INFOX
educación sin fronteras

Clase 4: Unsupervised Learning

Mg. Gloria Rivas

Agenda

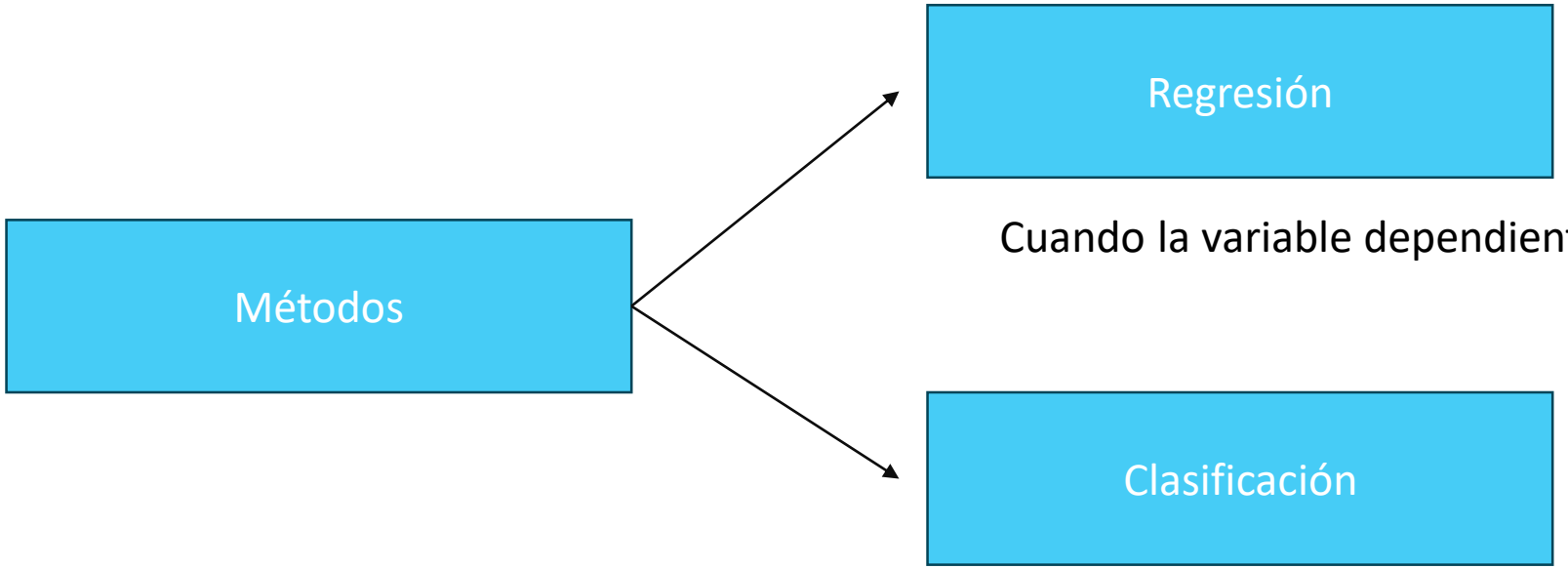
1. **Introducción**
2. **Principal Component Analysis**
3. **Clustering**

Agenda

1. Introducción

INFOX
educación sin fronteras

Supervised Learning



Cuando la variable dependiente es continua.

Cuando la variable dependiente es categórica.

Unsupervised Learning

- No estamos interesados en la predicción de alguna variable, mas bien nuestro objetivo es descubrir patrones en la data.
- Estamos interesado en responder preguntas como: hay una forma informativa de visualizar la data? Podemos descubrir subgrupos entre las variables o entre las observaciones?
- En este capítulo aprenderemos técnicas para responder este tipo de preguntas
- En este capítulo nos centraremos en 2 técnicas en particular: Principal Component Analysis (PCA) y Clustering.
- Cabe mencionar que este tema es mucho más retador que supervised Learning porque aquí no hay herramientas para contrastar el output.

Unsupervised Learning

- Unsupervised Learning actúa muchas veces como parte de la exploración de análisis de datos.
- Además puede ser difícil de evaluar los resultados, la razón es simple: no se sabe la respuesta verdadera.
- Mientras que en supervised Learning es mucho más sencillo saber si lo que estamos haciendo va por buen o mal camino dependiendo del fit de nuestras predicciones.
- Las técnicas de Unsupervised Learning están creciendo en diferentes campos: medicina, marketing, economía, entre otros.

Agenda

2. Principal Component Analysis

2. Principal Component Analysis

2.1. Qué son Componentes Principales?

- Supongamos que como parte de un análisis exploratorio queremos visualizar n observaciones y tenemos p variables $X_1, X_2, \ldots, X_p$

- Podríamos realizar este análisis usando un scatterplot de 2 dimensiones. Sin embargo tendríamos $\binom{p}{2} = p(p-1)/2$ scatterplots. Por ejemplo, si tenemos 10 variables, tendríamos 45 gráficos!

- Si p es muy grande, entonces será imposible de analizar todos los gráficos. En sí, ninguno de ellos será informativo porque solo contendrá una fracción de todo el Dataset.

- Claramente, necesitamos un método más eficiente para visualizar las n observaciones cuando p es muy grande.

- En particular, nos gustaría encontrar una forma de representar todo el Dataset con el menor número de dimensiones.

2. Principal Component Analysis

2.1. Qué son Componentes Principales?

- Por ejemplo, si obtenemos 2 dimensiones para representar la data y que capture la mayor parte de la información, entonces podemos graficar nuestras observaciones en este plano.

- Principal Component Analysis (PCA) nos da una herramienta para generar esto que buscamos.

- Encuentra el menor número de dimensiones para representar el Dataset que contiene la mayor variación.

- La idea es que cada una de las n observaciones vive en una dimensión p, pero no todas las dimensiones son igual de interesantes. Por eso PCA busca el menor número de dimensiones que sean lo más interesante posibles.

* Como interesante se mide la cantidad de observaciones que varía por dimensión.

2. Principal Component Analysis

2.1. Qué son Componentes Principales?

- Cada dimensión encontrada por el método PCA es una combinación lineal las p variables que tenemos.

- A continuación, explicamos la manera en que estas dimensiones, o componentes principales son encontradas.

- El primer componente principal de un set de variables $X_1, X_2, \ldots, X_p$ es normalizado como una combinación lineal de variables que contenga la mayor varianza.

$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \ldots + \phi_{p1} X_p \qquad (1)$$

- Por normalizado nos referimos a $\sum_{j=1}^{p} \phi_{j1}^2 = 1$. Nos referimos a estos elementos $\phi_{11}, \ldots, \phi_{p1}$ como los ´loadings´ del primer componente.

- Estos ´loadings´ forman el vector del primer componente $\phi_1 = (\phi_{11} \ \phi_{21} \ \ldots \ \phi_{p1})^T$. Construimos estos ´loadings´ tal que la suma del vector sea igual a 1, dado que de otra manera estos elementos que pueden ser arbitrariamente grandes en valor absoluto podrían afectar la varianza total.

2. Principal Component Analysis

2.1. Qué son Componentes Principales?

- Dado un Dataset **X** con dimensiones **n x p**, ¿cómo computamos el primer componente principal?
- Dado que solamente estamos enfocados en la varianza, vamos a asumir que todas las variables tienen una media igual a cero. Luego, revisamos nuestra combinación lineal sobre estas variables:

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \ldots + \phi_{p1} x_{ip} \qquad (2)$$

- Y vemos cual de ellas tiene la mayor varianza sujeto a la restricción $\sum_{j=1}^{p} \phi_{j1}^2 = 1$. En otras palabras, el vector del primer componente principal resuelve este problema de optimización:

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^2 \right\} \text{ subject to } \sum_{j=1}^{p} \phi_{j1}^2 = 1 \qquad (3)$$

- De (2) podemos escribir el objetivo en (3) co $\frac{1}{n} \sum_{i=1}^{n} z_{i1}^2$ $\frac{1}{n} \sum_{i=1}^{n} x_{ij} = 0$, el promedio de $z_{11}, \ldots, z_{n1}$ será cero también.

