix(datos_a$V3, corte_a1)

evaluation(matrix_single_a, nrow(datos_a))
```

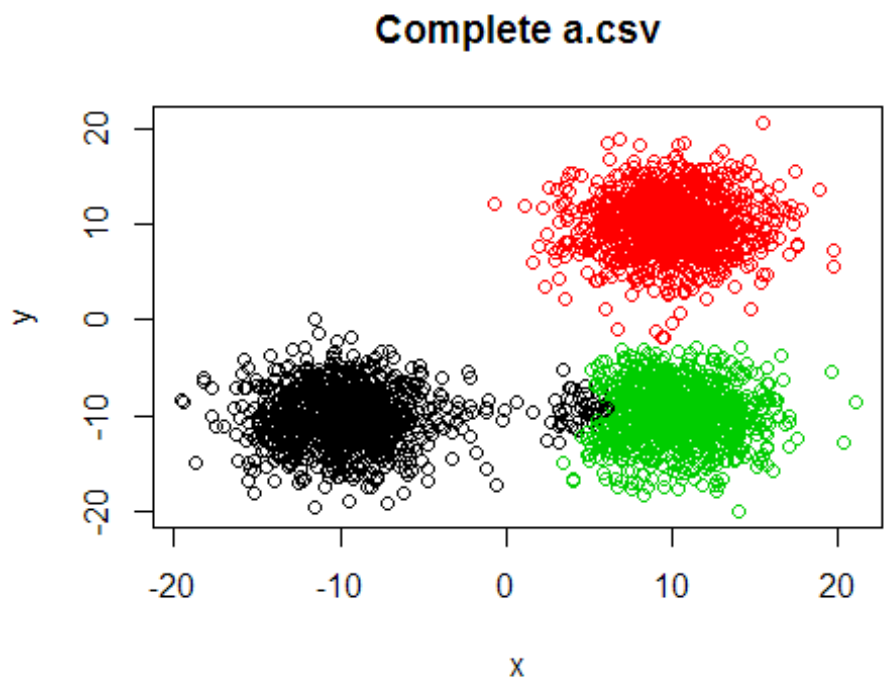
```
## [1] "Tasa de aciertos:  33.3333333333333"
## [1] "Tasa de fallos:   66.6666666666667"
```

Observaciones:

H clust simple parece no funcionar bien del todo para este data set, vemos una baja tasa de aciertos, además de que no está asignando bien los datos a los distintos cluster

```
###############HCLUST#########################
```

```r
#modelo Complete

modelo_complete_a = hclust(distance_a, method="complete")
corte_a2= cutree(modelo_complete_a, k = 3)

plot(x = a[,1], y = a[,2], main = paste(c("Complete", "a.csv") , collapse
= " "),
     xlab = "x", ylab = "y",
     col = corte_a2)
```



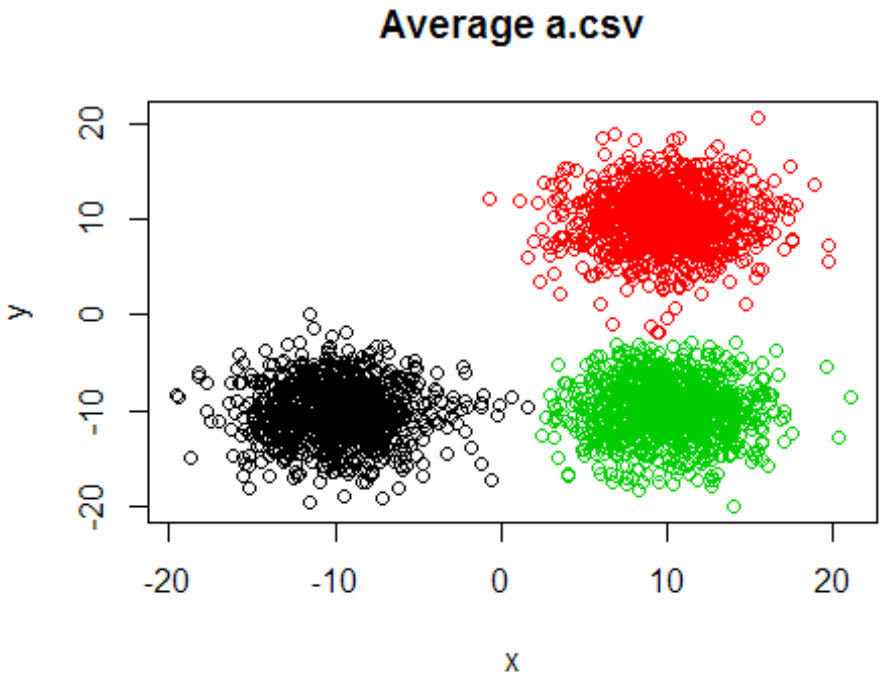

```r
#matriz de confusion para complete y evaluacion
matrix_complete_a = confusion_matrix(datos_a$V3, corte_a2)

evaluation(matrix_complete_a , nrow(datos_a))
```

```
## [1] "Tasa de aciertos:  31.5666666666667"
## [1] "Tasa de fallos:   68.4333333333333"
```

Observaciones:

H clust complete tambien muestra una baja tasa de aciertos, sin embargo asigna los elementos de los cluster de forma casi óptima

```
#modelo average

modelo_average_a = hclust(distance_a, method="average")
corte_a3= cutree(modelo_average_a, k = 3)

plot(x = a[,1], y = a[,2], main = paste(c("Average", "a.csv") , collapse
= " "),
     xlab = "x", ylab = "y",
     col = corte_a3)
```




```
#matriz de confusion para average y evaluacion
matrix_single_a = confusion_matrix(datos_a$V3, corte_a3)

evaluation(matrix_single_a, nrow(datos_a))
```

```
## [1] "Tasa de aciertos:  33.0666666666667"
## [1] "Tasa de fallos:   66.9333333333333"
```

Observaciones:

H clust average muestra un incremento en la tasa de aciertos respecto a h clust complete, además realiza una correcta asignación de los elementos a los clusters

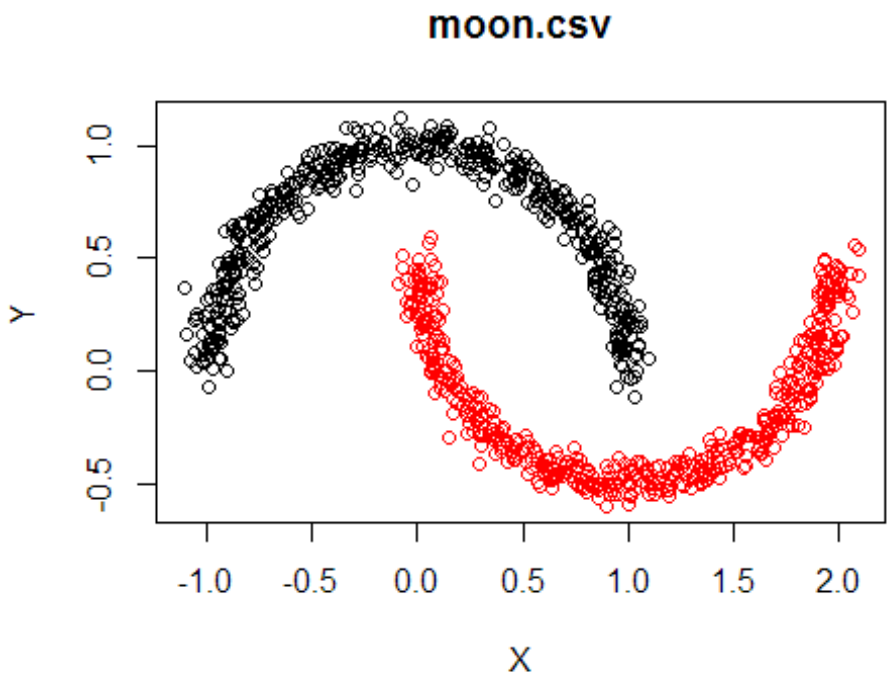
MOON.CSV

```
datos_moon = read.csv("moon.csv",header = F)

#analisis exploratorio
summary(datos_moon)
```

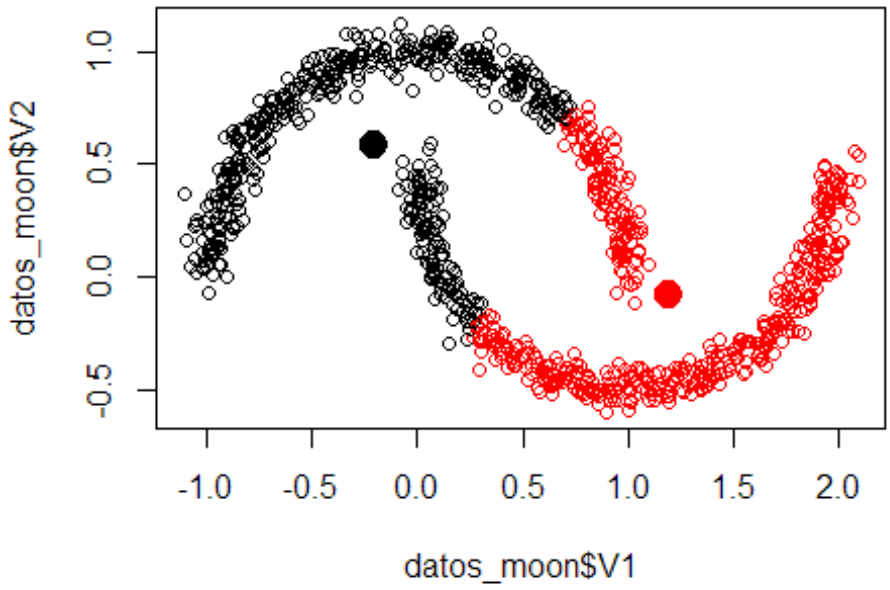
```
##        V1                  V2                 V3
##  Min.   :-1.11299   Min.   :-0.5994   Min.   :0.0
##  1st Qu.:-0.03567   1st Qu.:-0.2145   1st Qu.:0.0
##  Median : 0.48495   Median : 0.2462   Median :0.5
##  Mean   : 0.49760   Mean   : 0.2498   Mean   :0.5
##  3rd Qu.: 1.03710   3rd Qu.: 0.7040   3rd Qu.:1.0
##  Max.   : 2.09638   Max.   : 1.1232   Max.   :1.0
```

