UNIVERSIDAD DE GRANADA

Práctica 2:
Clustering

Autor
Juan Manuel Castillo Nievas

Máster Profesional en Ingeniería Informática

—

Granada, 15 de diciembre de 2020

Índice

1. Introducción

En esta práctica se van a aplicar distintas técnicas de *clustering* sobre el conjunto **wine.data**. Este conjunto contiene datos reales de 178 vinos de una misma región de Italia. Cada instancia se compone de **13 atributos numéricos** y una clase que determina el nivel de alcohol de vino (tipo 1, 2 ó 3), y es la solución del proceso *clustering*.

2. Clustering jerárquico

Todas las variables son numéricas, con lo cual se ha utilizado la **distancia euclídea** para calcular la similitud de los datos. Como método de aglomeración se ha usado **"complete"**, que es un método bastante común y que produce clusters más compactos [1]. Se ha creado un **dendrograma** (ver Figura 1).

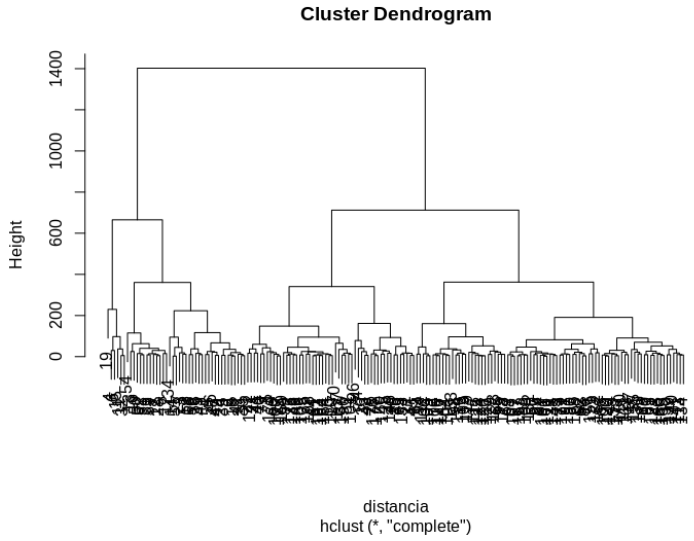

Figura 1: Dendrograma

A pesar de ya saber que hay 3 clusters, al ver el dendrograma puedo analizar diferentes formas de agrupar el dataset:

- **3 clusters:** en la Figura 2 se pueden ver claramente tres clusters diferenciados.
- **4 clusters:** en la Figura 3 se puede ver que el primer cluster se podría dividir en dos subgrupos, haciendo un total de 4 clusters.
- **7 clusters:** de forma un poco menos intuitiva, en la Figura 4 puede verse que pueden diferenciarse 7 clusters.

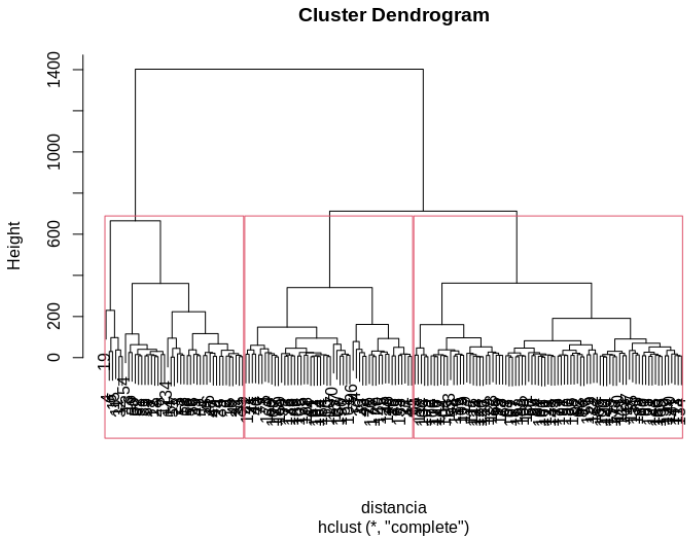

Figura 2: Agrupamiento en 3 clusters

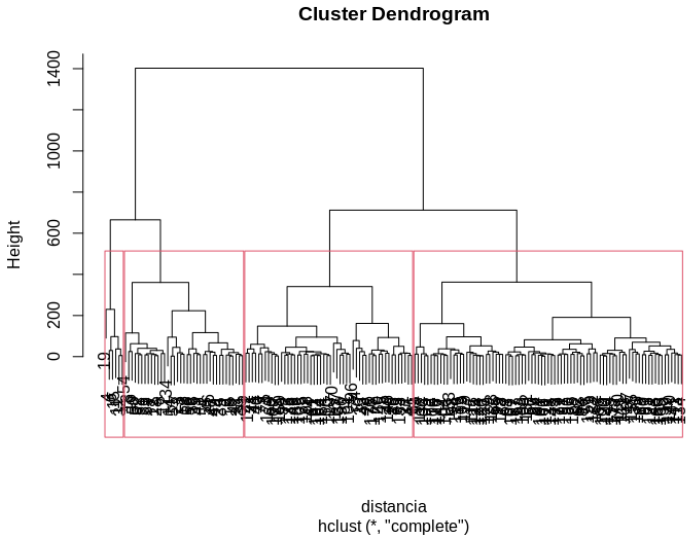

Figura 3: Agrupamiento en 4 clusters

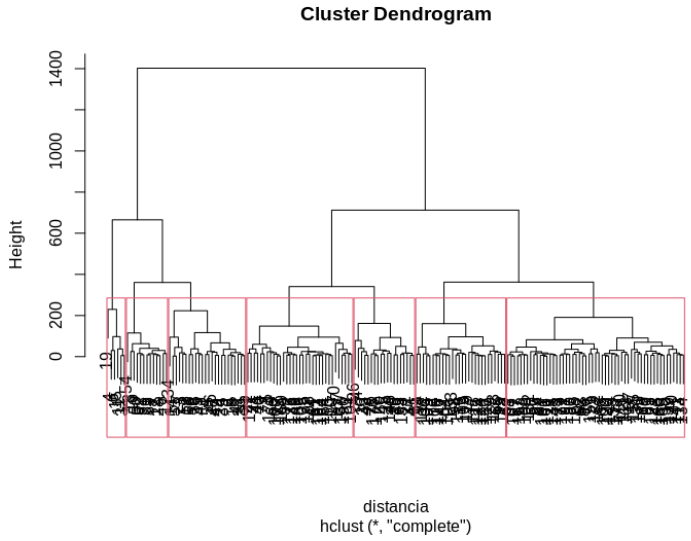

Figura 4: Agrupamiento en 7 clusters

El mejor clustering obtenido de acuerdo a mi criterio se consigue con **3 clusters**.

