U-tad
CENTRO UNIVERSITARIO DE TECNOLOGÍA Y ARTE DIGITAL

Modelos de Descripción

APRENDIZAJE AUTOMÁTICO I

Curso académico:
2022 – 2023

Modelos de Descripción: **Índice**

Clustering

Modelos de Descripción

- Clustering
  - Técnicas no supervisadas de agrupamiento
    - Tarea de modelado descriptivo que divide un conjunto de datos en grupos homogéneos
    - Los datos no tienen asociada ninguna etiqueta
    - Se diferencia de la tarea de clasificación en que no se conocen los grupos, ni el número de grupos que puede haber
    - El objetivo es encontrar grupos de datos u observaciones que sean muy similares entre sí, y muy diferentes con las observaciones de fuera, es decir, con otros grupos

Modelos de Descripción: **Clustering**

- Métodos de agrupamiento
  - **Jerárquicos**
    - Los datos se agrupan en una estructura en forma de árbol
  - **No jerárquicos**
    - Los datos se agrupan en un nivel determinado
- Aplicaciones del *clustering*
  - Minería de datos
  - Procesamiento de imágenes digitales
  - Bioinformática
  - Segmentar clientes en grupos con demografía similar
  - Detección de comportamientos anómalos
  - …

Modelos de Descripción: **Clustering Jerárquico**

- Agrupamiento Jerárquico
  - Generación de un árbol donde las hojas corresponden a los ejemplos u observaciones y el resto de los nodos son grupos de ejemplos
  - Se agrupan sucesivamente los datos uniendo de forma progresiva los ejemplos en clústeres, que a su vez se unen entre si hasta definir dos subconjuntos
- Tipos de algoritmo de agrupación
  - **Aglomerativos**
    - Parten de tantos grupos como observaciones y van fusionando los grupos más similares formando clústeres cada vez más grandes hasta formar un único clúster. El árbol se construye desde las hojas hacia la raíz
  - **Divisivos**
    - Parten de un solo grupo que se va particionando en distintos clústeres. Se conoce como segmentación. El árbol se construye desde la raíz hacia las hojas

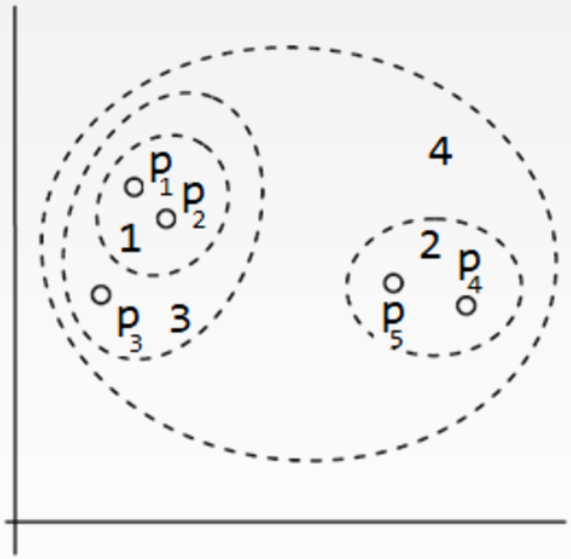
Modelos de Descripción: **Clustering Jerárquico**

- Algoritmo Aglomerativo

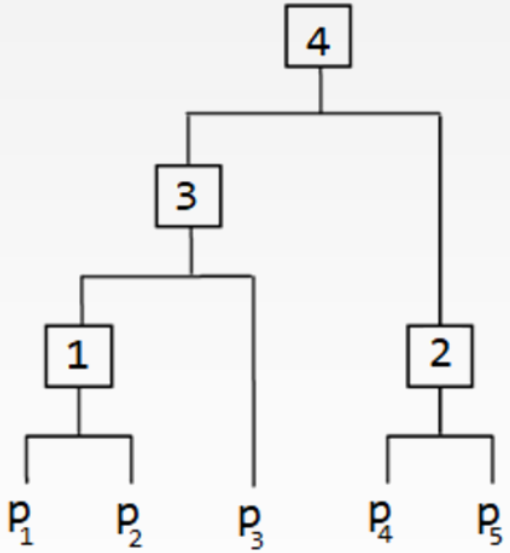
1. Definir cada observación (fila, caso, ejemplo) como un clúster (grupo)
2. Calcular todas las distancias por pares
3. Combinar los dos clústeres que tienen la menor distancia. Esto reduce en uno el número de grupos o clústeres
   - Encontrar los dos grupos que están más cerca uno del otro
   - Fusionar los dos grupos formando uno nuevo
   - Calcular la distancia del nuevo grupo al resto de los grupos
4. Repetir los pasos 2 y 3 hasta que todos los clústeres se hayan fusionado en uno solo que contiene todas las observaciones

Modelos de Descripción: **Clustering Jerárquico**

- Algoritmo Aglomerativo



Clústeres Anidados



Dendrograma

Modelos de Descripción: **Clustering Jerárquico**

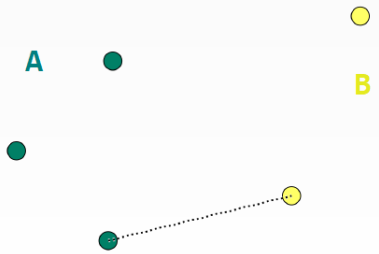
- Dendrograma
  - Diagrama que resume el proceso de composición de los clústeres
    - Forma muy intuitiva de representar el proceso de construcción de los grupos, pero no ofrece información sobre la distancia entre los distintos objetos
    - Ayuda a decidir el número óptimo de clústeres en base a su medida de similitud 'cortando' el dendrograma por donde se observe un salto importante en la longitud de las líneas que definen los grupos, es decir, cuando se detecta una separación considerable en los niveles de similitud
- Centroide
  - Media de cada variable para cada clúster
  - A partir de los centroides se puede interpretar cada clúster

Clustering Jerárquico Aglomerativo

- Criterio de enlazado
  - Dependiendo de cómo se calcule la distancia de enlace entre los grupos se pueden distinguir cinco métodos

- Métodos de fusión de los clústeres
  - **Enlace simple o de distancias mínimas** (*single linkage*)
    - Dados dos clústeres A y B, se calcula la distancia entre todos los puntos de los dos grupos y se toma la distancia mínima como la distancia entre grupos
    - Forma clústeres encadenados

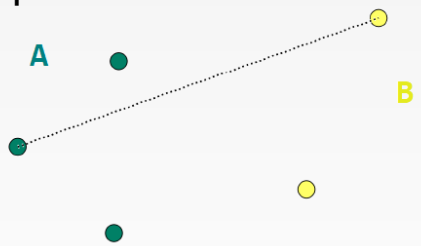
$$min\{dist(x, y \mid x \in A, y \in B)\}$$



Clustering Jerárquico Aglomerativo

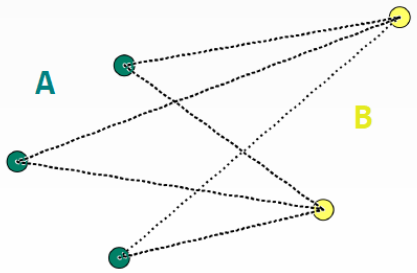
- Métodos de fusión de los clústeres
  - **Enlace completo o de distancias máximas** (*complete linkage*)
    - Igual que el anterior pero tomando la mayor distancia. Se escogen los elementos más lejanos o que menos se parecen. Forma clústeres compactos

$$max\{dist(x, y \mid x \in A, y \in B)\}$$



  - **Enlace de distancia media** (*average link*age)
    - La distancia entre dos clústeres es la distancia media de todos los pares de distancias entre los puntos de datos en ambos clústeres

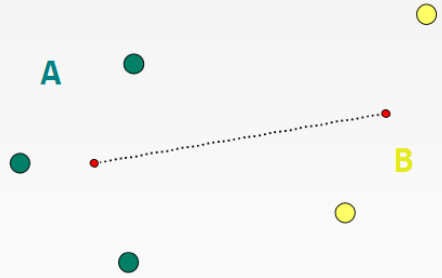
$$mean\{dist(x, y \mid x \in A, y \in B)\}$$



Clustering Jerárquico Aglomerativo

- Métodos de fusión de los clústeres
  - **Método Centroide**
    - La distancia entre dos cúmulos es la distancia entre sus centroides

$$dist(C_A, C_B)$$



  - **Método de Ward**
    - Minimiza las sumas de cuadrados intra-grupo. Forma los clústeres más compactos y de igual tamaño y forma que el resto de métodos. Menos sensible a valores atípicos

$$min\{dist^2(x, y \mid x \in A, y \in B)\}$$

Clustering Jerárquico Aglomerativo

Cluster Method	Definición de la distancia entre dos clústeres
Single linkage	Distancia más corta entre un punto de un clúster y un punto de otro clúster
Complete linkage	Distancia más larga entre un punto de un clúster y un punto de otro clúster
Average linkage	Distancia media entre cada punto de un clúster y cada punto del otro clúster
Centroid	Distancia entre los centroides (vector de medias de las variables) de los dos clústeres. Para una única observación, el centroide son los valores de la variable
Ward	La distancia entre dos clústeres se define como el incremento de la suma de errores al cuadrado (distancias) desde la de dos clústeres a la de un clúster fusionado

Modelos de Descripción: **Clustering Jerárquico**

- Función de Disimilitud o Distancia

Euclídea	Euclídea al cuadrado	Manhattan o City-Block
$d_E(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$	$d_{E2}(x,y) = \sum_{i=1}^{n}(x_i - y_i)^2$	$d_M(x,y) = \sum_{i=1}^{n}\lvert x_i - y_i\rvert$

La distancia Euclídea se suele usar cuando cada dimensión mide propiedades similares y la Manhattan en caso contrario

La más usual es la distancia Euclídea

Modelos de Descripción: **Clustering Jerárquico**

- Función de Disimilitud o Distancia
  - Tanto la distancia Euclídea como la Manhattan satisfacen los siguientes requisitos matemáticos de una función de distancia:

1) $d(i,j) \geq 0$
2) $d(i,i) = 0$
3) $d(i,j) = d(j,i)$
4) $d(i,j) \leq d(i,h) + d(h,j)$

para todos los objetos *i, j y h*

  - Una distancia que satisface estas propiedades se llama métrica

Modelos de Descripción: **Clustering Jerárquico**

- Resumen
  - Utiliza un algoritmo de iteración no supervisado jerárquico y aglomerativo
  - Es útil cuando queremos obtener distintos grupos con características similares de un mismo conjunto de datos
  - La distancia entre las observaciones permitirá realizar grupos con datos homogéneos pero heterogéneos entre sí
  - La media de las variables que conforman cada clúster ayudará a la interpretación de cada grupo
  - Dibujar los resultados determinará el número de clústeres a escoger como paso previo al *clustering* no jerárquico para seleccionar el valor de *k*