```
#redefinir las clases, para que esten en un rango de [1:2] , en vez de [0
:2]
datos_moon$V3 = as.numeric(datos_moon$V3)
datos_moon$V3 = datos_moon$V3 + 1
```

```
# visualizar datos:
plot(datos_moon[,1:2], col = datos_moon$V3,
     xlab = "X", ylab = "Y",
     main = "moon.csv")
```



```r
#se observan dos cluster con forma de parabola de similar longitud


#aplicamos el k medias

moon_kmean = kmeans(x = datos_moon[, c("V1", "V2")],
                    centers = 2)


plot(x = datos_moon$V1,
     y = datos_moon$V2,
     col = moon_kmean$cluster)


points(x = moon_kmean$centers[, c("V1", "V2")],
       col = 1:4, pch = 19, cex = 2)
```



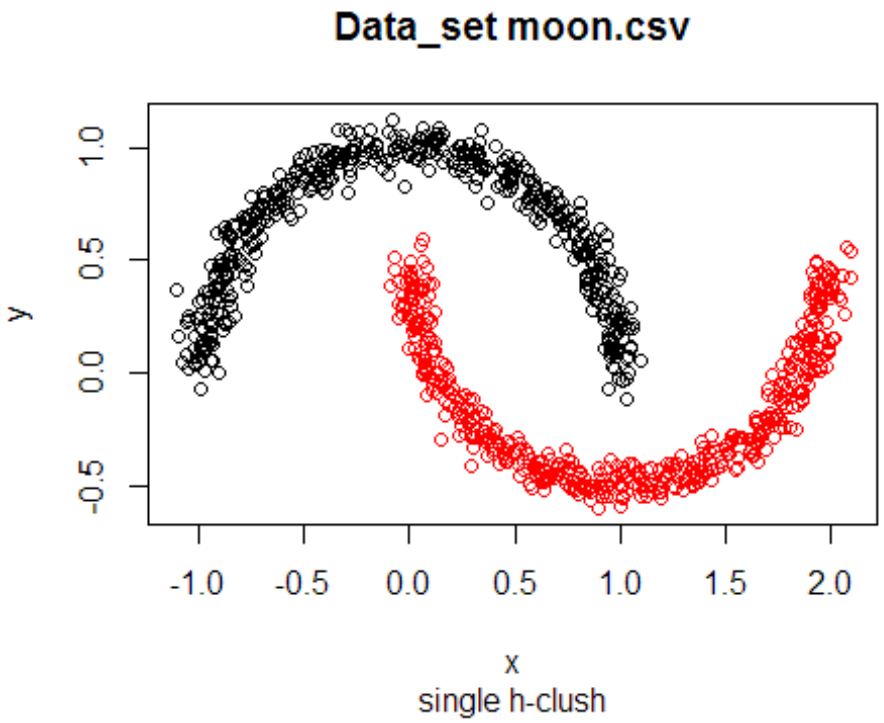

```r
#matriz de confusion para kmeans y evaluacion

matrix_kmeans_moon = confusion_matrix(datos_moon$V3, moon_kmean$cluster)

evaluation(matrix_kmeans_moon, nrow(datos_moon))
```

```
## [1] "Tasa de aciertos:  75.6"
## [1] "Tasa de fallos:   24.4"
```

Observaciones:

Podemos apreciar que k medias funciona con una tasa de aciertos alta, y asigna de buena manera los elementos en cada cluster

```
###############HCLUST#########################

#matriz de distancias

moon = datos_moon
moon$V3 = NULL
moon = as.matrix(moon)
distance_moon = dist(moon)



#modelo single

modelo_single_moon = hclust(distance_moon, method="single")
corte_moon1= cutree(modelo_single_moon, k = 2)

plot(x = moon[,1], y = moon[,2], main = paste(c("Data_set", "moon.csv") ,
collapse = " "),
     sub = paste(c("single", "h-clush"), collapse= " "),
     xlab = "x", ylab = "y",
     col = corte_moon1)
```



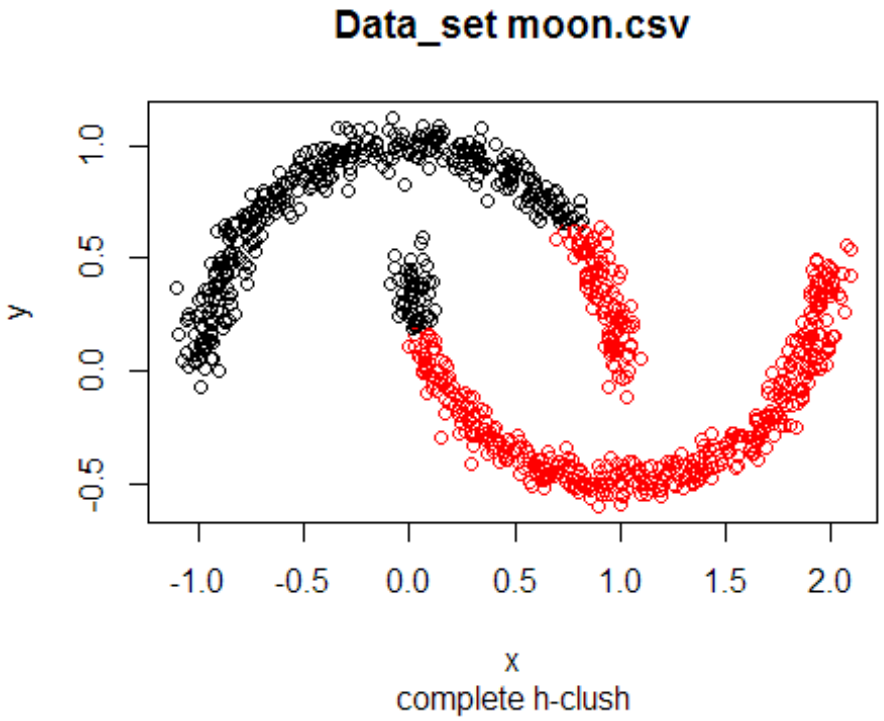
```r
#matriz de confusion para single y evaluacion

matrix_single_moon = confusion_matrix(datos_moon$V3, corte_moon1)

evaluation(matrix_single_moon, nrow(datos_moon))
```

```
## [1] "Tasa de aciertos:  100"
## [1] "Tasa de fallos:   0"
```

Observaciones:

H clust simple parece funciona perfectamente para este data set, presenta una alta tasa de aciertos y asigna correctamente los elementos a los clusters

```r
#modelo Complete

modelo_complete_moon = hclust(distance_moon, method="complete")
corte_moon2= cutree(modelo_complete_moon, k = 2)

plot(x = moon[,1], y = moon[,2], main = paste(c("Data_set", "moon.csv") ,
collapse = " "),
     sub = paste(c("complete", "h-clush"), collapse= " "),
     xlab = "x", ylab = "y",
     col = corte_moon2)
```



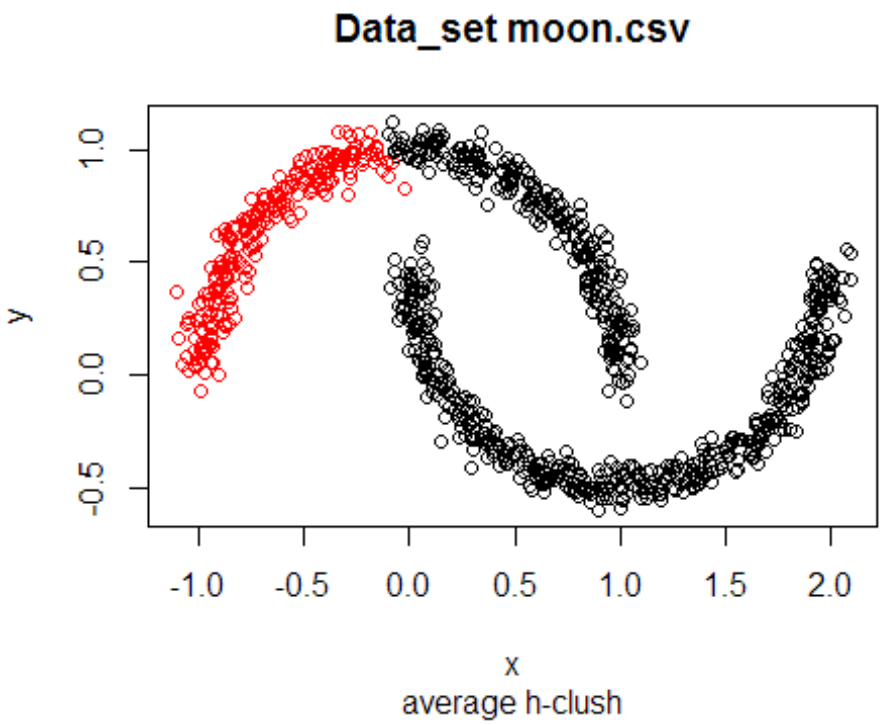
```r
#matriz de confusion para complete y evaluacion

matrix_complete_moon = confusion_matrix(datos_moon$V3, corte_moon2)

evaluation(matrix_complete_moon, nrow(datos_moon))
```

```
## [1] "Tasa de aciertos:  83.6"
## [1] "Tasa de fallos:   16.4"
```

Observaciones:

H clust complete tambien muestra una tasa de aciertos menor a H clust simple, sin embargo sigue siendo bastante alta