En la Listing 1 se puede ver el código usado en R para realizar este clustering jerárquico.

```
# Elijo las columas con datos numericos (todas las columnas menos la primera
# que es la solución del cluster)
wine_data=data.frame(wine[,2:14])
distancia=dist(wine_data)
# Aplico clustering jerárquico
clustering_jerarquico=hclust(distancia,method="complete")
clustering_jerarquico
plot(clustering_jerarquico)
```

Listing 1: Clustering jerárquico

3. Detección de outliers

Para la detección de outliers se ha usado un **diagrama de caja y bigotes** para cada variable y, en caso de detectar posibles outliers, se han visualizado los datos de la variable para ver cuáles son esos posibles outliers.

El código usado en RStudio para este fin se puede ver en la Listing 2.

A continuación, en las siguientes secciones se van a presentar aquellas variables que tienen posibles outliers de acuerdo a su diagrama de caja y bigotes y se va a analizar para eliminarlos en caso de que sea necesario.

```
1 boxplot(wine_data$Ash)
2 plot(wine_data$Ash)
```

Listing 2: Visualizar posibles outliers de una variable (variable **Ash** como ejemplo)

3.1. Ash

En las Figuras 5 y 6 se muestra el diagrama de cajas y bigotes y la visualización de esta variable, respectivamente.

Hay 3 valores que pueden ser posibles outliers, pero realmente **no los considero como tal** porque tampoco suponen una gran distorsión en los datos, por lo que no los consideraría valores atípicos.

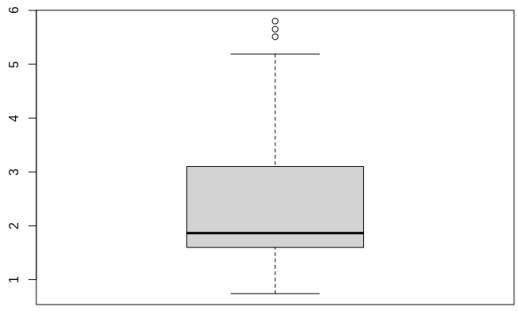

Figura 5: Diagrama de caja y bigotes de la variable **Ash**

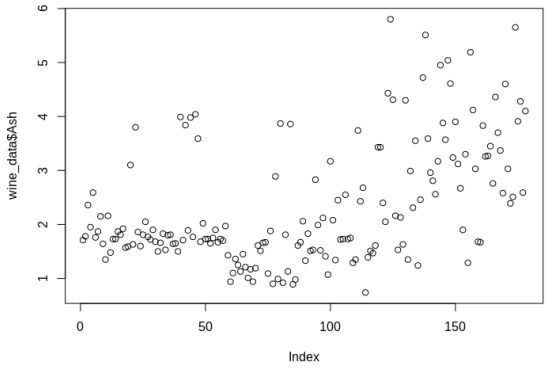

Figura 6: Visualización de los datos de la variable **Ash**

3.2. Alcalinity of ash

En las Figuras 7 y 8 se muestra el diagrama de cajas y bigotes y la visualización de esta variable, respectivamente.

Se detectan 3 posibles outliers y **voy a proceder a eliminarlos** ya que están alejados de los valores más frecuentes. 2 de ellos superan el valor 3.0 y 1 de ellos es inferior a 1.5.

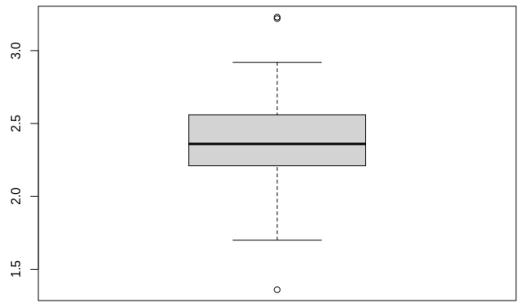

Figura 7: Diagrama de caja y bigotes de la variable **Alcalinity of ash**

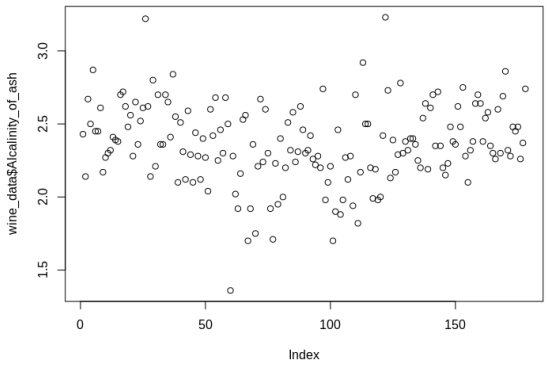

Figura 8: Visualización de los datos de la variable **Alcalinity of ash**

3.3. Magnesium

En las Figuras 9 y 10 se muestra el diagrama de cajas y bigotes y la visualización de esta variable, respectivamente.

Se detectan 3 posibles outliers. Realmente **sólo 1 de ellos lo considero un verdadero outlier**, y es aquel que tiene un valor de 30.

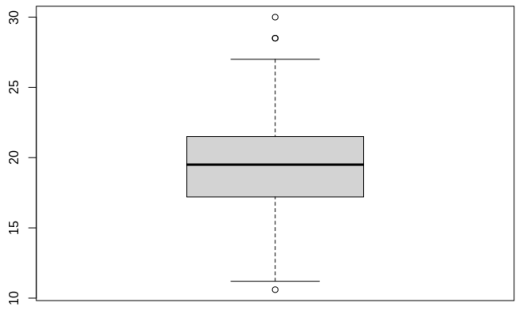

Figura 9: Diagrama de caja y bigotes de la variable **Magnesium**

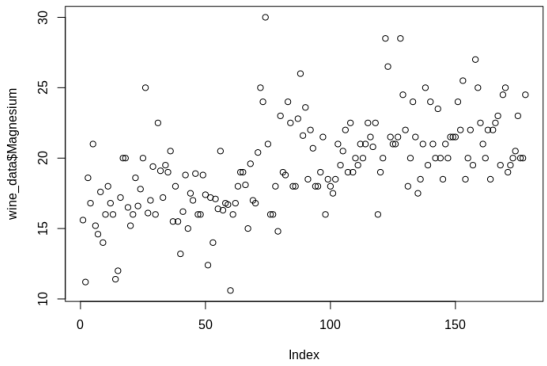

Figura 10: Visualización de los datos de la variable **Magnesium**

3.4. Total phenols

En las Figuras 11 y 12 se muestra el diagrama de cajas y bigotes y la visualización de esta variable, respectivamente.

Hay varios posibles outliers. Sólo voy a considerar como outliers **los dos que tienen los valores más altos**, ya que los demás están bastante cerca de los valores habituales.