Modelos de Descripción: **Clustering Jerárquico**

- Función en R
  - **hclust()**
    - hclust(data, method)
      - **data**
        - La matriz de datos es una **matriz de distancias**
      - **method**
        - Método aglomerativo que se quiere usar
        - Por defecto es *'complete'*

Clustering Jerárquico: Ejemplo

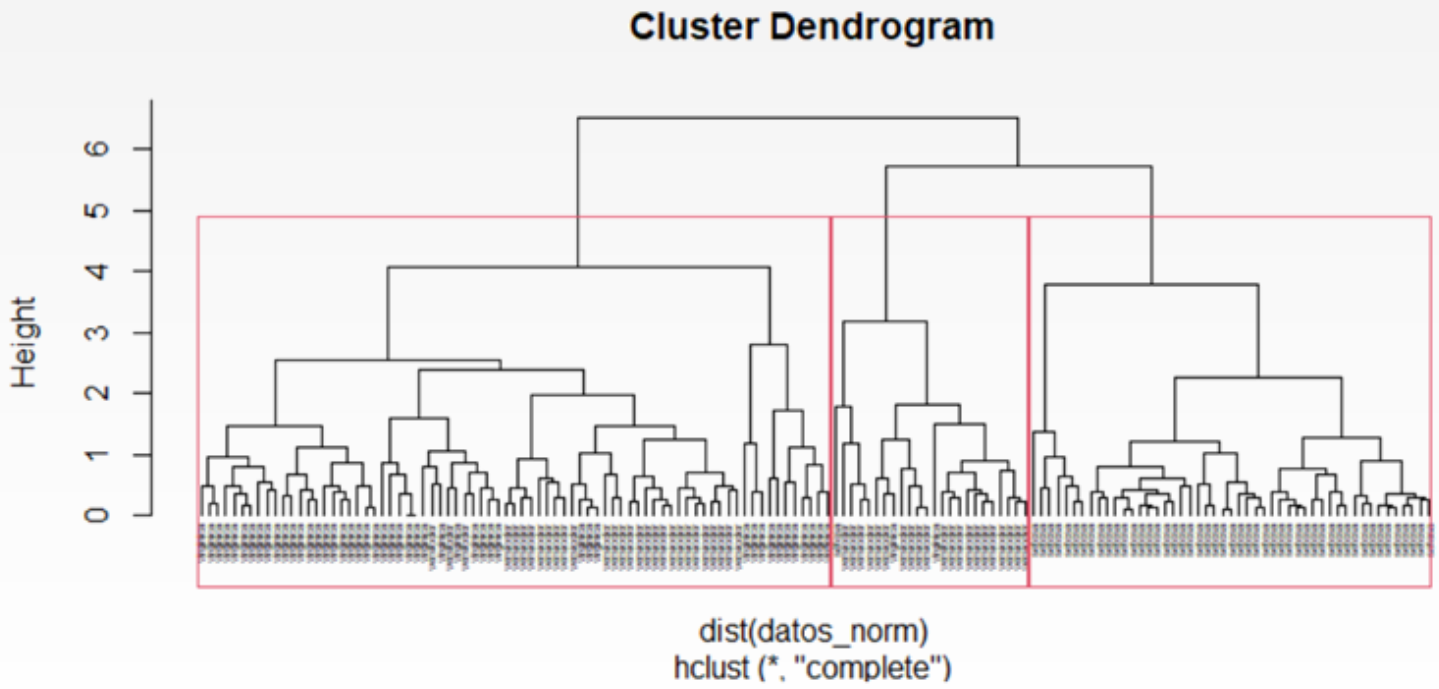
- R

```
> datos <- iris[, -5]
> head(datos)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
3          4.7         3.2          1.3         0.2
4          4.6         3.1          1.5         0.2
5          5.0         3.6          1.4         0.2
6          5.4         3.9          1.7         0.4
> datos_norm <- scale(datos)
> clustering <- hclust(dist(datos_norm)) #recibe una matriz de distancias
> clustering

Call:
hclust(d = dist(datos_norm))

Cluster method   : complete
Distance         : euclidean
Number of objects: 150
```

Clustering Jerárquico: Ejemplo

- R

```
> plot(clustering, hang = -1, cex=.4, labels=iris$Species)
> rect.hclust(clustering, k=3)
```



Modelos de Descripción: **Clustering No Jerárquico**

- Técnicas de agrupamiento basadas en vecindad
  - k-means
  - k-medoids

- k-means
  - Método de agrupamiento que tiene como objetivo la partición de un conjunto de *n* observaciones en *k* grupos, o clústeres, en el que cada observación pertenece al clúster cuyo valor medio, o centroide, sea más pequeño, es decir, esté más cercano
  - Es necesario conocer a priori el número de centroides, por lo que el resultado puede variar mucho en función del parámetro *k*
  - Si un grupo es de muy pequeño tamaño se puede considerar como un clúster de valores atípicos

Clustering No Jerárquico: **k-means**

- Objetivo
  - Obtener una partición de un conjunto de *n* elementos, con *p* dimensiones, en *k* subconjuntos o grupos disjuntos, S_i, minimizando el error cometido

$$Error = \sum_{i=1}^{k} \sum_{x_j \epsilon S_i} dist(x_j, \mu_i)$$

  - *dist(x,y)* es la función de distancia deseada (normalmente la distancia Euclídea) y μ_i es el centroide de cada grupo S_i
  - Centroide
    - Centro geométrico de los elementos del espacio *p*-dimensional

$$\mu_i = \frac{1}{|S_i|} \sum_{x_j \epsilon S_i} x_j$$

Clustering No Jerárquico: **k-means**

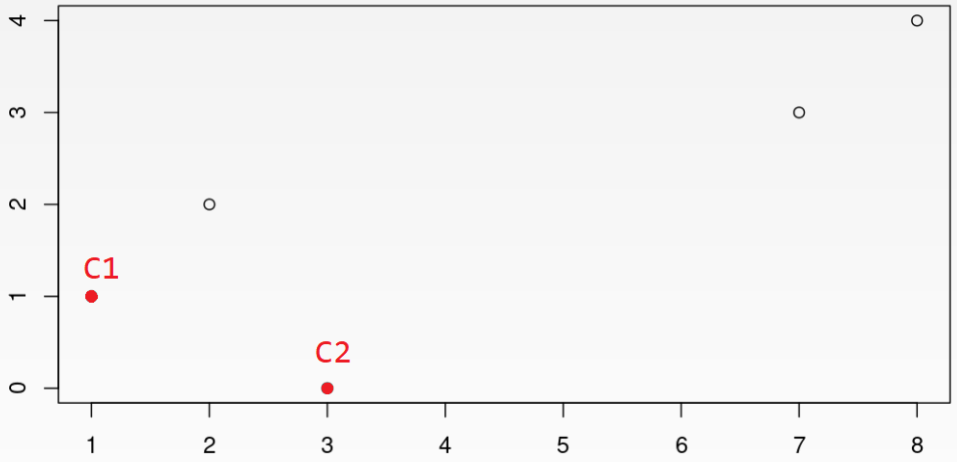
- Algoritmo de Lloyd

1. Generar *k* centroides iniciales
   - Dividir aleatoriamente los ejemplos en *k* conjuntos y calcular la media (el punto medio) de cada conjunto
2. Asignar cada elemento del conjunto original al grupo más cercano, de acuerdo a la distancia entre el elemento y el centroide de cada grupo
3. Calcular los nuevos centroides para cada grupo en función de los elementos asignados en el paso 2
4. Repetir desde el paso 2 hasta que no se produzca ningún cambio de grupo, es decir, los conjuntos no varíen

Clustering No Jerárquico: **k-means**

- Ejemplo

$$X = \begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 0 \\ 7 & 3 \\ 8 & 4 \end{bmatrix}$$



  - Paso 1
    - k = 2
    - Centroides
      - C1 = (1,1) fila 1
      - C2 = (3,0) fila 3

Clustering No Jerárquico: **k-means**

- Ejemplo
  - Paso 2
    - Asignar filas 2, 4 y 5 al clúster cuyo centroide esté a menor distancia
      - $d(x_2, c_1) = \sqrt{(2-1)^2+(2-1)^2} = \mathbf{1,41}$
      - $d(x_2, c_2) = \sqrt{(2-3)^2+(2-0)^2} = 2{,}23$
      - $d(x_4, c_1) = \sqrt{(7-1)^2+(3-1)^2} = 6{,}32$
      - $d(x_4, c_2) = \sqrt{(7-3)^2+(3-0)^2} = \mathbf{5}$
      - $d(x_5, c_1) = \sqrt{(8-1)^2+(4-1)^2} = 7{,}61$
      - $d(x_5, c_2) = \sqrt{(8-3)^2+(4-0)^2} = \mathbf{6,4}$

Clustering No Jerárquico: **k-means**

- Ejemplo
  - Paso 3
    - Calcular los nuevos centroides
      - Grupo 1 {x1, x2} ⇨ c1 = {(1+2)/2 (1+2)/2} = {1,5 1,5}
      - Grupo 2 {x3, x4, x5} ⇨ c2 = {(3+7+8)/3 (0+3+4)/3} = {6 2,3}
    - Vuelta al paso 2
      - Reasignar las filas. Calcular las distancias a los nuevos centroides