```r
modelo_average_moon = hclust(distance_moon, method="average")
corte_moon3= cutree(modelo_average_moon, k = 2)

plot(x = moon[,1], y = moon[,2], main = paste(c("Data_set", "moon.csv") ,
collapse = " "),
     sub = paste(c("average", "h-clush"), collapse= " "),
     xlab = "x", ylab = "y",
     col = corte_moon3)
```




```r
#matriz de confusion para average y evaluacion

matrix_average_moon = confusion_matrix(datos_moon$V3, corte_moon3)
```

```r
evaluation(matrix_average_moon, nrow(datos_moon))
```

```
## [1] "Tasa de aciertos:  26.6"
## [1] "Tasa de fallos:   73.4"
```

Observaciones:

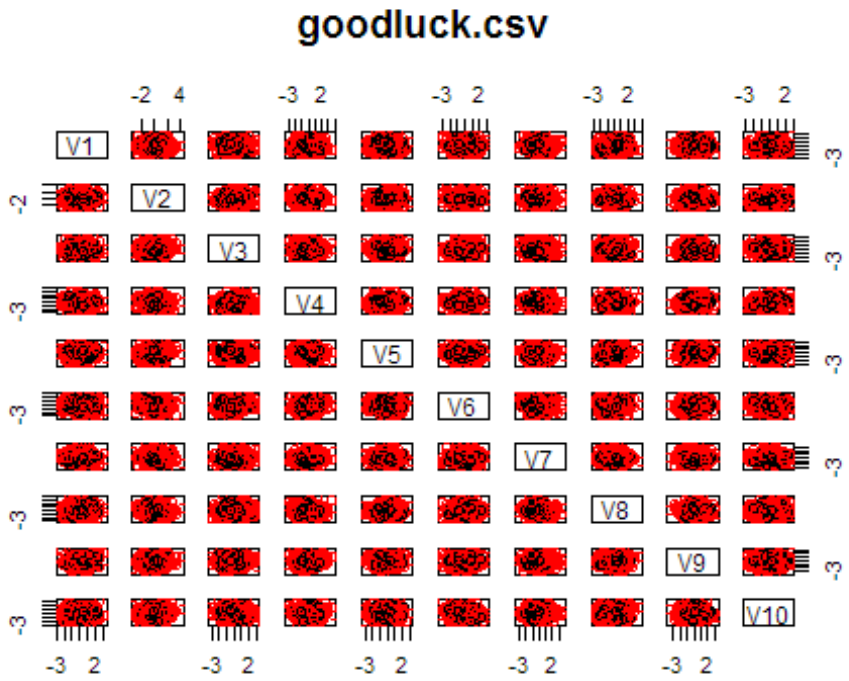
H clust average no muestra una buena tasa de aciertos, era de esperar por la forma en que se distribuyen los datos. El método óptimo para este data set parece ser h clust simple

good_luck.csv

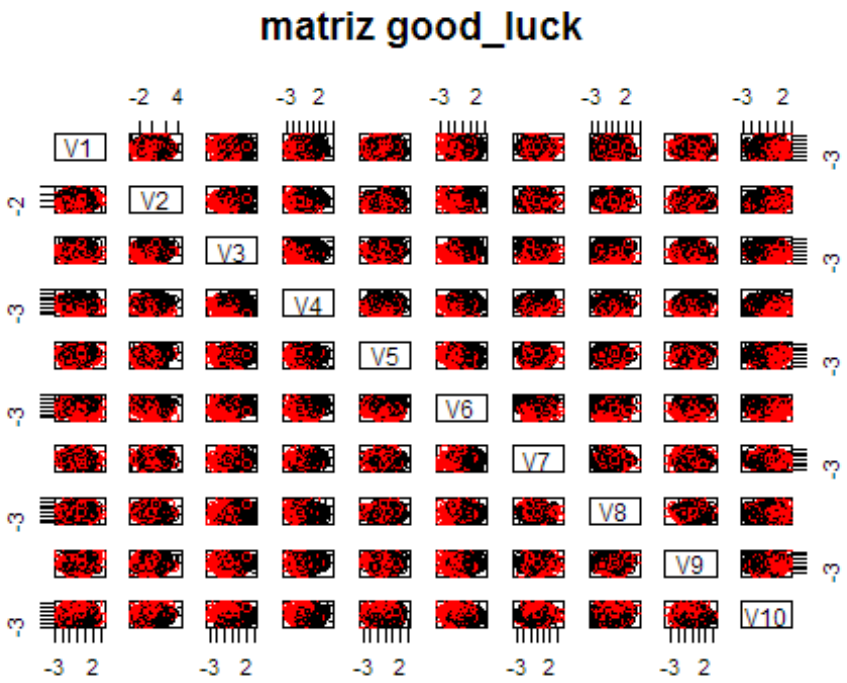
```r
datos_good = read.csv("good_luck.csv",header = F)

#analisis exploratorio
summary(datos_moon)
```

```
##        V1                  V2                 V3
##  Min.   :-1.11299   Min.   :-0.5994   Min.   :1.0
##  1st Qu.:-0.03567   1st Qu.:-0.2145   1st Qu.:1.0
##  Median : 0.48495   Median : 0.2462   Median :1.5
##  Mean   : 0.49760   Mean   : 0.2498   Mean   :1.5
##  3rd Qu.: 1.03710   3rd Qu.: 0.7040   3rd Qu.:2.0
##  Max.   : 2.09638   Max.   : 1.1232   Max.   :2.0
```

```r
#redefinir las clases, para que esten en un rango de [1:2], en vez de  [0
:2]
datos_good$V11 = as.numeric(datos_good$V11)
datos_good$V11 = datos_good$V11 + 1


# visualizar datos:
plot(datos_good[,1:10], col = datos_good$V11,
     main = "goodluck.csv")
```




```
good_kmean = kmeans(x = datos_good[, 1:10],
                    centers = 2)

plot(datos_good[,1:10], col= good_kmean$cluster,main="matriz good_luck")
```



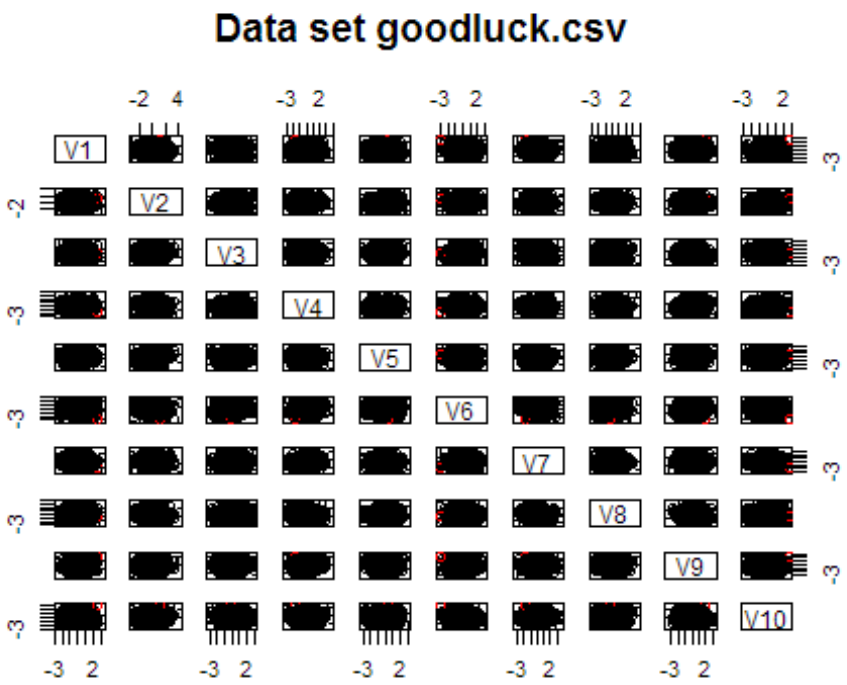

```
#matriz de confusion para kmean y evaluacion

matrix_kmeans_good = confusion_matrix(datos_good$V11, good_kmean$cluster)

evaluation(matrix_kmeans_good, nrow(datos_good))

## [1] "Tasa de aciertos:  47.1"
## [1] "Tasa de fallos:   52.9"
```

Observaciones:

Podemos apreciar que k medias tiene una tasa de aciertos ligeramente superior al 50%, el modelo no funciona del todo bien

```
###############  HCLUST  ########################

#matriz de distancias

good = datos_good
good$V11 = NULL
good = as.matrix(good)
distance_good = dist(good)
```

```
#modelo single

modelo_single_good = hclust(distance_good, method="single")
corte_good1= cutree(modelo_single_good , k = 2)
good= as.data.frame(good)

plot(good[,1:10],
     main = paste(c("Data set", "goodluck.csv"), collapse = " "),
     col = corte_good1)
```