INFOX
educación sin fronteras

2. Principal Component Analysis

2.1. Qué son Componentes Principales?

- Nuestro objetivo es maximizar (3) sobre la varianza de los n valores de z_{i1}.
- Nos referimos a $z_{11}, \ldots, z_{n1}$ como los scores del primer componente principal. El problema (3) se puede resolver por descomposición de valores propios (algebra lineal) pero detalles están fuera del objetivo de este curso.
- El vector loading ϕ_1 con componente $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ define la dirección en el espacio donde la data varía mas. Si proyectamos las n observaciones $x_1, \ldots, x_n$ en esa dirección, los valores proyectados son los componentes principales $z_{11}, \ldots, z_{n1}$.
- Después que hemos calculado el primer componente principal Z_1 , podemos calcular el segundo componente principal Z_2. El segundo componente principal es la combinación linear de $X_1, \ldots, X_p$ que tenga la máxima varianza no correlacionadas con el vector Z_1 .
- Los scores del segundo componente principal toman la siguiente forma:

$$z_{i2} = \phi_{12} x_{i1} + \phi_{22} x_{i2} + \ldots + \phi_{p2} x_{ip}$$

2. Principal Component Analysis

2.1. Qué son Componentes Principales?

- Donde ϕ_2 es el vector loadings del segundo componente principal con elementos $\phi_{12}, \phi_{22}, \ldots, \phi_{p2}$
- Para calcular ϕ_2 resolvemos un problema similar al presentado en la ecuación (3) con ϕ_2 remplazando a ϕ_1, con una adicional restricción, que ϕ_2 sea ortogonal a ϕ_1.
- Una vez que tenemos computados los componentes principales, podemos graficarlos (los vectores) para producir un gráfico el menor número de dimensiones.
- Por ejemplo, podríamos graficar el vector Z_1 sobre el vector Z_2, el vector Z_1 sobre el vector Z_3 y así sucesivamente.
- Geométricamente, las cantidades se proyectan sobre la data original explicada por los vectore ϕ_1 y ϕ_2.

2. Principal Component Analysis

2.1. Qué son Componentes Principales?

- Ilustramos el uso de PCA con el Dataset: USArrests para cada uno de los 50 estados de USA. La data contiene el número de arrestos por 100 000 residentes por los crímenes de: asalto, matanza o violación.

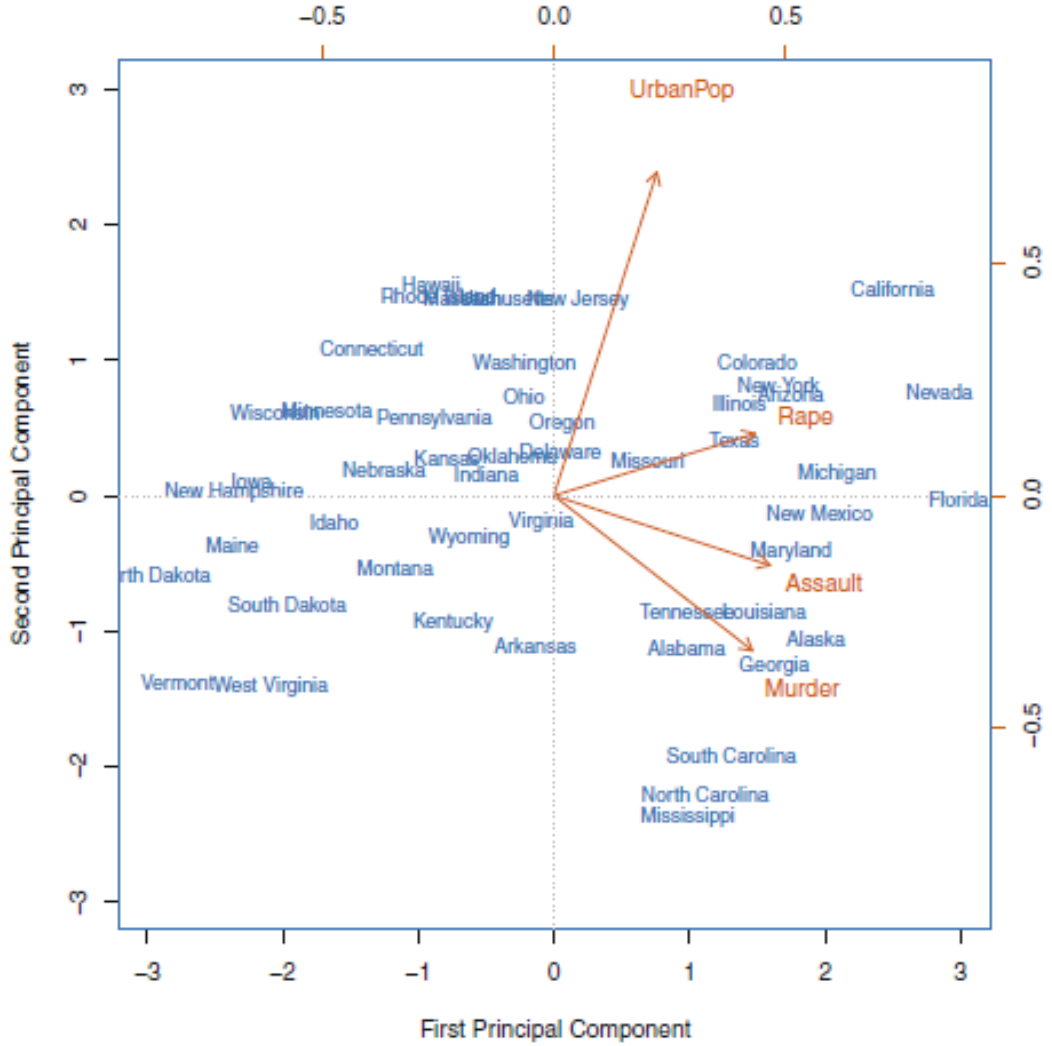
- Asimismo, tenemos la variable Urbanpop que el es porcentaje de la población por estado que vive en áreas urbanas.

- El score de componentes principales tiene una longitud de n=50 y los loading vectors son p=4. PCA es estimado después de estandarizar las variables para que tengan media 0 y desviación estándar de 1.

- En el siguiente gráfico vemos el ploteo de los 2 primeros componentes principales de esta data.

2. Principal Component Analysis

2.1. Qué son Componentes Principales?



- La figura representa el componente de scores y los loading vectors en un biplot. Se le conoce como un biplot porque muestra los scores y los loadings.

	PC1	PC2
Murder	0.5358995	−0.4181809
Assault	0.5831836	−0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

2. Principal Component Analysis
2.2. Escalando las variables

- Como ya se ha mencionado, cuando se realiza un análisis con PCA las variables deben ser centradas y tener una media 0.

