

Tarea 2

MÓDULO VI. APRENDIZAJE NO SUPERVISADO

Integrantes:

Luis Rodrigo Santamaría Rodríguez
Jonathan Vargas González
Marisol Estrada Téllez

Equipo 8

Integrantes:

- Luis Rodrigo Santamaría Rodríguez
- Jonathan Vargas González
- Marisol Estrada Téllez

En esta tarea se abordará el análisis de una base de datos relacionada con la calidad del agua, la cual contiene diversas variables físico-químicas que permiten evaluar su potabilidad. El objetivo principal es aplicar técnicas de agrupamiento (clustering) para identificar patrones ocultos y posibles grupos naturales dentro de los datos, sin necesidad de una variable de salida en este caso nuestra variable llamada “Potability”.

La base de datos utilizada incluye atributos como pH, dureza, sólidos disueltos, cloraminas, sulfatos, conductividad eléctrica, carbono orgánico, trihalometanos y turbidez. Cada registro representa una muestra de agua, y aunque se cuenta con la variable Potability como referencia, el enfoque será completamente no supervisado.

Para lograr una segmentación significativa, se emplearán métodos de reducción de dimensionalidad como PCA, t-SNE y UMAP (los cuáles fueron analizados en la tarea 1), que facilitarán la visualización y comprensión de la estructura interna de los datos. Posteriormente, se aplicarán algoritmos de clustering como clustering jerárquico, AGNES, DIANA, K-means, DBSCAN y Model-Based Clustering (EM), con el fin de descubrir agrupamientos que puedan aportar valor en la clasificación de la calidad del agua.

Cargamos nuestras bases de datos.

```
df<-read.csv("./water_potability.csv")
head(df)
```

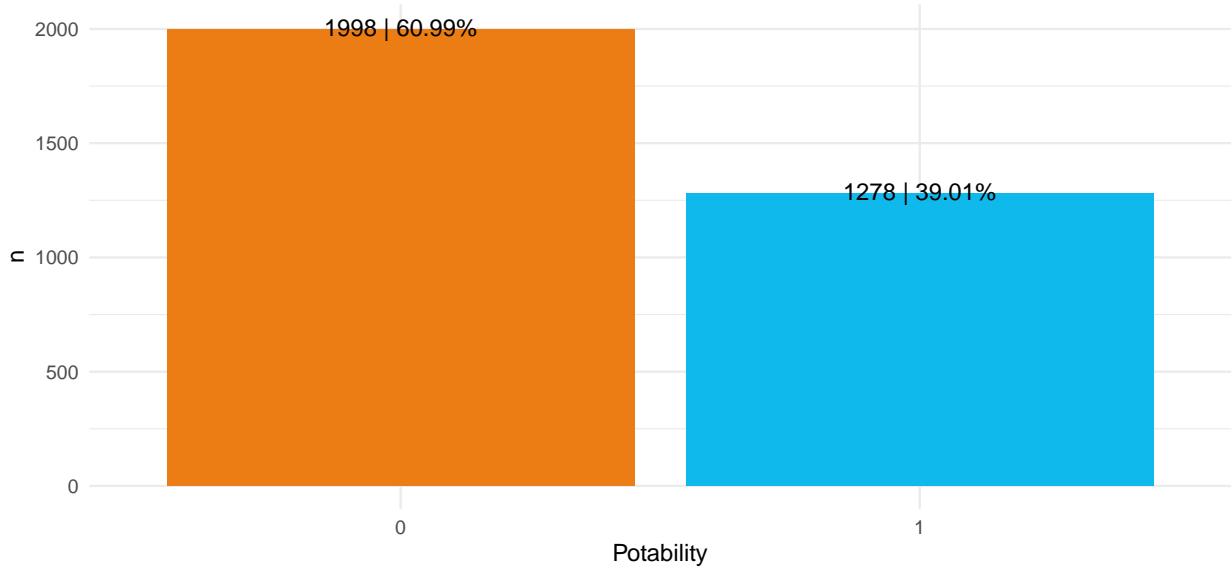
```
##          ph Hardness   Solids Chloramines   Sulfate Conductivity Organic_carbon
## 1        NA 204.8905 20791.32     7.300212 368.5164      564.3087      10.379783
## 2 3.716080 129.4229 18630.06     6.635246      NA      592.8854      15.180013
## 3 8.099124 224.2363 19909.54     9.275884      NA      418.6062      16.868637
## 4 8.316766 214.3734 22018.42     8.059332 356.8861      363.2665      18.436525
## 5 9.092223 181.1015 17978.99     6.546600 310.1357      398.4108      11.558279
## 6 5.584087 188.3133 28748.69     7.544869 326.6784      280.4679      8.399735
##   Trihalomethanes Turbidity Potability
## 1        86.99097  2.963135       0
## 2        56.32908  4.500656       0
## 3        66.42009  3.055934       0
## 4       100.34167  4.628771       0
## 5       31.99799  4.075075       0
## 6       54.91786  2.559708       0
```

Imputamos datos ya que, tenemos datos faltantes:

```
imputed_data <- mice(df, m = 5, method = 'pmm', maxit = 5, seed = 123)
```

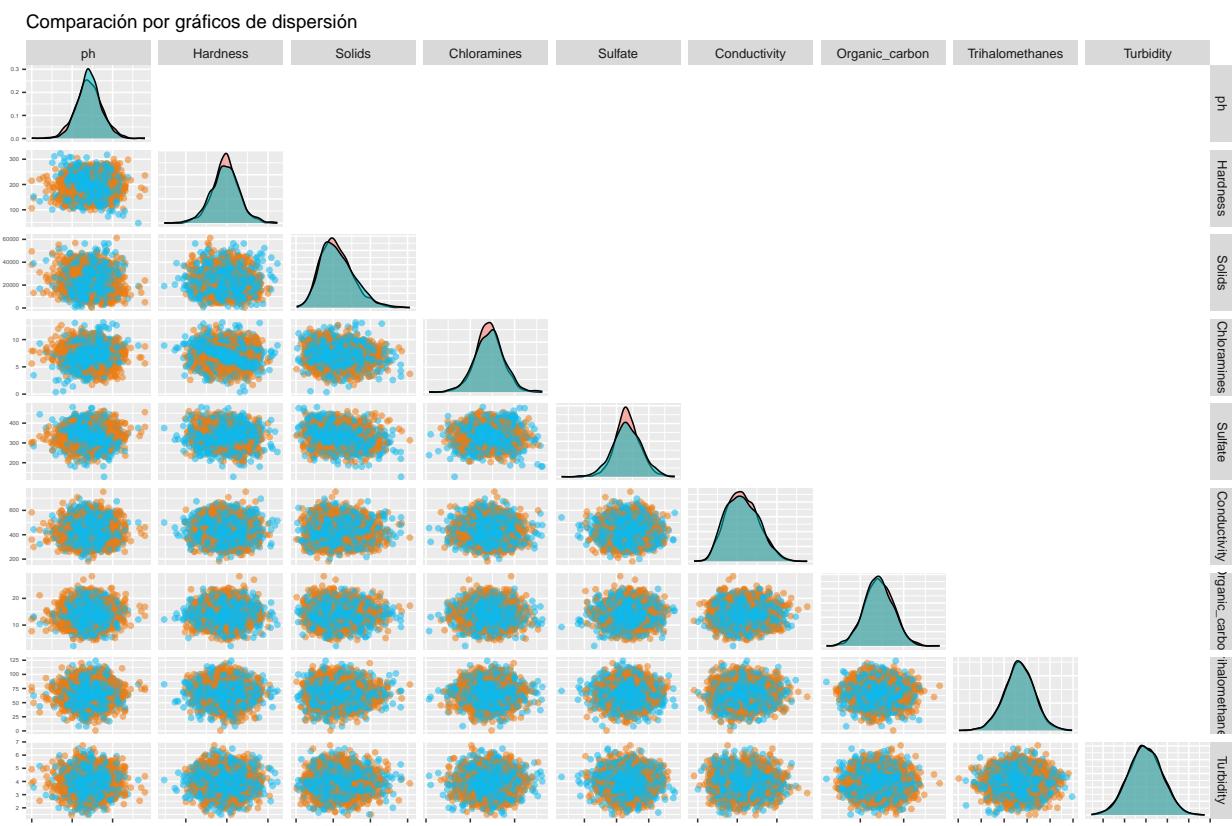
Hacemos un gráfico de nuestra variable “Potability”.