```
#matriz de confusion para single y evaluacion

matrix_single_good = confusion_matrix(datos_good$V11, corte_good1)

evaluation(matrix_single_good, nrow(datos_good))
```

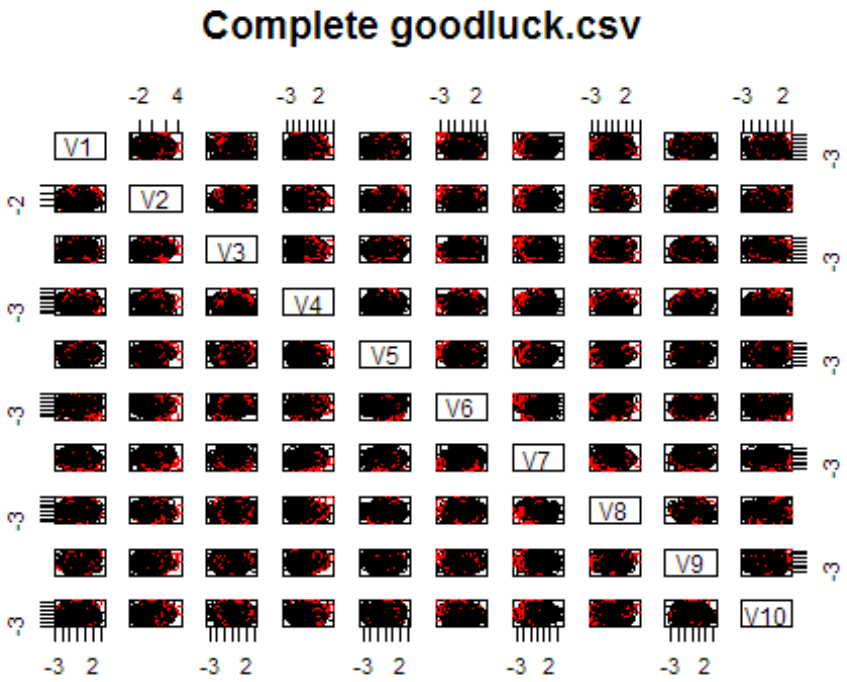
```
## [1] "Tasa de aciertos:  51.4"
## [1] "Tasa de fallos:   48.6"
```

Observaciones:

H clust simple presenta una tasa de aciertos similar al k-medias

```
#modelo Complete

modelo_complete_good = hclust(distance_good, method="complete")
```

```
corte_good2= cutree(modelo_complete_good, k = 2)
good= as.data.frame(good)

plot(good[,1:10],
     main = paste(c("Complete", "goodluck.csv"), collapse = " "),
     col = corte_good2)
```



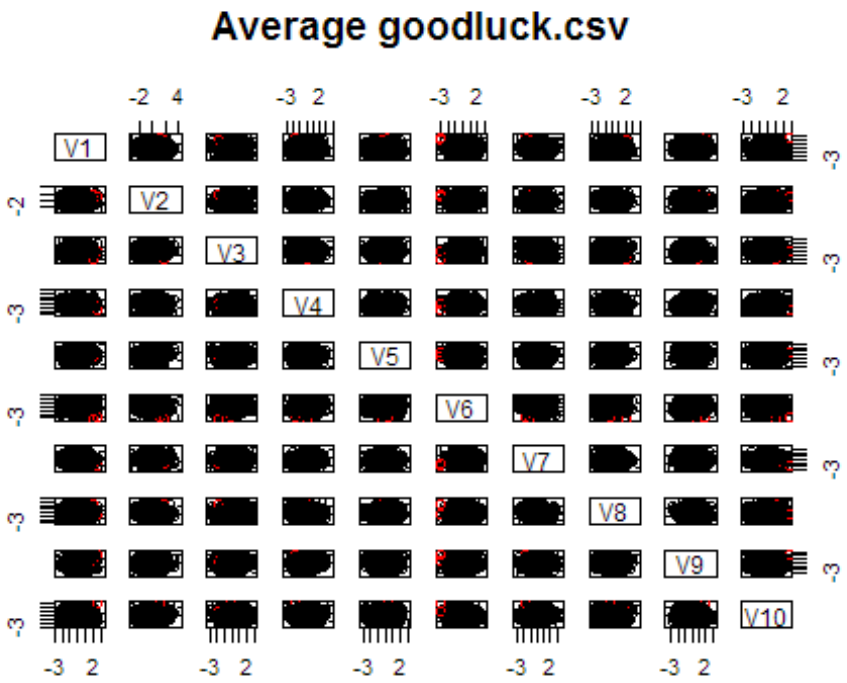

```
#matriz de confusion para complete y evaluacion

matrix_complete_good = confusion_matrix(datos_good$V11, corte_good2)

evaluation(matrix_complete_good, nrow(datos_good))

## [1] "Tasa de aciertos:  54.5"
## [1] "Tasa de fallos:   45.5"
```

Observaciones:

H clust complete muestra un desempeño ligeramente superior a los dos modelos anteriores

```
modelo_average_good = hclust(distance_good, method="average")
corte_good3= cutree(modelo_average_good, k = 2)
good= as.data.frame(good)

plot(good[,1:10],
```

```
    main = paste(c("Average", "goodluck.csv"), collapse = " "),
    col = corte_good3)
```




```
#matriz de confusion para average y evaluacion

matrix_average_good = confusion_matrix(datos_good$V11, corte_good3)

evaluation(matrix_average_good, nrow(datos_good))

## [1] "Tasa de aciertos:  51.5"
## [1] "Tasa de fallos:   48.5"
```

Observaciones:

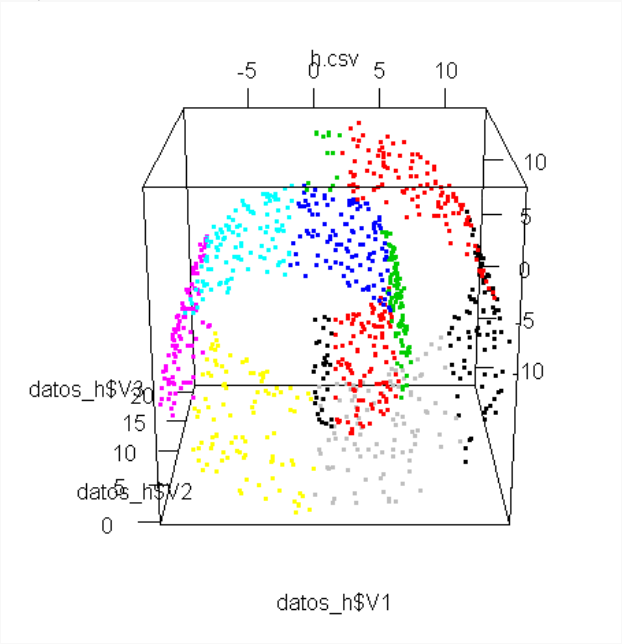
H clust average muestra una tasa de aciertos cercana al 50% , igual que los modelos anteriores. Al parecer la complejidad del data set dificulta que alguno de los métodos seleccionados tenga un desempeño óptimo, sin embargo todas tienen un desempeño ligeramente por encima de un 50% , por lo que no se puede considerar malo del todo

H.csv

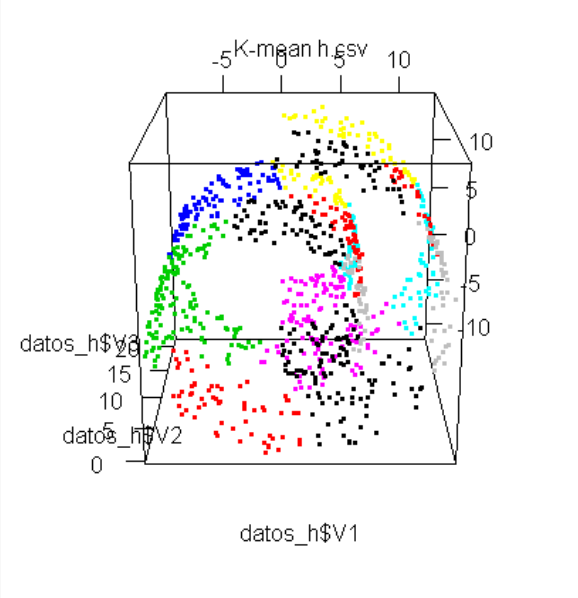
```
datos_h = read.csv("h.csv",header = F)

#analisis exploratorio

summary (datos_h)
```

```
##        V1                V2                 V3                 V4        
##  Min.   :-9.608   Min.   :-0.02557   Min.   :-11.3510   Min.   : 4.718  
##  1st Qu.:-3.311   1st Qu.: 5.37268   1st Qu.: -4.7355   1st Qu.: 7.046  
##  Median : 3.112   Median :10.84431   Median :  0.7600   Median : 9.249  
##  Mean   : 1.957   Mean   :10.73390   Mean   :  0.6411   Mean   : 9.386  
##  3rd Qu.: 6.199   3rd Qu.:16.33213   3rd Qu.:  6.6741   3rd Qu.:11.661  
##  Max.   :12.777   Max.   :21.16627   Max.   : 14.2422   Max.   :14.135
```

```
#llevar el rango a [1,11]

datos_h$V4 = floor(datos_h$V4) - 3

plot3d(datos_h$V1, datos_h$V2, datos_h$V3, col = datos_h$V4, main = "h.cs
v")
```



```
h_kmean= kmeans(x = datos_h[, 1:3], centers = 11)
plot3d(datos_h$V1, datos_h$V2, datos_h$V3, col = h_kmean$cluster, main =
"h.csv")
```



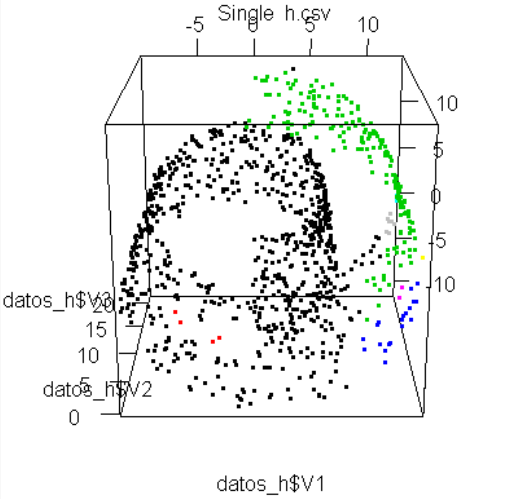