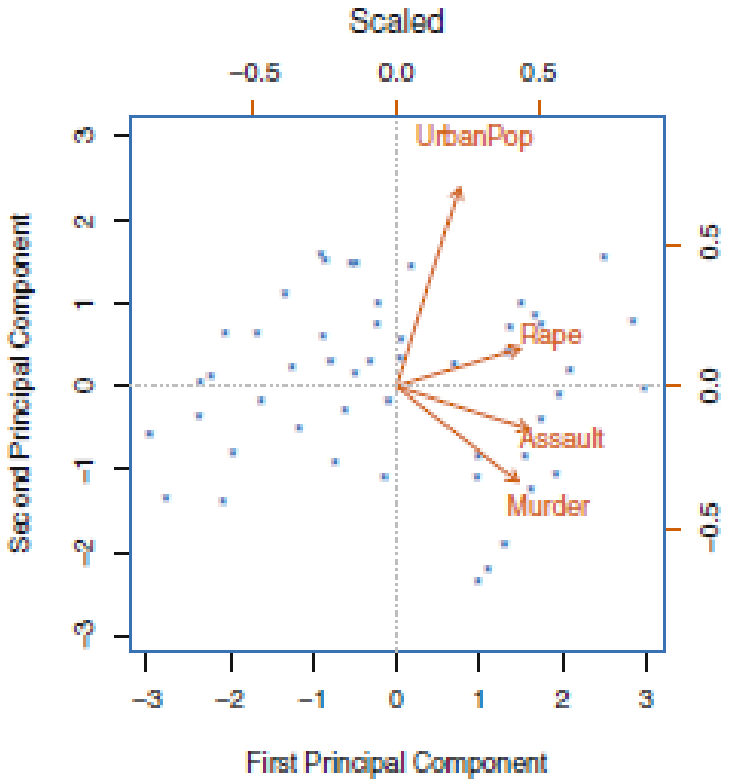
- Además los resultados obtenidos también va a depender de si las variables han sido individualmente escaladas (cada una multiplicada por una constante diferente).

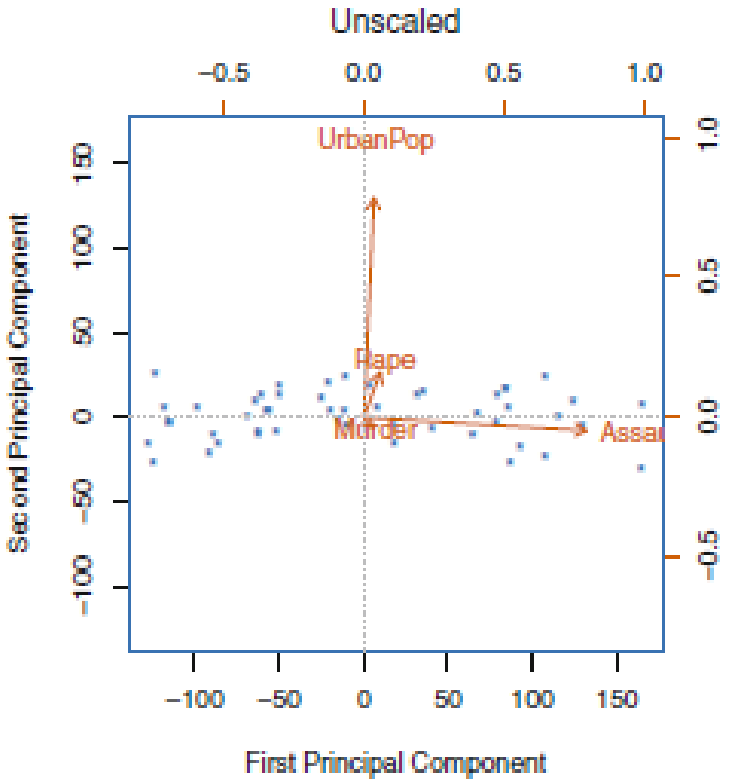
- Por ejemplo, en la primera figura se escalo cada una de las variables para tener una desviación estándar de 1.

- Qué pasaría si no escalamos?

2. Principal Component Analysis
2.2. Escalando las variables

- En este Dataset, las variables están en diferentes unidades.
- Murder, Rape and Assault está como número de ocurrencias por cada 1000 personas mientras que UrbanPop es un ratio.
- Las cuatro variables tienen una varianza de 18.97, 87.73, 6945.16 y 209.5 respectivamente.
- Consecuentemente, si calculamos un PCA para variables no escaladas, el primer vector principal será el que tiene la mayor varianza.





INFOX
educación sin fronteras

2. Principal Component Analysis
2.3. Principales componentes únicos

- Cada loading del componente principal es único, solo podría variar el signo. Esto se refiere a que dos softwares estadísticos diferentes (i.e. Python y R) nos va a dar el mismo vector de loadings de componentes principales.

- Los signos pueden diferir porque va a depender de la dimensión en la que se proyecte, pero el cambio de signo no va a afectar la dirección del espacio P.

- Asimismo, los scores de un vector son únicos (salvo puede variar el signo).

2. Principal Component Analysis
2.4. Proporción de la varianza explicada

- Cuanta data perdemos proyectando la información en pocos componentes principales?

- Esto se resuelve respondiendo cuanta varianza no estamos incluyendo en los componentes principales. En general estamos interesados en conocer la proporción de varianza explicada (PVE) por cada componente principal.

- El total de varianza de un Dataset (asumiendo que las variables han sido centradas y tienen media cero) se define como:

$$\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2 \qquad (4)$$

- Y la varianza explicada por el m componente principal es:

$$\frac{1}{n} \sum_{i=1}^{n} z_{im}^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2 \qquad (5)$$

2. Principal Component Analysis
2.4. Proporción de la varianza explicada

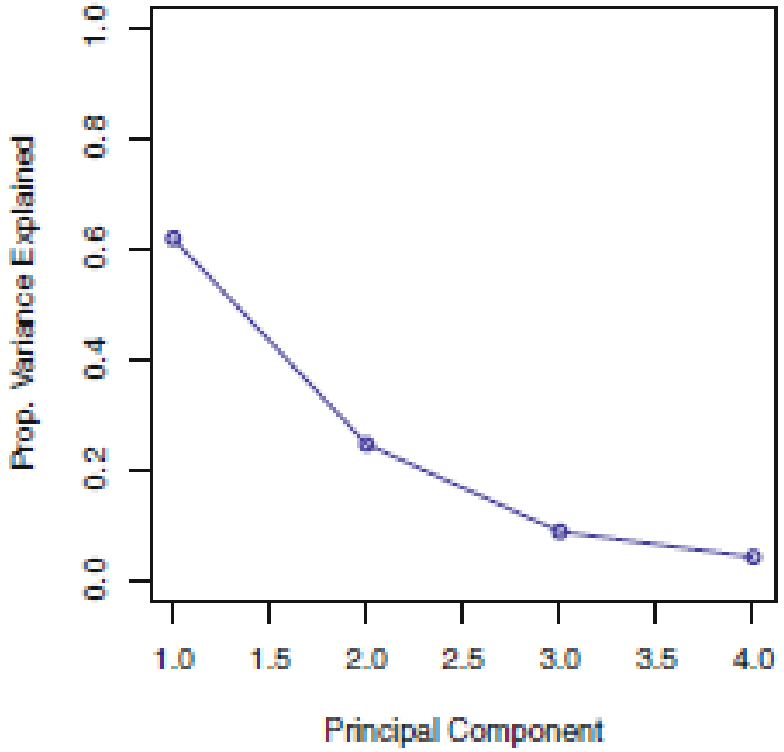
- En ese sentido, el PVE del m componente principal es dado por:

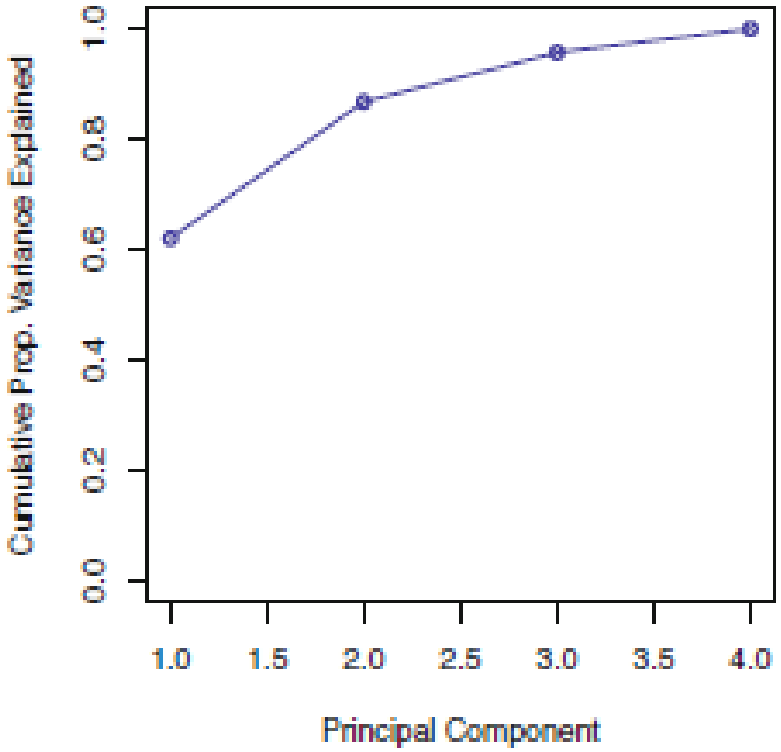
$$\frac{\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{jm}x_{ij}\right)^2}{\sum_{j=1}^{p}\sum_{i=1}^{n}x_{ij}^2} \qquad (6)$$

