PECL4 - Fundamentos de la ciencia de datos

Marcos Barranquero Adrián Montesinos
Eduardo Graván

12 de noviembre de 2019

1. Introducción - Clasificación no supervisada

En esta práctica se estudia el método de clasificación k-means como algoritmo para realizar clasificación no supervisada. Partiendo de una muestra, y utilizando la media aritmética para calcular centroides de los datos, conseguimos clasificar los elementos de la muestra en varios conjuntos, en base al número de centroides utilizados.

2. Apartado 1 - Análisis de clasificación no supervisada

2.1. Introducción

En este apartado vemos el ejercicio de clasificación no supervisada realizados en clase, donde clasificamos un conjunto de datos de la forma (x,y) en dos clusters. Se tienen los siguientes datos:

Cuadro 1: Muestra datos

4	4
3	5
1	2
5	5
0	1
2	2
4	5
2	1

Para realizar el análisis, el primer paso es cargar los datos. No tenemos que importar ninguna librería ya que trabajamos con una librería base. Para cargar los datos, se han introducido directamente en la declaración de la matriz:

Tras esto, leemos el fichero de texto con la información de las calificaciones, y las insertamos en un *dataframe*:

```
> matriz_datos_iniciales<-matrix(c(4,4,
+                                  3,5,
+                                  1,2,
```

```
+                                    5,5,
+                                    0,1,
+                                    2,2,
+                                    4,5,
+                                    2,1),
+                                  # Filas, columnas
+                                  2,8)
> matriz_datos_iniciales<-t(matriz_datos_iniciales)
```

Ahora podemos establecer los centroides:

```
> centroides <- matrix(c(0,1,2,2),2,2)
> centroides <- t(centroides)
```

Realizamos la clasificación llamando al método kmeans. Con 4 iteraciones es suficiente para clasificar los datos:

```
> clasificacion_kmeans = kmeans(matriz_datos_iniciales, centroides, 4)
```

Si imprimimos, vemos una tabla cuya primera columna indica el centroide asociado al par de datos, mostrados en la segunda y tercera columna:

```
> matriz_datos_clasificados <- cbind(clasificacion_kmeans$cluster,
+                                    matriz_datos_iniciales)
```

Como queremos separar los datos en función del clúster al que pertenecen, creamos dos vectores y añadimos los datos en función del cluster:

```
> matriz_cluster1 <- subset(matriz_datos_clasificados,
+                           matriz_datos_clasificados[,1]==1)
> matriz_cluster2 <- subset(matriz_datos_clasificados,
+                           matriz_datos_clasificados[,1]==2)
> matriz_cluster1<- matriz_cluster1[,-1]
> matriz_cluster2<- matriz_cluster2[,-1]
> print(matriz_cluster1)

     [,1] [,2]
[1,]    1    2
[2,]    0    1
[3,]    2    2
[4,]    2    1

> print(matriz_cluster2)

     [,1] [,2]
[1,]    4    4
[2,]    3    5
[3,]    5    5
[4,]    4    5
```

3. Apartado 2 - Ejercicio propio

Para este apartado vamos a realizar una clasificación del color de una imagen con kmeans. Para ello, vamos a necesitar instalar las librerías *jpeg* y *ggplot2*.

Tras establecer el directorio de trabajo, leemos la imagen, obtenemos sus dimensiones, y la convertimos a una imagen RGB.

```
> library(jpeg)
> library(ggplot2)
> # Leemos imagen
> imagen <- readJPEG(normalizePath('amarillo.jpg'))
> # Obtenemos dimensión de la imagen
> dimensionesImg <- dim(imagen)
> # Asignamos colores RGB de la imagen
> imgRGB <- data.frame(
+   x = rep(1:dimensionesImg[2], each = dimensionesImg[1]),
+   y = rep(dimensionesImg[1]:1, dimensionesImg[2]),
+   R = as.vector(imagen[,,1]),
+   G = as.vector(imagen[,,2]),
+   B = as.vector(imagen[,,3])
+   )
```

Realizamos la clasificación supervisada con - por ejemplo - 8 clústers. En función del número de clusters utilizados, la imagen de salida tendrá más o menos canales de color.

```
> # Realizamos el clustering
> nClusters <- 8
> kMeans <- kmeans(imgRGB[, c("R", "G", "B")], centers = nClusters)
> nColores <- rgb(kMeans$centers[kMeans$cluster,])
```

Finalmente, renderizamos la imagen y la escribimos en disco: **Nota:** el renderizado toma algo menos de un minuto en mi ordenador. Tener en cuenta en caso de querer ejecutar el *rnw*. Aunque parezca que se queda colgado, al final consigue procesar la imagen.

```
> # Creamos plot
> show(ggplot(data = imgRGB, aes(x = x, y = y)) +
+   geom_point(colour = nColores) +
+   labs(title = paste("K-Means realizado con ", nClusters, " clusters. ")) +
+   xlab("x") +
+   ylab("y"))
> # Dumpeamos en una imagen
> ggsave("amarillo_renderizado.png")
```

Podemos comparar la imagen orginal con la imagen tras realizar la clasificación:

Figura 1: Imagen original

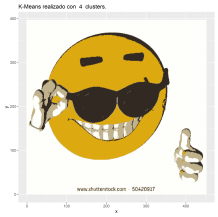

Figura 2: Imagen con colores clasificados en 4 clústeres.

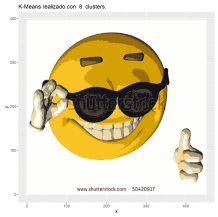

Figura 3: Imagen con colores clasificados en 8 clústeres.

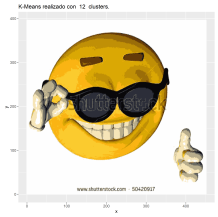

Figura 4: Imagen con colores clasificados en 12 clústeres.

4. Conclusiones

El lenguaje R trae herramientas potentes que permiten realizar análisis de forma sencilla y eficiente. Con ayuda de librerías externas, podemos aplicar estos análisis a casos pragmáticos tales como análisis de imágenes, como hemos mostrado en la práctica.