#CampusVivo

Universidad® de Medellín

Ciencia y Libertad

Vigilada MinEducación

RDay:

Introducción a Ciencia de datos

Grupo de Investigación en Ingeniería Financiera (GINIF)

Nini Johana Marín-Rodríguez

Profesora

Fuente: https://uc-r.github.io/kmeans_clustering

Aprendizaje no supervisado

Introducción

En el aprendizaje no supervisado (UML), no se proporcionan etiquetas y el algoritmo de aprendizaje se centra únicamente en detectar la estructura en los datos de entrada no etiquetados. En general, se distingue entre

- Agrupación, cuyo objetivo es encontrar subgrupos homogéneos dentro de los datos; la agrupación se basa en la distancia entre las observaciones.
- Reducción de la dimensionalidad, cuyo objetivo es identificar patrones en las características de los datos. La reducción de la dimensionalidad se utiliza a menudo para facilitar la visualización de los datos, así como un método de preprocesamiento antes del aprendizaje supervisado.

Clustering

- La agrupación es un amplio conjunto de técnicas para encontrar subgrupos de observaciones dentro de un conjunto de datos.
- Cuando agrupamos observaciones, queremos que las observaciones del mismo grupo sean similares y que las observaciones de grupos diferentes sean disímiles.
- Al no haber una variable de respuesta, se trata de un método no supervisado, lo que implica que trata de encontrar relaciones entre las n observaciones sin estar entrenado por una variable de respuesta.

Clustering

- El clustering nos permite identificar qué observaciones son similares y, potencialmente, clasificarlas en una categoría. El clustering de K-means es el método de clustering más sencillo y el más utilizado para dividir un conjunto de datos en un conjunto de k grupos.

Clustering

Para aplicar el método de clustering k-means se requiere lo siguiente:

1. Paquetes requeridos
2. Preparación de datos: Preparación de nuestros datos para el análisis de cluster
3. Medidas de Distancia de Clustering: Entender cómo medir las diferencias en las observaciones
4. K-Means Clustering: Cálculos y métodos para crear K subgrupos de los datos
5. Determinación de clusters óptimos: Identificación del número correcto de conglomerados para agrupar los datos

1. Paquetes requeridos

- library(tidyverse) # data manipulation
- library(cluster)   # clustering algorithms
- library(factoextra) # clustering algorithms & visualization

2. Preparación de datos

Para realizar un análisis de conglomerados en R, por lo general, los datos deben prepararse de la siguiente manera:

- Las filas son observaciones (individuos) y las columnas son variables
- Cualquier valor que falte en los datos debe eliminarse o estimarse.
- Los datos deben estandarizarse (es decir, escalarse) para que las variables sean comparables. Recordemos que la normalización consiste en transformar las variables de forma que tengan media cero y desviación típica uno.
- En este caso, utilizaremos el conjunto de datos incorporado en R USArrests, que contiene estadísticas en arrestos por cada 100.000 residentes por agresión, asesinato y violación en cada uno de los 50 estados de EE.UU. en 1973. Incluye también el porcentaje de la población que vive en zonas urbanas.

```
df <- USArrests
```

2. Preparación de datos

- Para eliminar cualquier valor omitido que pudiera estar presente en los datos, escriba esto

```
df <- na.omit(df)
```

- Como no queremos que el algoritmo de agrupación dependa de una unidad de variable arbitraria, empezamos por escalar/estandarizar los datos utilizando la función de R scale:

```
df <- scale(df)
head(df)
```

3. Medidas de Distancia de Clustering

- La clasificación de observaciones en grupos requiere algunos métodos para calcular la distancia o la (des)similitud entre cada par de observaciones. El resultado de este cálculo se conoce como matriz de disimilitud o distancia.
- Existen muchos métodos para calcular esta información de distancia; la elección de las medidas de distancia es un paso crítico en la agrupación.
  - Define cómo se calcula la similitud de dos elementos (x, y) e influirá en la forma de los conglomerados.
  - La elección de las medidas de distancia es un paso crítico en la agrupación. Define cómo se calcula la similitud de dos elementos (x, y) e influirá en la forma de los conglomerados.
  - Los métodos clásicos para las medidas de distancia son las distancias euclídea y de Manhattan.