- El PVE de cada componente principal es una cantidad positiva. Para calcular el PVE acumulado de los primero M componentes, simplemente sumamos (6) sobre los primero M PVE.

- En la data de Usarrests, el primer componente principal explica el 62% de la varianza, y el siguiente explica el 24.7% de la varianza. En total, esos 2 componentes explican casi el 87% de la varianza, y los últimos 2 solo el 13%. Esto significa que la figura mostrada láminas anteriores es bastante precisa.

2. Principal Component Analysis
2.4. Proporción de la varianza explicada





Izquierda: Un scree plot mostrando la varianza explicada por cada componente.
Derecha: La proporción acumulada de la varianza por las cuatro variables.

2. Principal Component Analysis
2.5. Cuántos componentes usar

- En general, en una matriz X de nxp dimensiones, tiene min(n-1, p) dimensiones diferentes a los componentes principales. Sin embargo, casi nunca estamos interesando en todos ellos. En vez de eso solo nos interesa un par de componentes principales para visualizar o interpretar la data.

- Cuántos componentes necesitamos? Desafortunadamente, no hay una sola respuesta.

- Típicamente se decide viendo el scree plot que mostramos en el gráfico anterior. Escogemos el menos número de componentes principales que se requieren para explicar una cantidad considerable de la varianza (60% - 80%).

Agenda

3. Clustering

3. Métodos de Clustering

- Clustering se refiere a diferentes técnicas para encontrar subgrupos o clusters en un Dataset.
- Cuando encontramos un subgrupo de observaciones, estamos buscando diferentes grupos en que las observaciones dentro de cada grupo sean similar dentro de ellas, mientras que observaciones en diferentes grupos son diferentes.
- Para hacer esto mucho más concreto, debemos definir que significa que dos o más observaciones sean similares o diferentes. En sí, esto va a depender del dominio del tema o del conocimiento sobre la data estudiada.
- Por ejemplo, supongamos que tenemos n observaciones y p variables. Las observaciones corresponden a muestras de tejido de pacientes con cáncer de mama y las p variables a medidas recolectada para esas muestras.
- Clustering podría ayudarnos a identificar tipos de muestras similares dentro de este Dataset.

3. Métodos de Clustering

- Tanto Clustering como PCA busca simplificar la data a través de un resumen gráficos, pero sus mecanismos son diferentes:

  - PCA looks to find a low-dimensional representation of the observations that explain a good fraction of the variance;
  - Clustering looks to find homogeneous subgroups among the observations.

- Otra aplicación de Clustering la podemos ver en marketing. Imaginemos que tenemos a un gran número de variables (ingreso promedio por hogar, ocupación, distancia al área urbana más cercana, entres otras) para un gran número de personas.

- El objetivo es realizar una segmentación de mercado para identificar subgrupos de personas que pueden ser más perceptivos a una forma particular de publicidad.

3. Métodos de Clustering

- Dado que el método Clustering es muy popular en diferentes ramas, existe un gran número de métodos.

- En este capítulo nos vamos a enfocar en las dos técnicas más conocidas: K means Clustering y Hierarchical Clustering.

- En K means Clustering buscamos particionar las observaciones en un número determinado de clusters, por otro lado, en Hierarchical Clustering no sabemos cuantos clusters queremos entonces visualizamos un dendograma que nos va ayudar a tomar esa decisión.

- Hay ventajas y desventajas por cada método que ya iremos aprendiendo.

Agenda

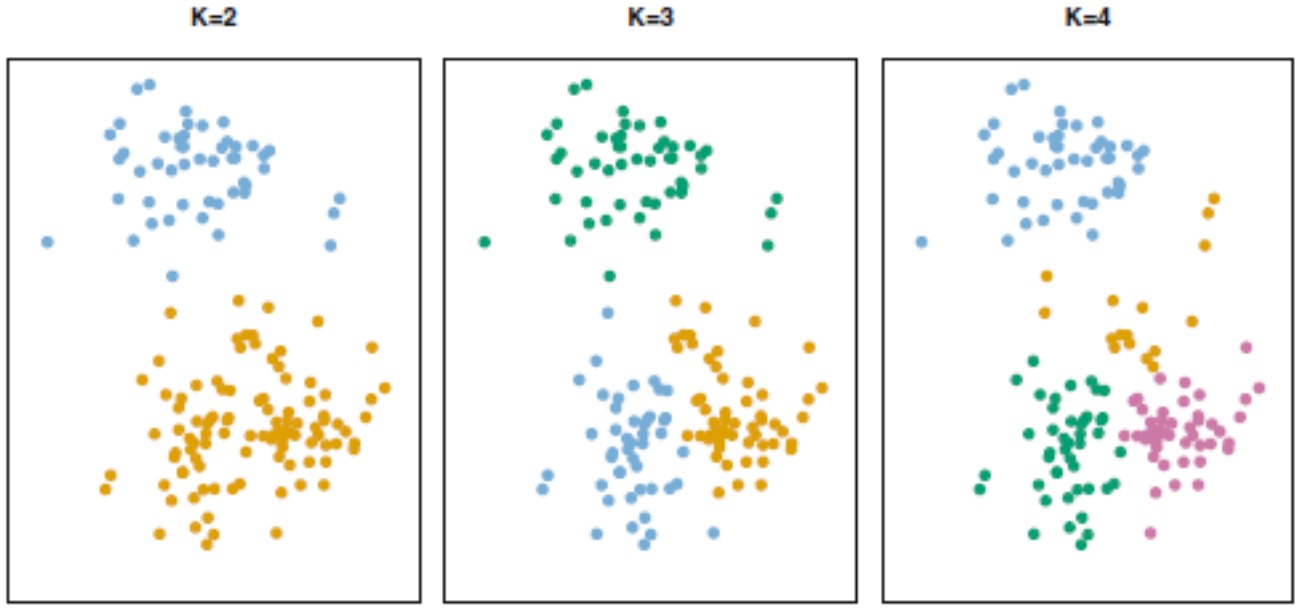
3.1. K means clustering

3.1. K means clustering

- Es una técnica simple y elegante para particionar el Dataset en K diferentes y no entrelazados clusters.

- Para realizar esta técnica, debemos especificar el número de clusters K deseados, después el algoritmo K means va asignar cada observación a cada cluster establecido.

- En el siguiente gráfico vemos el resultado obtenido de realizar K means en un data de 150 observaciones en 2 dimensiones usando 3 valores diferentes de K.



3.1. K means clustering

- El procedimiento K-means viene de un problema matemático sencillo e intuitivo.

- Primero empezamos con la notación, sea $C_1, \ldots, C_K$ que denotan los conjuntos que contienen los índices de las observaciones en cada grupo. Estos conjuntos satisfacen las siguientes propiedades:

1. $C_1 \cup C_2 \cup \ldots \cup C_K = \{1, \ldots, n\}$. In other words, each observation belongs to at least one of the K clusters.