Porcentajes de agua potable y no potable

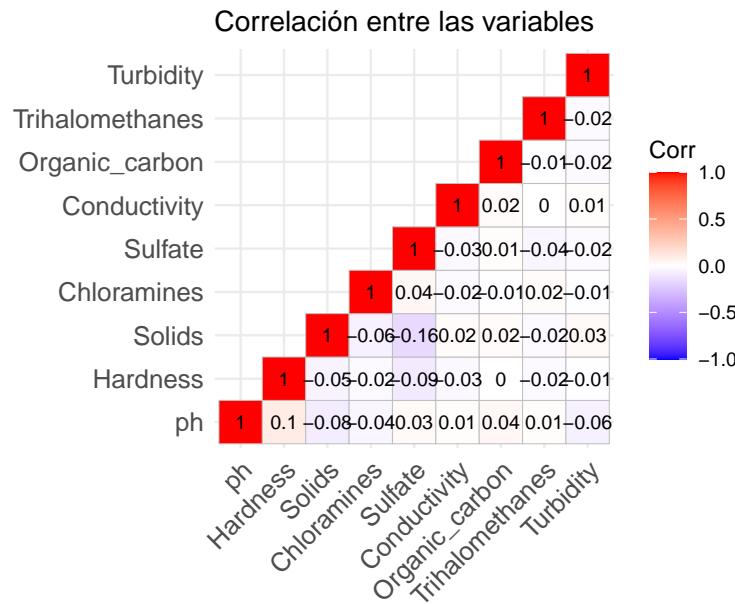


Podemos observar que hay 1998 registros que no son aptos para el consumo humano y 1278 registros sí son aptos para el consumo humano, es decir, el **60.99% de registros no son aptos para el consumo humano y un 39.01% sí lo son.**

Gráficos de dispersión y densidad:



Veamos la estructura de correlación de Spearman que tienen nuestras variables.



Hay que destacar que, como se observa en los gráficos de dispersión y densidad, en cada par de variable parece no haber diferencia entre los grupos de potabilidad pues las densidades son muy similares y los puntos en los gráficos de dispersión están muy mezclados.

Marginalmente no se puede observar grupos con claridad pero tal vez tomando todas las variables sí se puedan encontrar con los métodos presentados más adelante.

Métodos Jerárquicos

Primero comenzamos con los **Métodos Jerárquicos**. Así se ven las variables estandarizadas que usaremos para formar clusters:

```
head(df1)
```

```
##          ph   Hardness      Solids Chloramines     Sulfate Conductivity
## 1  0.01145878  0.2591551 -0.1394495831  0.1123977  0.8540734  1.70869338
## 2 -2.08685969 -2.0361028 -0.3859277336 -0.3076467 -0.7685191  2.06226017
## 3  0.63518203  0.8475354 -0.2400106970  1.3603862 -0.5567942 -0.09401776
## 4  0.77034603  0.5475678  0.0004932291  0.5919175  0.5695506 -0.77871108
## 5  1.25193549 -0.4643582 -0.4601783193 -0.3636424 -0.5741473 -0.34388641
## 6 -0.92675460 -0.2450192  0.7680379558  0.2669421 -0.1694499 -1.80314114
##          Organic_carbon Trihalomethanes Turbidity
## 1       -1.1804704    1.274612313 -1.2861012
## 2        0.2705559   -0.617366189  0.6841135
## 3        0.7809976    0.005295466 -1.1671873
## 4        1.2549429    2.098411543  0.8482820
## 5       -0.8242313   -2.118704845  0.1387643
## 6       -1.7790047   -0.704444546 -1.8030621
```

Vamos a calcular la distancia. Como las variables son continuas utilizaremos la distancia **Euclíadiana**. Además, la distancia euclíadiana funciona mejor para datos normales como los nuestros (ver gráficas de densidad anteriores).

```
dist_df <- dist(df1, method = "euclidean")
```

Calculamos las distancias entre clusters con las ligas promedio, simple, completa, Ward, Ward cuadrada y centroide:

```
hc_ave <- hclust(dist_df, method = "average")
hc_single<-hclust(dist_df, method = "single")
hc_comp<-hclust(dist_df, method = "complete")
hc_ward<-hclust(dist_df, method = "ward.D")
hc_ward2<-hclust(dist_df, method = "ward.D2")
hc_centroid <- hclust(dist_df, method = "centroid")
```

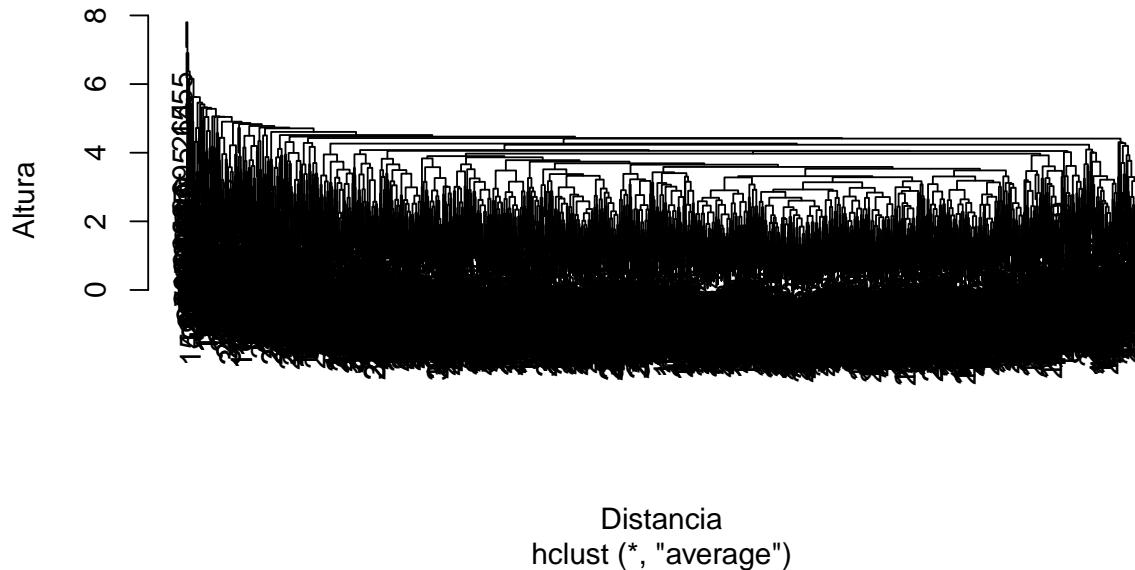
Calculamos la distancia cophenética y calculamos la correlación que tienen las distancias.

```
d_ave <- cophenetic(hc_ave)
d_single <- cophenetic(hc_single)
d_comp<- cophenetic(hc_comp)
d_ward<- cophenetic(hc_ward)
d_ward2<-cophenetic(hc_ward2)
d_centroid<-cophenetic(hc_centroid)
```

