Clase 4.0
Análisis

Marcos Rosetti y Luis Pacheco-Cobos

Estadística y Manejo de Datos con R (EMDR) — Virtual

Clustering

- Una técnica de clasificar una serie de datos en grupos.

How many clusters?

Six Clusters

Two Clusters

Four Clusters

Clustering: `kmeans`

- Uno de los métodos más viejos de agregación.
- Se eligen el número de clusters que uno cree que hay en los datos (`k`).
- El algoritmo elige k puntos al azar y calcula la distancia a todos los puntos. Las observaciones se clasifican en “agregados”.
- Se recalculan los centros y se vuelve a comenzar.
- El proceso continua hasta que las observaciones no cambian de grupo.

Clustering: **kmeans**

```
iris.kc <- kmeans(iris[1:4],3)
print(iris.kc)
```

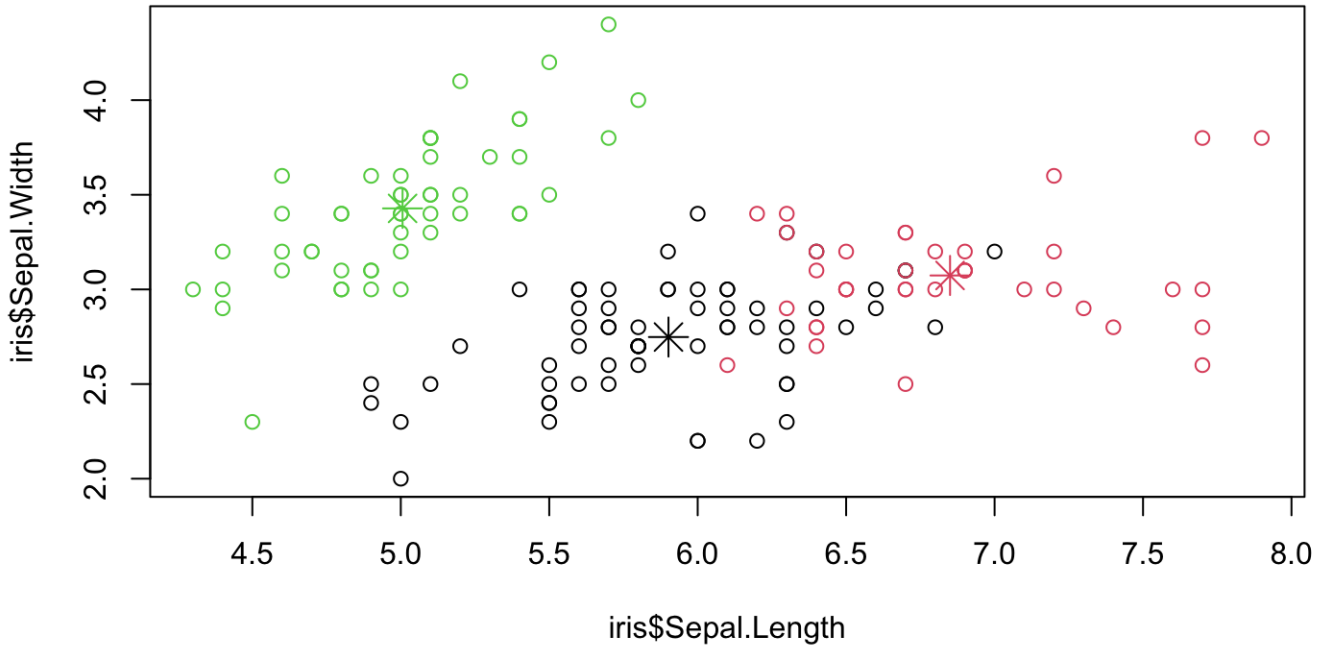
```
## K-means clustering with 3 clusters of sizes 62, 38, 50
##
## Cluster means:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     5.901613    2.748387     4.393548    1.433871
## 2     6.850000    3.073684     5.742105    2.071053
## 3     5.006000    3.428000     1.462000    0.246000
##
## Clustering vector:
##   [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2 2 2 1 2 2 2 2
## [112] 2 2 1 1 2 2 2 2 1 2 1 2 1 2 2 1 1 2 2 2 2 2 1 2 2 2 2 1 2 2 2 1 2 2 2 1 2
## [149] 2 1
##
## Within cluster sum of squares by cluster:
## [1] 39.82097 23.87947 15.15100
##  (between_SS / total_SS =  88.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
table(iris$Species, iris.kc$cluster)
```

```
##
##              1  2  3
```