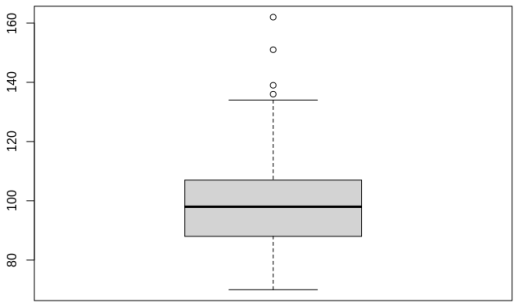

Figura 11: Diagrama de caja y bigotes de la variable **Total phenols**

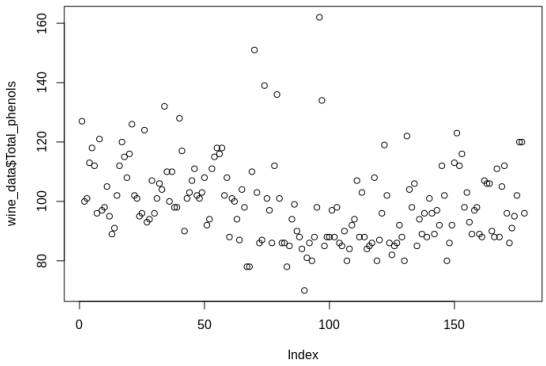

Figura 12: Visualización de los datos de la variable **Total phenols**

3.5. Color intensity

En las Figuras 13 y 14 se muestra el diagrama de cajas y bigotes y la visualización de esta variable, respectivamente.

Hay 2 posibles outliers. **Voy a considerar como outlier únicamente el valor más alto** ya que el otro me parece que puede ser un ruido más que un outlier.

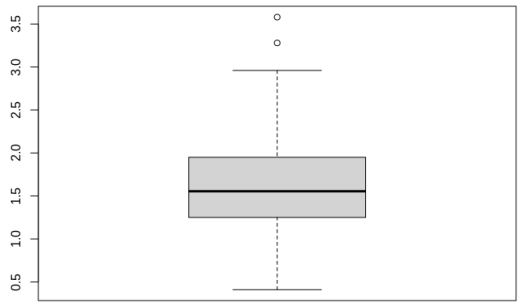

Figura 13: Diagrama de caja y bigotes de la variable **Color intensity**

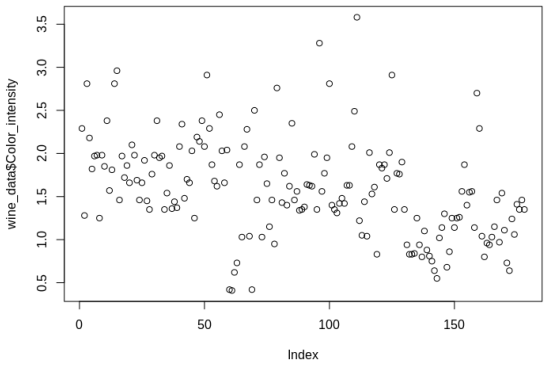

Figura 14: Visualización de los datos de la variable **Color intensity**

3.6. Hue

En las Figuras 15 y 16 se muestra el diagrama de cajas y bigotes y la visualización de esta variable, respectivamente.

Hay 3 posibles outliers. **Voy a considerar como outlier únicamente el valor más alto**, ya que tiene un valor más grande que 12 y sí es un valor más atípico que los otros dos posibles outliers.

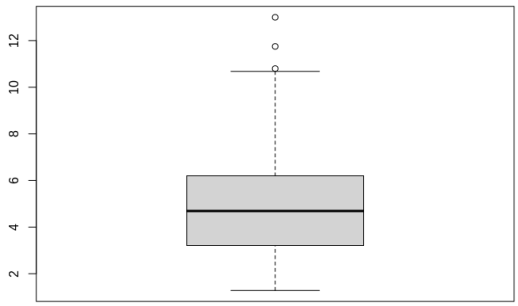

Figura 15: Diagrama de caja y bigotes de la variable **Hue**

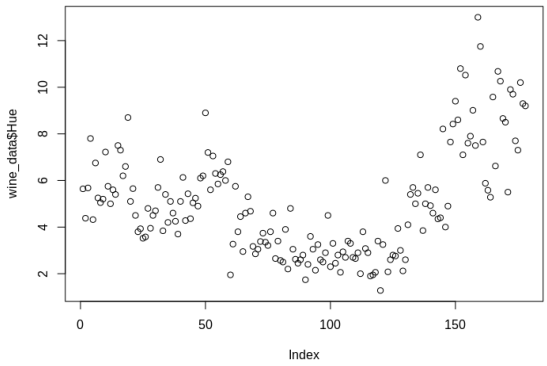

Figura 16: Visualización de los datos de la variable **Hue**

3.7. OD280.OD315 of diluted wines

En las Figuras 17 y 18 se muestra el diagrama de cajas y bigotes y la visualización de esta variable, respectivamente.

Está muy claro que **hay un valor que es un outlier** ya que está muy alejado de todos los demás datos.

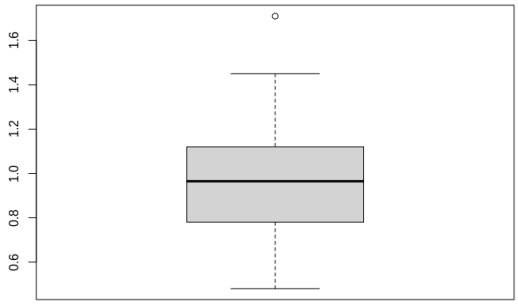

Figura 17: Diagrama de caja y bigotes de la variable **OD280.OD315 of diluted wines**

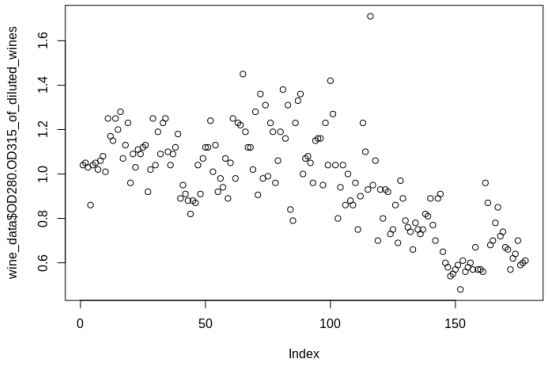

Figura 18: Visualización de los datos de la variable **OD280.OD315 of diluted wines**

3.8. Conclusiones y outliers definitivos

De acuerdo al análisis que se ha hecho en esta sección de los posibles outliers y las conclusiones sacadas en cada variable, se van a eliminar los siguientes datos:

- **Filas 26, 122 y 60** por ser outliers en la variable **Alcalinity of ash**
- **Fila 74** por ser outlier en la variable **Magnesium**
- **Filas 70 y 96** por ser outliers en la variable **Total phenols**
- **Fila 111** por ser outlier en la variable **Color intensity**
- **Fila 159** por ser outlier en la variable **Hue**
- **Fila 116** por ser outlier en la variable **OD280.OD315 of diluted wines**

Se pasa de tener 178 datos a tener 169 datos. En la Figura 19 se puede ver el dendrograma que ha salido como resultado de aplicar el clustering jerárquico. De nuevo se ha utilizado la **distancia euclídea** y el método **"complete"**.

