Máster Ciencia de Datos – *Data Science*

Minería de datos / Profesor colaborador: Raúl Montoliu Colás

Fernando Antonio Barbeiro Campos – fbarbeiro@uoc.edu

PEC 4 – Prueba de Evaluación Continua 4 – Modelos de Agregación

Entrega prevista: 25 de Abril de 2018

Índice General

Introducción

Competencias

Las competencias que se trabajan en esta prueba son:

- Uso y aplicación de las TIC en el ámbito académico y profesional.
- Capacidad para innovar y generar nuevas ideas.
- Capacidad para evaluar soluciones tecnológicas y elaborar propuestas de proyectos teniendo en cuenta los recursos, las alternativas disponibles y las condiciones de mercado.
- Conocer las tecnologías de comunicaciones actuales y emergentes así como saberlas aplicar convenientemente para diseñar y desarrollar soluciones basadas en sistemas y tecnologías de la información.
- Aplicación de las técnicas específicas de ingeniería del software en las diferentes etapas del ciclo de vida de un proyecto.
- Capacidad para aplicar las técnicas específicas de tratamiento, almacenamiento y administración de datos.
- Capacidad para proponer y evaluar diferentes alternativas tecnológicas para resolver un problema concreto.

Objetivos

La correcta asimilación del Módulo 5:
En esta PEC trabajaremos la generación e interpretación de un **modelo de agregación** basado en particiones de datos con la herramienta de prácticas. No perderemos de vista las fases de preparación de los datos y extracción inicial de conocimiento.

Ejercicio 1

Contextualizad los ejemplos de las siguientes preguntas respecto al proyecto que has definido en la PEC1. Si lo deseáis, podéis redefinir o ajustar el proyecto.

1.

- *¿Creéis que los métodos de agregación son el método más adecuado por conseguir algún de los objetivos que os habíais propuesto?*

- *Justificad la respuesta razonándola. Si lo deseáis, podéis redefinir o adecuar la propuesta de proyecto*

- *Proponed un posible ejemplo relacionado con vuestro proyecto.*

RESPUESTA:

La recapitulación para poner un contexto - mi proyecto era ***baggage-propensity***: predecir la propensión de los clientes a comprar equipaje adicional durante sus vuelos.

Absolutamente los métodos de agregación **no son los más adecuados** para elegir en el caso de *baggage-propensity*. La **justificativa** es que, conforme hemos comentado en la PEC3, tratase de un modelo de aprendizaje supervisado, esto es, conocemos a *priori* las categorías (*labels*). Mientras tanto, modelos de agregación (que dan como resultado modelos descriptivos) buscan obtener una primera aproximación con relación al dominio de la información, o sea, son modelos de aprendizaje no supervisados.

Un posible ejemplo de la aplicación de un modelo no supervisado de agregación, podría ser si quisiéramos desde la misma base de datos / DW que utilizamos para *baggage*, reunir informaciones de los clientes partiendo de un punto de relativo desconocimiento, el modelo de agregación encajaría perfectamente para la necesidad y el resultado nos daría N grupos que nos permitiría conocer mejor los clientes y quizás, de alguna manera, servir como input para el otro modelo supervisado que hemos definido en la PEC3. En la figura 1, traigo una visualización de un *k-means* para agregar los clientes.