Clustering: **kmeans**

```
plot(iris$Sepal.Length, iris$Sepal.Width, col=iris.kc$cluster)
points(iris.kc$centers[, c("Sepal.Length", "Sepal.Width")], col=1:3, pch=8, cex=2)
```



Clustering: `pam`

- Una alternativa más moderna de agregación es PAM (*Partición Alrededor de Medias*).
- Al igual que en `kmeans`, es necesario especificar el número estimado de clusters.
- Este algoritmo recalcula las distancias entre los objeto dentro de un cluster en cada ciclo, lo que puede dar como resultado clústers más robustos.
- Como `kmeans`, el resultado puede cambiar al cambiar `k`.

Clustering: pam

```
library(cluster)
cars.clus <- pam(scale(cars), k = 3)
plot(cars.clus)
```

Clustering: `pam`

- El gráfico de siluetas produce una medida para cada valor que nos dice que también encaja en el clúster.
- Valores cerca de 1 indican un buen ajuste, mientras que valores cerca de 0 o negativos indican que ese caso probablemente pertenezca a otro clúster.
- En cada clúster, los valores están organizados de mayor a menor.
- Criterios para elegir el número de clústers o a saber si el algoritmo está haciendo un buen trabajo.

Clustering: `hclust`

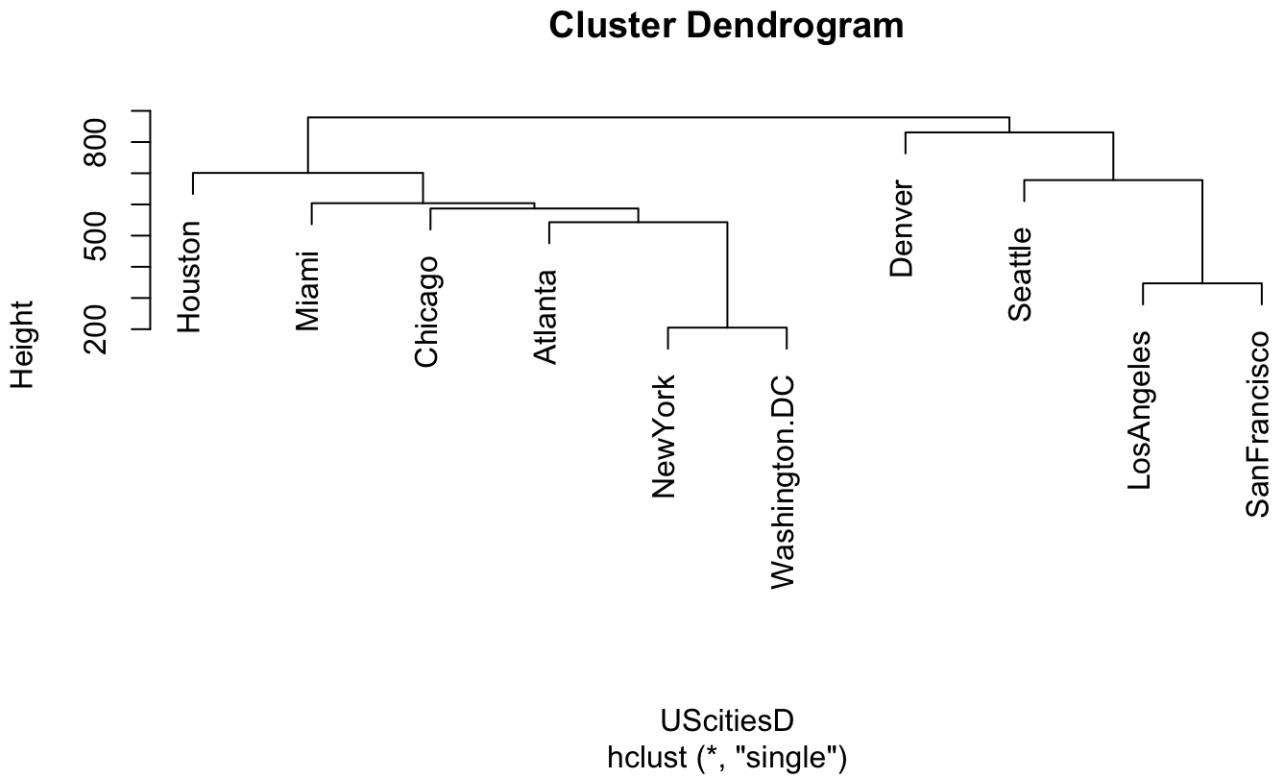
- Cómo se agregan más puntos a ese clúster inicial puede depender del método: distancia mínima, máxima, promedio, etc.
- Cada uno de estos métodos puede mostrar distintos aspectos de la estructura de los datos.
- Usar la distancia al punto mínimo genera dendogramas tipo “serpiente”.
- Usar la distancia al punto máximo genera grupos más chicos y densos.
- Usar el promedio es un compromiso entre estos dos puntos.
- El método de Ward intenta usar distancia mínima, pero que no queden grupos demasiado pequeños.