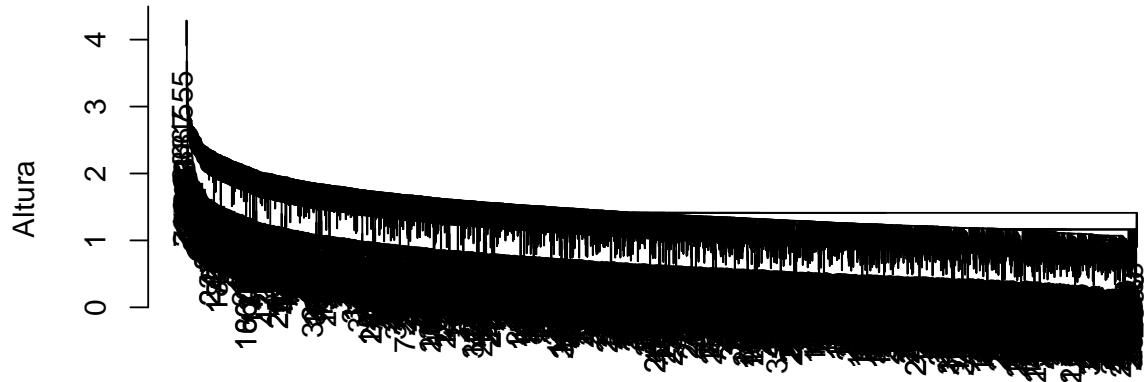
```
##      Promedio     Simple    Completa      Ward    Ward D2 Centroide
## [1,] 0.5519751 0.6507382 0.2713331 0.1489466 0.1768797 0.5648073
```

Podemos observar que las mejores ligas que mantienen la distancia son la promedio, la simple y la centroide.

Liga Promedio: Dendrograma

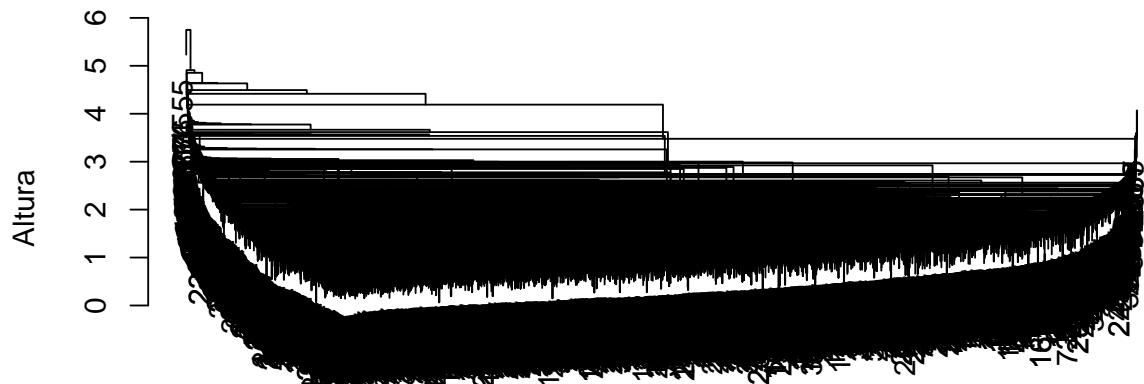


Liga Simple: Dendrograma



Distancia
hclust (*, "single")

Liga Centroide: Dendrograma

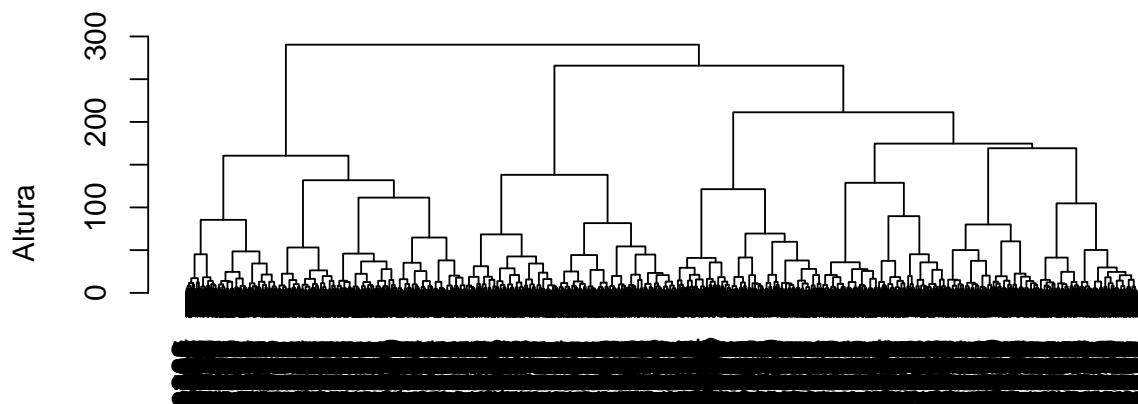


Distancia
hclust (*, "centroid")

Podemos observar que con los dendrogramas con las ligas simple, promedio y centroide no es claro cuántos grupos podemos tener de toda nuestra base de datos haciendo algún corte.

Las ligas que funcionan mejor son las de Ward.

Liga Ward: Dendrograma

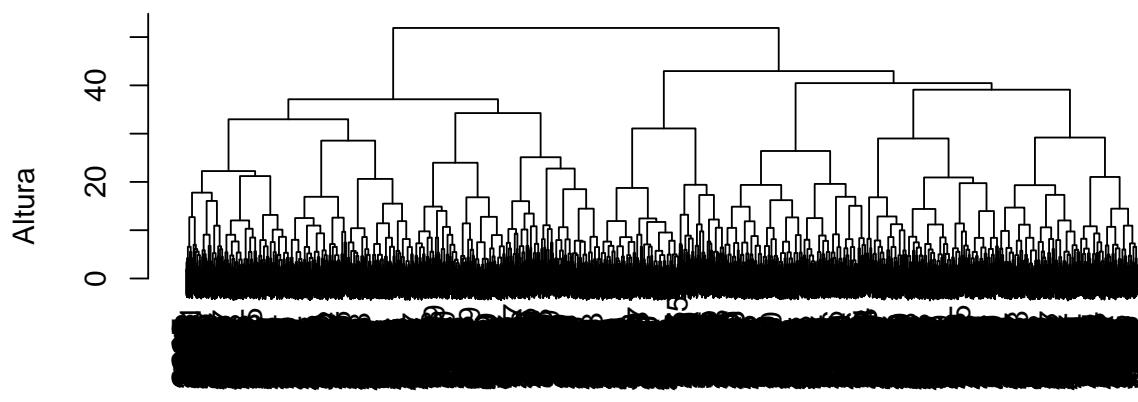


Distancia
hclust (*, "ward.D")

Podemos notar en la gráfica anterior que si hacemos un corte a la altura 215 aproximadamente tenemos un total de 4 clusters y si hacemos un corte a una altura de 250 tenemos 3 clusters.