Clustering No Jerárquico: **k-means**

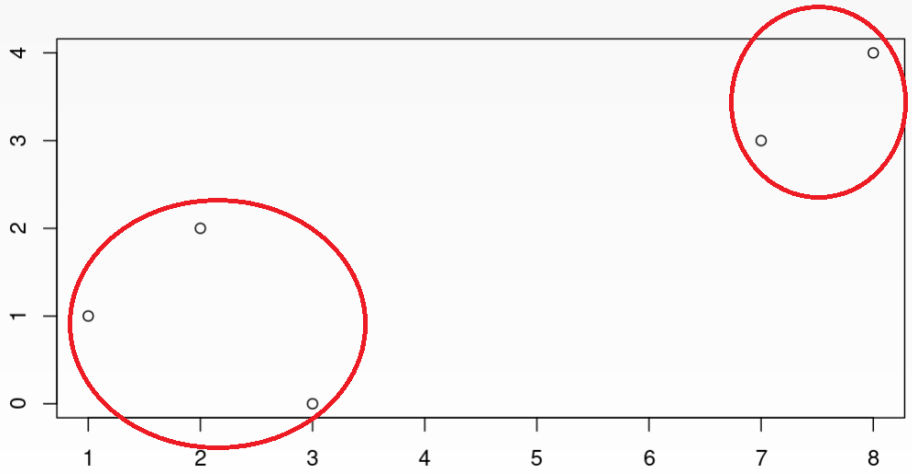
- Ejemplo
  - Paso 2
    - Reasignar las filas. Calcular las distancias a los nuevos centroides
      - $d(x_2, c_1) = \sqrt{(2-1{,}5)^2+(2-1{,}5)^2} = \mathbf{0,7}$
      - $d(x_2, c_2) = \sqrt{(2-6)^2+(2-2{,}3)^2} = 4{,}01$
      - $d(x_3, c_1) = \sqrt{(3-1{,}5)^2+(0-1{,}5)^2} = \mathbf{2,12}$
      - $d(x_3, c_2) = \sqrt{(3-6)^2+(0-2{,}3)^2} = 3{,}78$
      - $d(x_4, c_1) = \sqrt{(7-1{,}5)^2+(3-1{,}5)^2} = 5{,}7$
      - $d(x_4, c_2) = \sqrt{(7-6)^2+(3-2{,}3)^2} = \mathbf{1,22}$
      - $d(x_5, c_1) = \sqrt{(8-1{,}5)^2+(4-1{,}5)^2} = 6{,}96$
      - $d(x_5, c_2) = \sqrt{(8-6)^2+(4-2{,}3)^2} = \mathbf{2,62}$

Clustering No Jerárquico: **k-means**

- Ejemplo
  - Paso 3
    - Calcular los nuevos centroides
      - Grupo 1 {x1, x2, x3} ⇨ c1 = {2 1}
      - Grupo 2 {x4, x5} ⇨ c2 = {7,5 3,5}
    - Vuelta al paso 2
      - Reasignar las filas. Calcular las distancias a los nuevos centroides

Clustering No Jerárquico: **k-means**

- Ejemplo
  - Paso 4
    - Repetir desde el paso 2 hasta que no se produzca ningún cambio de grupo, es decir, los conjuntos no varíen

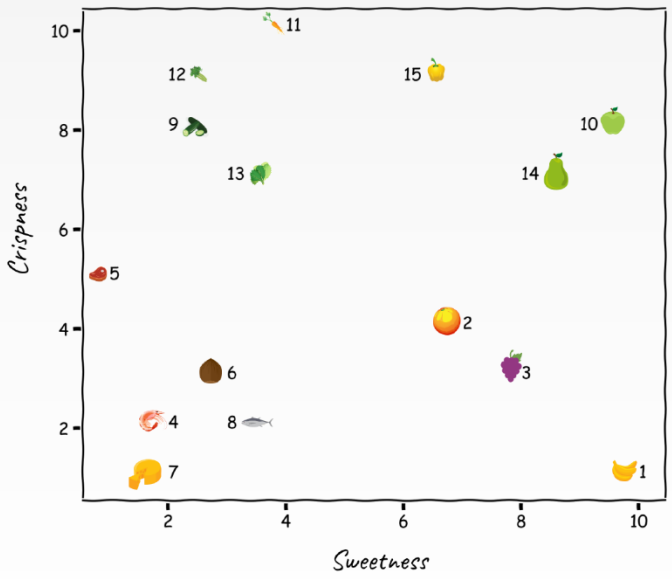


Clustering No Jerárquico: **k-means**

- El algoritmo de Lloyd no asegura encontrar la solución óptima global, sino solamente una local
  - Es muy sensible al conjunto inicial de centroides, que depende del valor de *k*
  - Hay que seleccionar bien los centroides iniciales y repetir varias veces, con distintos centroides y distintos *k*
  - Comparar las soluciones obtenidas y quedarse con aquella que minimice el error cometido
- El valor de *k* se suele determinar heurísticamente
  - Si *k* se elige muy pequeño puede dar lugar a agrupaciones heterogéneas
  - Si *k* se elige muy grande, los grupos serán demasiado específicos para ser útiles

Clustering No Jerárquico: **k-means**

- Ejercicio
  - Dado el siguiente conjunto de datos utilizar el algoritmo *k-means* para obtener los distintos grupos en los que pueden agruparse. La primera tarea, y la más crucial, es elegir el número correcto de *clusters*. Se puede resolver probando diferentes valores. Sin embargo, como los datos son intuitivos se podrían considerar tres grupos distintos, uno formado por fruta, otro por verdura y un tercero por proteínas.



Fuente: *High School of Digital Culture - ITMO University*

No.	Product	Sweetness	Crispness
1	banana	10	1
2	orange	7	4
3	grapes	8	3
4	shrimp	2	2
5	bacon	1	5
6	nuts	3	3
7	cheese	2	1
8	fish	3	2
9	cucumber	2	8
10	apple	9	8
11	carrot	4	10
12	celery	2	9
13	lettuce	3	7
14	pear	8	7
15	bell pepper	6	9

Clustering No Jerárquico: **k-means**

- Función en R
  - **kmeans()**
    - myclusters <- kmeans(mydata, centers = k)
      - myclusters**$cluster**
        - Vector que indica el número de clúster, o grupo, para cada observación
      - myclusters**$center**
        - Matriz que muestra los valores medios de las características de cada centro
      - myclusters**$size**
        - Número de observaciones asignadas a cada grupo

Modelos de Descripción: **Clustering**

- Características

Ventajas	Inconvenientes
Técnica muy intuitiva y consistente con la percepción humana	Costoso computacionalmente si hay muchos datos, tanto en número de elementos como de dimensiones
Es bastante eficiente y funciona bien al dividir los datos en grupos útiles	No son capaces de determinar por sí mismos el número óptimo de grupos a generar
Es muy flexible y puede adaptarse para abordar casi todas sus deficiencias con simples ajustes	Es necesario fijar el número de grupos de antemano o bien definir algún criterio de parada en el proceso de construcción jerárquica
Utiliza principios sencillos para identificar los grupos que pueden explicarse en términos no estadísticos	El algoritmo es sensible a los valores atípicos y además funciona peor que los métodos basados en densidad

Modelos de Descripción: **Clustering**

- Pasos típicos en un análisis completo de *clustering*

1. Elegir los atributos apropiados
2. Escalar los datos
3. Detección de valores atípicos
4. Calcular distancias
5. Seleccionar el algoritmo de agrupación
6. Obtener una o más soluciones de agrupamiento
7. Determinar el número final de grupos
8. Obtener una solución final
9. Visualización de los datos
10. Interpretación de los grupos
11. Validar los resultados

Ejercicio 1 – Teen Market Segments

- Objetivo
  - Identificar segmentos de mercado en adolescentes que comparten gustos similares, para que los clientes puedan evitar dirigir anuncios a adolescentes sin interés en el producto que se vende
  - Paso 1: Cargar datos
    - Archivo Social Networking Service *'snsdata.csv'*
  - Paso 2: Exploración y preparación de los datos
    - str(), summary(), table()
    - Tratamiento NAs. Estandarizar: scale()
  - Paso 3: Aplicar algoritmo de clustering
    - clusters <- kmeans(data, centers = k)
  - Paso 4: Evaluar el rendimiento
    - Número de ejemplos que caen en cada uno de los grupos,…
  - Paso 5: Mejora del rendimiento
    - Probar diferentes valores de *k*

Ejercicio 1 – Teen Market Segments

- Paso 1: Cargar datos
  - teens <- read.csv("snsdata.csv")

- Paso 2: Exploración y preparación de los datos

```
> str(teens)
'data.frame':   30000 obs. of  40 variables:
 $ gradyear    : int  2006 2006 2006 2006 2006 2006 2006 2006 2006 2006 ...
 $ gender      : chr  "M" "F" "M" "F" ...
 $ age         : num  19 18.8 18.3 18.9 19 ...
 $ friends     : int  7 0 69 0 10 142 72 17 52 39 ...
 $ basketball  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ football    : int  0 1 1 0 0 0 0 0 0 0 ...
 $ soccer      : int  0 0 0 0 0 0 0 0 0 0 ...
 $ softball    : int  0 0 0 0 0 0 0 1 0 0 ...
 $ volleyball  : int  0 0 0 0 0 0 0 0 0 0 ...
                           ...
```

Ejercicio 1 – Teen Market Segments

- Paso 2: Exploración y preparación de los datos

```
> table(teens$gender, useNA = "ifany")

    F     M  <NA>
22054  5222  2724

> summary(teens$age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  3.086  16.312  17.287  17.994  18.259 106.927    5086

> teens$age <- ifelse(teens$age >= 13 & teens$age < 20, teens$age, NA)

> summary(teens$age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  13.03   16.30   17.27   17.25   18.22   20.00    5523
```

Ejercicio 1 – Teen Market Segments

- Paso 2: Exploración y preparación de los datos

```
> summary(teens$age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.  NA's
  13.03   16.30   17.27   17.25   18.22   20.00  5523

> aggregate(data = teens, age ~ gradyear, mean, na.rm = TRUE)
  gradyear      age
1     2006 18.65586
2     2007 17.70617
3     2008 16.76770
4     2009 15.81957

> #Se eliminan los NA sustituyéndolos por la edad media del año de graduación
> summary(teens$age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  13.03   16.05   17.05   17.10   18.00   20.00
```

Ejercicio 1 – Teen Market Segments

- Paso 3: Aplicar el algoritmo de clustering

```
#PASO 3

#Se Consideran sólo las 36 características que representan el número
#de veces que los distintos intereses aparecen en los perfiles

interests <- teens[5:40]

#Estandarización de los datos

interests_norm <- as.data.frame(scale(interests))

#Modelo k-means. k=5

teen_clusters <- kmeans(interests_norm, 5)
```

Ejercicio 1 – Teen Market Segments

- Paso 4: Evaluar el rendimiento

```
> teen_clusters$size
[1]  1050   551  5235 22579   585

> #coordenandas de los centroides
> teen_clusters$centers
  basketball   football      soccer    softball  volleyball
1  0.3447570  0.3741235  0.12365965  0.04086561  0.09630328
2  0.7729004  0.2289524  0.02029233  5.86679413  0.72356065
3  0.2526940  0.2743935  0.21259584 -0.06039772  0.30527363
4 -0.1199329 -0.1235496 -0.06257103 -0.13296349 -0.09542407
5  1.0209366  1.4259732  0.27150154  0.07325148  0.09687568
```