Clustering: `hclust`

- Un aspecto que hace a este algoritmo interesante es que las soluciones con muchos clústers están incluidas (anidadas) dentro de las soluciones con menos grupos.
- Esto hace que las observaciones no salten de un grupo a otro como lo hacen en `kmeans` o `pam`.
- Tampoco es necesario especificar el número de grupos.

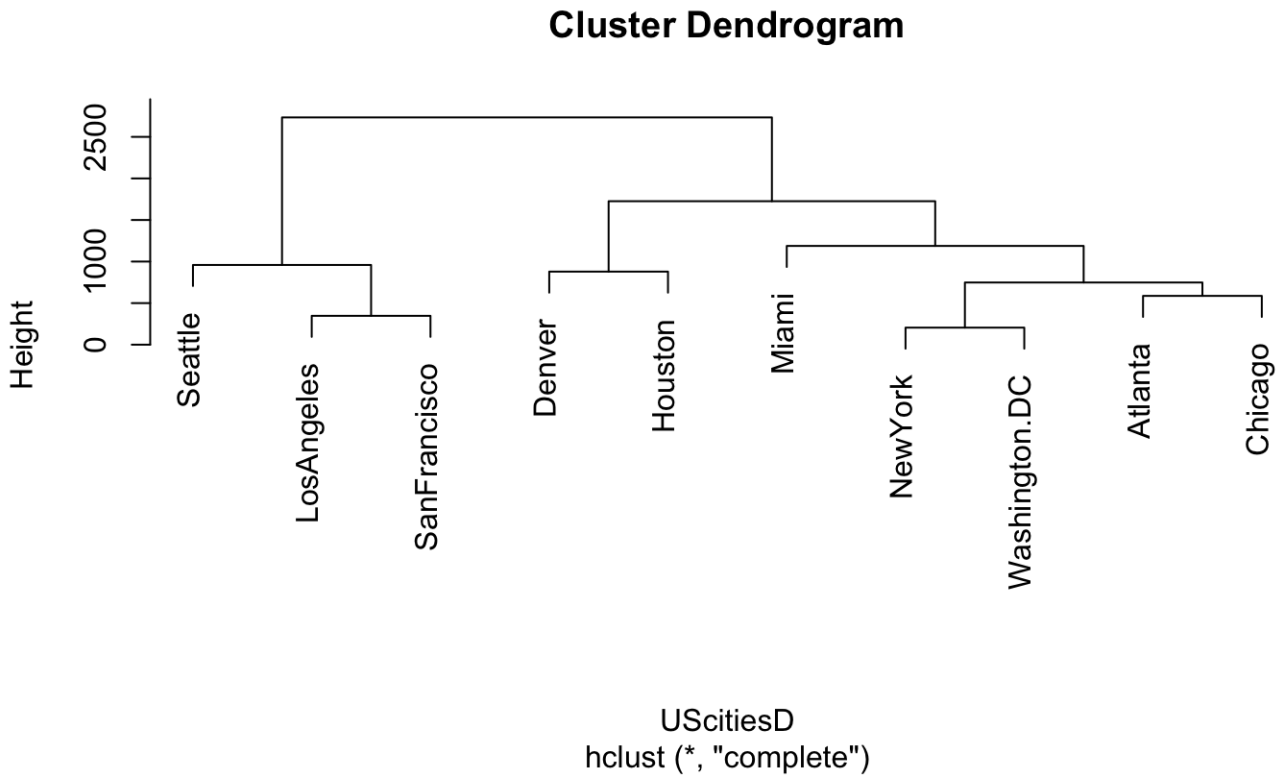
Clustering: `hclust`

```
uscities.single.link <- hclust(UScitiesD, method = "single")
plot(uscities.single.link)
```