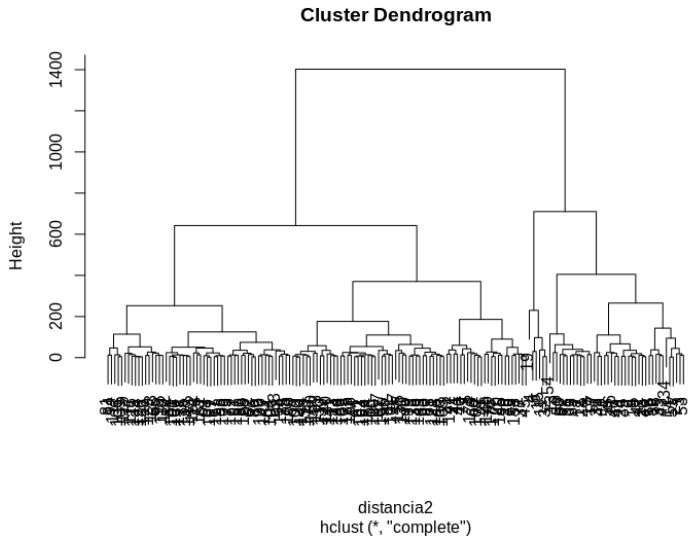

Figura 19: Dendrograma sin outliers

En este caso agruparía los datos en **3 ó 4 grupos**.

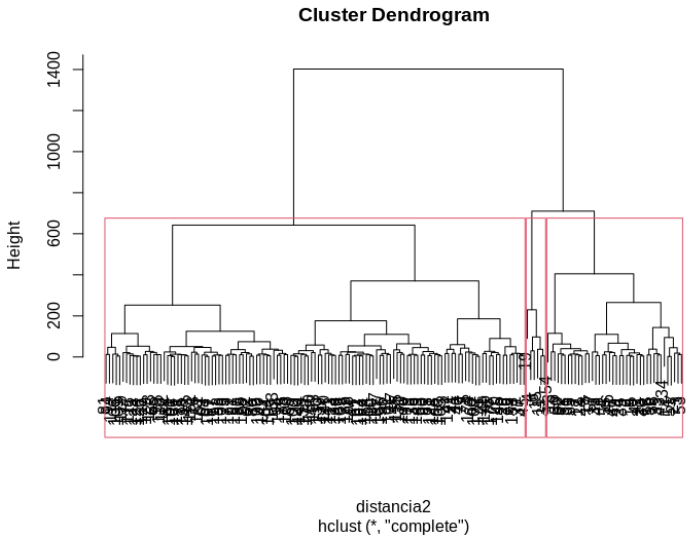

Figura 20: Agrupación en 3 clusters con los datos sin outliers

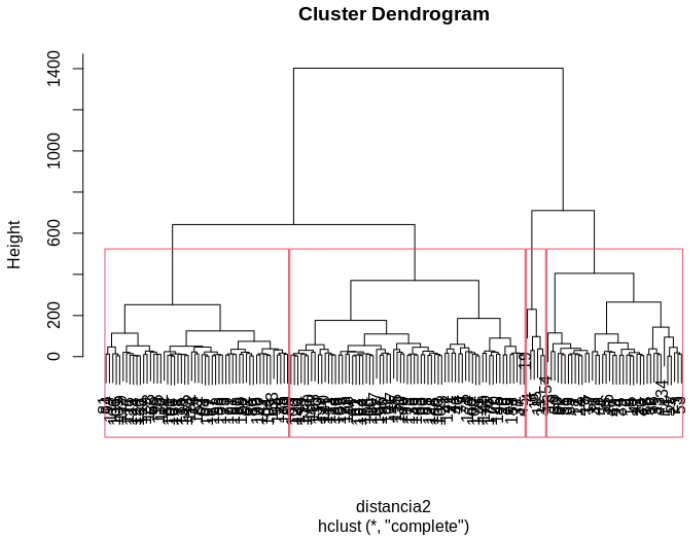

Figura 21: Agrupación en 4 clusters con los datos sin outliers

En las Figura 20 se puede ver la separación en 3 clusters que hace R en el dendrograma. Esta separación yo la haría diferente, pues hay un cluster que está formado por muy pocos datos. El tercer cluster en el que yo separaría este dendrograma sería el segundo cluster que aparece en la Figura 21, que es el que contiene la agrupación en 4 clusters.

Al ver esta separación, he probado a realizar el clustering jerárquico con el método **"ward.D2"** y he obtenido un dendrograma en el que puedo hacer una agrupación en 3 clusters (ver Figura 22) o en 4 clusters (ver Figura 23). La agrupación en 3 clusters en este caso es más coherente, al contrario que pasó en la Figura 20.