3. Medidas de Distancia de Clustering

- Distancia euclídea

$$d_{euc}(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \ (1)$$

- Distancia de Manhattan

$$d_{man}(x,y) = \sum_{i=1}^{n} |(x_i - y_i)| \ (2)$$

Donde, x e y son dos vectores de longitud n.

3. Medidas de Distancia de Clustering

Existen otras medidas de disimilitud, como las distancias basadas en la correlación, muy utilizadas para el análisis de datos de expresión génica. La distancia basada en la correlación se define restando el coeficiente de correlación de 1. Se pueden utilizar diferentes tipos de métodos de correlación como:

- **Distancia de correlación de Pearson:**

$$d_{cor}(x,y) = 1 - \frac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\overline{x})^2 \sum_{i=1}^{n}(y_i-\overline{y})^2}} \quad (3)$$

- **Distancia de correlación de Spearman:**

$$d_{spear}(x,y) = 1 - \frac{\sum_{i=1}^{n}(x_i'-\overline{x}')(y_i'-\overline{y}')}{\sqrt{\sum_{i=1}^{n}(x_i'-\overline{x}')^2 \sum_{i=1}^{n}(y_i'-\overline{y}')^2}} \quad (4)$$

3. Medidas de Distancia de Clustering

- **Distancia de correlación de Kendall:**

El método de correlación de Kendall mide la correspondencia entre la ordenación de las variables x e y. El número total de emparejamientos posibles de x con observaciones de y es n(n – 1)/2, donde n es el tamaño de x e y.

Empiece ordenando los pares por los valores de x. Si x e y están correlacionadas, tendrán el mismo orden relativo. Ahora, para cada y_i cuente el número de pares (pares concordantes (c)) $y_j > y_i$ y el número de (pares discordantes (d)) $y_j < y_i$.

La distancia de correlación de Kendall se define como sigue:

$$d_{kend}(x, y) = 1 - \frac{n_c - n_d}{\frac{1}{2}n(n-1)} \quad (5)$$

3. Medidas de Distancia de Clustering

- La elección de las medidas de distancia es muy importante, ya que influye mucho en los resultados de la agrupación. En los programas de clustering más comunes, la medida de distancia por defecto es la distancia euclídea. Sin embargo, dependiendo del tipo de datos y de las preguntas de la investigación, pueden preferirse otras medidas de disimilitud, por lo que debe conocer las opciones.
- En R es sencillo calcular y visualizar la matriz de distancias utilizando las funciones get_dist y fviz_dist del paquete factoextra de R. De esta forma se empieza a ilustrar qué estados tienen una mayor distancia entre sí. Esto empieza a ilustrar qué estados tienen grandes disimilitudes (rojo) frente a los que parecen bastante similares (verde azulado).
- get_dist: para calcular una matriz de distancias entre las filas de una matriz de datos. La distancia calculada por defecto es la euclídea; sin embargo, get_dist también admite las distancias descritas en las ecuaciones 2-5 y otras
- .fviz_dist: para visualizar una matriz de distancias.

```
distance <- get_dist(df)
fviz_dist(distance, gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07"))
```

Agrupación k-means

El clustering de K-Means es el algoritmo de aprendizaje automático no supervisado más utilizado para dividir un conjunto de datos dado en un conjunto de k grupos (es decir, k clústeres), donde k representa el número de grupos especificado previamente por el analista. Clasifica los objetos en varios grupos (es decir, clusters), de forma que los objetos de un mismo clúster sean lo más parecidos posible (es decir, tengan una similitud intraclase alta), mientras que los objetos de clusters diferentes sean lo más distintos posible (es decir, tengan una similitud interclase baja).

En el clustering de k-means, cada clúster está representado por su centro (es decir, centroide), que corresponde a la media de los puntos asignados al clúster.