```
# matriz confusion para kmeans

matrix_kmeans_h = confusion_matrix(datos_h$V4, h_kmean$cluster)

evaluation(matrix_kmeans_h, nrow(datos_h))

## [1] "Tasa de aciertos:  14"
## [1] "Tasa de fallos:   86"
```

Observaciones:

Podemos apreciar que k medias tiene una tasa de aciertos bastante baja

```
###############  HCLUST  ########################

#matriz de distancias

h = datos_h
h$V4 = NULL
h = as.matrix(h)
distance_h = dist(h)
```

```
#single h

modelo_single_h = hclust(distance_h, method="single")
corte_h1= cutree(modelo_single_h, k = 11)

plot3d(datos_h$V1, datos_h$V2, datos_h$V3, col = corte_h1, main = " Singl
e  h.csv")
```

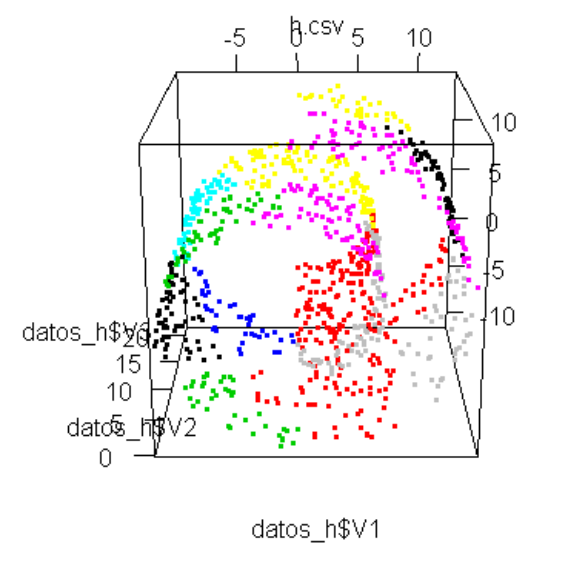



```
#matriz confusion single h

table_single_h = confusion_matrix(datos_h$V4, corte_h1)

evaluation(table_single_h, nrow(datos_h))

## [1] "Tasa de aciertos:  3"
## [1] "Tasa de fallos:   97"
```

Observaciones:

H clust simple presenta una tasa de aciertos muy baja, además no asigna correctamente a los elementos

```
#complete h


modelo_h2 = hclust(distance_h, method="complete")
corte_h2= cutree(modelo_h2, k = 11)

plot3d(datos_h$V1, datos_h$V2, datos_h$V3, col = corte_h2, main = "Comple
te h.csv")
```



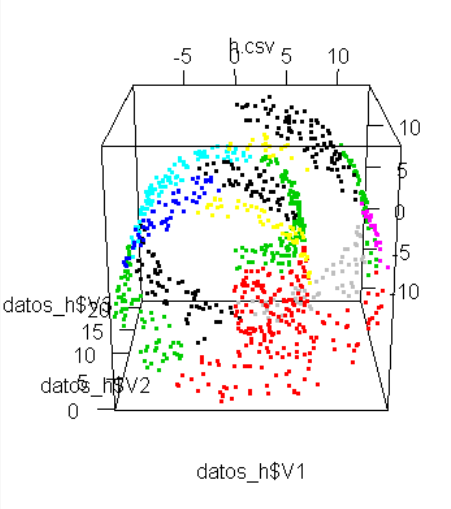

```
#matriz confusion complete h

matrix_complete_h = confusion_matrix(datos_h$V4, corte_h2)


evaluation(matrix_complete_h, nrow(datos_h))
```

```
## [1] "Tasa de aciertos:  10.4"
## [1] "Tasa de fallos:   89.6"
```

Observaciones:

H clust complete muestra un desempeño ligeramente superior a los dos modelos anteriores, y realiza una mejor asignación de los elementos a los clusters

```r
#average h


modelo_h3 = hclust(distance_h, method="average")
corte_h3= cutree(modelo_h3, k = 11)

plot3d(datos_h$V1, datos_h$V2, datos_h$V3, col = corte_h3, main = "Averag
e h.csv")
```




```r
#matriz confusion average h

matrix_average_h = confusion_matrix(datos_h$V4, corte_h3)

evaluation(matrix_average_h, nrow(datos_h))
```

```
## [1] "Tasa de aciertos:  9.3"
## [1] "Tasa de fallos:   90.7"
```

Observaciones:

H clust average muestra una tasa de acierto similar a los modelos anteriores para este data set, bastante baja, tampoco asigna bien a los elementos en los clusters. Por lo que el número de cluster elegidos para evaluar este modelo ocasiona un mal rendimiento

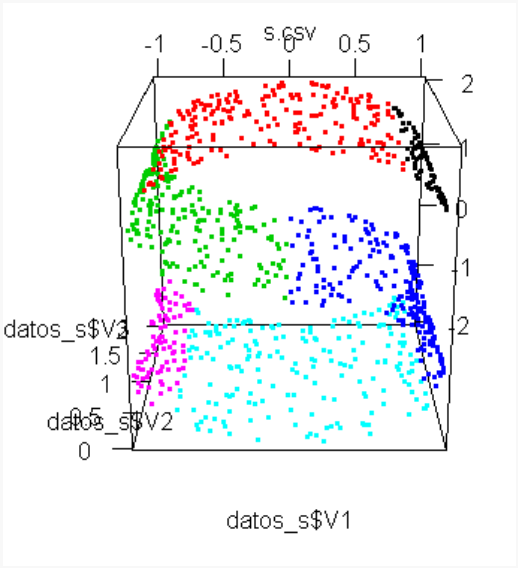
S.cvs

```
datos_s = read.csv("s.csv",header = F)

#analisis exploratorio
summary(datos_s$V4)
```

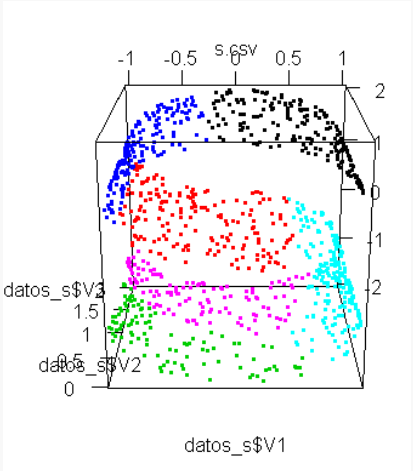
```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -4.70700 -2.37900 -0.17600 -0.03844  2.23600  4.71100
```

```
# fijamos las clases que estan en un rango de [-5,5]  a [1,6]
for(i in 1:length(datos_s$V4)){

  if (datos_s$V4[i]< -4.0)
    datos_s$V4[i]=as.numeric(1)

  else if(datos_s$V4[i]< -2.0)
    datos_s$V4[i]=as.numeric(2)

  else if(datos_s$V4[i]< -0.0)
    datos_s$V4[i]=as.numeric(3)

  else if(datos_s$V4[i]< 2.0)
    datos_s$V4[i]=as.numeric(4)

  else if(datos_s$V4[i]< 4.0)
    datos_s$V4[i]=as.numeric(5)

  else
    datos_s$V4[i]=as.numeric(6)


}
```

```r
#visualizar datos

plot3d(datos_s$V1, datos_s$V2, datos_s$V3, col = datos_s$V4, main = "s.cs
v")
```




```r
#aplicando k medias

s_kmean= kmeans(x = datos_s[, 1:3], centers = 6)

plot3d(datos_s$V1, datos_s$V2, datos_s$V3, col = s_kmean$cluster, main =
"K mean s.csv")
```



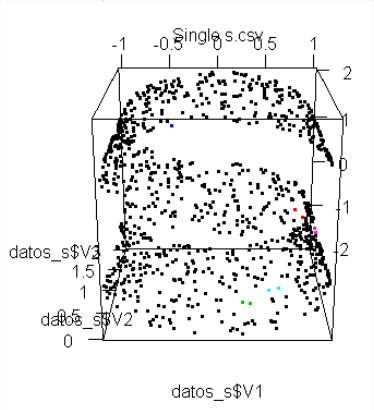
```r
# matriz confusion para kmeans y evaluacion


matrix_kmeans_s = confusion_matrix(datos_s$V4, s_kmean$cluster)

evaluation(matrix_kmeans_s, nrow(datos_s))
```

```
## [1] "Tasa de aciertos:  11.8"
## [1] "Tasa de fallos:   88.2"
```

Observaciones:

Podemos apreciar que k medias tiene una tasa de aciertos cercana al 50%, y hace una correcta asignación de los elementos de las clases

```r
###############  HCLUST  ########################

#matriz de distancias


s = datos_s
s$V4 = NULL
s = as.matrix(s)
distance_s = dist(s)


#single s

modelo_single_s = hclust(distance_s, method="single")
corte_s1= cutree(modelo_single_s, k = 6)

plot3d(datos_s$V1, datos_s$V2, datos_s$V3, col = corte_s1, main = "Single
s.csv")
```