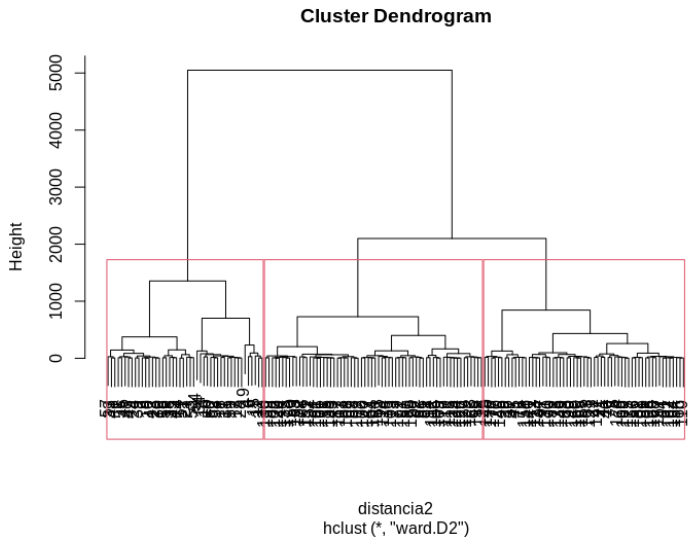

Figura 22: Dendrograma usando el método **ward.D2** y agrupación en 3 clusters

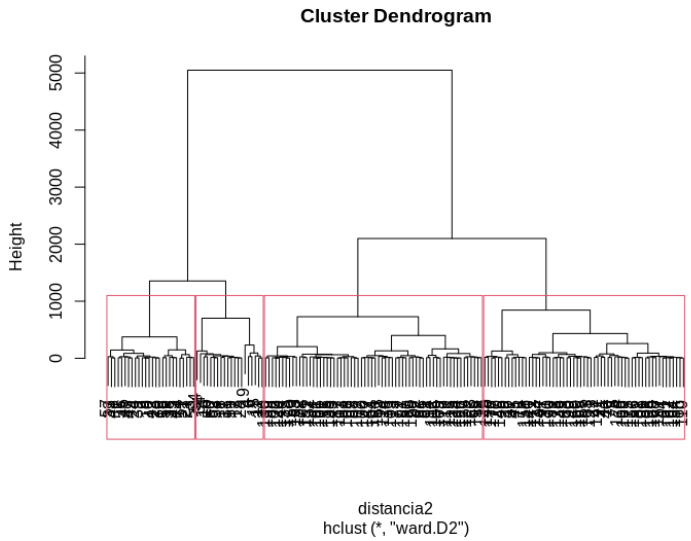

Figura 23: Dendrograma usando el método **ward.D2** y agrupación en 4 clusters

En el Listing 3 se muesta el código usado para este clustering jerárquico.

```r
# Elimino las filas que son outliers
wine_no_outliers <- wine_data[-c(26,122,60,74,70,96,111,159,116),]
# Calculo distancia euclídea
distancia2=dist(wine_no_outliers)
# Aplico clustering jerárquico con metodo complete
clustering_jerarquico2=hclust(distancia2,method="complete")
clustering_jerarquico2
plot(clustering_jerarquico2)
rect.hclust(clustering_jerarquico2,k=3)
rect.hclust(clustering_jerarquico2,k=4)
# Aplico clustering jerárquico con metodo ward.D2
clustering_jerarquico2=hclust(distancia2,method="ward.D2")
clustering_jerarquico2
plot(clustering_jerarquico2)
rect.hclust(clustering_jerarquico2,k=3)
```

Listing 3: Clustering jerárquico sin outliers

4. Algoritmo k-medias

En la Sección 3 se concluyó que había 3 ó 4 clusters. En esta sección se ha aplicado el algoritmo k-medias para agrupar los datos usando ***k=3*** y ***k=4***. En la Listing 4 se muestra el código usado. Se puede ver que se ha usado el dataset sin los outliers que se detectaron anteriormente.

```
set.seed(1234)
distancia=dist(wine_no_outliers)
# K-means con k=3
kmeans3=kmeans(distancia,3)
grupo3=kmeans3$cluster
# K-means con k=4
kmeans4=kmeans(distancia,4)
grupo4=kmeans4$cluster
```

Listing 4: Algoritmo k-means

4.1. Visualización de los clusters con plotcluster

La función **plotcluster** está dentro de la librería **fpc** y sirve para visualizar los clusters que se han encontrado.

En la Figura 24 se muestran los 3 clusters que se han encontrado usado k-means con **k=3**. En la Figura 25 se muestran los 4 clusters que han surgido con **k=4**. Por último, en la Figura 26 se muestran la agrupación de los clusters en el dataset original, en la primera columna.

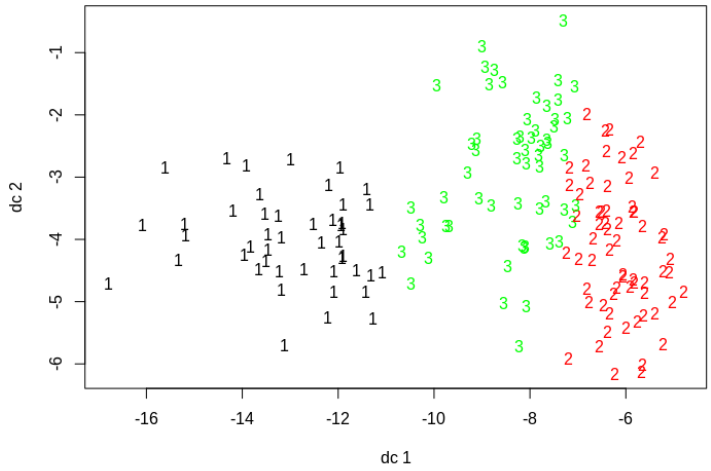

Figura 24: Visualización de los clusters usando k-means con k=3

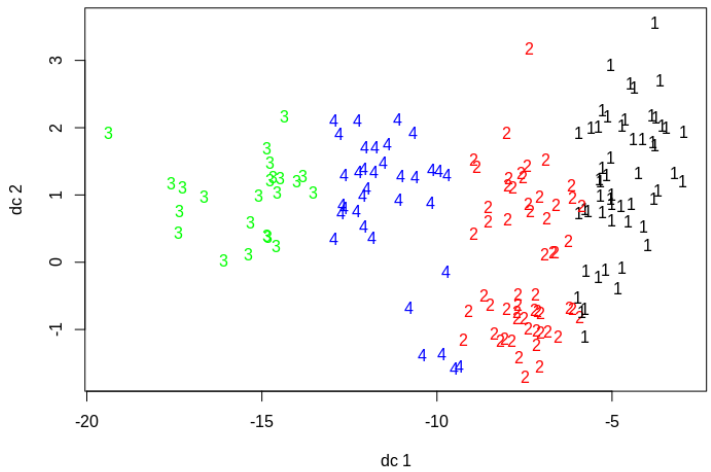

Figura 25: Visualización de los clusters usando k-means con k=4

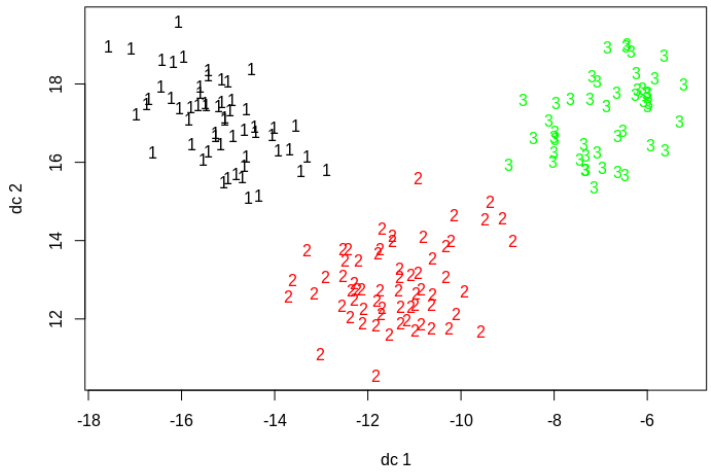

Figura 26: Visualización de los clusters correctos

4.2. Visualización de los clusters con plot

Se ha usado la función **plot** para ver el cluster asignado a cada instancia. En la Figura 27 se puede ver que:

- Las instancias 1-59 pertenecen al **cluster 1**
- Las instancias 60-130 pertenecen al **cluster 2**
- Las instancias 131-178 pertenecen al **cluster 3**

En la Figura 28 se pueden ver los clusters asignados a cada instancia de acuerdo a la agrupación realizada por el algoritmo k-means con **k=3**. Los grupos 1 y 2 se clasifican de forma muy acertada con la Figura 27. Sin embargo, el grupo 3 está bastante distanciado de su verdadera agrupación y está bastante distribuido.

En la Figura 29 se pueden ver los clusters asignados a cada instancia de acuerdo a la agrupación realizada por el algoritmo k-means con **k=4**.