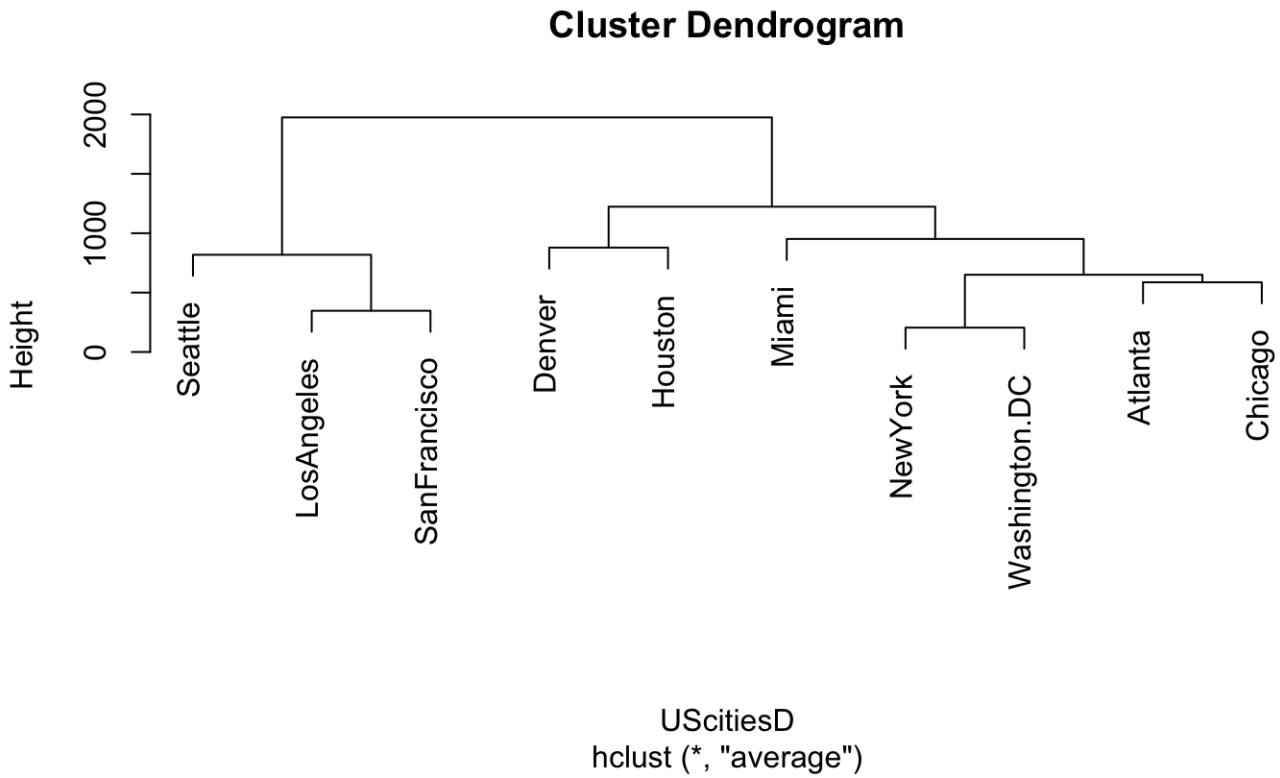
Clustering: **hclust**

```
uscities.complete.link <- hclust(UScitiesD, method = "complete")
plot(uscities.complete.link)
```



Clustering: **hclust**

```
uscities.average.link <- hclust(UScitiesD, method = "average")
plot(uscities.average.link)
```



Licencia CC BY