```
#matriz confusion single s y evaluacion

matrix_single_s = confusion_matrix(datos_s$V4, corte_s1)

evaluation(matrix_single_s, nrow(datos_s))
```

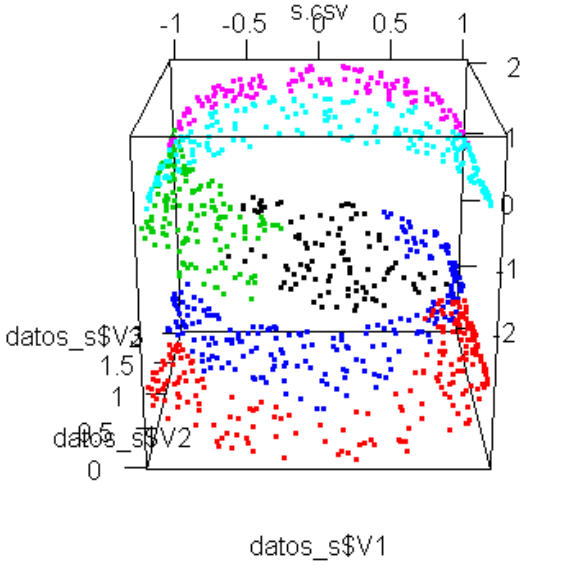
```
## [1] "Tasa de aciertos:  7.7"
## [1] "Tasa de fallos:   92.3"
```

Observaciones:

H clust simple presenta un desempeño deficiente, con una tasa de aciertos muy baja, además no asigna bien los elementos en los clusters

#complete

```
modelo_complete_s = hclust(distance_s, method="complete")
corte_s2= cutree(modelo_complete_s, k = 6)

plot3d(datos_s$V1, datos_s$V2, datos_s$V3, col = corte_s2, main = "s.csv"
)
```



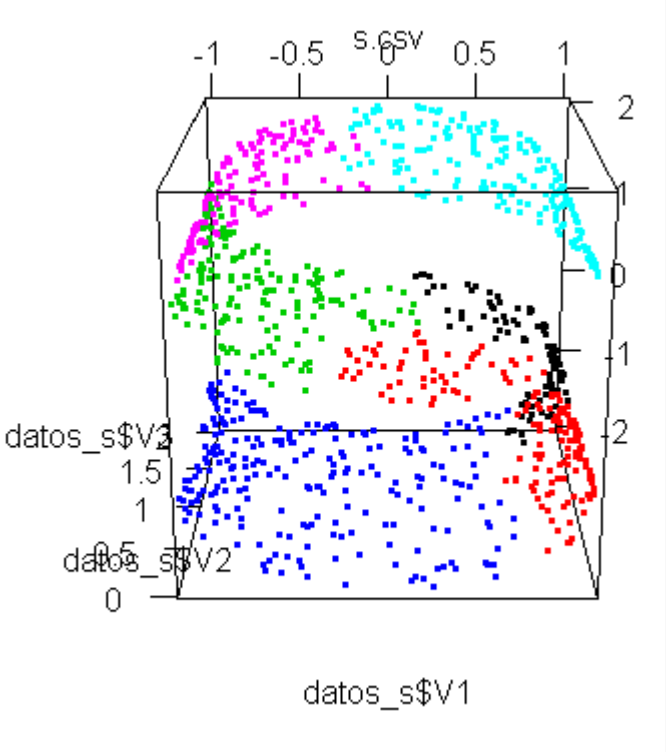
```
#matriz confusion complete s y evaluacion

matrix_complete_s = confusion_matrix(datos_s$V4, corte_s2)

evaluation(matrix_complete_s, nrow(datos_s))
```

```
## [1] "Tasa de aciertos:  21.9"
## [1] "Tasa de fallos:   78.1"
```

Observaciones:

H clust complete muestra un desempeño superior a hclust simple en cuanto a la asignación de elementos en los cluster, sin embargo tiene una tasa de aciertos baja en su matriz de confusión

```
#average s


modelo_average_s = hclust(distance_s, method="average")
corte_s3= cutree(modelo_average_s, k = 6)

plot3d(datos_s$V1, datos_s$V2, datos_s$V3, col = corte_s3, main = "s.csv"
)
```



```r
#matriz confusion average s  y evaluacion

matrix_average_s = confusion_matrix(datos_s$V4, corte_s3)

evaluation(matrix_average_s, nrow(datos_s))
```

```
## [1] "Tasa de aciertos:  16.8"
## [1] "Tasa de fallos:   83.2"
```

Observaciones:

H clust average muestra una tasa de algo baja , al igual que los anteriores métodos de clustering, asigna relativamente bien a los elementos en los clusters,. Sin embargo al parecer ningún método de h clust funciona mejor que k medias para este data set

HELP.

csv

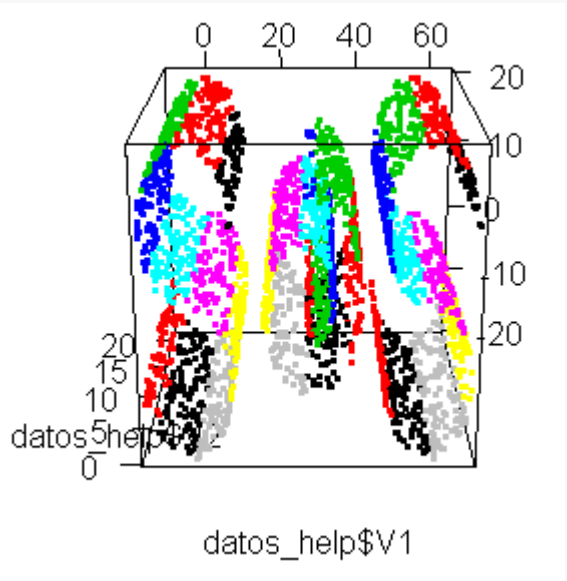
```r
datos_help = read.csv("help.csv",header = F)

#analisis exploratorio

summary(datos_help)
```

```
##        V1                V2                  V3                   V4
##  Min.   :-10.000   Min.   :-0.02557   Min.   :-19.99995   Min.   :-4.7
12
##  1st Qu.:  6.864   1st Qu.: 4.93492   1st Qu.:-10.21263   1st Qu.:-1.1
80
##  Median : 30.112   Median : 9.98509   Median :  0.09956   Median : 2.4
26
##  Mean   : 27.848   Mean   :10.07264   Mean   :  0.10830   Mean   : 3.1
41
##  3rd Qu.: 47.801   3rd Qu.:15.20300   3rd Qu.:  9.93020   3rd Qu.: 7.0
45
##  Max.   : 65.000   Max.   :21.16627   Max.   : 19.99975   Max.   :14.1
35
```

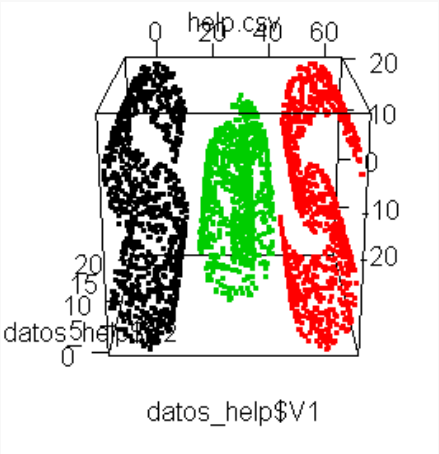
```r
#establecer el rango a partir de 1

#rango de [1,20]

datos_help$V4 = floor(datos_help$V4) + 6

summary(datos_help$V4)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   4.000   8.000   8.637  13.000  20.000
```

```r
plot3d(datos_help$V1, datos_help$V2, datos_help$V3, col = datos_help$V4)
#se observan 3 cluster

#al asignar las clases se observa una distribucion, que  forma algo parec
ido a las letras 'S' y 'O'.
#Esta distribución es parecida a los datasets h.csv y s.csv. Donde los mé
todos no tuvieron un rendimiento óptimo
```



```r
#Podemos definir 3 cluster, basandonos en los valores del eje X, que deli
mitan cada letra según el gráfico observado

for(i in 1:length(datos_help$V4)){

  if (datos_help$V1[i]< 10.0)
    datos_help$V4[i]=as.numeric(1)

  else if(datos_help$V1[i]< 45.0)
    datos_help$V4[i]=as.numeric(2)

  else
    datos_help$V4[i]=as.numeric(3)


}



#aplicando k medias para help

help_kmean= kmeans(x = datos_help[, 1:3], centers = 3)

plot3d(datos_help$V1, datos_help$V2, datos_help$V3, col = help_kmean$clus
ter, main="help.csv")
```



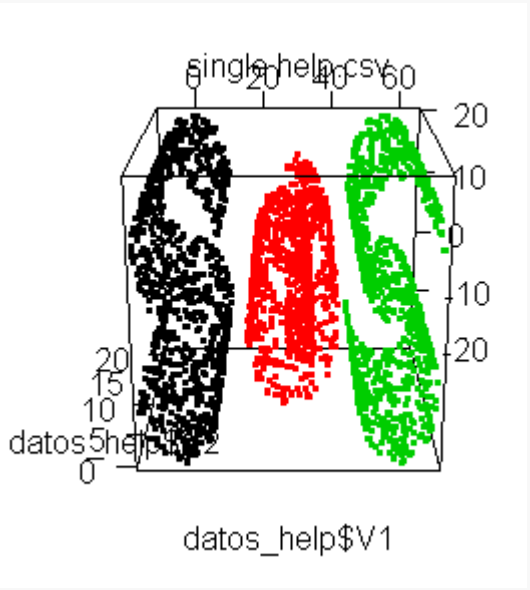