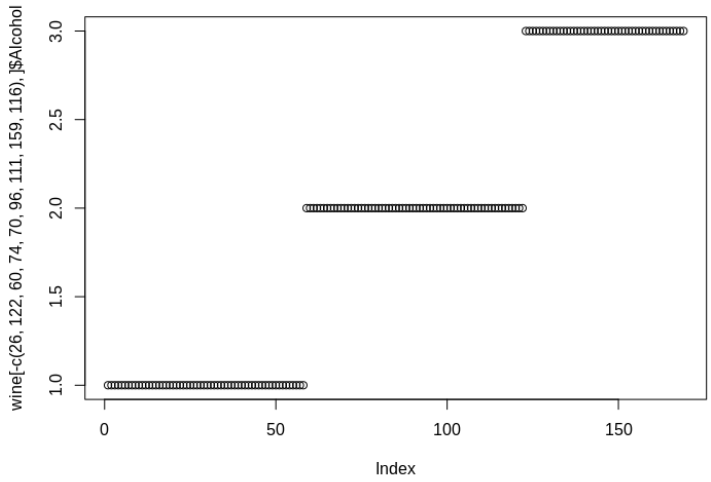

Figura 27: Cluster asignado a cada instancia en los datos originales

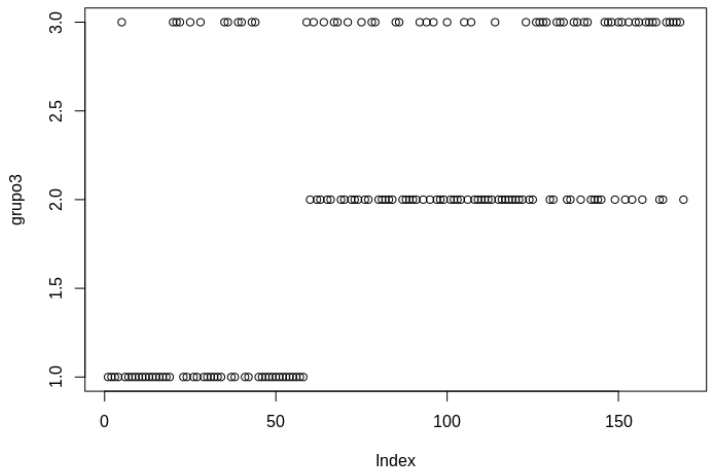

Figura 28: Cluster asignado a cada instancia de acuerdo al algoritmo k-means con **k**=3

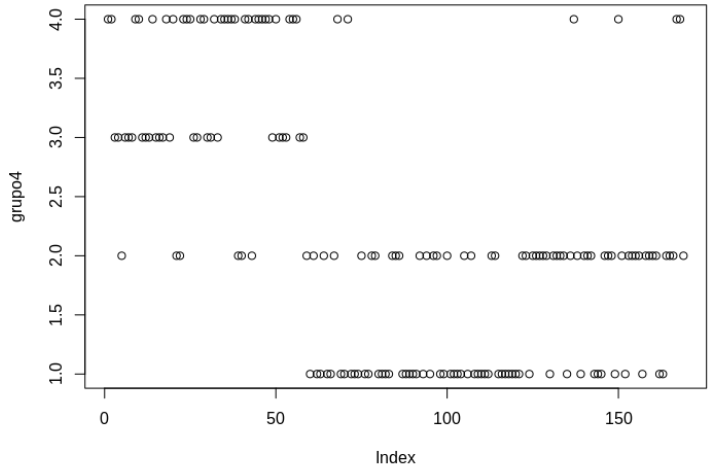

Figura 29: Cluster asignado a cada instancia de acuerdo al algoritmo k-means con **k=4**

5. Reducción de dimensionalidad y k-means con k=3

En la Figura 30 se puede ver la matriz de correlación del dataset completo (las 178 instancias). Se ha usado la matriz de correlación para ver qué variables están mas relacionadas entre sí y eliminar aquellas que tengan una relación muy fuerte.

Se va a eliminar la variable **Flavanoids** porque tiene una fuerte relación con **Nonflavanoid_phenols**.

La variable **Proline** tiene una fuerte relación con **Flavanoids** (que ya está eliminada) y con **Nonflavanoid_phenols**, pero realmente esta última no se puede eliminar porque se ha mantenido debido a su relación con **Flavanoids**.

En la Listing 5 se muestra el código usado para esto.

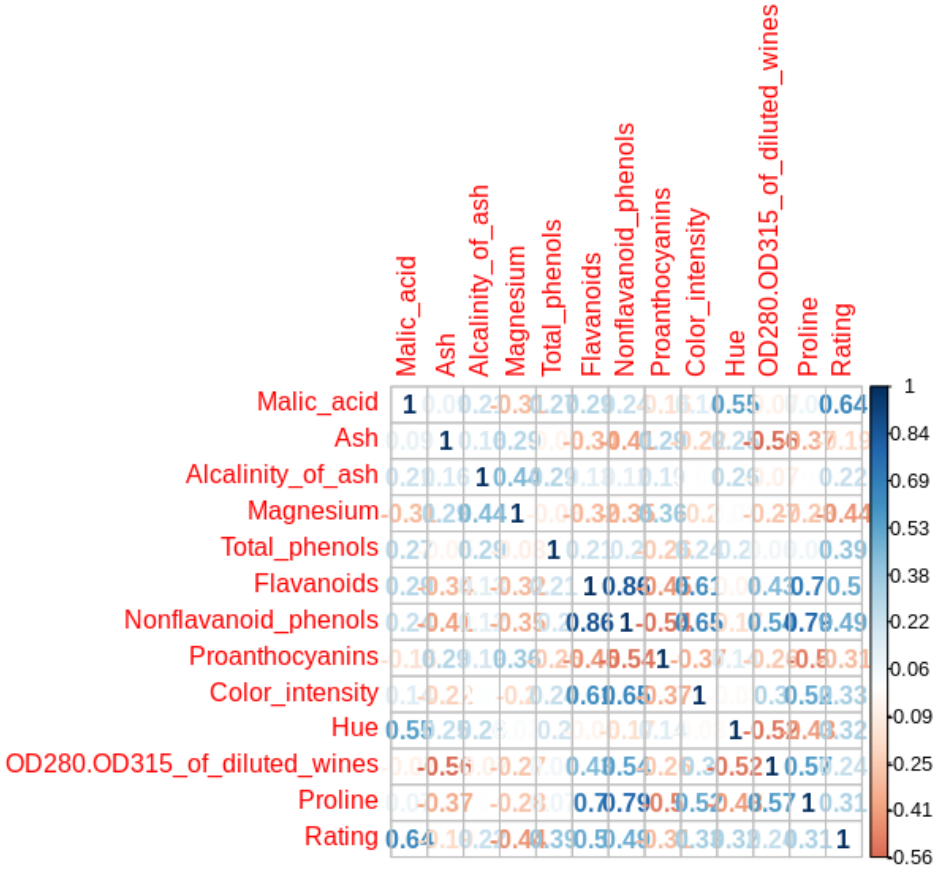

Figura 30: Matriz de correlación

```r
# Visualizar la matriz de correlación
library(corrplot)
corrplot(cor(wine_data), method="number", is.corr=FALSE)
# Elimino la columna Flavanoids
wine_reduction <- wine_data[c(-6)]
```

Listing 5: Matriz de correlación y eliminación de la variable **Flavanoids**

5.1. Visualización de los clusters con plotcluster

En la Figura 31 se muestra la visualización de los clusters generados por el algoritmo k-means con **k=3**.