2. $C_k \cap C_{k'} = \emptyset$ for all $k \neq k'$. In other words, the clusters are non-overlapping: no observation belongs to more than one cluster.

- Por ejemplo, si la observación ith está en el cluster kth y entonces $i \in C_k$. La idea detrás de K means es que un buen clustering es aquel en que la variación dentro del cluster es la más pequeña posible.

- La variación dentro del cluster C_k la vamos a medir con $W(C_k)$ que será la cantidad por la que las observaciones dentro del cluster difieren de las otras.

3.1. K means clustering

- De ese modo, el problema que queremos resolver es:

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} W(C_k) \right\} \qquad (1)$$

- Esta fórmula nos dice que si queremos particionar las observaciones en K clusters dado que la variación total dentro del cluster, sumado en todos los clusters, sea la menor posible.

- Resolver esa ecuación parece factible pero nos falta definir qué es la variación dentro del cluster. Hay muchas maneras de definir este concepto, pero por mayormente se elige la "squared Euclidean distance". Que la definimos como:

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2. \qquad (2)x$$

Donde $|C_k|$ denota el número de observaciones dentro del cluster $k\text{th}$

3.1. K means clustering

- En otras palabras la variación dentro del cluster Kth es la suma de todas las parejas de “squared Euclidean distances” entre las observaciones del cluster Kth dividido por el numero total de observaciones dentro del cluster Kth.

- Combinando las 2 ecuaciones anteriores, tenemos:

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\} \qquad (3)$$

- Ahora nos enfocaremos en el algoritmo pa K^n olver esa ecuación. Esto puede ser un problema bastante difícil de resolver ya que hay casi formas para particionar las n observaciones en K clusters. Este es un número enorme a no ser K y n sean pequeños.

3.1. K means clustering

- Afortunadamente, un algoritmo bastante simple muestra como dar un óptimo local. Es una buena solución al problema de optimización de K means. El algoritmo es:

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.
   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

- Este algoritmo tiene la garantía que va a decrecer la función (3). Para entender porque acá presentamos un ejemplo:

3.1. K means clustering

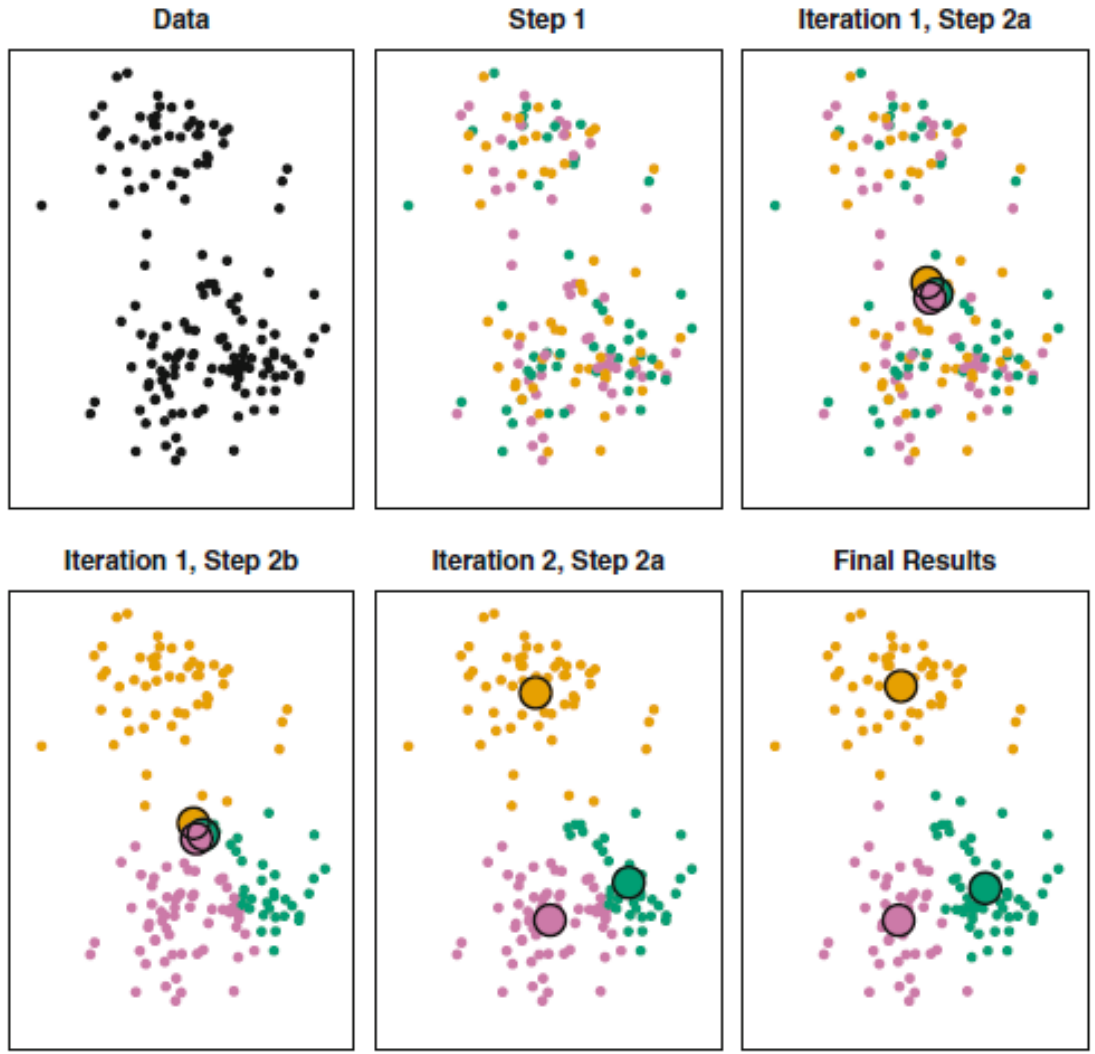
$$\frac{1}{|C_k|}\sum_{i,i' \in C_k}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2 = 2\sum_{i \in C_k}\sum_{j=1}^{p}(x_{ij}-\bar{x}_{kj})^2 \qquad (4)$$

- Donde $\bar{x}_{kj} = \frac{1}{|C_k|}\sum_{i \in C_k} x_{ij}$ es la media para la observación j en el cluster C_k. En la etapa 2(a) la media del cluster para cada observación son las constantes que minimizan la suma al cuadrado de las deviaciones.

- Y en la etapa 2(b), reasignar las observaciones solo puede mejorar (4). Esto significa que mientras el algoritmo está corriendo, el cluster obtenido va a mejorar continuamente hasta que el resultado no cambie más. Cuando el resultado no cambie más significa que el óptimo local ha sido alcanzado.

- En la siguiente figura el progreso del algoritmo en un ejemplo.

3.1. K means clustering



- En la etapa 1 del algoritmo cada observación es asignada de forma aleatoria a un cluster.
- En 2(a) los centros de los clusters son computados.
- En 2(b) cada observación es asignada a su centro más cercano.
- En 2(a) se vuelven a computar los clusters, llevándonos a nuevos centros.
- Los resultados finales se obtienen después de 10 iteraciones.

3.1. K means clustering

- El nombre de K means clustering viene del hecho que en la etapa 2(a) los centros del cluster se computan como las medias de las observaciones a cada cluster.

- Este algoritmo encuentra un óptimo local pero no uno global, los resultados obtenidos van a depender del cluster aleatorio inicial asignado en la etapa 1 del algoritmo. Por esta razón, es importante correr el algoritmo múltiples veces desde diferentes punto iniciales.

- En la siguiente figura se muestra el óptimo local obtenido corriendo K means 6 veces, usando 6 puntos iniciales diferentes.

- Así como hemos, para correr este algoritmo es necesario especificar el número de clusters desde un inicio, esto puede tener algunas desventajas que ya veremos.