Ejercicio 2 – Pediatric Trauma

- Paso 1: Cargar datos
  - Archivo *'trauma_data.csv'*
    - Datos sobre la utilización de servicios por niños expuestos a traumas en los EE.UU. Examina las asociaciones entre la psicopatología post-traumática y la utilización de servicios por parte de niños expuestos a un trauma concreto
    - Tipos de trauma
      - **dvexp**: exposición a la violencia doméstica o a la violencia de género
      - **neglect**: abandono
      - **physabuse**: abuso físico
      - **psychabuse**: abuso psicológico o emocional
      - **sexabuse**: abuso sexual

Ejercicio 2 – Pediatric Trauma

- Paso 2: Exploración y preparación de los datos

```
> head(trauma, n = 10)
   id sex age ses  race traumatype ptsd dissoc service
1   1   1   6   0 black   sexabuse    1      1      17
2   2   1  14   0 black   sexabuse    0      0      12
3   3   0   6   0 black   sexabuse    0      1       9
4   4   0  11   0 black   sexabuse    0      1      11
5   5   1   7   0 black   sexabuse    1      1      15
6   6   0   9   0 black   sexabuse    1      0       6
7   7   0  12   0 black   sexabuse    1      1       9
8   8   1   9   0 black   sexabuse    0      1      10
9   9   0   9   1 black   sexabuse    1      1      11
10 10   1  13   0 black   sexabuse    1      0      13
```

- **ses**: el niño fue expuesto a un trauma de maltrato infantil - (1 = yes, 0 = no)
- **ptsd**: el niño tiene actualmente un trastorno de estrés postraumático. (1 = yes, 0 = no)
- **dissoc**: el niño tiene actualmente un trastorno disociativo (1 = yes, 0 = no)
- **service**: número de servicios que el niño ha utilizado en los últimos 6 meses, como atención primaria, sala de emergencias, psiquiatra ambulatorio, tratamiento de acogida, centro de detención o cárcel, oficial de libertad condicional…

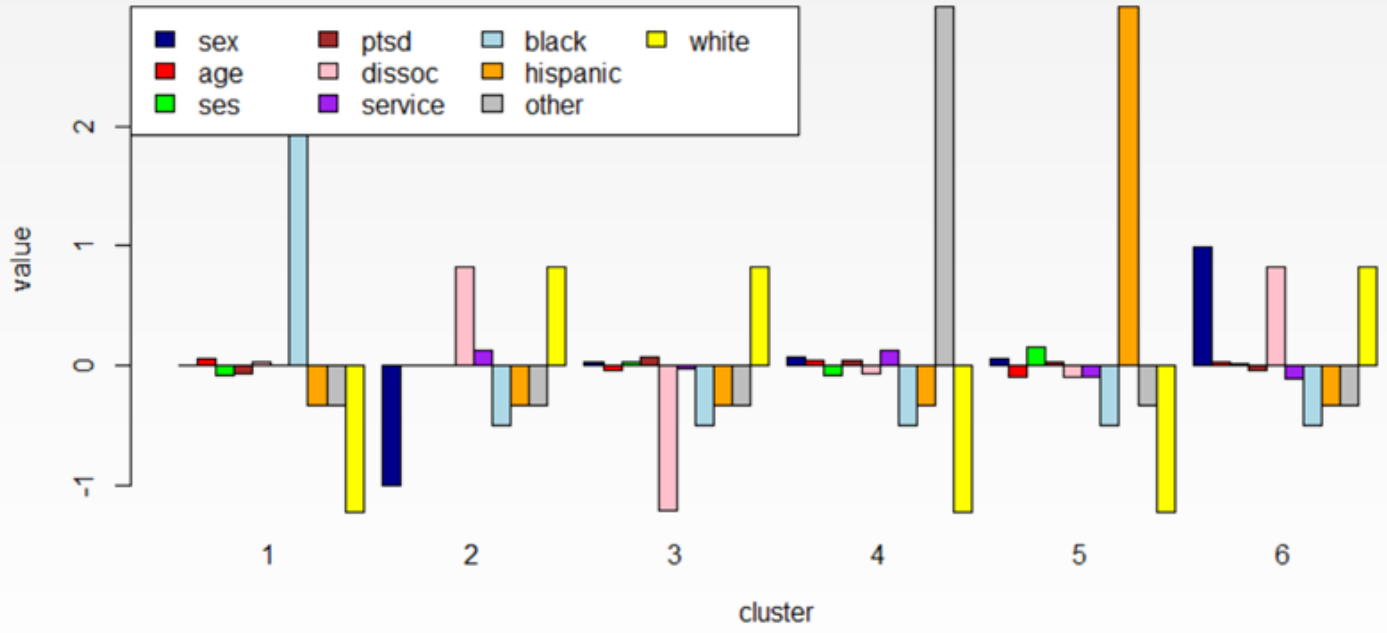
Ejercicio 2 – Pediatric Trauma

- Paso 3: Entrenar el modelo con los datos

```
> trauma_clusters<-kmeans(datos_norm, 6)

> myColors <- c("darkblue","red","green","brown","pink","purple",
                "lightblue","orange","grey","yellow")
> barplot(t(trauma_clusters$centers), beside = TRUE, xlab="cluster",
                ylab="value", col = myColors)
> legend("topleft", ncol=4, legend = c("sex", "age", "ses", "ptsd",
  "dissoc", "service", "black","hispanic",  "other", "white"), fill = myColors)
```

Ejercicio 2 – Pediatric Trauma

- Paso 3: Entrenar el modelo con los datos



Ejercicio 2 – Pediatric Trauma

- Paso 4: Evaluar el rendimiento del modelo

```
> trauma_clusters$centers
           sex          age          ses         ptsd      dissoc
1 -0.001999144  0.061154336 -0.091055799 -0.077094361  0.02446247
2 -1.011566709 -0.006361734 -0.000275301  0.002214351  0.81949287
3  0.026286613 -0.043755817  0.029890657  0.064206246 -1.21904661
4  0.067970886  0.046116384 -0.078047828  0.044053921 -0.07746450
5  0.047979449 -0.104263129  0.156095655  0.022026960 -0.09784989
6  0.987576985  0.028799955  0.019511957 -0.038046568  0.81949287
       service      black   hispanic      other      white
1  0.001308569  1.9989997 -0.3331666 -0.3331666 -1.2241323
2  0.126332303 -0.4997499 -0.3331666 -0.3331666  0.8160882
3 -0.030083167 -0.4997499 -0.3331666 -0.3331666  0.8160882
4  0.128894052 -0.4997499 -0.3331666  2.9984996 -1.2241323
5 -0.103376956 -0.4997499  2.9984996 -0.3331666 -1.2241323
6 -0.111481162 -0.4997499 -0.3331666 -0.3331666  0.8160882
```

Ejercicio 2 – Pediatric Trauma

- Paso 4: Evaluar el rendimiento del modelo
  - Comparación con las etiquetas reales (tipo de trauma)

```
> trauma$clusters<-trauma_clusters$cluster
> table(trauma$clusters, trauma$traumatype)

  dvexp neglect physabuse psychabuse sexabuse
1     0       0       100          0      100
2    10     118         0         61        0
3    23     133         0         79        0
4   100       0         0          0        0
5   100       0         0          0        0
6    17      99         0         60        0
```

PCA
Principal Component Analysis

PCA: Análisis de Componentes Principales

- Técnica Estadística de Reducción de la Dimensionalidad
  - Transformación ortogonal que reduce el número de dimensiones del conjunto de datos
    - Es una de las técnicas más conocida y utilizada en todas las áreas científicas, para la extracción de características y reducción de la dimensión por transformación lineal
    - Se basa en el supuesto de que la mayor parte de la información de un conjunto de datos puede ser explicada por un número menor de variables o atributos
    - Sólo tiene sentido si existen correlaciones entre las variables, ya que esto es indicativo de que existe información redundante y, por lo tanto, un número menor de variables serán capaces de explicar gran parte de la variabilidad total del conjunto de datos
    - Puede aplicarse a cualquier tipo de datos, aunque está especialmente indicado para datos normales (Gaussianos)
    - Sólo datos cuantitativos o numéricos. Los datos categóricos deben ser transformados

PCA: Análisis de Componentes Principales

- Objetivo
  - El propósito suele ser visualizar datos para tratar de detectar patrones
  - Proporcionar una reducción de la dimensión en los datos facilitando así la tarea de clasificación
  - El análisis normalmente sólo transforma los datos y no les añade nada
- Extracción de Características
  - Obtener un menor número de atributos que representan globalmente la información presente en el conjunto de datos original
    - Explotar la correlación inherente en los datos
    - Seleccionar los factores (CP - Componentes Principales) según la proporción de la varianza que preservan

PCA: Análisis de Componentes Principales

- Algoritmo

1. Sustraer la media de cada atributo o variable
2. Calcular la matriz de covarianza
3. Calcular los valores y vectores propios de la matriz de covarianza
4. Proyección de los datos sobre los autovectores ordenados de mayor a menor valor propio

$$A_{rxn} = M_{rxp} \cdot X_{pxn}$$

- $X = Matriz\ de\ datos$
- $M = Matriz\ de\ proyección\ lineal$

PCA: Análisis de Componentes Principales

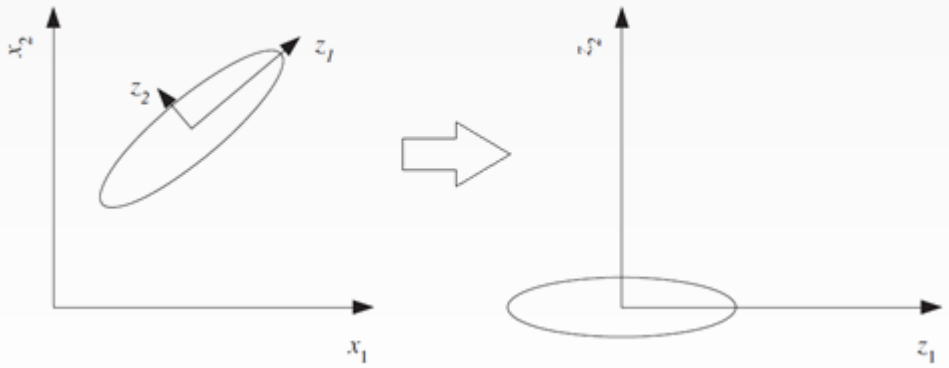
- Conceptos previos
  - Media: $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$
  - Varianza: $Var(x) = E[(x - \bar{x})^2]$
  - Covarianza: $Cov(x, y) = E[(x - \bar{x})(y - \bar{y})]$
  - Valores y vectores propios: $A \cdot v = \lambda \cdot v$
    - $\lambda\ es\ un\ valor\ propio\ de\ A \Leftrightarrow det(A - \lambda I) = 0$
      - Los valores propios representan las varianzas de las CP
      - Cada CP es un autovector o vector propio