```r
# matriz confusion help


matrix_kmeans_help = confusion_matrix(datos_help$V4, help_kmean$cluster)

evaluation(matrix_kmeans_help, nrow(datos_help))
```

```
## [1] "Tasa de aciertos:  99.9666666666667

## [1] "Tasa de fallos:   0.0333333333333314
```

Observaciones:

Podemos apreciar para k-medias un buen desempeño , ya que se clasifican los puntos correctamente. Con una tasa de aciertos de la matriz de confusión de casi 100%

```
help = datos_help
help$V4 = NULL
help = as.matrix(help)
distance_help = dist(help)


#single help


modelo_single_help = hclust(distance_help, method="single")
corte_help1= cutree(modelo_single_help, k = 3)

plot3d(datos_help$V1, datos_help$V2, datos_help$V3, col = corte_help1, ma
in = "single help.csv")
```



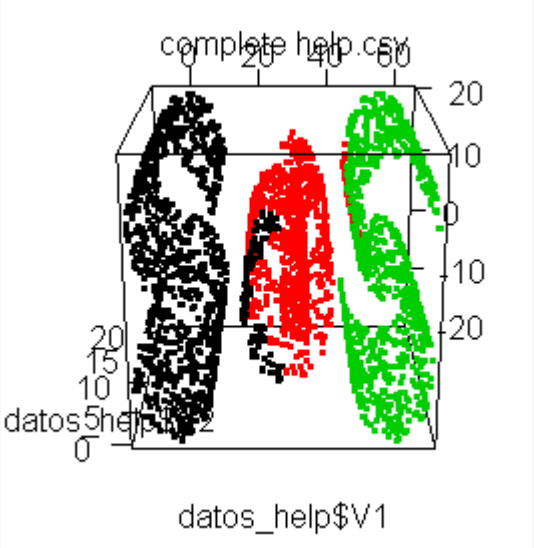
```r
#matriz confusion single help  y evaluacion

matrix_single_help = confusion_matrix(datos_help$V4, corte_help1)

evaluation(matrix_single_help, nrow(datos_help))
```

```
## [1] "Tasa de aciertos:  99.9666666666667"
## [1] "Tasa de fallos:   0.0333333333333314"
```

Observaciones:

H clust simple, al igual que k medias presenta un rendimiento óptimo.

```r
#Modelo complete

modelo_complete_help = hclust(distance_help, method="complete")
corte_help2= cutree(modelo_complete_help, k = 3)

plot3d(datos_help$V1, datos_help$V2, datos_help$V3, col = corte_help2, ma
in = "complete help.csv")
```



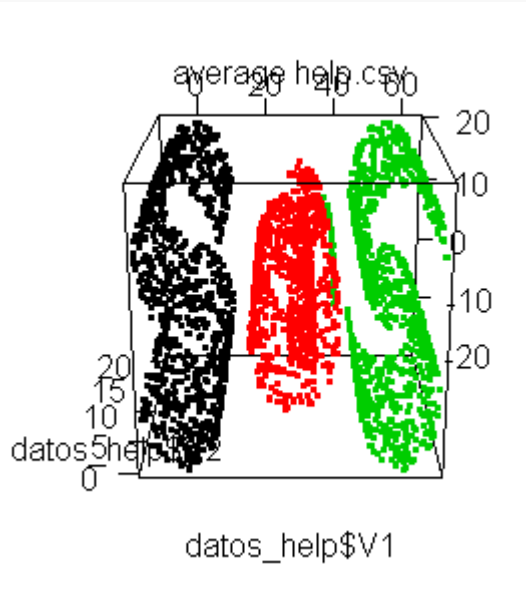

```r
#matriz confusion complete help y evaluacion

matrix_complete_help = confusion_matrix(datos_help$V4, corte_help2)

evaluation(matrix_complete_help, nrow(datos_help))
```

```
## [1] "Tasa de aciertos:  94.1666666666667"
## [1] "Tasa de fallos:   5.83333333333333"
```

Observaciones:

H clust complete muestra una alta tasa de aciertos en su matriz de confusion, sin embargo algunos elementos del data set no se estan clasificando correctamente

```
#modelo average


modelo_average_help = hclust(distance_help, method="average")
corte_help3= cutree(modelo_average_help, k = 3)

plot3d(datos_help$V1, datos_help$V2, datos_help$V3, col = corte_help3, ma
in = "average help.csv")
```




```
#matriz confusion average h

matrix_average_help = confusion_matrix(datos_help$V4, corte_help3)

evaluation(matrix_average_help, nrow(datos_help))

## [1] "Tasa de aciertos:  98.1"
## [1] "Tasa de fallos:   1.90000000000001"
```

Observaciones:

H clust average muestra una tasa de acierto similar a los modelos anteriores, es decir bastante alta, y hace una clasificación casi óptima de sus elementos.

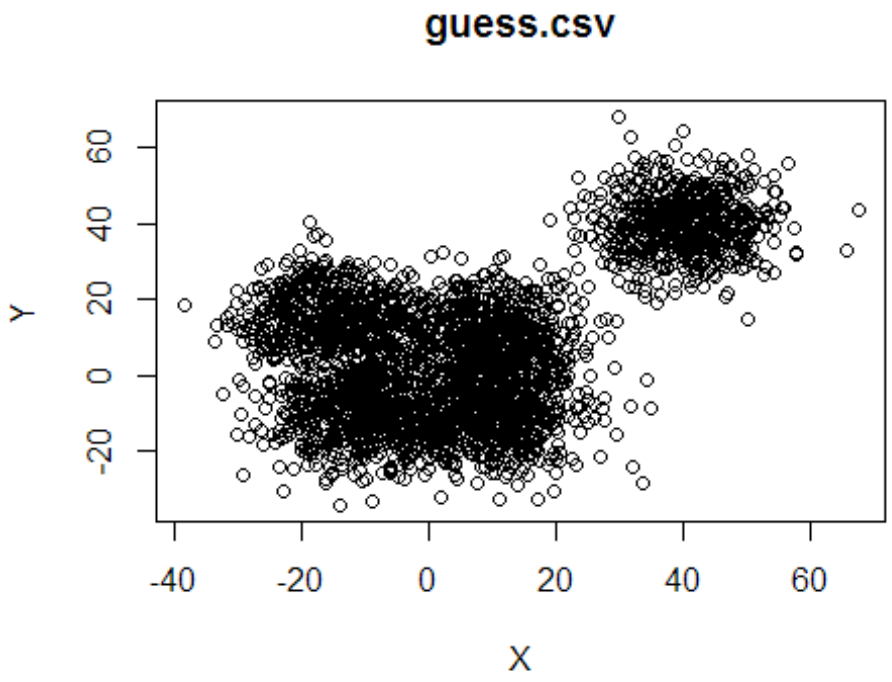
Al parecer k medias, y h cluster simple son los que tienen un desempeño óptimo para este data set, sin embargo h clust complete y average también muestran un buen desempeño

Guess.csv

```
datos_guess = read.csv("guess.csv",header = F)


#analisis exploratorio

summary(datos_guess)
```

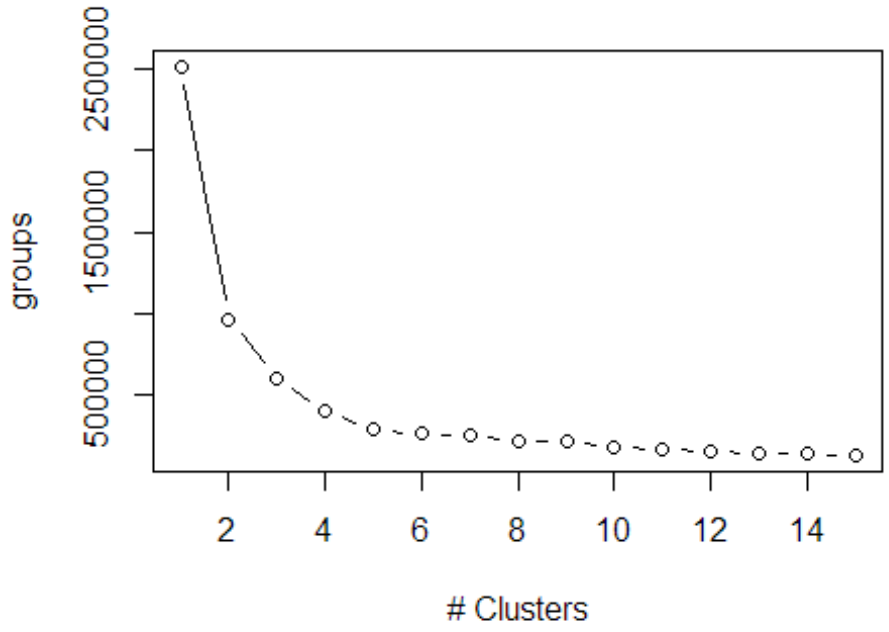
```
##        V1                 V2          
##  Min.   :-38.575   Min.   :-34.207  
##  1st Qu.:-10.144   1st Qu.: -8.089  
##  Median :  5.529   Median :  7.426  
##  Mean   :  6.903   Mean   :  8.853  
##  3rd Qu.: 17.894   3rd Qu.: 21.857  
##  Max.   : 67.537   Max.   : 68.029
```