A continuación veamos la grafica de Ward D2:

Liga Ward D2: Dendrograma

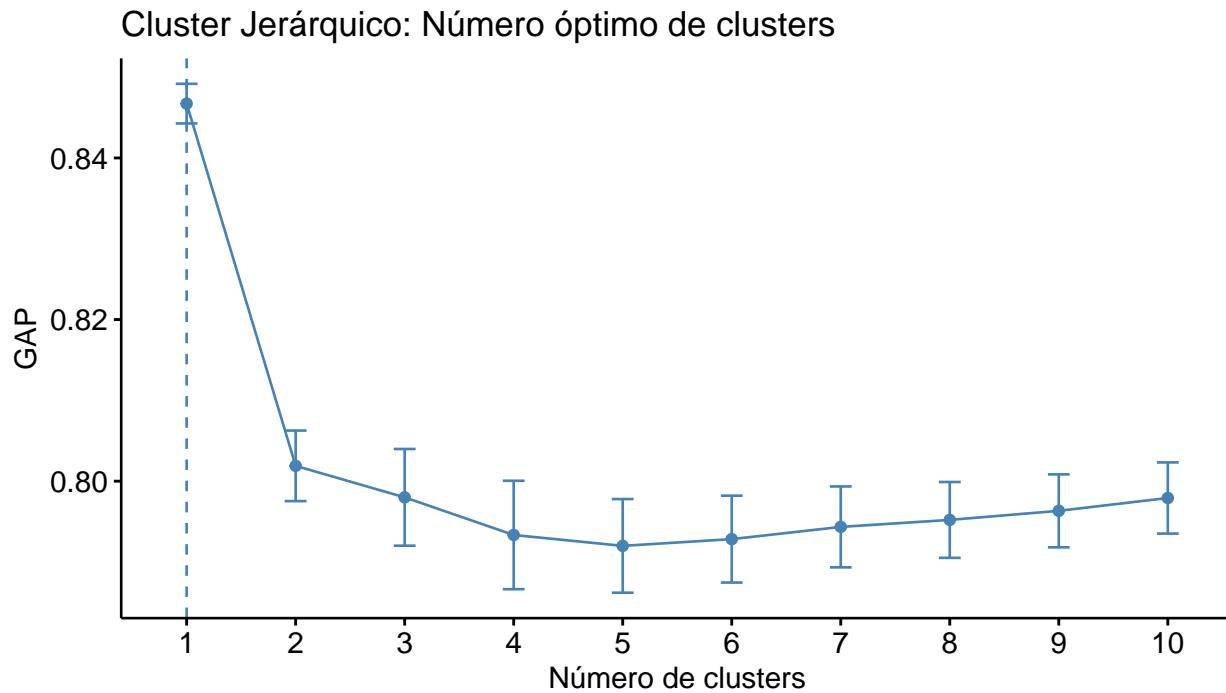


Distancia
hclust (*, "ward.D2")

Podemos observar del dendrograma anterior que si hacemos un corte con altura 45 tenemos 2 clusters, pero si hacemos un corte con una altura de 40 tenemos 4 clusters.

Con los dendrogramas anteriores no podemos hacer una afirmación de cuántos cluster podemos tener en nuestra base de datos.

Ahora revisaremos el estadístico **GAP** para tener otra forma de confirmar el número de clusters que podrían encontrarse en los datos.



Según el estadístico GAP, el número de cluster óptimo es 1, es decir, no encuentra más de un solo grupo en los datos. Este resultado está relacionado con lo mostrado en los siguientes métodos explorados.

AGNES

Trabajemos con otro método aglomerativo llamado AGNES.

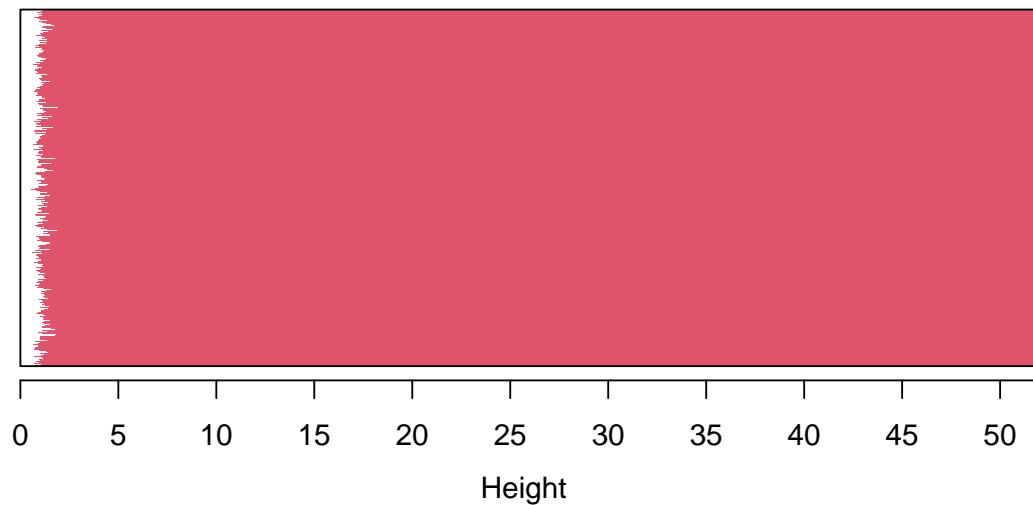
```
ac_metric <- list(  
  complete_ac = agnes(df1, metric = "euclidean", method = "complete")$ac,  
  average_ac = agnes(df1, metric = "euclidean", method = "average")$ac,  
  single_ac = agnes(df1, metric = "euclidean", method = "single")$ac,  
  ward_ac = agnes(df1, metric = "euclidean", method = "ward")$ac  
)  
  
ac_metric  
  
## $complete_ac  
## [1] 0.8748141  
##  
## $average_ac
```

```
## [1] 0.8049079
##
## $single_ac
## [1] 0.6765082
##
## $ward_ac
## [1] 0.9705284
```

Podemos observar que la mejor liga con el método agnes es la Ward

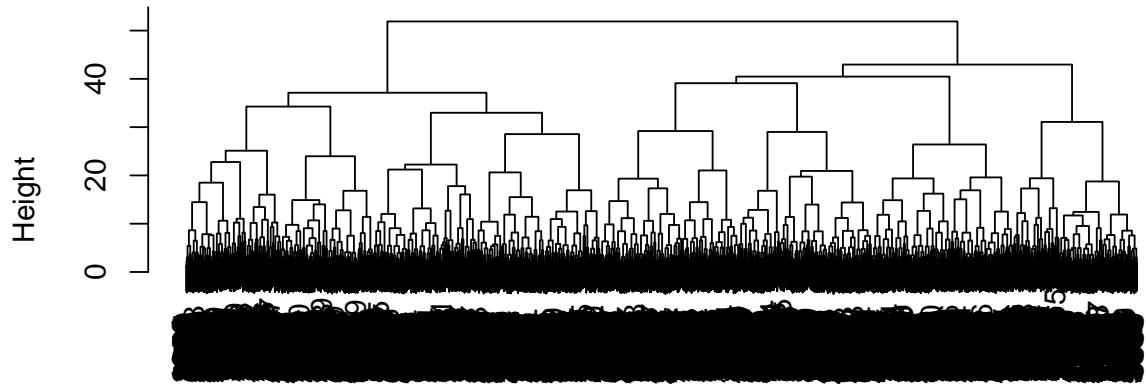
Veamos su gráfica:

Banner of agnes(x = df1, metric = "euclidean", method = "ward")