PCA: Análisis de Componentes Principales

- Algoritmo
  - Convierte un conjunto de observaciones correlacionadas, en un conjunto de variables sin correlación lineal, llamadas ‘Componentes Principales’
  - Encuentra la proyección que minimiza el error cuadrático de los datos reconstruidos, o lo que es lo mismo, intenta maximizar la varianza en el espacio proyectado
  - Estos ejes de proyección se corresponden con los primeros vectores propios de mayor valor propio, asociado a la matriz de covarianza de los datos centrados
  - La selección de componentes se realiza de manera que el primero recoja la mayor proporción posible de la variabilidad original, el segundo componente debe recoger la máxima variabilidad posible no recogida por el primero, y así sucesivamente
    - 1ª Componente: Combinación lineal de las variables originales con la mayor varianza
    - 2ª Componente: Combinación lineal de las variables originales con la segunda varianza más grande
    - …

PCA: Análisis de Componentes Principales

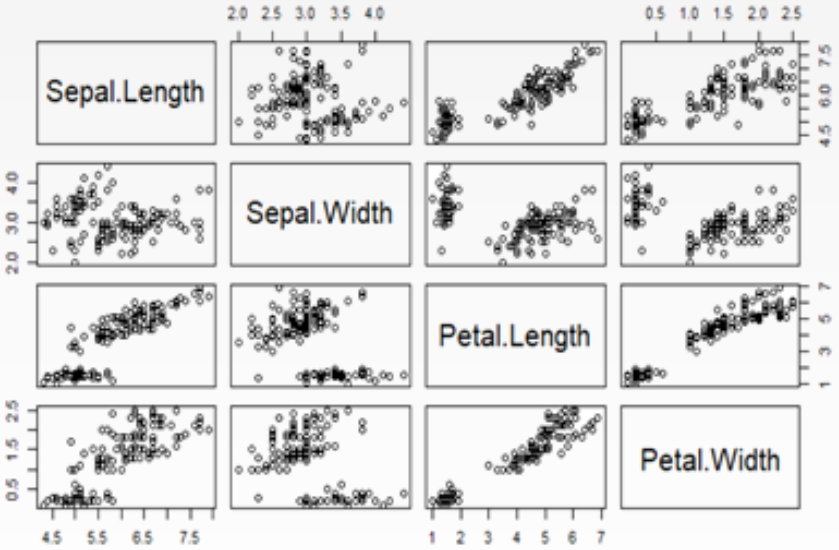
- De forma gráfica, PCA consiste en una rotación de los datos sobre los ejes, para conseguir que la mayor parte de los datos caigan sobre ellos
- Centra la muestra y luego rota los ejes para alinearse con las direcciones de mayor variación
- Si la varianza en el eje *z2* es demasiado pequeña, puede ser alineada y tenemos una reducción de la dimensión de dos a uno



PCA: Análisis de Componentes Principales

- Ejemplo en R
  - Data Iris

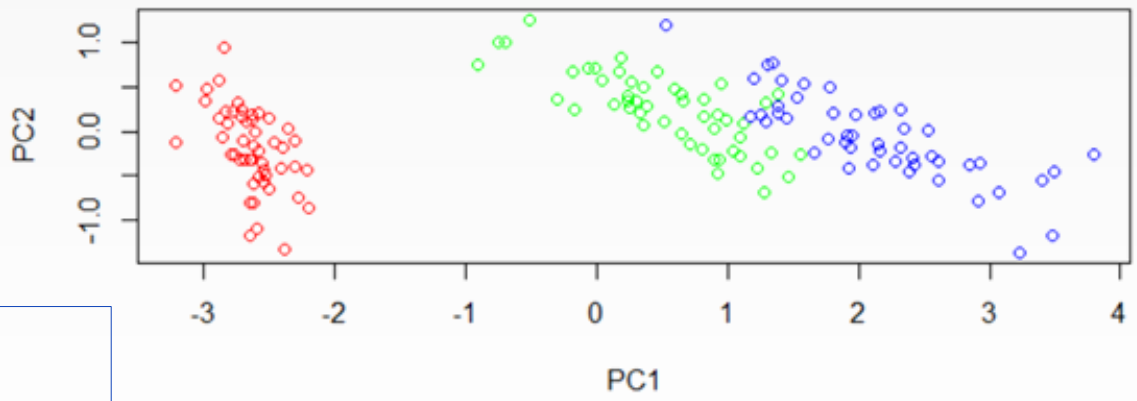
```
> #1- Media de cada columna
> m1 <- mean(iris[,1])
> m2 <- mean(iris[,2])
> m3 <- mean(iris[,3])
> m4 <- mean(iris[,4])
> #Sustracción de la media de cada aributo
> data <- cbind(iris[,1]-m1,iris[,2]-m2)
> data <- cbind(data,iris[,3]-m3)
> data <- cbind(data,iris[,4]-m4)
> cor(data)
           [,1]       [,2]       [,3]       [,4]
[1,]  1.0000000 -0.1175698  0.8717538  0.8179411
[2,] -0.1175698  1.0000000 -0.4284401 -0.3661259
[3,]  0.8717538 -0.4284401  1.0000000  0.9628654
[4,]  0.8179411 -0.3661259  0.9628654  1.0000000
```



PCA: Análisis de Componentes Principales

- Ejemplo en R
  - Data Iris

```
> #2
> #Cálculo de la matriz de covarianza
> covData <- cov(data)
> #3
> #Cálculo de los autovectores y autovalores de la matriz de covarianza
> pcas <- eigen(covData)
> #4
> datos_proyectados <- data%*%pcas$vectors
> 
```




```
colores <- function(vec){
  col <- rainbow(length(unique(vec)))
  return(col[as.numeric(as.factor(vec))])
}
plot(datos_proyectados, col = colores(iris$Species))
```

PCA: Análisis de Componentes Principales

- Ejemplo en R
  - Data Iris

```
> pca
Standard deviations (1, .., p=4):
[1] 2.0562689 0.4926162 0.2796596 0.1543862

Rotation (n x k) = (4 x 4):
                    PC1         PC2         PC3         PC4
Sepal.Length  0.36138659 -0.65658877  0.58202985  0.3154872
Sepal.Width  -0.08452251 -0.73016143 -0.59791083 -0.3197231
Petal.Length  0.85667061  0.17337266 -0.07623608 -0.4798390
Petal.Width   0.35828920  0.07548102 -0.54583143  0.7536574

> summary(pca)
Importance of components:
                          PC1     PC2    PC3     PC4
Standard deviation     2.0563 0.49262 0.2797 0.15439
Proportion of Variance 0.9246 0.05307 0.0171 0.00521
Cumulative Proportion  0.9246 0.97769 0.9948 1.00000
```

- La 1ª CP será la combinación lineal formada por las variables originales
  - ***0.36*Sepal.Length-0.08*Sepal.Width+0.85Petal.Length+0.35*Petal.Width***

PCA: Análisis de Componentes Principales

- Propiedades

1. La proyección de los datos sobre la primera componente principal (CP) tiene una varianza máxima de todas las posibles proyecciones unidimensionales
2. Las proyecciones en las CPs tienen media cero
3. Las proyecciones en las CP son mutuamente incorreladas, es decir, son ortogonales. Por lo tanto, los coeficientes de correlación entre las proyecciones de las observaciones sobre las componentes principales son cero
4. La varianza de la muestra de la proyección k-ésima es igual al k-ésimo valor propio de la matriz de covarianza de la muestra
5. Las componentes principales se clasifican de forma decreciente según sus valores propios, que son las varianzas en las direcciones de las componentes principales
6. Las primeras CPs minimizan el error cuadrático medio

PCA: Análisis de Componentes Principales

- Criterios de selección del número de Componentes Principales

1. Proporción total de la varianza explicada. Se fija un nivel mínimo y nos quedamos el número de componentes necesario para alcanzar ese valor mínimo
2. Establecer que una componente no pueda tener una desviación típica menor que una de las variables originales. Si los datos están escalados, la desviación típica debe ser mayor que uno
3. Otro criterio puede ser ver en qué momento se produce un descenso de la desviación estándar muy notable, y quedarse con las componentes previas

PCA: Análisis de Componentes Principales

- Resumen
  - Se trata de una transformación ortogonal que reduce la dimensión del conjunto de datos, mediante la proyección según la cual los datos queden mejor representados en términos de mínimos cuadrados
  - Convierte un conjunto de observaciones correlacionadas en un conjunto de variables sin correlación lineal llamadas componentes principales
  - Se realiza una descomposición de autovalores de la matriz de covarianza, de modo que la primera componente será una combinación lineal de las variables originales con mayor varianza, la segunda componente será una combinación lineal de las variables originales con la segunda varianza más grande, y así sucesivamente
  - Este proceso se puede ver geométricamente como un cambio de ejes en la proyección
  - Reducción de la dimensionalidad no es exactamente lo mismo que selección de características, ya que se usan combinaciones lineales de variables que reduzcan la dimensión de los datos, pero se consideran todas las variables

PCA: Análisis de Componentes Principales

- Función en R
  - **princomp()**
    - pca <- princomp(data)
  - **prcomp()**
    - A diferencia de *princomp()*, las varianzas se calculan con el divisor habitual N - 1
    - pca <- prcomp(data)
    - **predict(pca)**
      - Devuelve una matriz con los datos mapeados al nuevo espacio de las componentes principales

PCA: Análisis de Componentes Principales

- Características

Ventajas	Inconvenientes
Adecuado para datos de grandes dimensiones	Dado que cada CP es una combinación lineal de todas las variables originales, es difícil analizar la importancia de cada variable original en el nuevo espacio
Retiene aquellas características del conjunto de datos que contribuyen más a su varianza	
No es necesario conocer la distribución de probabilidad del conjunto de datos	
Método lineal. Es computacionalmente rápido	
Puede usarse como técnica de detección de atípicos multivariantes	Es muy sensible a valores atípicos