3.1. K means clustering

- Cada plot tiene un valor objetivo.
- 3 puntos locales diferentes hemos encontrado, uno de ellos resulta en un valor pequeño.
- El valor de 235.8 nos da la mejor solución.

Agenda

3.2. Hierarchical Clustering

3.2. Hierarchical Clustering

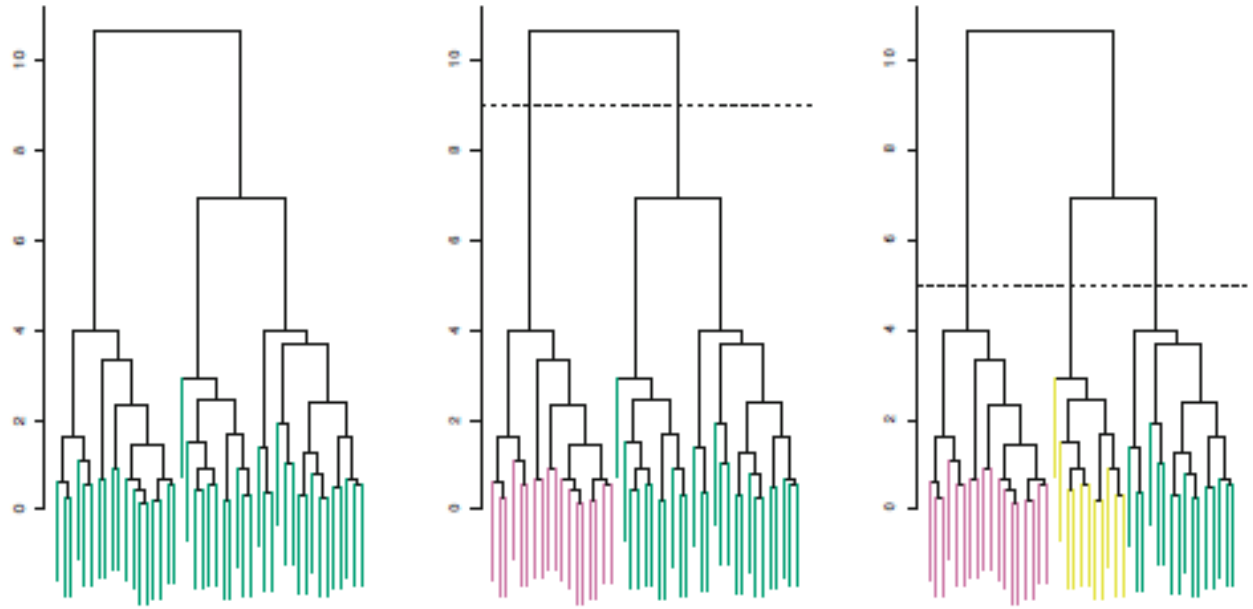
- Una desventaja potencial de K-means clustering es que requieres que especifiquemos el número de cluster que usaremos.

- Hierarchical clustering es una alternativa a este problema ya que no requiere esta pre especificación.

- Tiene una ventaja sobre K means y es que vemos las observaciones en una representación de árbol que llamamos dendograma.

- En esta sección describimos el cluster aglomerado. Este es el mas común dentro de Hierarchical clustering y se refiere al hecho que el dendograma empieza desde las hojas del “árbol” y combina clusters en el “tronco”.

3.2. Hierarchical Clustering

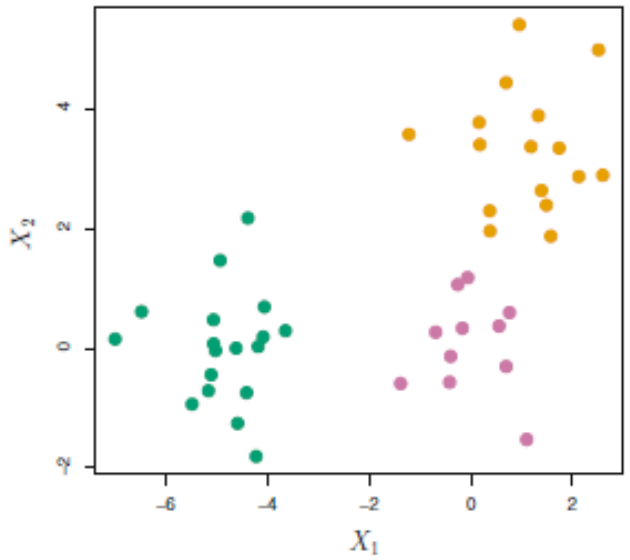


Ejemplo de dendograma

- En la segunda figura se corta en 9, llevándonos a 2 cluster.
- En la tercera figura se corta en 5, llevándonos a 3 clusters.

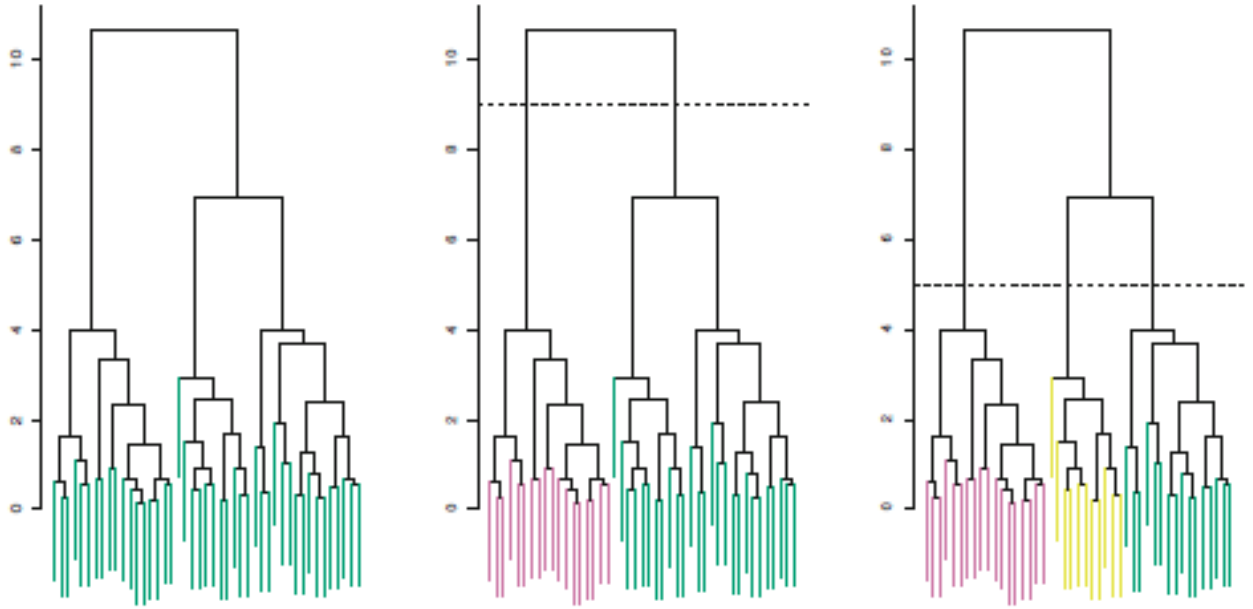
3.2. Hierarchical Clustering Interpretando un dendograma

- Empezamos con la data mostramos a continuación, contiene 45 observaciones, la data fue generada de un modelo de 3 clases, las verdaderas clases tienen un color diferente.



- Sin embargo, que observamos la data sin esa clasificación de colores y que queremos realizar Hierarchical clustering. Hierarchical Clustering, nos va a llevar al siguiente dendograma.