En la Figura 32 se muestra la agrupación de los clusters usando la primera columna del dataset.

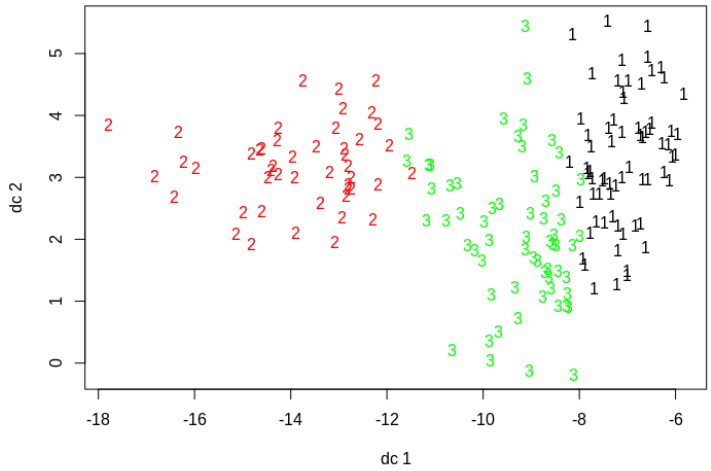

Figura 31: Visualización de los clusters generados por el algoritmo k-means con **k=3**

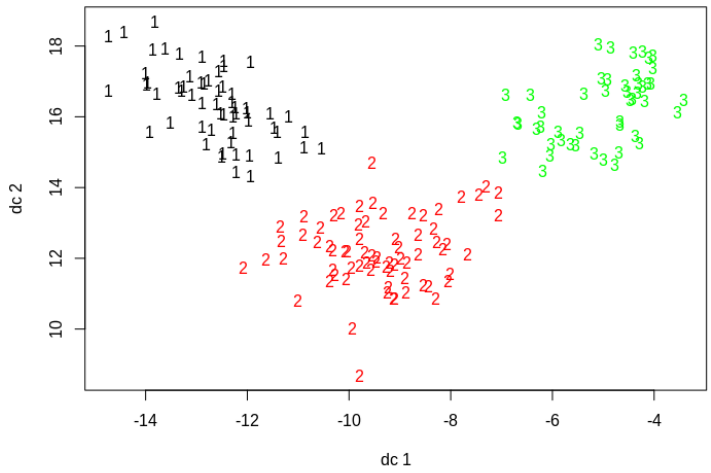

Figura 32: Visualización de los clusters generados por el algoritmo k-means con **k=4**

5.2. Visualización de los clusters con plot

En la Figura 33 se muestran el cluster asignado a cada una de las 178 instancias en el dataset original.

En la Figura 34 se muestran los clusters asignados a las 178 instancias de acuerdo al algoritmo k-means con **k=3**.

Se puede observar que en la Figura 34, los clusters 1 y 2 están cambiados, es decir, el cluster 2 realmente sería el cluster 1 de la Figura 33 y es el cluster que mejor agrupa el algoritmo k-means. Sin embargo, el cluster 3 sigue siendo el cluster peor agrupado, ya que está muy distribuido.

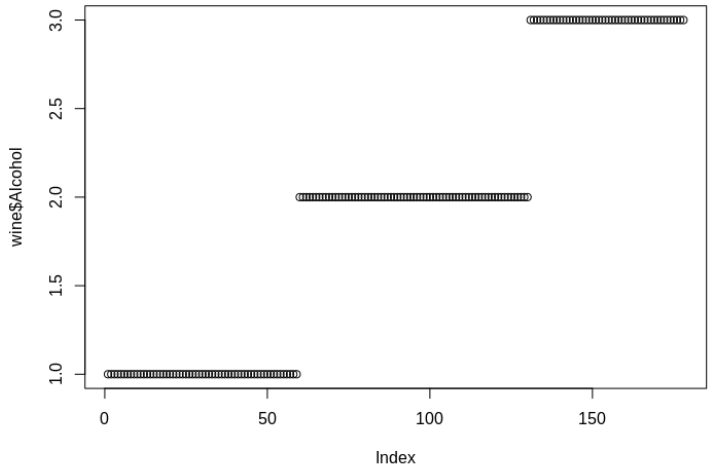

Figura 33: Clusters asignados a cada instancia en el dataset original

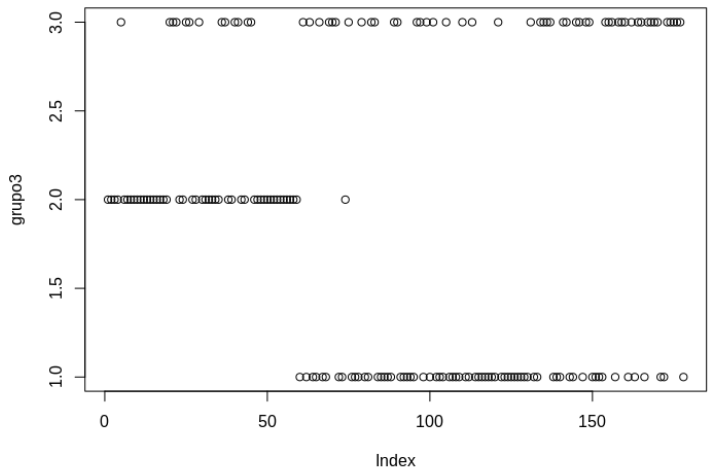

Figura 34: Clusters asignados a cada instancia en k-means con **k=3**

6. DBSCAN

Para este apartado se ha usado la librería **DBSCAN** disponible en R. Para utilizar este algoritmo es necesario averiguar cuál es el radio que se va a usar. Para ello, se ha usado la función **kNNdistplot**. Se necesita saber el número mínimo de puntos, y una heurística aconseja usar el logarítmo neperiano del número de instancias a ser agrupadas, en este caso **ln(178) que es aproximadamente 5.** En la Listing 6 se muestra el código usado para esto.

```
1 library(dbscan)
2 kNNdistplot(wine_data, k = 5)
3 abline(h = 50, lty = 2)
```

Listing 6: Averiguando el radio óptimo a usar

En la Figura 35 se puede ver la salida de la función **kNNdistplot**, y se puede ver como hay un punto de rodilla en y=50, con lo cual se va a usar este radio.