PCA: Ejercicio 1 – Occupancy Detection

- Paso 1: Cargar datos
  - Occupancy Detection Data Set - UCI Machine Learning Repository
  - Archivo *'occupancy_data.csv'* (Muestra de 8143 mediciones para saber si una habitación se encuentra vacía u ocupada en función de sus variables)
- Paso 2: Exploración y preparación de los datos
  - str(), summary()
  - cor(), pairs()
- Paso 3: Aplicar el algoritmo de PCA
  - pca <- princomp(data)
  - pca <- prcomp(data, center = TRUE, scale = TRUE)
- Paso 4: Evaluar el rendimiento
  - plot(PC1, PC2)
  - biplot(pca)

PCA: Ejercicio 1 – Occupancy Detection

- Paso 1: Cargar datos

```
> data <- read.csv2('ocupancy_data.csv', header = TRUE)
> head(data, n = 10)
```

	date	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
1	04/02/2015 17:51	23.180	27.2720	426.0	721.2500	0.004792988	1
2	04/02/2015 17:51	23.150	27.2675	429.5	714.0000	0.004783441	1
3	04/02/2015 17:53	23.150	27.2450	426.0	713.5000	0.004779464	1
4	04/02/2015 17:54	23.150	27.2000	426.0	708.2500	0.004771509	1
5	04/02/2015 17:55	23.100	27.2000	426.0	704.5000	0.004756993	1
6	04/02/2015 17:55	23.100	27.2000	419.0	701.0000	0.004756993	1
7	04/02/2015 17:57	23.100	27.2000	419.0	701.6667	0.004756993	1
8	04/02/2015 17:57	23.100	27.2000	419.0	699.0000	0.004756993	1
9	04/02/2015 17:58	23.100	27.2000	419.0	689.3333	0.004756993	1
10	04/02/2015 18:00	23.075	27.1750	419.0	688.0000	0.004745351	1

PCA: Ejercicio 1 – Occupancy Detection

- Paso 2: Exploración y preparación de los datos

```
> str(data)

'data.frame':   8143 obs. of  7 variables:
 $ date         : chr  "04/02/2015 17:51" "04/02/2015 17:51" "04/02/2015 17:53" ..
 $ Temperature  : num  23.2 23.1 23.1 23.1 23.1 ...
 $ Humidity     : num  27.3 27.3 27.2 27.2 27.2 ...
 $ Light        : num  426 430 426 426 426 ...
 $ CO2          : num  721 714 714 708 704 ...
 $ HumidityRatio: num  0.00479 0.00478 0.00478 0.00477 0.00476 ...
 $ Occupancy    : int  1 1 1 1 1 1 1 1 1 1 ...

> table(data$Occupancy)

   0    1
6414 1729
```

PCA: Ejercicio 1 – Occupancy Detection

- Paso 2: Exploración y preparación de los datos

```
> cov(data)
                Temperature      Humidity         Light           CO2 HumidityRatio
Temperature    1.034119e+00  -0.797364851 1.287212e+02 1.789634e+02  1.315394e-04
Humidity      -7.973649e-01  30.594295122 4.074960e+01 7.632740e+02  4.503209e-03
Light          1.287212e+02  40.749600817 3.792982e+04 4.064865e+04  3.824894e-02
CO2            1.789634e+02 763.274029176 4.064865e+04 9.879761e+04  1.678578e-01
HumidityRatio  1.315394e-04   0.004503209 3.824894e-02 1.678578e-01  7.264687e-07
> cor(data)
               Temperature    Humidity      Light       CO2 HumidityRatio
Temperature      1.0000000 -0.14175931 0.64994184 0.5598938     0.1517616
Humidity        -0.1417593  1.00000000 0.03782794 0.4390228     0.9551981
Light            0.6499418  0.03782794 1.00000000 0.6640221     0.2304202
CO2              0.5598938  0.43902276 0.66402206 1.0000000     0.6265559
HumidityRatio    0.1517616  0.95519808 0.23042021 0.6265559     1.0000000
> cov(scale(data))
               Temperature    Humidity      Light       CO2 HumidityRatio
Temperature      1.0000000 -0.14175931 0.64994184 0.5598938     0.1517616
Humidity        -0.1417593  1.00000000 0.03782794 0.4390228     0.9551981
Light            0.6499418  0.03782794 1.00000000 0.6640221     0.2304202
CO2              0.5598938  0.43902276 0.66402206 1.0000000     0.6265559
HumidityRatio    0.1517616  0.95519808 0.23042021 0.6265559     1.0000000
```

PCA: Ejercicio 1 – Occupancy Detection

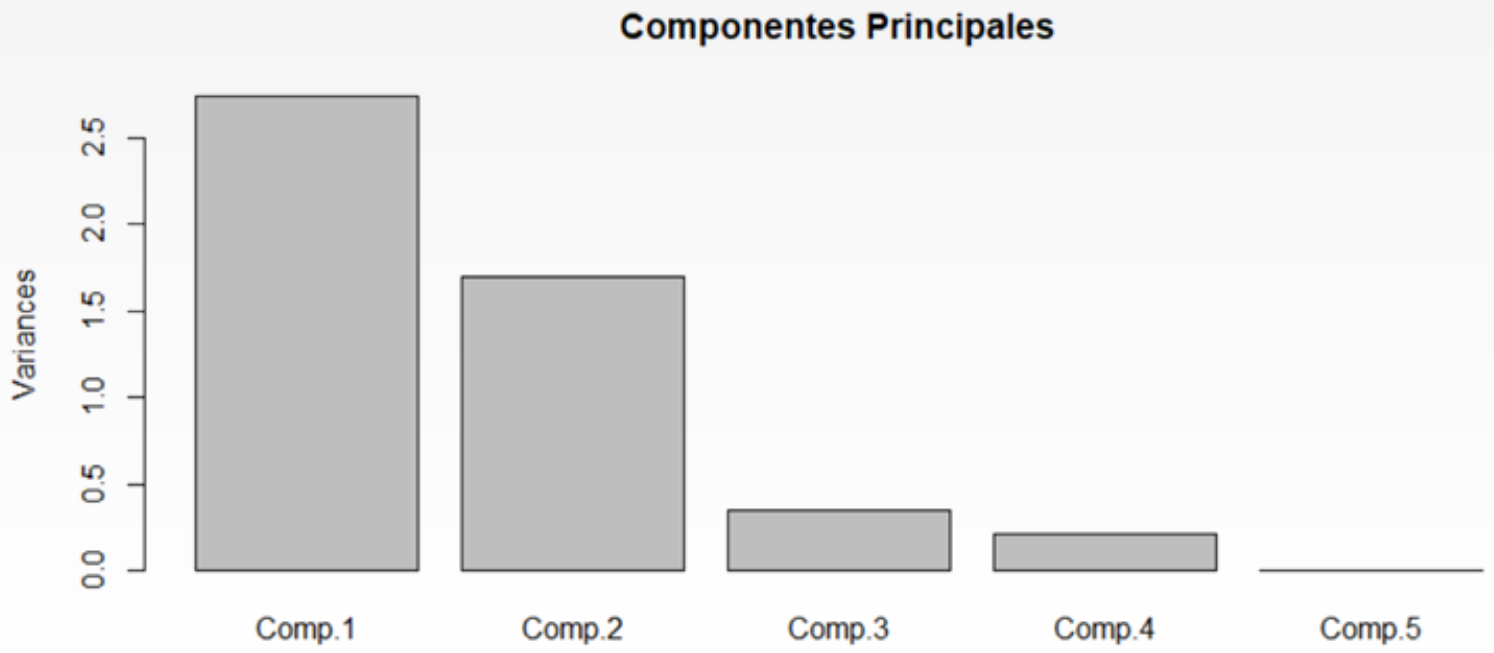
- Paso 3: Aplicar el algoritmo de PCA

```
> cp <- princomp(data, cor=TRUE) #princomp(scale(data))
> summary(cp)
Importance of components:
                          Comp.1    Comp.2     Comp.3     Comp.4       Comp.5
Standard deviation     1.6542442 1.3036375 0.59061773 0.46299795 0.0284474830
Proportion of Variance 0.5473047 0.3398941 0.06976586 0.04287342 0.0001618519
Cumulative Proportion  0.5473047 0.8871989 0.95696473 0.99983815 1.0000000000
> cp$sdev
    Comp.1     Comp.2     Comp.3     Comp.4     Comp.5
1.65424415 1.30363746 0.59061773 0.46299795 0.02844748
> cp$loadings

Loadings:
             Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
Temperature   0.344  0.536  0.713  0.225  0.187
Humidity      0.396 -0.574         0.227  0.680
Light         0.414  0.445 -0.665  0.433
CO2           0.550  0.120 -0.111 -0.817
HumidityRatio 0.501 -0.414  0.190  0.205 -0.707
```

PCA: Ejercicio 1 – Occupancy Detection

- Paso 3: Aplicar el algoritmo de PCA

```
> plot(cp)
```



PCA: Ejercicio 1 – Occupancy Detection

- Paso 3: Aplicar el algoritmo de PCA

```
> cp$scores
              Comp.1        Comp.2        Comp.3        Comp.4        Comp.5
 [1,]  2.3758098742  1.481548e+00  9.132344e-01  1.2383709564 -1.228806e-01
 [2,]  2.3544846332  1.476060e+00  8.806723e-01  1.2558748447 -1.225875e-01
 [3,]  2.3422183947  1.472145e+00  8.919613e-01  1.2475087240 -1.217927e-01
 [4,]  2.3251336992  1.478671e+00  8.921207e-01  1.2573977232 -1.216053e-01
 [5,]  2.2931278569  1.457935e+00  8.551390e-01  1.2525706169 -1.193810e-01
 [6,]  2.2721159492  1.440616e+00  8.802928e-01  1.2461011204 -1.192756e-01
 [7,]  2.2732827053  1.440870e+00  8.800575e-01  1.2443676162 -1.191640e-01
 [8,]  2.2686156813  1.439851e+00  8.809988e-01  1.2513016326 -1.196105e-01
 [9,]  2.2516977198  1.436157e+00  8.844108e-01  1.2764374416 -1.212289e-01
[10,]  2.2322766881  1.430719e+00  8.647957e-01  1.2705337601 -1.194630e-01
[11,]  2.2318395701  1.436309e+00  8.630651e-01  1.2625979154 -1.185109e-01
[12,]  2.2370996449  1.455728e+00  8.800712e-01  1.2637563462 -1.186265e-01
[13,]  2.2356502838  1.440240e+00  8.852193e-01  1.2888231572 -1.214311e-01
[14,]  2.2143276246  1.425589e+00  8.448916e-01  1.2624687375 -1.175767e-01
[15,]  2.1819581607  1.410382e+00  8.062071e-01  1.2497358889 -1.145040e-01
                                    ...
```