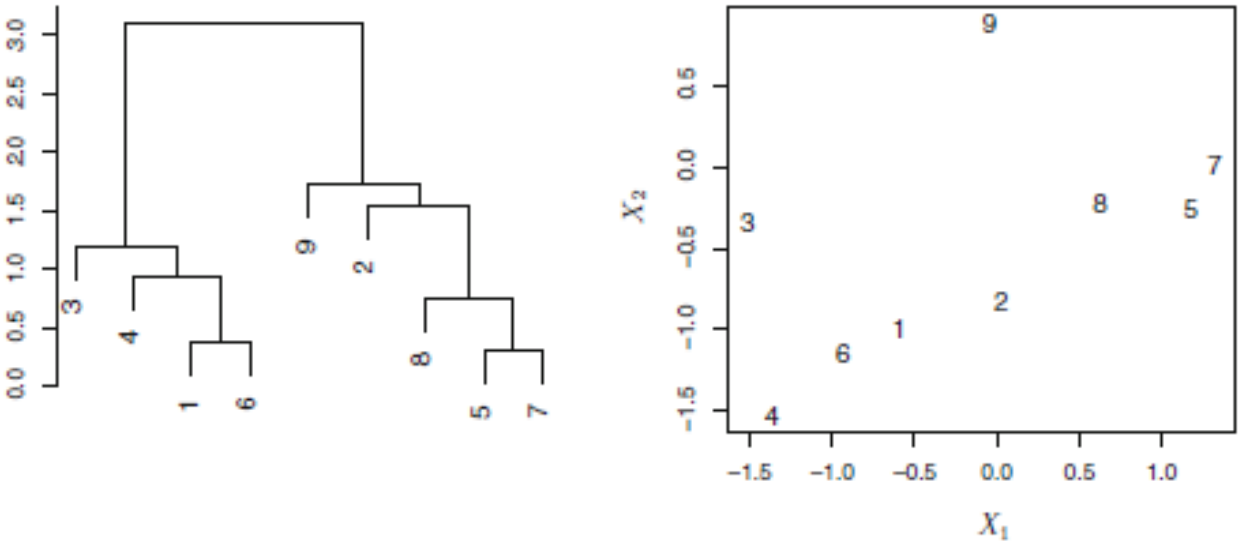
3.2. Hierarchical Clustering



Dendograma

- En este dendograma, cada hoja representa una de las 45 observaciones de la data con la que estamos trabajando.
- Sin embargo, a medida que vamos subiendo en el árbol, vemos ramas que contiene esas observaciones.
- La rama indica que son observaciones similares (grupos), por otro lado las ramas cerca al tope del árbol puede ser más diferentes.
- La altura del árbol los vemos en el eje vertical.

3.2. Hierarchical Clustering



Dendograma

Ejemplo de dendograma con solo 9 observaciones.

- Este es un dendograma bastante sencillo, vemos que las observaciones 6 y 7 (1 y 6) son muy similares dado que se unen en el punto más bajo.
- Sin embargo, no podríamos decir que las observaciones 9 y 2 son similares solo porque se encuentran cerca.
- De hecho, la observación 9 no es más similar que la observación 2 que observaciones 8, 5 y 7.
- Podemos generar conclusiones sobre el eje vertical mas no sobre el horizontal.

3.2. Hierarchical Clustering

- Dado que entendemos el dendograma, ahora podemos identificar clusters. Para hacer esto debemos hacer un corte horizontal sobre el dendograma.

- Los distintos conjuntos de observaciones debajo de la línea pueden ser interpretados como clusters, si lo cortamos en la altura 9 tendremos 2 clusters, y si lo cortamos en altura 5 tendremos 3 clusters. Si cortamos a la altura de 0 cada observación tendrá su propio cluster.

- La altura en la que cortemos el dendograma nos da el mismo que K en K-means, controla el número de clusters obtenidos.

- La figura anterior nos da a entender que un simple dendograma puede ser usado para obtener el número de clusters.

- Sin embargo, muchas veces la elección del número de clusters no es muy clara.

3.2. Hierarchical Clustering
El algoritmo

- El dendograma visto se obtiene mediante un algoritmo bastante simple.
- Empezamos con definir la medida “dissimilarity” entre cada par de observaciones. En la mayoría de casos se usa la distancia euclidiana. El algoritmo procede iterativamente, empezando en la parte baja del dendograma, cada observación es tratada como un cluster diferente.
- Los dos clusters que son más similares se les trata como un nuevo cluster, entones ahora hay n-1 clusters, luego otros dos clusters que son similares se juntan y ya tenemos n-2 cluster y así sucesivamente.
- Esto se realiza hasta que solamente hay un cluster y con eso se completa el dendograma.

3.2. Hierarchical Clustering
El algoritmo

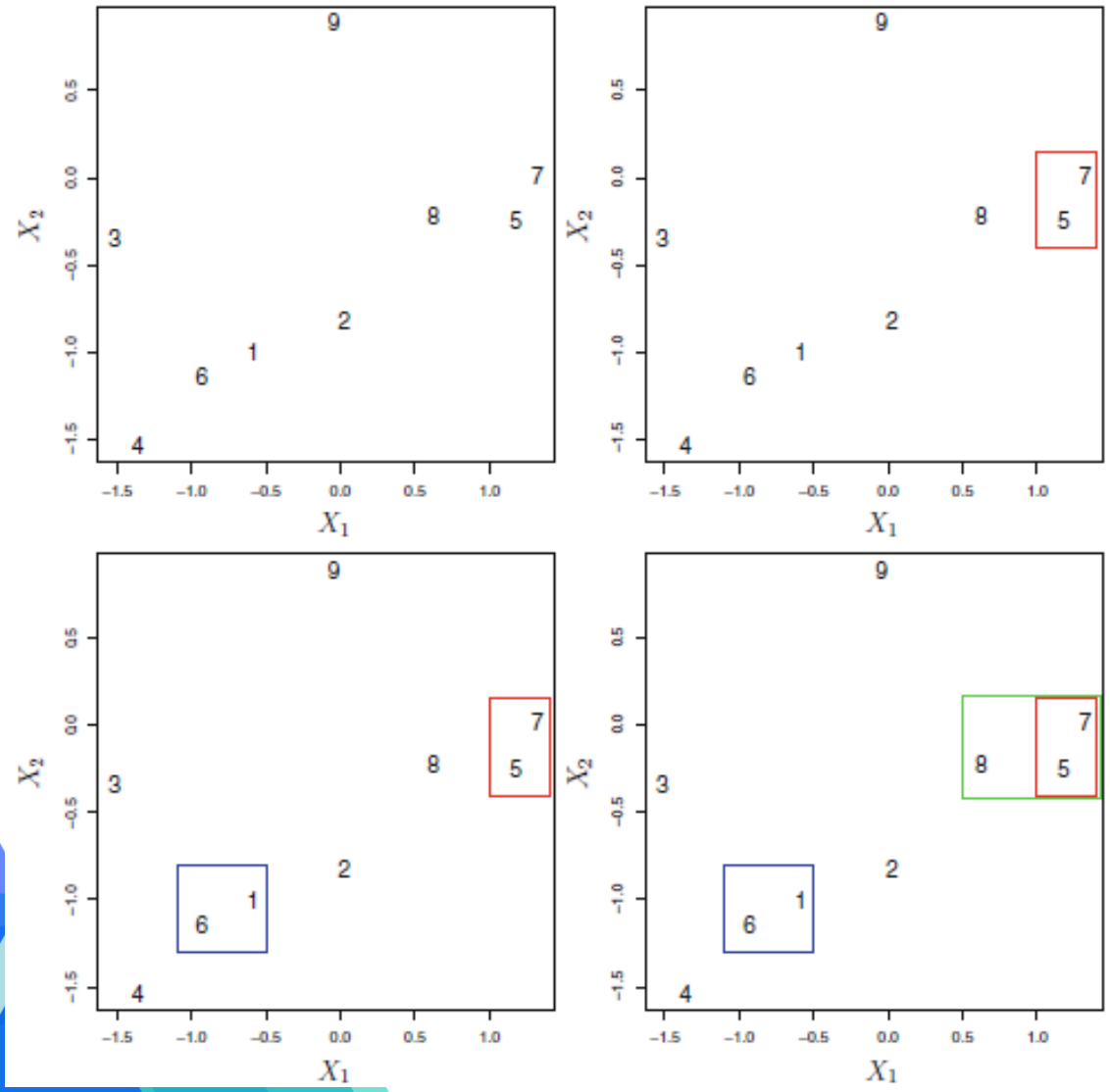


FIGURE 10.11. *An illustration of the first few steps of the hierarchical clustering algorithm, using the data from Figure 10.10, with complete linkage and Euclidean distance.* Top Left: *initially, there are nine distinct clusters,* $\{1\}, \{2\}, \ldots, \{9\}$. Top Right: *the two clusters that are closest together,* $\{5\}$ *and* $\{7\}$, *are fused into a single cluster.* Bottom Left: *the two clusters that are closest together,* $\{6\}$ *and* $\{1\}$, *are fused into a single cluster.* Bottom Right: *the two clusters that are closest together using* complete linkage, $\{8\}$ *and the cluster* $\{5, 7\}$, *are fused into a single cluster.*