Agrupación k-means

La idea básica de la agrupación k-means consiste en definir clústeres de modo que se minimice la variación total intraclúster (conocida como variación total dentro del clúster). Existen varios algoritmos k-means. El algoritmo estándar es el de Hartigan-Wong (1979), que define la variación total intracluster como la suma de las distancias euclídeas al cuadrado entre los elementos y el centroide correspondiente:

$$W(C_k) = \sum_{x_i \in C_k} (x_i - \mu_k)^2 \ (6)$$

Donde:

x_i es un punto de datos perteneciente al conglomerado C_k

μ_k es el valor medio de los puntos asignados al clúster C_k

Cada observación (x_i) se asigna a un cluster determinado tal que la distancia suma de cuadrados (SS) de la observación a sus centros de cluster asignados (μ_k) son minimizados.

Agrupación k-means

Definimos la variación total dentro del conglomerado de la siguiente manera:

$$tot.withiness = \sum_{k=1}^{k} W(C_k) = \sum_{k=1}^{k} \sum_{x_i \in C_k} (x_i - \mu_k)^2 \quad (7)$$

La suma cuadrática total dentro del clúster mide la compacidad (es decir, la bondad) de la agrupación y queremos que sea lo más pequeña posible.

Algoritmo K-means

El algoritmo K-means puede resumirse del siguiente modo:

1. Especificar el número de clusters (K) a crear (por el analista).
2. Selecciona aleatoriamente k objetos del conjunto de datos como centros o medias iniciales de los clusters.
3. Asigna cada observación a su centroide más cercano, basándose en la distancia euclídea entre el objeto y el centroide.
4. Para cada uno de los k clusters actualiza el centroide del cluster calculando los nuevos valores medios de todos los puntos de datos del cluster. El centroide de un k-ésimo cluster es un vector de longitud p que contiene las medias de todas las variables para las observaciones en el k-ésimo cluster; p es el número de variables.
5. Iterativamente minimizar el total dentro de la suma de cuadrados (Ec. 7). Es decir, itere los pasos 3 y 4 hasta que las asignaciones de conglomerados dejen de cambiar o se alcance el número máximo de iteraciones. Por defecto, el software R utiliza 10 como valor por defecto para el número máximo de iteraciones.

Cálculo de k-means en R

- Podemos calcular k-means en R con la función kmeans. Aquí agruparemos los datos en dos clusters (centros = 2). La función kmeans también tiene una opción nstart que intenta múltiples configuraciones iniciales e informa sobre la mejor.
- Por ejemplo, añadir nstart = 25 generará 25 configuraciones iniciales. Este método se recomienda a menudo.

```
k2 <- kmeans(df, centers = 2, nstart = 25)
str(k2)
```

Resultado de k-means

- El resultado de kmeans es una lista con varios datos. La más importante es:
  - cluster: Un vector de enteros (de 1:k) que indica el cluster al que se asigna cada punto.
  - centers: Una matriz de centros de cluster.
  - totss: La suma total de cuadrados.
  - withinss: Vector de la suma de cuadrados dentro del conglomerado, un componente por conglomerado.
  - tot.withinss: Suma total de cuadrados dentro del conglomerado, es decir, sum(withinss).
  - betweenss: La suma de cuadrados entre conglomerados, es decir, $totss-tot.withinss$.
  - size: El número de puntos de cada conglomerado.

Resultado de k-means

- También podemos ver nuestros resultados utilizando fviz_cluster. Esto proporciona una buena ilustración de los clusters. Si hay más de dos dimensiones (variables) fviz_cluster realizará un análisis de componentes principales (ACP) y trazará los puntos de datos de acuerdo con los dos primeros componentes principales que explican la mayor parte de la varianza.

```
fviz_cluster(k2, data = df)
```

- También puede utilizar gráficos de dispersión por pares estándar para ilustrar los conglomerados en comparación con las variables originales.

```
df %>%
  as_tibble() %>%
  mutate(cluster = k2$cluster,
         state = row.names(USArrests)) %>%
  ggplot(aes(UrbanPop, Murder, color = factor(cluster), label = state)) +
  geom_text()
```