Agglomerative Coefficient = 0.97

Dendrogram of agnes(x = df1, metric = "euclidean", method = "ward")



df1
Agglomerative Coefficient = 0.97

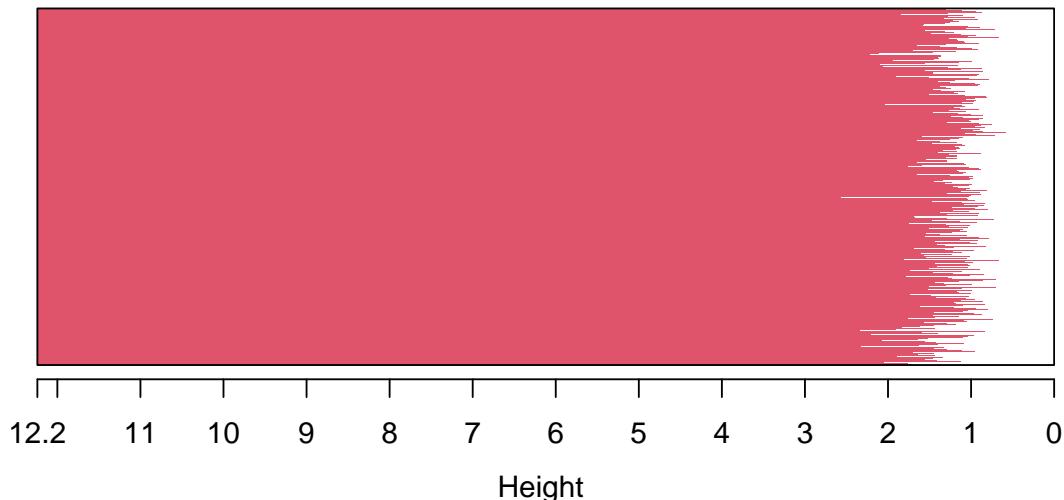
Podemos observar que tenemos un comportamiento similar a las dos ligas Ward anteriores. Por lo que podemos decir que si hacemos un corte con altura alrededor de 40 tenemos tres clusters.

DIANA

Veamos con otro método divisivo llamado DIANA.

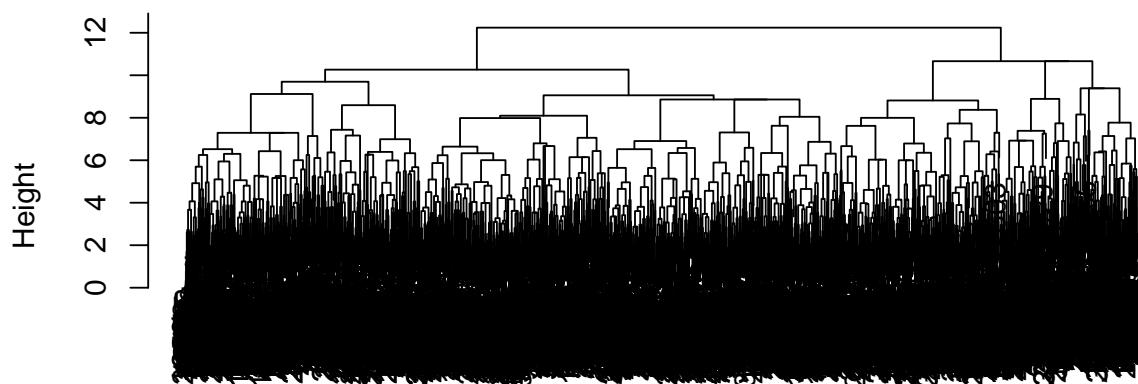
```
diana_clus <- diana(df1, metric = "euclidean")
```

Banner of `diana(x = df1, metric = "euclidean")`



Divisive Coefficient = 0.85

Dendrogram of `diana(x = df1, metric = "euclidean")`



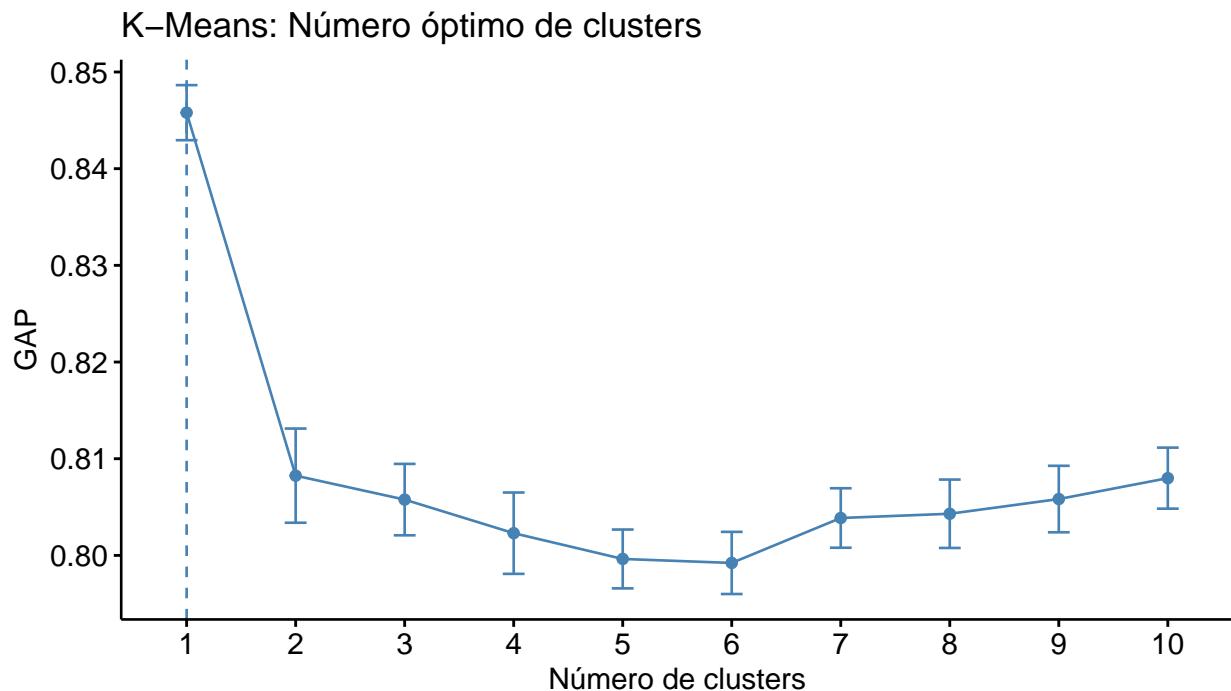
`df1`
Divisive Coefficient = 0.85

Podemos observar que con DIANA tenemos 2 clusters si hacemos un corte alrededor de altura 11.
Con los métodos jerárquicos podemos afirmar que tenemos entre 2 y 3 clusters.