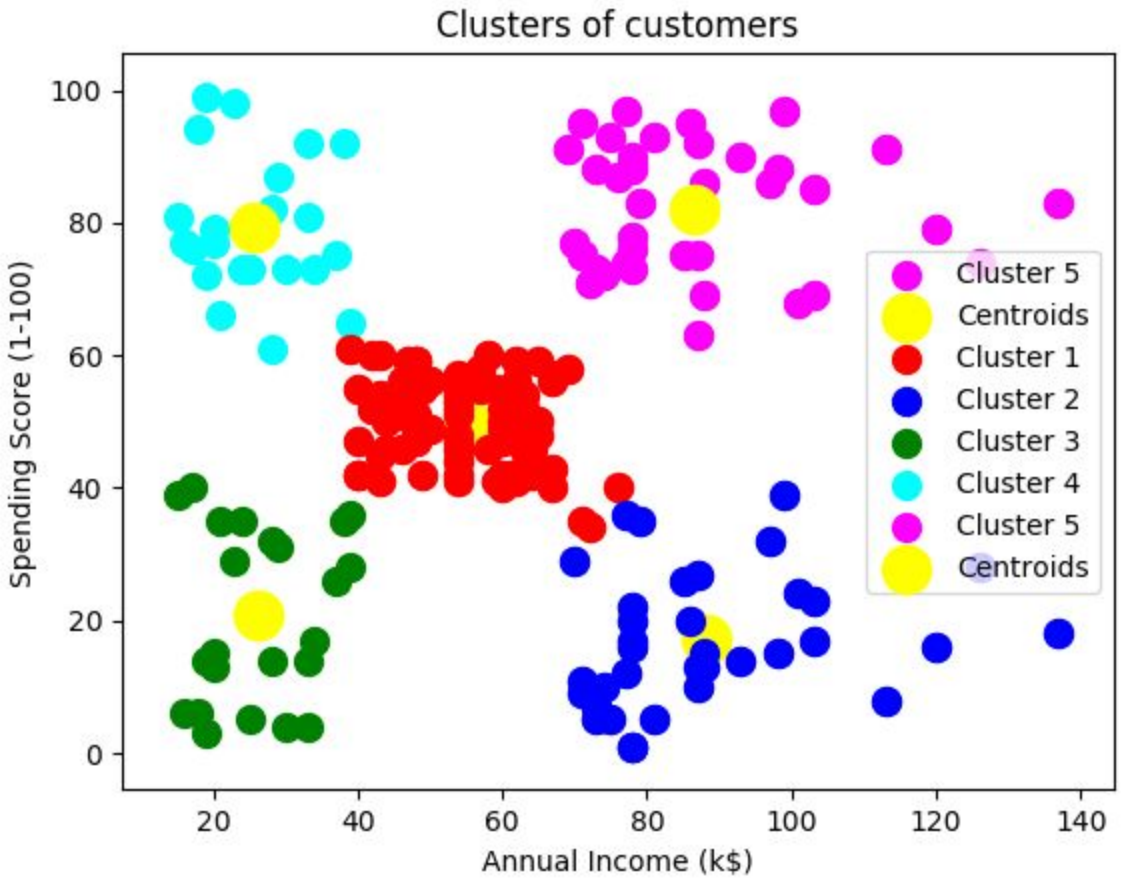

Figura 1: Clustering de clientes por apenas 2 atributos usando *k-means* [1].

Podemos ver que el muestreo de clientes se agrupan en 5 cluster distintos. Además por aquí queda claro el uso de centroides (se representa inicialmente como una *seed* como un punto inicial fijo y luego al transcurrir de las iteraciones se van agrupando y formando clusters alrededor - basado en la distancia - de tales centroides). Probablemente volveremos a *k-means* en los otros ejercicios, el intuito por aquí era solo proveer una idea

Ejercicio 2

2.

En este ejercicio vais a seguir los pasos del ciclo de vida de un proyecto de minería de datos para el caso de un algoritmo de agregación y más concretamente de un clúster con el algoritmo kmeans. Lo haréis con el archivo seguros2.csv. Que se encuentra en la wiki. Este archivo contiene un registro por cada incidente que gestiona una compañía de seguros. Estos incidentes están caracterizados por variables de negocio: coste, potencia, antigüedad del coche y sociodemográficas: sexo edad del asegurado.