Resultado de k-means

- Dado que el número de conglomerados (k) debe fijarse antes de iniciar el algoritmo, a menudo resulta ventajoso utilizar varios valores diferentes de k y examinar las diferencias en los resultados. Podemos ejecutar el mismo proceso para 3, 4 y 5 conglomerados, y los resultados se muestran en la figura:

```
k3 <- kmeans(df, centers = 3, nstart = 25)
k4 <- kmeans(df, centers = 4, nstart = 25)
k5 <- kmeans(df, centers = 5, nstart = 25)

# plots to compare
p1 <- fviz_cluster(k2, geom = "point", data = df) + ggtitle("k = 2")
p2 <- fviz_cluster(k3, geom = "point",  data = df) + ggtitle("k = 3")
p3 <- fviz_cluster(k4, geom = "point",  data = df) + ggtitle("k = 4")
p4 <- fviz_cluster(k5, geom = "point",  data = df) + ggtitle("k = 5")

library(gridExtra)
grid.arrange(p1, p2, p3, p4, nrow = 2)
```

Determinar los clústeres óptimos

- Aunque esta evaluación visual nos indica dónde se producen verdaderas dilataciones (o no se producen, como en los conglomerados 2 y 4 del gráfico k = 5) entre los conglomerados, no nos dice cuál es el número óptimo de conglomerados.
- Como recordará, el analista especifica el número de conglomerados que debe utilizar; preferiblemente, el analista desearía utilizar el número óptimo de conglomerados.
- Para ayudar al analista, a continuación, se explican los tres métodos más populares para determinar los conglomerados óptimos, que incluyen:
  1. Método del codo (Elbow method)
  2. Método de la silueta (Silhouette method)
  3. Método Estadístico de Gap (Gap statistic method)

Determinar los clústeres óptimos

1. Método del codo (Elbow method)

Recordemos que la idea básica de los métodos de partición de clusters, como el clustering k-means, es definir clusters de forma que se minimice la variación total intra-cluster (conocida como variación total intra-cluster o suma cuadrática total intra-cluster):

$$minimize(\sum_{k=1}^{k} W(C_k)) \ (8)$$

donde C_k es el k^{th} y $W(C_k)$ es la variación dentro del cluster. La suma cuadrada total dentro del cluster (wss) mide la compacidad de la agrupación y queremos que sea lo más pequeña posible.

Determinar los clústeres óptimos

Así, podemos utilizar el siguiente algoritmo para definir los conglomerados óptimos:

1. Calcule el algoritmo de clustering (por ejemplo, clustering k-means) para diferentes valores de k. Por ejemplo, variando k de 1 a 10 clusters.
2. Para cada k, calcule la suma cuadrática total dentro del conglomerado (wss)Trace la curva de wss en función del número de conglomerados k.
3. La ubicación de una curva (rodilla) en el gráfico se considera generalmente como un indicador del número apropiado de conglomerados.

Determinar los clústeres óptimos

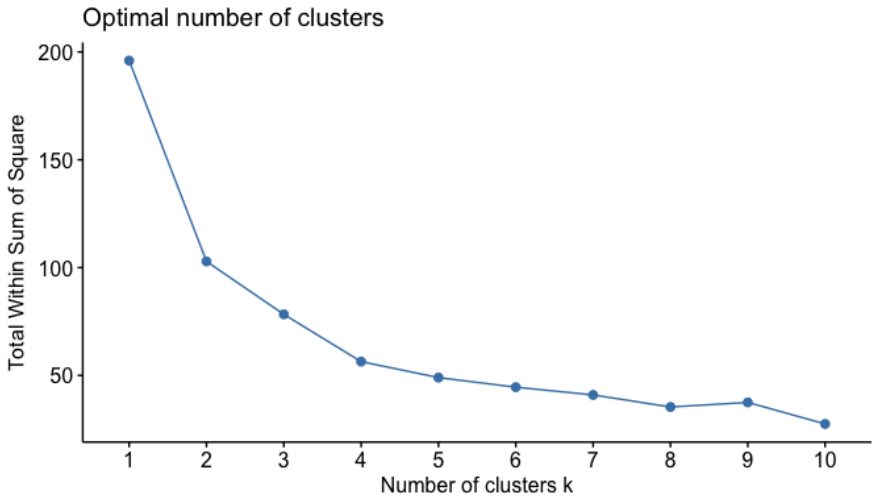
- Podemos implementar esto en R con el siguiente código. Los resultados sugieren que 4 es el número óptimo de conglomerados, ya que parece ser la curva de la rodilla (o codo).

```
set.seed(123)

# function to compute total within-cluster sum of square
wss <- function(k) {
  kmeans(df, k, nstart = 10 )$tot.withinss
}

# Compute and plot wss for k = 1 to k = 15
k.values <- 1:15

# extract wss for 2-15 clusters
wss_values <- map_dbl(k.values, wss)

plot(k.values, wss_values,
     type="b", pch = 19, frame = FALSE,
     xlab="Number of clusters K",
     ylab="Total within-clusters sum of squares")
```