Métodos no Jerárquicos

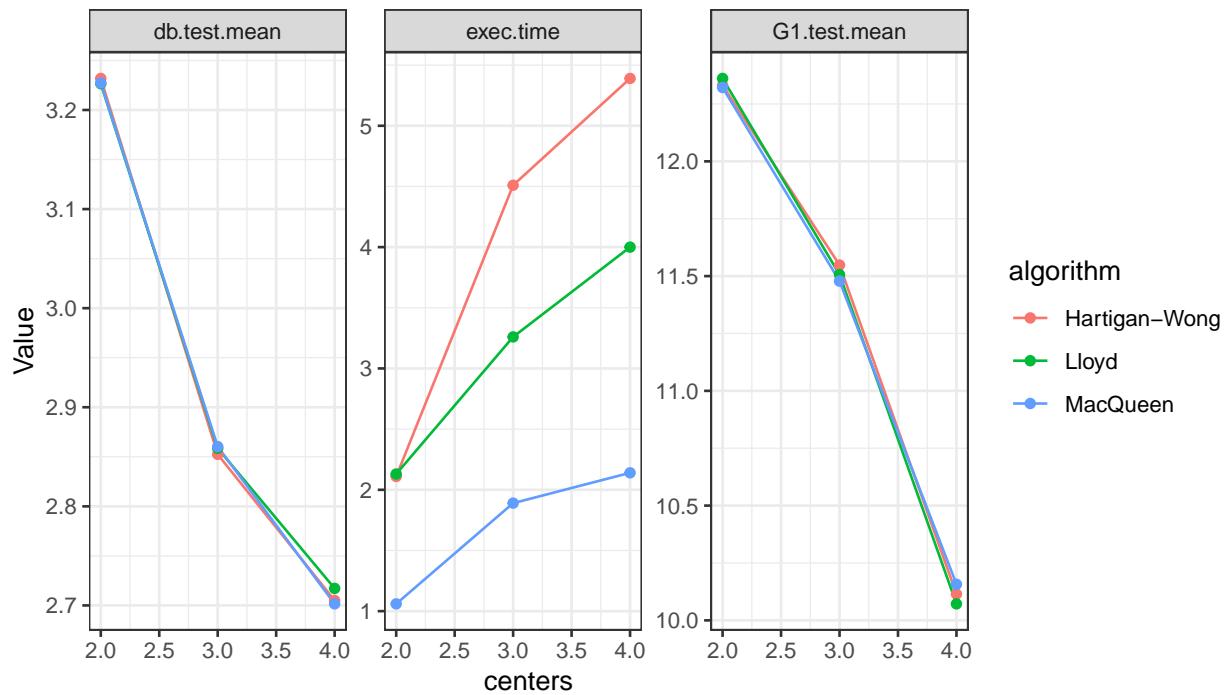
K-MEANS

Primero veremos el estadístico **GAP** para ver el número de clusters que puede ser adecuado.

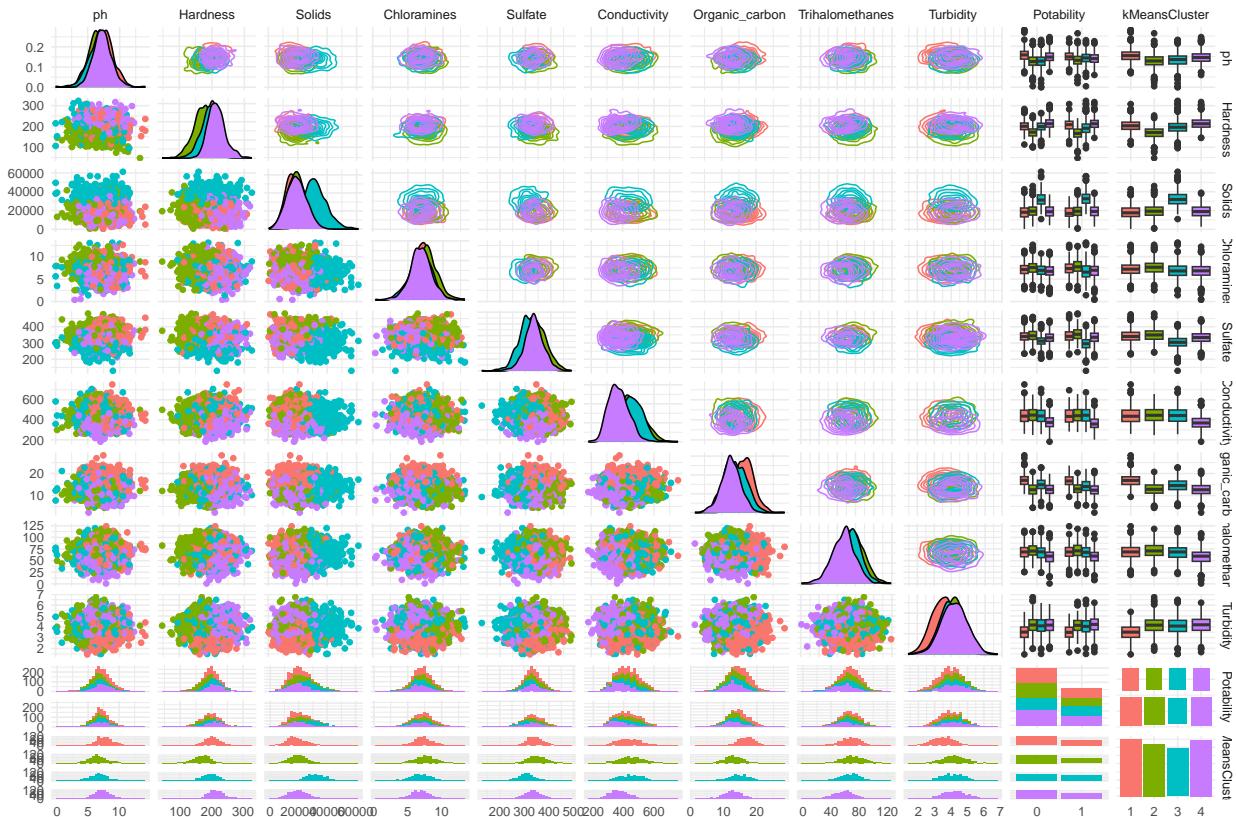


Con K-Means indica que el número óptimo de clusters es 1, es decir que no encuentra más de un cluster como óptimo.

(Aún así) Para el método de K-Means probaremos con valores de K de 2, 3 y 4, y mediante validación cruzada de 25 iteraciones escogeremos el valor óptimo de K.



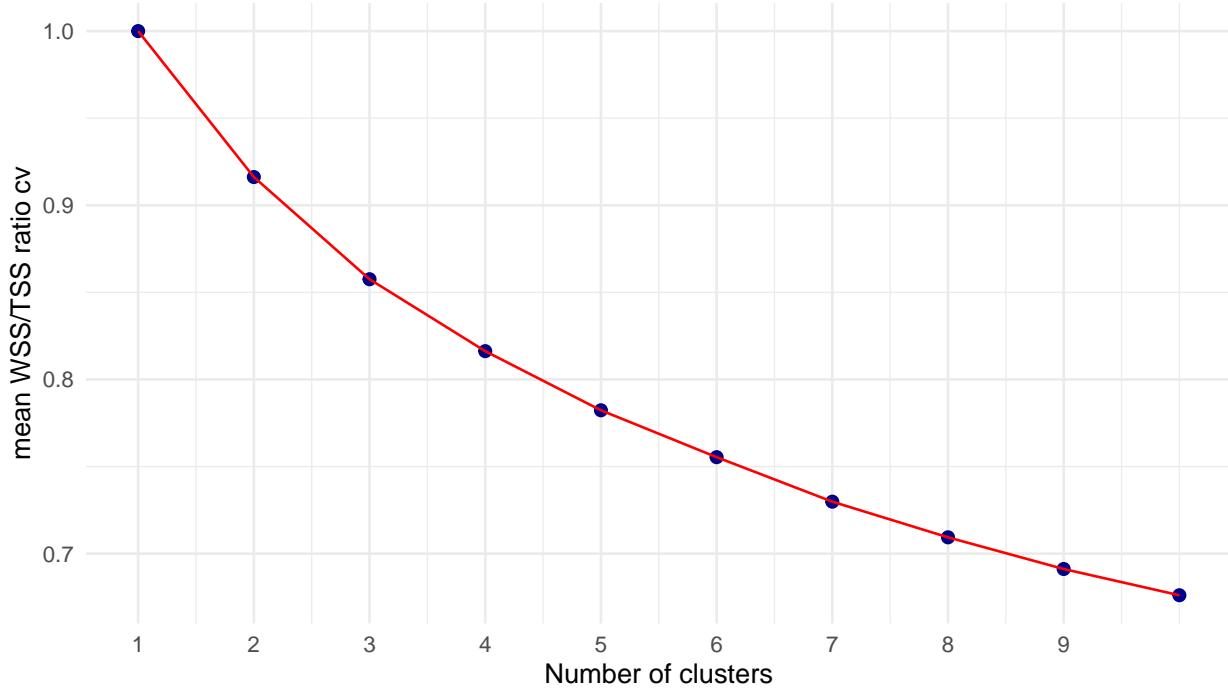
En la gráfica anterior vemos que el mejor modelo es con $K = 4$.