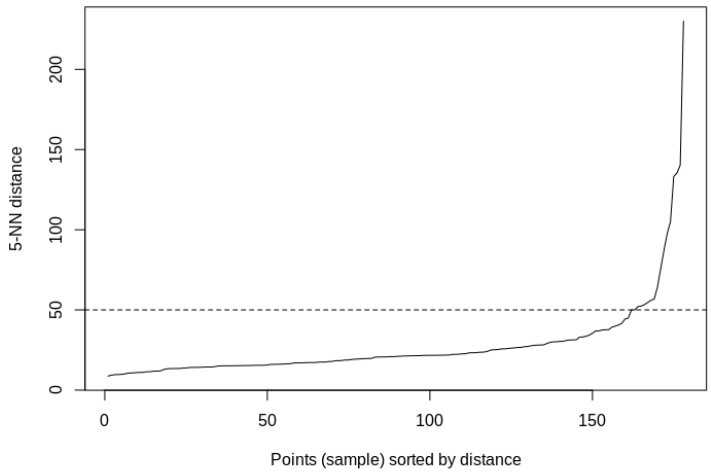

Figura 35: Salida de **kNNdistplot**

En la Listing 7 se puede ver el código usado para aplicar el algoritmo **DBSCAN** y visualizar los resultados.

```r
# Aplico DBSCAN
dbscan<-dbscan(wine_data,eps=50,minPts = 5)
plotcluster(wine_data,dbscan$cluster)
plotcluster(wine_data,wine$Alcohol)
plot(dbscan$cluster)
plot(wine$Alcohol)
```

Listing 7: DBSCAN y visualización de clusters

6.1. Visualización de los clusters con plotcluster

En la Figura 36 se pueden ver los clusters que da como resultado el algoritmo **DBSCAN**. **Hay 7 outliers que no pertenecen a ningún grupo.**

En la Figura 37 se pueden ver los clusters en los que se clasifican originalmente los datos.

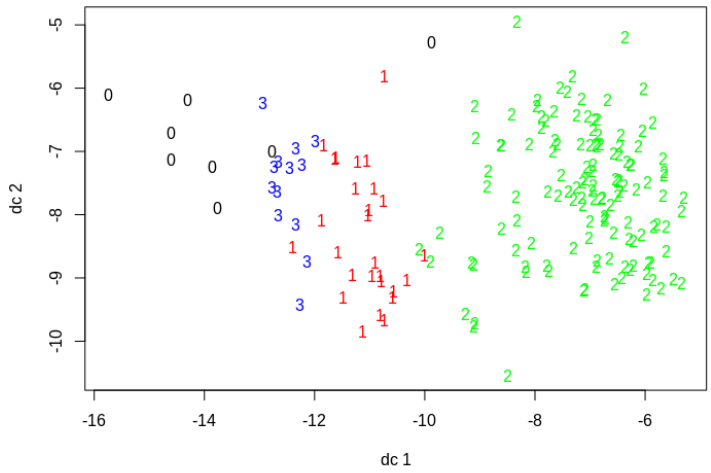

Figura 36: Clusters como resultado de **DBSCAN**

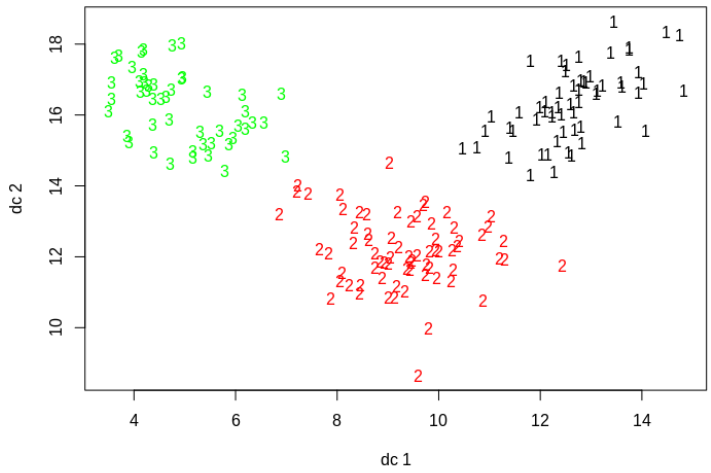

Figura 37: Clusters originales

6.2. Visualización de clusters con plot

En la Figura 38 se puede ver el grupo al que pertenece cada instancia de acuerdo al algoritmo **DBSCAN** usado. Se pueden ver las 7 instancias que no pertenecen a ningún cluster (outliers para el algoritmo). El cluster 2 tiene muchas más instancias de las que debería tener.

En la Figura 39 se puede ver la clasificación de cada instancia de acuerdo al dataset original.

Al igual que en las secciones anteriores, el cluster 1 se clasifica bastante bien. El cluster 2 se clasifica bien pero se le asignan más instancias de las que debería. El cluster 3 no se clasifica nada bien.

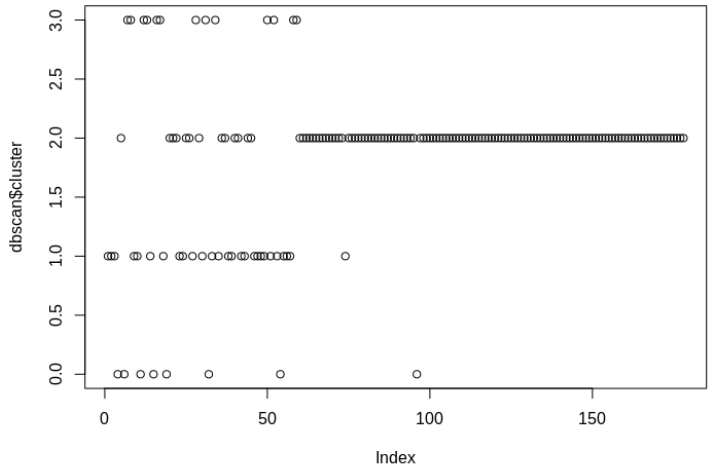

Figura 38: Clase a la que pertenece cada instancia según **DBSCAN**

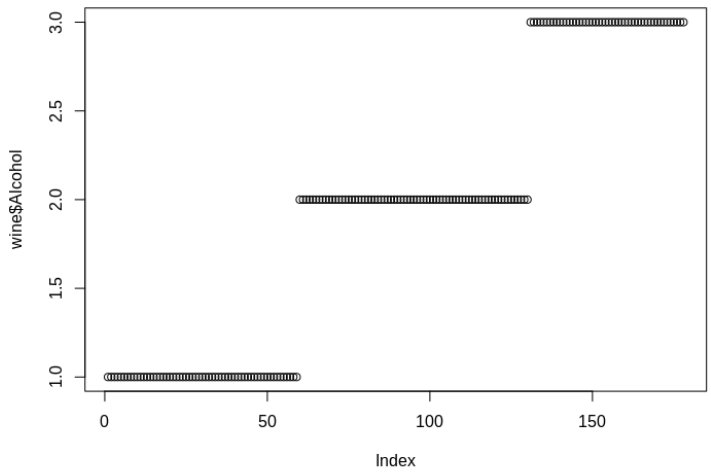

Figura 39: Clase a la que pertenece cada instancia en el dataset original

Referencias

[1] Alboukadel Kassambara. Hierarchical clustering in r: The essentials. https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/#:~:text=The%20agglomerative%20clustering%20is%20the,object%20as%20a%20singleton%20cluster.