Determinar los clústeres óptimos

- Afortunadamente, este proceso para calcular el "método del codo" se ha englobado en una única función (fviz_nbclust):

```
set.seed(123)

fviz_nbclust(df, kmeans, method = "wss")
```



Determinar los clústeres óptimos

2. Método de la silueta (Silhouette method)

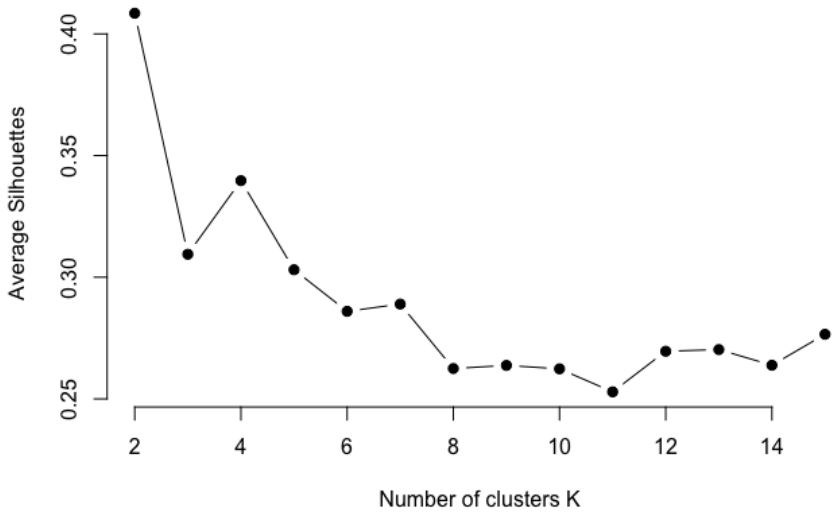
el enfoque de la silueta media mide la calidad de una agrupación. Es decir, determina hasta qué punto cada objeto se encuentra dentro de su agrupación. Una anchura de silueta media elevada indica una buena agrupación.

El método de la silueta media calcula la silueta media de las observaciones para distintos valores de k. El número óptimo de conglomerados k es el que maximiza la silueta media en un rango de posibles valores de k.

Determinar los clústeres óptimos

- Podemos utilizar la función silhouette del paquete cluster para calcular la anchura de la silueta media. El código siguiente calcula esta aproximación para 1-15 conglomerados. Los resultados muestran que 2 conglomerados maximizan los valores medios de la silueta y que 4 conglomerados son el segundo número óptimo de conglomerados.