3.2. Hierarchical Clustering
El algoritmo

- ¿Cómo determinamos el cluster 5,7? Porque no está también ahí el cluster 8?
- Cómo determinamos el concepto de “dissimilarity” entre dos clusters? Si uno o ambos contiene múltiples observaciones?
- Este concepto debe ser extendido a un par de grupos de observaciones, esto lo resolvemos generando el concepto de “linkage” que define el “dissmilarity” entre un grupo de observaciones.
- Los 4 tipos más comunes de “linkage” son: complete, average, single and centroid.
- Average y complete linkage son prefereido sobre single linkage, dado que dan dendogramas mucho más balanceados.

3.2. Hierarchical Clustering
El algoritmo

Linkage	*Description*
Complete	Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *largest* of these dissimilarities.
Single	Minimal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *smallest* of these dissimilarities. Single linkage can result in extended, trailing clusters in which single observations are fused one-at-a-time.
Average	Mean intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *average* of these dissimilarities.
Centroid	Dissimilarity between the centroid for cluster A (a mean vector of length p) and the centroid for cluster B. Centroid linkage can result in undesirable *inversions*.

Algorithm 10.2 *Hierarchical Clustering*

1. Begin with n observations and a measure (such as Euclidean distance) of all the $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities. Treat each observation as its own cluster.
2. For $i = n, n-1, \ldots, 2$:
   (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.
   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.

3.2. Hierarchical Clustering
El algoritmo

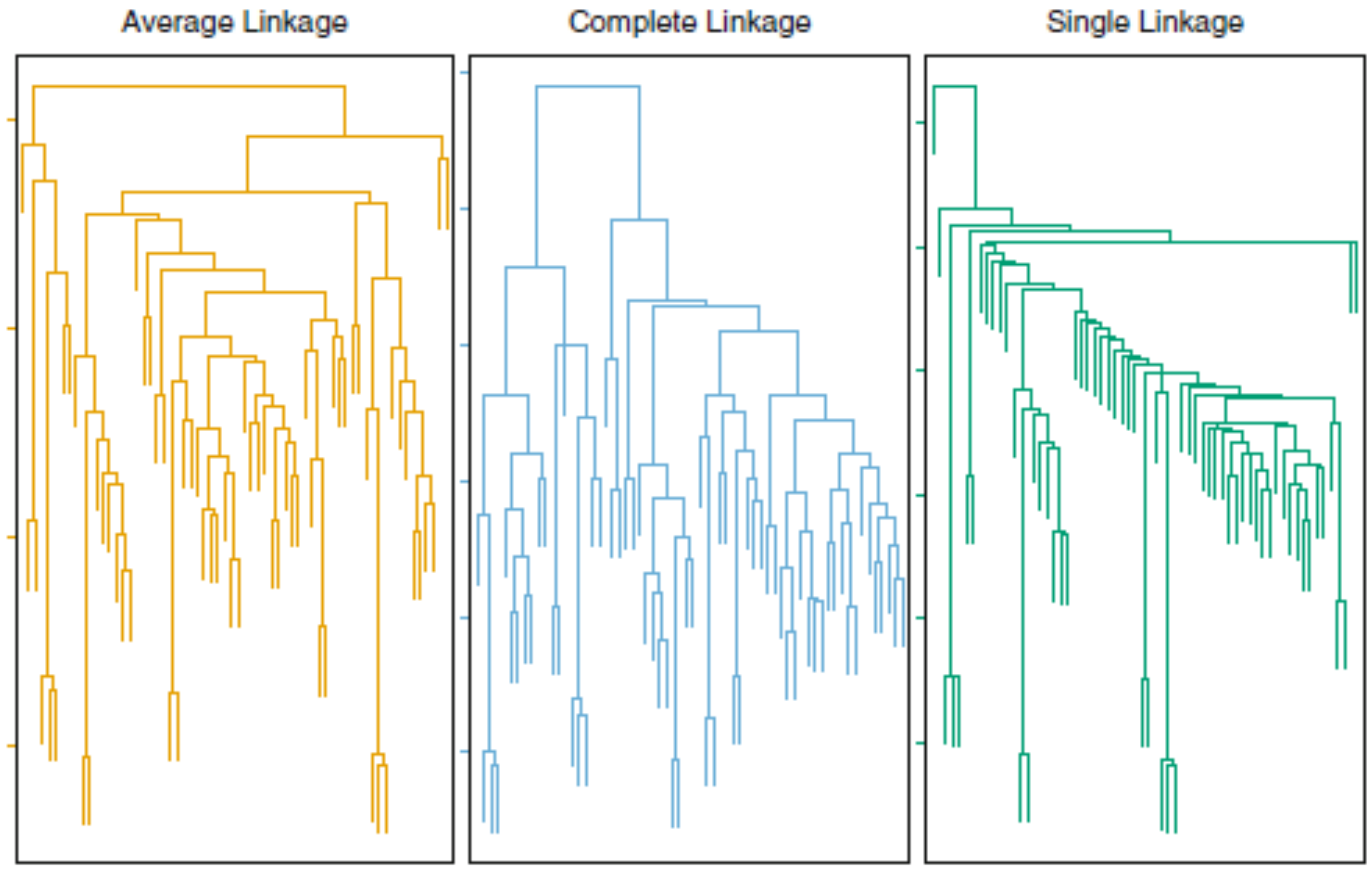


FIGURE 10.12. *Average, complete, and single linkage applied to an example data set. Average and complete linkage tend to yield more balanced clusters.*

Agenda

3.3. Practical issues

3.3. Practical issues

- Clustering es una herramienta bastante potente en Unsupervised Learning. Sin embargo hay que tener en cuentas ciertas consideraciones.

Pequeñas decisiones con grandes consecuencias:

❖ Las observaciones deben estandarizarse? Por ejemplo, algunas variables debemos centrarlas para que tengan media 0 y desviación estándar de 1.

❖ En el caso de Hierarchical clustering:
✓ Qué tipo de linkage deberíamos usar?
✓ Dónde debemos cortar el dendograma para obtener los clusters?

❖ En caso de K means, cuántos clusters debemos usar?

- Cada una de estas decisión va a tener un gran impacto en los resultados que vamos a obtener.
- En la práctica debemos tratar con diferentes decisiones, y buscar la que nos da la mejor interpretación posible. Con estos métodos no hay solo una respuesta verdadera.

3.3. Practical issues

Validando los clusters obtenidos:

- No hay una clara aproximación en la academia de cómo validar que el número de clusters sea el correcto, pero hay bastantes técnicas para realizar esto.

- Por ejemplo asignar un p-value al cluster para evaluar si solo uno es necesario, sin embargo no hay consenso sobre cual es la mejor aproximación.

- Para más detalles, revisar Hastie et al. (2009).

¡GRACIAS!

INFOX
educación sin fronteras