- *Estudiar y entender los datos, por ejemplo: ¿Número de registros del archivo? ¿Distribuciones de valores para variables? ¿Hay campos mal informados o vacíos? Extraed información tangible.*
- *Preparar los datos. En este caso ya están en el formato correcto y no es necesario discretizar ni generar atributos nuevos. Obsérvese, pero que hay un archivo Seguros.csv. Este es el fichero maestro de donde se ha generado el 2. Tener en cuenta que se han preparado los datos para poder aplicar el algoritmo. Es importante que lo reviséis porque os dará pistas de cara al ejercicio 3.*
- *Instalar, si es necesario, el paquete stats R. Este paquete, documentado en la wiki contiene una implementación del algoritmo kmeans visto en el Módulo. Es una implementación sencilla para poder contrastar lo visto en la teoría. R contiene implementaciones de clúster muy potentes, pero mejor trabajar algo sencillo y comprensible. Con este paquete, generar un modelo de minería. Observar que el número de clústeres a generar lo marcamos nosotros. Hay que ir probando diferentes números de particiones hasta que nos encontramos satisfechos del resultado. Los criterios pueden ser la homogeneidad de las particiones, pero también la lectura de los clústeres resultantes y si son muy parecidos o diferentes ...*
- *Generar un modelo de agregación*
- *¿Cuál es la calidad del modelo?*
- *Dibujar gráficamente los clústeres*
- *En función del modelo. ¿Cuál es el conocimiento que extraemos?*

RESPUESTA:

a) El estudio de los datos

Primeros observamos el código y las salidas. Al final, los comentarios pertinentes a tal análisis.

```
# Loading the dataframe
df <- read.csv2("seguros2.csv")
head(df)
```

```
##   Numero.incidente Sexo Edad Anyos.Coche Caballos  Costos
## 1          5126419    1   46           6      110 2436.86
## 2          9915975    1   77           3      100 1322.65
## 3          1250208    0   24          10       50 6871.83
## 4          1332233    0   50          11       70 1652.94
## 5          2558883    1   40          15       50 1495.18
## 6          2862115    0   56           9      102 2018.52
```

```
summary(df)
```

```
##  Numero.incidente       Sexo             Edad        Anyos.Coche
##  Min.   :1001809   Min.   :0.0000   Min.   :18.00   Min.   : 0.000
##  1st Qu.:3369286   1st Qu.:0.0000   1st Qu.:30.00   1st Qu.: 5.000
##  Median :5337270   Median :1.0000   Median :43.00   Median :10.000
##  Mean   :5422967   Mean   :0.5442   Mean   :44.06   Mean   : 9.896
##  3rd Qu.:7412620   3rd Qu.:1.0000   3rd Qu.:57.00   3rd Qu.:15.000
##  Max.   :9971262   Max.   :1.0000   Max.   :77.00   Max.   :19.000
##
##     Caballos          Costos
##  Min.   : 30.00   1003.45:  1
##  1st Qu.: 49.00   1016.32:  1
##  Median : 58.00   1025.02:  1
##  Mean   : 79.68   1029.29:  1
##  3rd Qu.:100.00   1031.02:  1
##  Max.   :354.00   1035.85:  1
##                   (Other):492
```

```
nrow(df)
```

```
## [1] 498
```

```
ncol(df)
```

```
## [1] 6
```

```
colnames(df)
```

```
## [1] "Numero.incidente" "Sexo"             "Edad"
## [4] "Anyos.Coche"      "Caballos"         "Costos"
```

```
res <- sapply(df, class)
kable(data.frame(variables=names(res),clase=as.vector(res)))
```

variables	clase
Numero.incidente	integer
Sexo	integer
Edad	integer
Anyos.Coche	integer
Caballos	integer
Costos	factor

Hemos encontrado con un dataframe de 498 filas por 6 columnas. Los tipos de las columnas son, en general, *factor* y numéricas. Diferentemente del fichero de Seguros.csv, no hay problemas de *encoding* (con excepción del *header*) en el muestreo ni hay valores NA's, como muestra la salida del comando `summary`.

b) Preparar los datos

En mi opinión, los atributos de seguros2.csv están listos para ser trabajados por nuestros modelos. Sin embargo, si fuéramos mirar Seguros.csv, primeramente, haría cambios en los nombres de los atributos que llevan tilde:

```
colnames(df)[which(names(df) == "Fecha.adquisicin")] <- "FA"
colnames(df)[which(names(df) == "A.o.compra.Coche")] <- "ACC"
colnames(df)[which(names(df) == "A.os.Coche")] <- "AC"
colnames(df)
```

```
##  [1] "ID.Asegurado"     "Sexo"             "Fecha.nacimiento"
##  [4] "Edad"             "Obligatorio"      "Todo.Riesgo"
##  [7] "Cristales"        "Incendio"         "Uso"
## [10] "ACC"              "AC"               "Modelo"
## [13] "Cubicaje"         "caballos"         "Color"
## [16] "FA"               "Numero.incidente" "Costos"
```