En los gráficos anteriores podemos ver que no se percibe una buena separación de los clusters. Además, tampoco se ve que los clusters estén relacionados con la potabilidad del agua, pues están casi igualmente

distribuidos en ambos grupos de potabilidad.

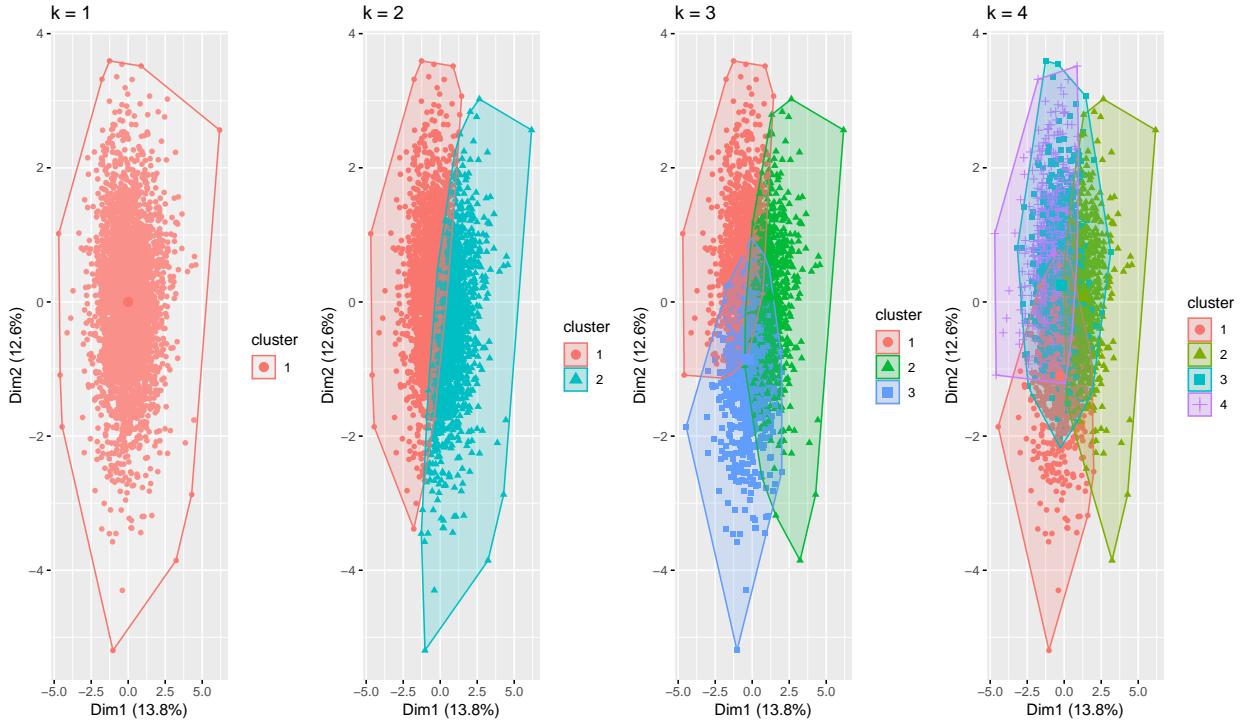
Ahora realizamos una gráfica de codo para visualizar la proporción WSS/TSS o SSE Ratio que es una medida de la proporción de la varianza total para buscar un clúster óptimo.



Observando la gráfica, el “codo” parece estar cerca del cluster número 1 a 4, donde hay un cambio más notable en la pendiente. Por lo que el número óptimo de clusters probablemente esté entre estos números. Más de 5 clústers no suele ser óptimo ya que a partir del cluster 5 las reducciones son cada vez menores, es decir, la curva se aplana, lo que sugiere que agregar más clusters ya no mejora significativamente.

Veamos cuáles de los clústers anteriores puede ser el más óptimo.

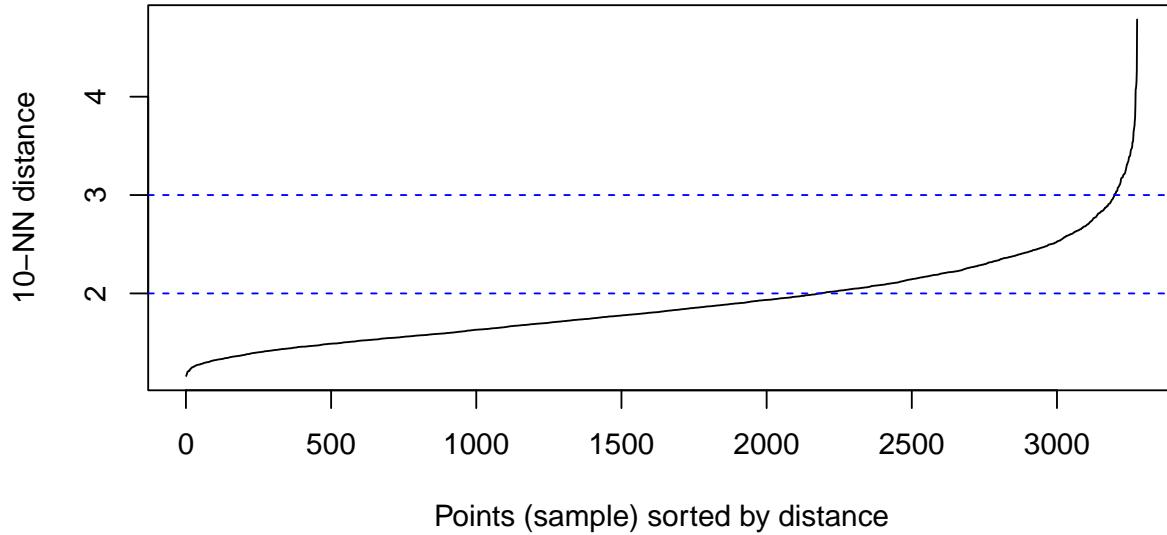
Creamos una visualización de los resultados del modelo anterior K-Means con k=1,2,3 y 4.



De la gráfica anterior concluimos que: con $k=2$ los grupos aún no son claramente separados, con $k=3$ se mejora la separación ya que algunos grupos están mejor definidos pero solamente en los extremos y con $k=4$ nuevamente no hay mucha claridad. Por lo que nuevamente con el modelo implementado de esta manera no existe para nuestro dataset una separación óptima.

DBSCAN

Para el parámetro del número de vecinos cercanos tomaremos 10 por tener 9 variables del modelo.



En la gráfica anterior podemos ver que un intervalo donde podemos ver el punto de inflexión en las distancias es entre 2 y 3.

Si calculamos los cuantiles, vemos que el 95% de las distancias quedamos debajo de 2.71. Por lo que un valor adecuado de epsilón podría ser 3.

```
quantile(kNNdist(df1, k = 10), c(0.70, 0.75, 0.8, 0.9, 0.95, 0.99, 0.995, 0.999))
```

```
##      70%      75%      80%      90%      95%      99%     99.5%     99.9%
## 2.044612 2.116021 2.212688 2.465946 2.718829 3.326106 3.518717 4.104107
```

```
df_clus <- dbSCAN(df1, eps = 3, minPts = 10)
df_clus
```

```
## DBSCAN clustering for 3276 objects.
## Parameters: eps = 3, minPts = 10
## Using euclidean distances and borderpoints = TRUE
## The clustering contains 1 cluster(s) and 7 noise points.
##
##      0      1
##      7 3269
##
## Available fields: cluster, eps, minPts, dist, borderPoints
```

Los resultados indican 10 observaciones de ruido y el resto pertenece a un solo grupo.