PCA: Ejercicio 1 – Occupancy Detection

- Paso 3: Aplicar el algoritmo de PCA

```
> pca <- prcomp(data, center = TRUE, scale. = TRUE)
> pca
Standard deviations (1, .., p=5):
[1] 1.65424415 1.30363746 0.59061773 0.46299795 0.02844748

Rotation (n x k) = (5 x 5):
                    PC1        PC2          PC3        PC4          PC5
Temperature   -0.3438562 -0.5358643  0.713373643  0.2253825  0.18685058
Humidity      -0.3956638  0.5741108 -0.009248538  0.2269663  0.67988804
Light         -0.4141489 -0.4446120 -0.665423661  0.4331769 -0.01923539
CO2           -0.5500699 -0.1201057 -0.110938448 -0.8172646  0.05262155
HumidityRatio -0.5011157  0.4136920  0.189516549  0.2052448 -0.70689468
> predict(pca)
                  PC1           PC2           PC3           PC4           PC5
   [1,] -2.3756639892 -1.481457e+00  9.131783e-01  1.238295e+00 -1.228730e-01
   [2,] -2.3543400577 -1.475970e+00  8.806182e-01  1.255798e+00 -1.225799e-01
   [3,] -2.3420745724 -1.472054e+00  8.919065e-01  1.247432e+00 -1.217852e-01
   [4,] -2.3249909260 -1.478580e+00  8.920659e-01  1.257321e+00 -1.215979e-01
   [5,] -2.2929870489 -1.457845e+00  8.550865e-01  1.252494e+00 -1.193737e-01
                                  ...
```

PCA: Ejercicio 2 – Wisconsin Breast Cancer Data

- Paso 1: Cargar datos
  - Archivo 'wisc_bc_data.csv' *(Diagnóstico de cáncer de mama)*
- Paso 2: Exploración y preparación de los datos
  - str(), summary()
  - cor(), pairs()
- Paso 3: Aplicar el algoritmo de PCA
  - cpa <- princomp(data)
  - cpa <- prcomp(data, center = TRUE, scale = TRUE)
- Paso 4: Evaluar el rendimiento
  - plot(PC1, PC2)
  - biplot(pca)

PCA: Ejercicio 2 – Wisconsin Breast Cancer Data

- Paso 1: Cargar datos

```
#PASO 1
wbcd <- read.csv2("wisc_bc_data.csv", header=TRUE, stringsAsFactors=FALSE)
head(wbcd)
```

- Paso 2: Exploración y preparación de los datos

```
#PASO 2
summary(wbcd)
str(wbcd)
table(wbcd$diagnosis)
cor(wbcd[,-1])
wbcd_norm <- scale(wbcd[,-1])
```

```
> table(wbcd$diagnosis)

Benigno Maligno
    357     212
```

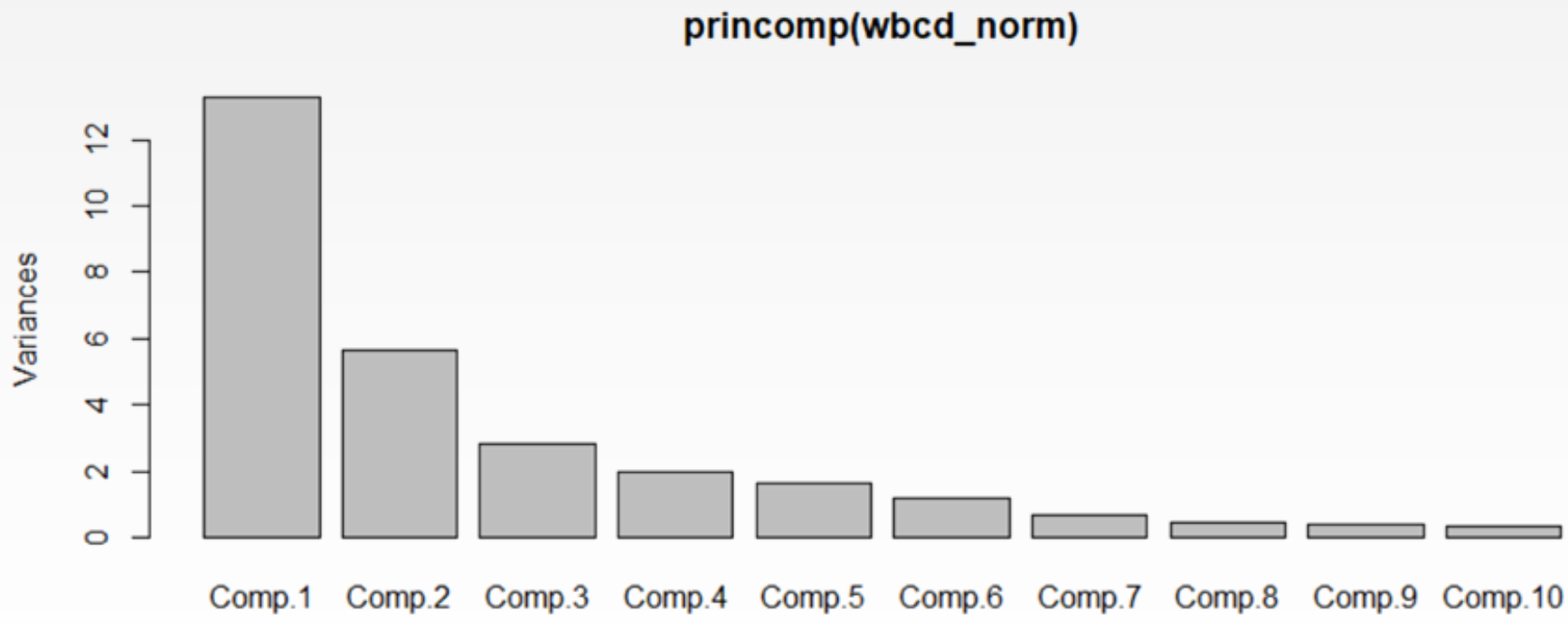
PCA: Ejercicio 2 – Wisconsin Breast Cancer Data

- Paso 3: Aplicar el algoritmo de PCA

```
> pca <- prcomp(wbcd_norm)
> summary(pca)

Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7     PC8    PC9    PC10
Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172 0.69037 0.6457 0.59219
Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251 0.01589 0.0139 0.01169
Cumulative Proportion  0.4427 0.6324 0.72636 0.79239 0.84734 0.88759 0.91010 0.92598 0.9399 0.95157

                          PC11    PC12    PC13    PC14    PC15    PC16    PC17    PC18    PC19    PC20
Standard deviation     0.5421 0.51104 0.49128 0.39624 0.30681 0.28260 0.24372 0.22939 0.22244 0.17652
Proportion of Variance 0.0098 0.00871 0.00805 0.00523 0.00314 0.00266 0.00198 0.00175 0.00165 0.00104
Cumulative Proportion  0.9614 0.97007 0.97812 0.98335 0.98649 0.98915 0.99113 0.99288 0.99453 0.99557

                          PC21    PC22    PC23   PC24    PC25    PC26    PC27    PC28    PC29    PC30
Standard deviation     0.1731 0.16565 0.15602 0.1344 0.12442 0.09043 0.08307 0.03987 0.02736 0.01153
Proportion of Variance 0.0010 0.00091 0.00081 0.0006 0.00052 0.00027 0.00023 0.00005 0.00002 0.00000
Cumulative Proportion  0.9966 0.99749 0.99830 0.9989 0.99942 0.99969 0.99992 0.99997 1.00000 1.00000
```