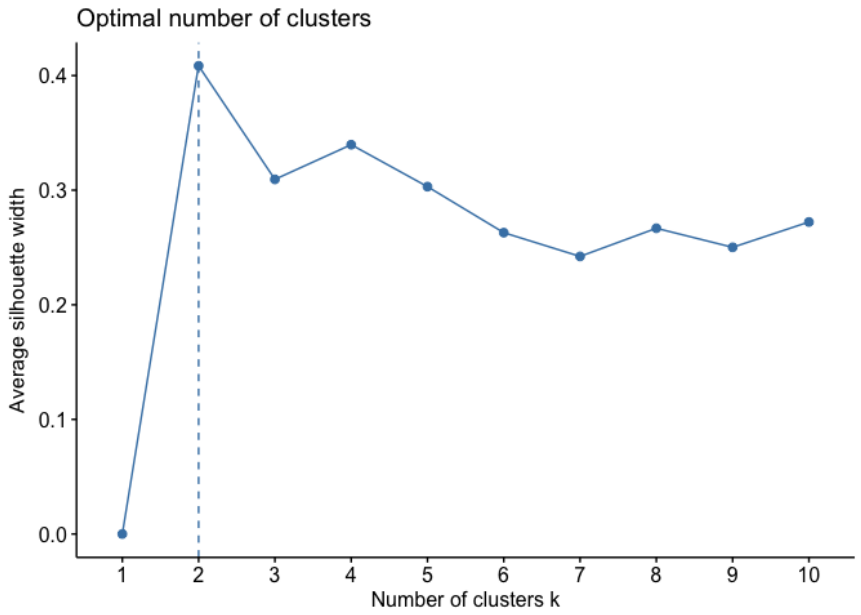
```
# function to compute average silhouette for k clusters
avg_sil <- function(k) {
  km.res <- kmeans(df, centers = k, nstart = 25)
  ss <- silhouette(km.res$cluster, dist(df))
  mean(ss[, 3])
}

# Compute and plot wss for k = 2 to k = 15
k.values <- 2:15

# extract avg silhouette for 2-15 clusters
avg_sil_values <- map_dbl(k.values, avg_sil)

plot(k.values, avg_sil_values,
       type = "b", pch = 19, frame = FALSE,
       xlab = "Number of clusters K",
       ylab = "Average Silhouettes")
```



Determinar los clústeres óptimos

- De forma similar al método del codo, este proceso para calcular el "método del silhoutte medio" se ha englobado en una única función (fviz_nbclust):

```
fviz_nbclust(df, kmeans, method = "silhouette")
```



Determinar los clústeres óptimos

3. Método Estadístico de Gap (Gap statistic method)

- R. Tibshirani, G. Walther y T. Hastie (Universidad de Standford, 2001) han publicado el método estadístico de gap. El enfoque puede aplicarse a cualquier método de clustering (es decir, clustering K-means, clustering jerárquico).
- El estadístico de gap compara la variación intracluster total para diferentes valores de k con sus valores esperados bajo una distribución de referencia nula de los datos (es decir, una distribución sin agrupación evidente).
- El conjunto de datos de referencia se genera mediante simulaciones Monte Carlo del proceso de muestreo. Es decir, para cada variable (xi) del conjunto de datos calculamos su intervalo $[min(x_i), max(x_j)]$ y generar valores para los n puntos uniformemente a partir del intervalo min a max.

Determinar los clústeres óptimos

- Para los datos observados y los datos de referencia, se calcula la variación intracluster total utilizando diferentes valores de k. El estadístico de brecha para un k dado se define como sigue:

$$Gap_n(k) = E_n^* log(W_k) - log(W_k)\ (9)$$

- Donde E_n^* denota la expectativa bajo una muestra de tamaño n de la distribución de referencia. E_n^* se define mediante bootstrapping (B) generando B copias de los conjuntos de datos de referencia y, calculando la media $log(W_k^*)$.
- El estadístico de la brecha mide la desviación de los valores observados de W_k observado con respecto a su valor esperado bajo la hipótesis nula. La estimación de los conglomerados óptimos ($\hat{k}$) será el valor que maximice $Gap_n(k)$. Esto significa que la estructura de agrupación se aleja de la distribución uniforme de puntos.