```
plot(datos_guess, xlab = "X", ylab = "Y",
     main = "guess.csv")
```

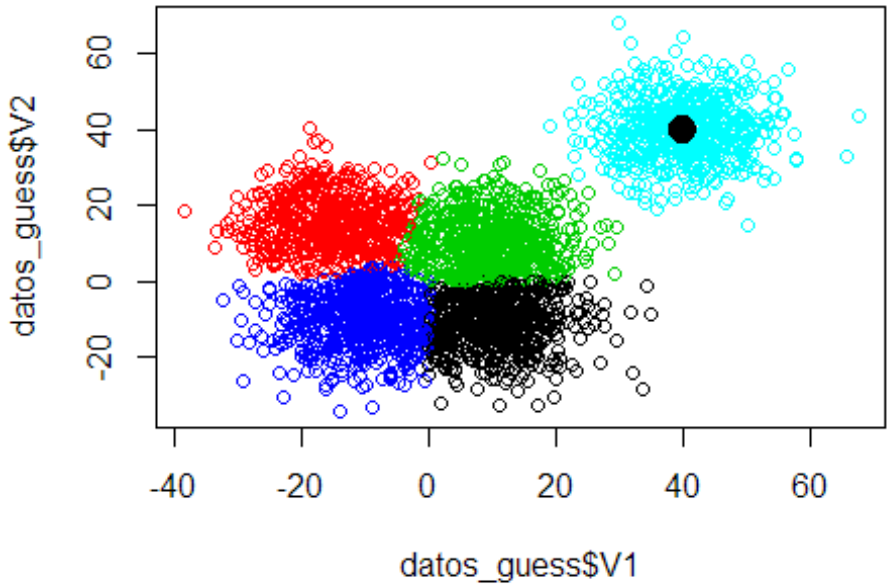



```
#En principio se observan dos cluster de forma esférica y  tamaños distin
tos
```

```
codo_jambu(datos_guess)
```



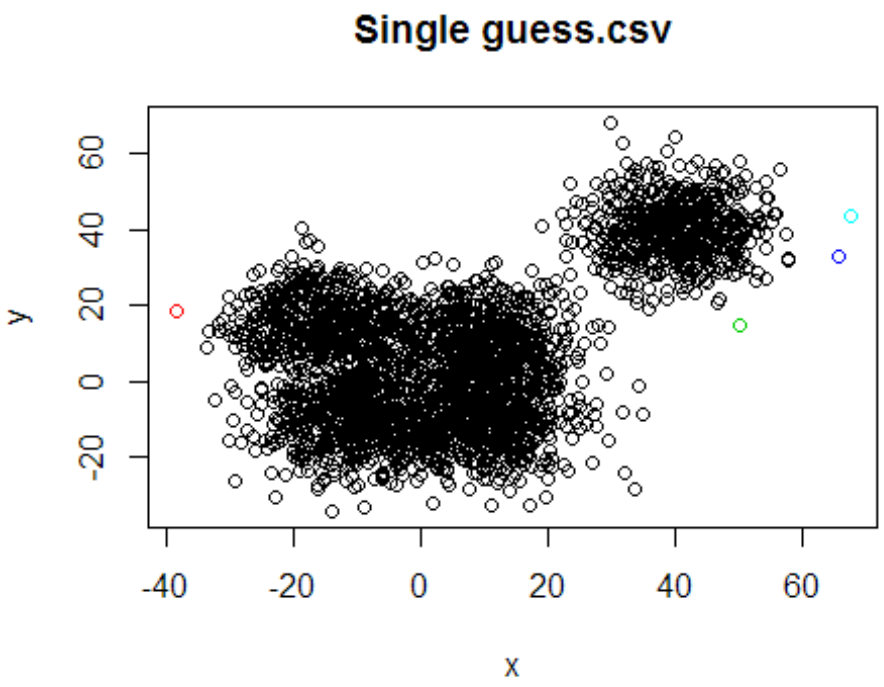

```
# la grafica de codo jambu, nos hace elegir un k=5 , ya que en este punto
empieza a variar la curva

#k means para guest.csv

guess_kmean = kmeans(x = datos_guess[, c("V1", "V2")],
                     centers = 5)


plot(x = datos_guess$V1,
     y = datos_guess$V2,
     col = guess_kmean$cluster)



points(x = guess_kmean$centers[, c("V1", "V2")],
       col = 1:4, pch = 19, cex = 2)
```



Observaciones:

Al aplicar k medias , se pueden ver las 5 clases bien definidas, lo que indica que se esta haciendo una buena clasificación

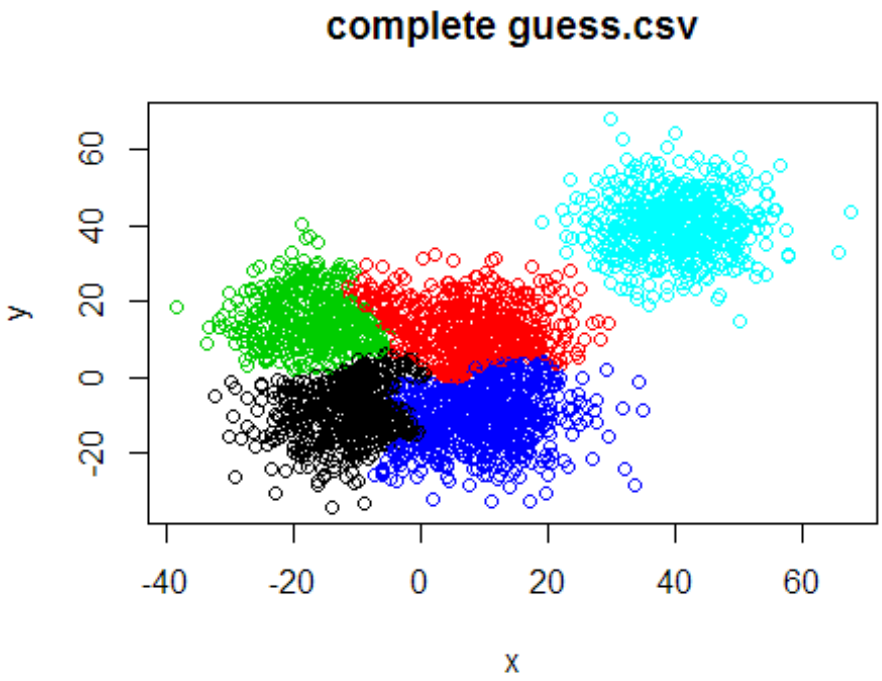
```
##############  HCLUST  ########################

#matriz de distancias

guess = datos_guess
guess = as.matrix(guess)
distance_guess = dist(guess)



#modelo single para guest.csv

modelo_single_guess = hclust(distance_guess, method="single")
corte_guess1= cutree(modelo_single_guess, k = 5)


plot(x = guess[,1], y = guess[,2], main = paste(c("Single", "guess.csv")
, collapse = " "),
     xlab = "x", ylab = "y",
     col = corte_guess1)
```



Observaciones:

H clust simple, no funciona para este data set, ya que no clasifica bien a los elementos

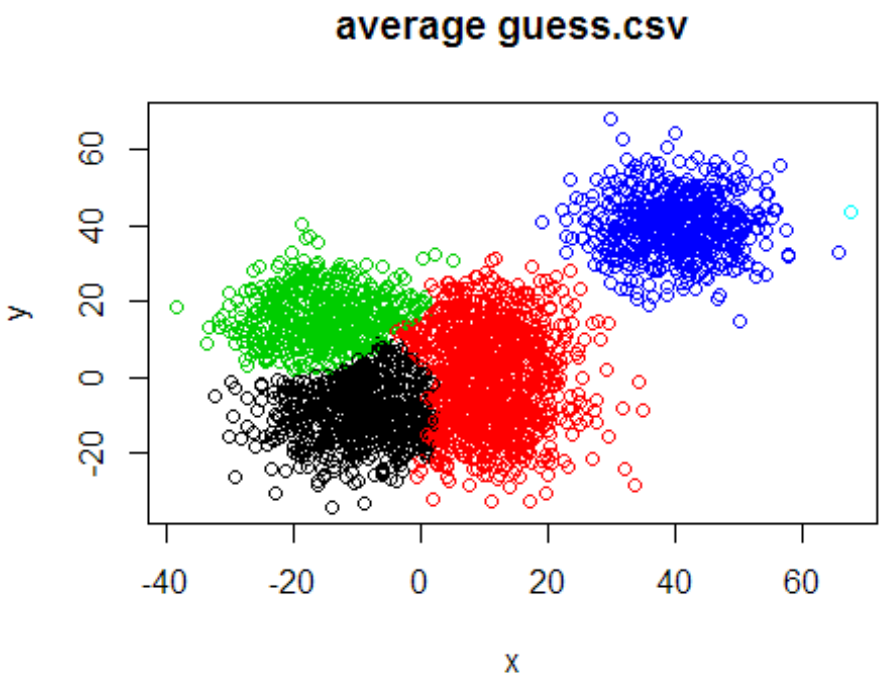
```
#modelo Complete

modelo_complete_guess = hclust(distance_guess, method="complete")
corte_guess2= cutree(modelo_complete_guess, k = 5)

plot(x = guess[,1], y = guess[,2], main = paste(c("complete", "guess.csv"
) , collapse = " "),
     xlab = "x", ylab = "y",
     col = corte_guess2)
```



Observaciones:

H clust complete si funciona correctamente para este set de datos, ya que hace una buena clasificación de sus elementos

```
#modelo average

modelo_average_guess = hclust(distance_guess, method="average")
corte_guess3= cutree(modelo_average_guess, k = 5)

plot(x = guess[,1], y = guess[,2], main = paste(c("average", "guess.csv")
, collapse = " "),
     xlab = "x", ylab = "y",
     col = corte_guess3)
```



Observaciones:

H clust average no funciona bien del todo ya que no clasifica correctamente en los 5 cluster, al parecer asigna en un mismo cluster un tipo de elementos, lo que hace que visualmente se observen 4 clusters definidos

Podemos concluir que k medias y h clust complete son los modelos que mejor desempeño tienen para este data set