Model-Based Clustering

Explorando el número de clusters:

```

set.seed(123)
K <- 5
models <- lapply(1:K, function(k) Mclust(df1, G = k))
bic_values <- sapply(models, function(model) BIC(model))
bic_values

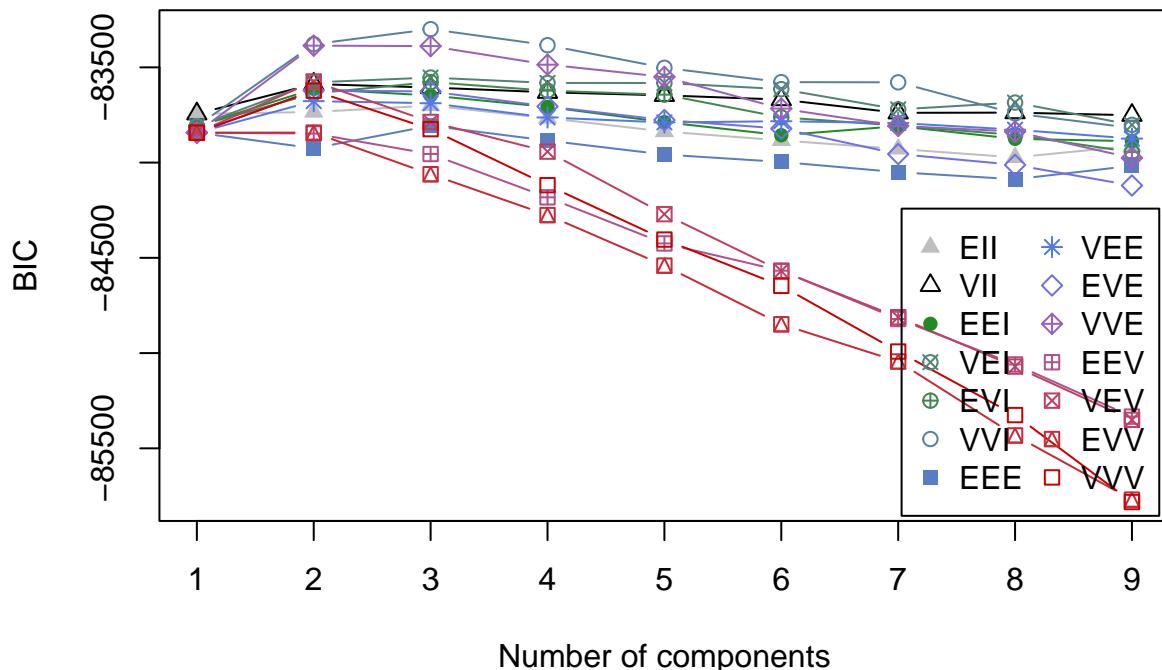
```

```
## [1] 83743.91 83377.79 83442.22 83373.39 83422.72
```

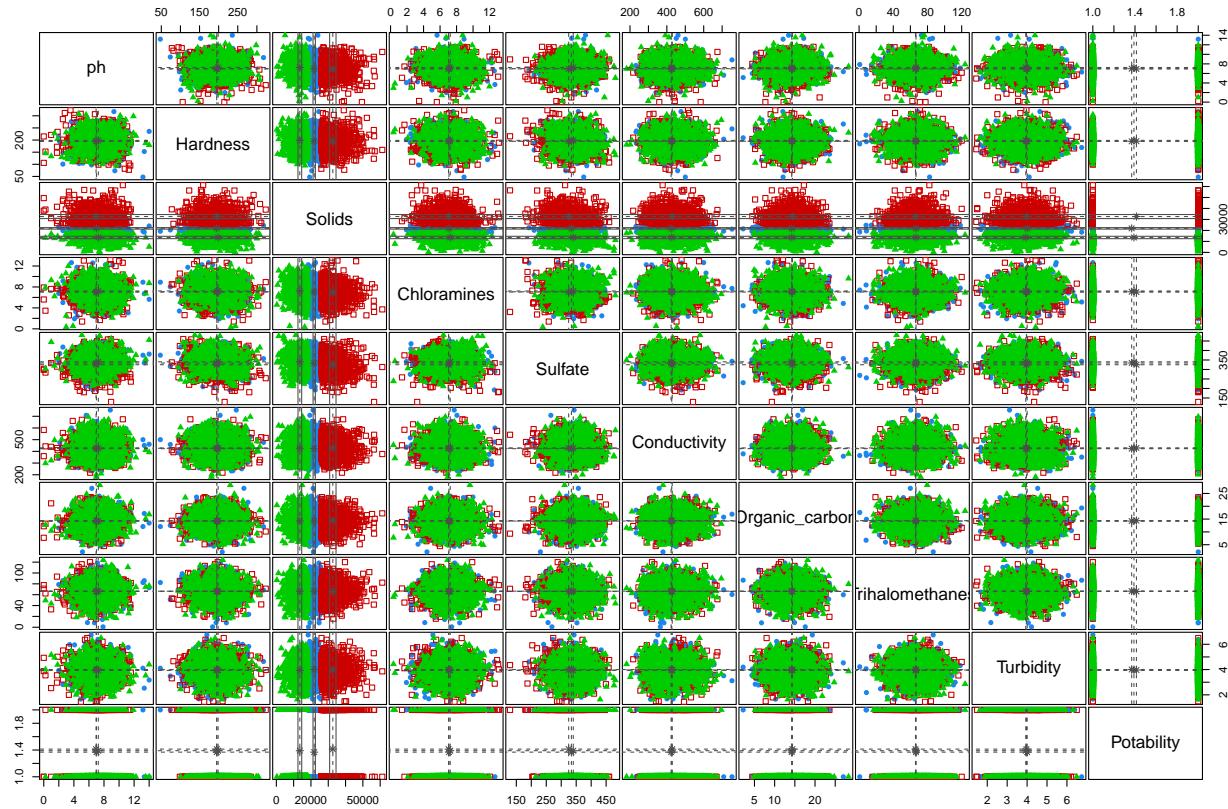
En esta salida vemos los valores del BIC (Bayesian Information Criterion) para los modelos ajustados con 1 a 5 clusters.

En mclust, el mejor modelo es aquel con el BIC mayor. En este caso el valor más alto es 83,743, correspondiente a $k = 1$ cluster lo que sugiere que, según el criterio BIC, no hay evidencia suficiente para justificar múltiples grupos.

Ahora vamos a determinar cuál modelo de mezcla gaussiana se ajusta mejor a los datos.



En esta gráfica vemos que en este caso es el modelo VVI con tres clusters.



De esta gráfica vemos que algunas combinaciones de variables muestran separación entre grupos, como: Solids vs Conductivity, Hardness vs Solids, ph vs Sulfate. Otras combinaciones tienen mezcla considerable de puntos, lo que indica solapamiento entre clusters.

Conclusión

Si bien con los métodos presentados (a excepción de DBSCAN) es posible generar clusters, hay una superposición significativa entre los mismos en muchas combinaciones de variables, lo que indica que no todos los atributos separan bien los grupos.

Además, las variables no presentan una estructura adecuada para poderlas separar en grupos de potabilidad y tampoco en clusters.

Concluimos que con ninguno de los métodos es posible generar clusters adecuados.