Determinar los clústeres óptimos

En resumen, el algoritmo estadístico de Gap consta de los siguientes pasos:

1. Agrupar los datos observados, variando el número de conglomerados de $k = 1, \dots, k_{max}$ y calcular la correspondiente W_k.
2. Generar B conjuntos de datos de referencia y agrupar cada uno de ellos con un número variable de conglomerados $k = 1, \dots, k_{max}$
3. Calcule las estadísticas de brecha estimadas que se presentan en la ec. 9.
4. Sea $\overline{w} = (1/B) \sum_b log(W_{kb}^*)$ calcula la desviación típica $sd(k) = \sqrt{(1/b) \sum_b (log(W_{kb}^*) - \overline{w})^2}$ y definimos $s_k = sd_k \times \sqrt{1 + 1/B}$.
5. Elija el número de conglomerados como el k más pequeño tal que $Gap(k) \geq Gap(k+1) - s_{k+1}$.

Determinar los clústeres óptimos

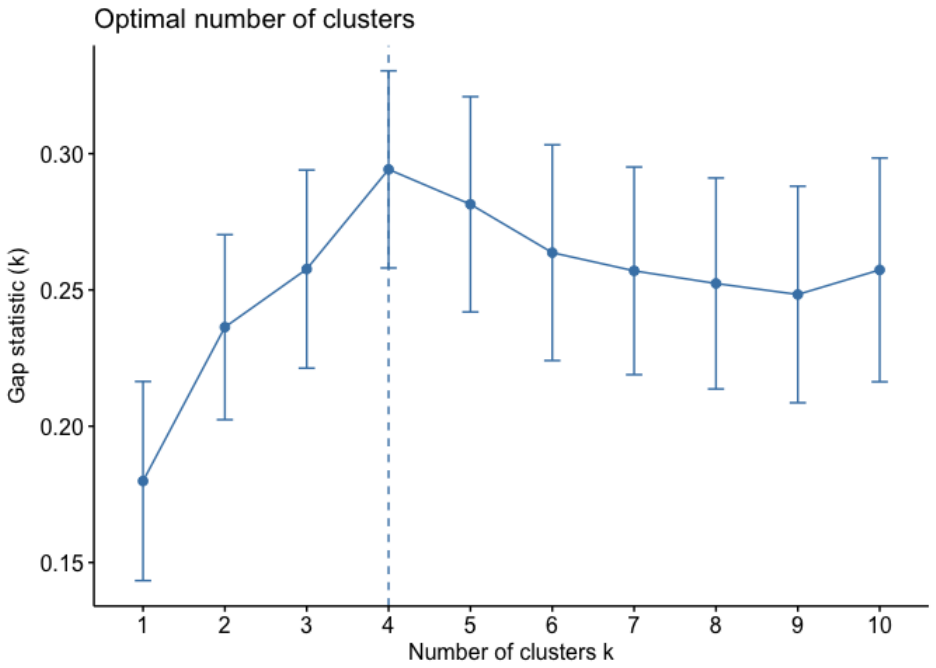
- Para calcular el método estadístico de la brecha podemos utilizar la función clusGap que proporciona el estadístico de la brecha y el error estándar para una salida.

```
# compute gap statistic
set.seed(123)
gap_stat <- clusGap(df, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
# Print the result
print(gap_stat, method = "firstmax")
```

Determinar los clústeres óptimos

- Podemos visualizar los resultados con fviz_gap_stat, que sugiere cuatro conglomerados como número óptimo de conglomerados.

```
fviz_gap_stat(gap_stat)
```



Determinar los clústeres óptimos

- Además de estos enfoques comúnmente utilizados, el paquete NbClust, publicado por Charrad et al., 2014, proporciona 30 índices para determinar el número relevante de clústeres y propone a los usuarios el mejor esquema de agrupación a partir de los diferentes resultados obtenidos variando todas las combinaciones de número de clústeres, medidas de distancia y métodos de agrupación.