PCA: Ejercicio 2 – Wisconsin Breast Cancer Data

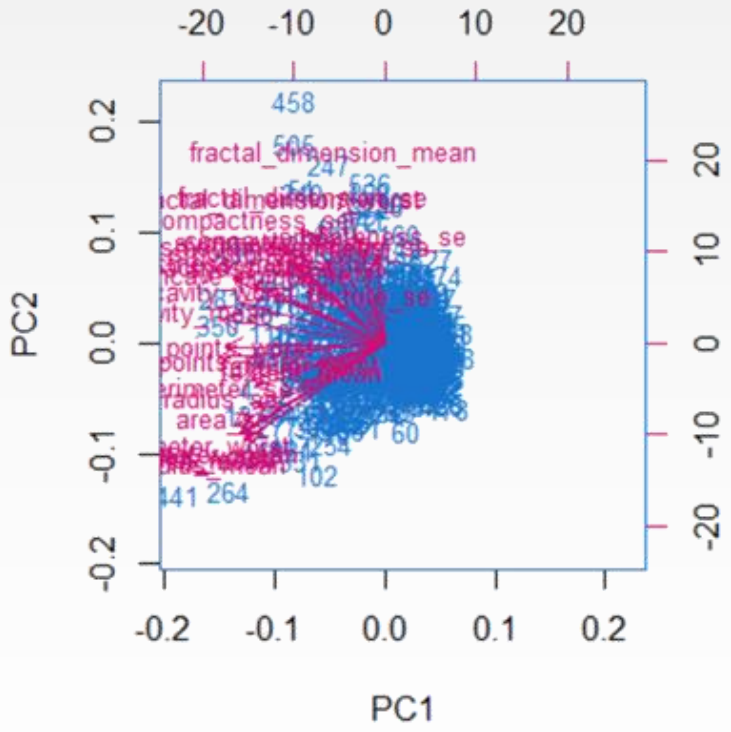
- Paso 3: Aplicar el algoritmo de PCA



PCA: Ejercicio 2 – Wisconsin Breast Cancer Data

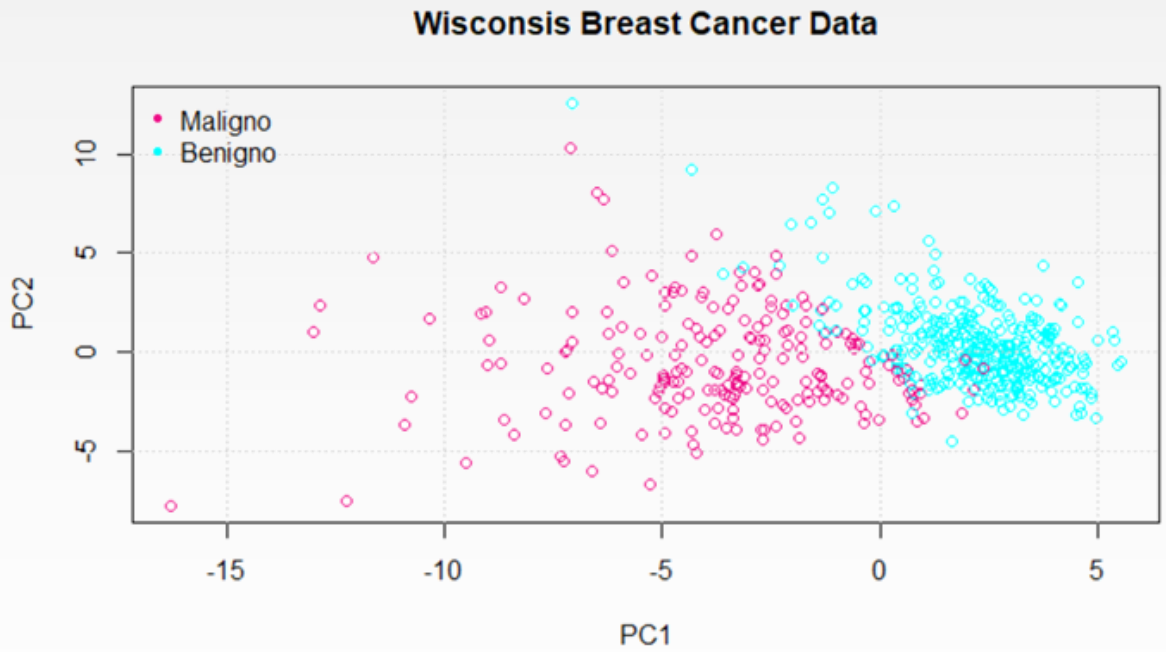
- Paso 4: Evaluar el rendimiento

```
biplot(pca, cex = 0.8, col =
  c("dodgerblue3", "deeppink3"))
```



PCA: Ejercicio 2 – Wisconsin Breast Cancer Data

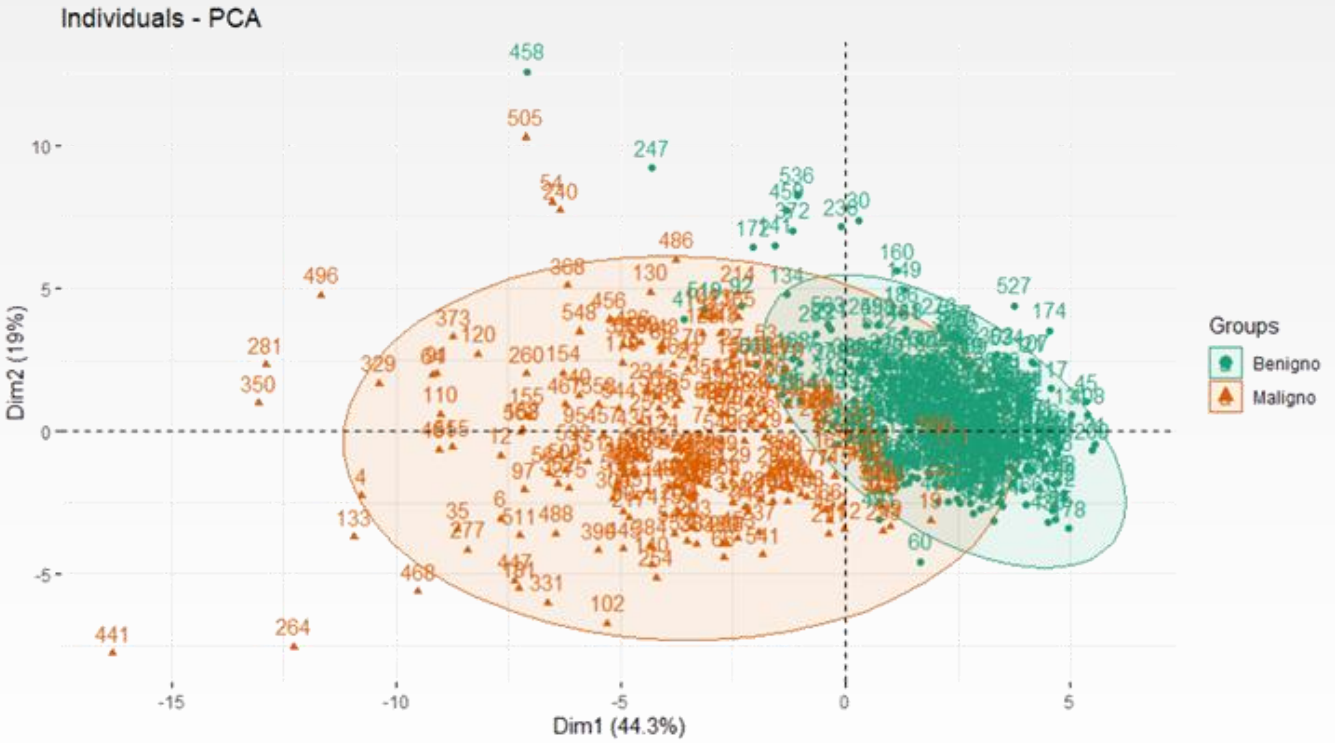
- Paso 4: Evaluar el rendimiento




```
color <- c('cyan', 'deeppink2')
colores <- color[as.numeric(as.factor(wbcd$diagnosis))]
plot(pca$x[,1:2], col = colores, main = "Wisconsis Breast Cancer Data")
legend("topleft",legend = c("Maligno","Benigno"),col=colores,bty='n',pch=20)
grid()
```

PCA: Ejercicio 2 – Wisconsin Breast Cancer Data

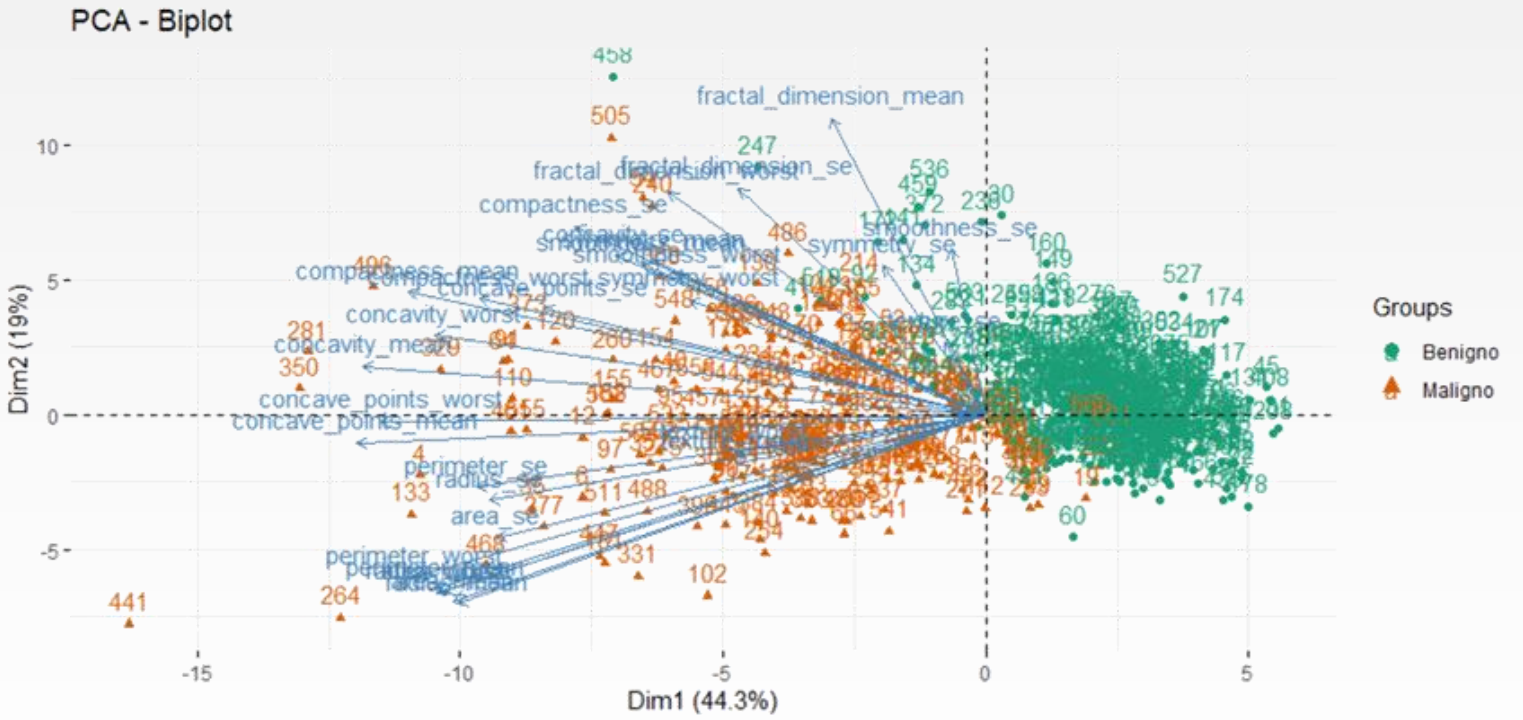
- Paso 4: Evaluar el rendimiento




```
library(factoextra)
fviz_pca_ind(pca, habillage=wbcd$diagnosis,addEllipses=TRUE,ellipse.level=0.95,palette = "Dark2")
```

PCA: Ejercicio 2 – Wisconsin Breast Cancer Data

- Paso 4: Evaluar el rendimiento




```
library(factoextra)
fviz_pca_biplot(pca, habillage=wbcd$diagnosis, palette = "Dark2")
```

PCA: Ejercicio Propuesto

- Cargar el dataset *HouseVotes84* del paquete '*mlbench*'
  - Los datos contienen los votos emitidos tanto de republicanos como de demócratas en 16 propuestas diferentes. Los tipos de votos son sí, no, y perdido/desconocido
  - La falta de datos aquí significa que alguien decidió activamente no votar
  - NAs: Un enfoque que podríamos tomar es traducir cada columna en tres columnas binarias que indican si un voto fue emitido como si, no, o no emitido. Otro enfoque sería considerar que si alguien se abstuvo de votar, entonces está con igual probabilidad de haber votado sí o no y traducir los datos que faltan en 0,5
  - Una pregunta interesante que se puede hacer es si hay diferencias en los patrones de votación entre republicanos y demócratas. Se espera eso, pero ¿podemos verlo en los datos?
  - Los datos no se ven muy diferentes entre los dos grupos, por lo que hay poca información en cada individuo. Hacer un PCA para corroborarlo

Continuará…