Extracción de resultados

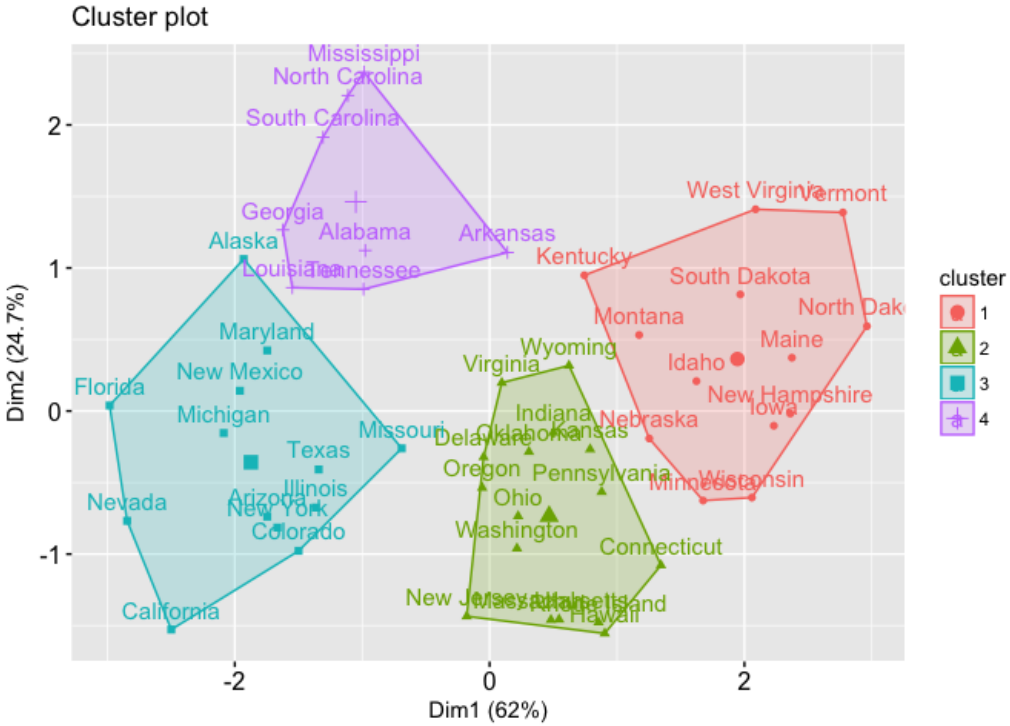
- Dado que la mayoría de estos enfoques sugieren 4 como número de clusters óptimos, podemos realizar el análisis final y extraer los resultados utilizando 4 clusters.

```
# Compute k-means clustering with k = 4
set.seed(123)
final <- kmeans(df, 4, nstart = 25)
print(final)
```

Determinar los clústeres óptimos

- Podemos visualizar los resultados utilizando fviz_cluster:

```
fviz_cluster(final, data = df)
```



Determinar los clústeres óptimos

- Y podemos extraer los conglomerados y añadirlos a nuestros datos iniciales para hacer algunas estadísticas descriptivas a nivel de conglomerado:

```
USArrests %>%
  mutate(Cluster = final$cluster) %>%
  group_by(Cluster) %>%
  summarise_all("mean")
```

Comentarios adicionales

- El clustering de K-means es un algoritmo muy sencillo y rápido. Además, puede tratar eficazmente conjuntos de datos muy grandes. Sin embargo, el método k-means presenta algunos puntos débiles.

- Una desventaja potencial del clustering de k-means es que requiere que especifiquemos previamente el número de clusters. El clustering jerárquico es un enfoque alternativo que no requiere que nos comprometamos con una elección concreta de clusters. El clustering jerárquico tiene una ventaja añadida sobre el clustering K-means y es que da como resultado una atractiva representación de las observaciones basada en un árbol, llamado dendrograma. Un tutorial futuro ilustrará el enfoque de clustering jerárquico.

Comentarios adicionales

- Una desventaja adicional de K-means es que es sensible a los valores atípicos y pueden producirse resultados diferentes si se cambia el orden de los datos. El enfoque de clustering Partitioning Around Medoids (PAM) es menos sensible a los valores atípicos y proporciona una alternativa robusta a k-means para hacer frente a estas situaciones. Un tutorial futuro ilustrará el enfoque de agrupación PAM.

¿Preguntas?
¿Dudas?
¿Sugerencias?

GRACIAS

#CampusVivo

Universidad® de Medellín
Ciencia y Libertad

CAMPUS VIVO
NUESTRA UNIVERSIDAD SOSTENIBLE