También he notado problemas de *encoding* en una serie de atributos, arreglando los mismos:

```
summary(df$Todo.Riesgo)
```

```
##          No S\355
##     1   340   165
```

```
str(df$Todo.Riesgo)
```

```
##  Factor w/ 3 levels "","No","S\355": 3 3 2 2 2 2 2 2 2 2 ...
```

```
# Se podria remplazar por un for
df$Todo.Riesgo <- as.ordered(gsub("S\355", "Si", df$Todo.Riesgo))
df$Obligatorio <- as.ordered(gsub("S\355", "Si", df$Obligatorio))
df$Cristales <- as.ordered(gsub("S\355", "Si", df$Cristales))
df$Incendio <- as.ordered(gsub("S\355", "Si", df$Incendio))
```

Finalmente, he detectado una línea con una NA's y atributos que no aportan valor - eliminando:

```
df[df$Todo.Riesgo == "", ]
```

```
##     ID.Asegurado Sexo Fecha.nacimiento Edad Obligatorio Todo.Riesgo
## 287                                       NA
##     Cristales Incendio Uso ACC AC Modelo Cubicaje caballos Color FA
## 287                          NA NA     NA                NA
##     Numero.incidente Costos
## 287
```

```
# Observamos que habia un registro que además de no tener Riesgo, tenia NAs - eliminando
df <- df[rowSums(is.na(df)) == 0,]
str(df)
```

```
## 'data.frame':    505 obs. of  18 variables:
##  $ ID.Asegurado    : Factor w/ 499 levels "","101.351","102.938",..: 85 86
114 115 140 141 160 161 184 185 ...
##  $ Sexo            : Factor w/ 3 levels "","H","M": 2 2 3 3 2 3 2 3 3 3
...
##  $ Fecha.nacimiento: Factor w/ 493 levels "","01/01/1937",..: 337 253 238
375 457 448 309 364 4 116 ...
##  $ Edad            : int  46 77 24 50 40 56 22 54 64 27 ...
##  $ Obligatorio     : Ord.factor w/ 2 levels ""<"Si": 2 2 2 2 2 2 2 2 2 2
...
##  $ Todo.Riesgo     : Ord.factor w/ 3 levels ""<"No"<"Si": 3 3 2 2 2 2 2 2
2 2 ...
##  $ Cristales       : Ord.factor w/ 3 levels ""<"No"<"Si": 2 2 3 3 3 3 3 3
3 3 ...
##  $ Incendio        : Ord.factor w/ 3 levels ""<"No"<"Si": 2 2 3 3 3 3 3 3
3 3 ...
```

```
##  $ Uso             : Factor w/ 3 levels "","Industrial",..: 3 3 3 3 3 3 3
2 3 3 ...
##  $ ACC             : int  1996 1999 1991 1990 1986 1993 1990 1986 1990
1988 ...
##  $ AC              : int  6 3 10 11 15 9 12 16 12 14 ...
##  $ Modelo          : int  19 35 10 3 10 37 10 11 9 2 ...
##  $ Cubicaje        : Factor w/ 32 levels "","1000","1100",..: 9 22 3 5 3
19 3 18 2 3 ...
##  $ caballos        : int  110 100 50 70 50 102 50 40 49 60 ...
##  $ Color           : Factor w/ 8 levels "","AMARILLO",..: 8 8 4 4 8 4 8 4
4 4 ...
##  $ FA              : Factor w/ 483 levels "","01/01/1993",..: 188 46 54
330 281 328 406 452 255 103 ...
##  $ Numero.incidente: Factor w/ 500 levels "","1.001.809",..: 239 495 15 17
77 95 212 121 237 79 ...
##  $ Costos          : Factor w/ 506 levels "","1003.45","1016.32",..: 310
73 404 147 112 220 24 94 18 379 ...
```

Otra vez, no tuve que hacer cambios en seguros2.csv que es el objetivo del estudio aquí.

c) Generando el modelo de agregación

```
set.seed(20)
k <- kmeans(df[, 2:3], 2, nstart = 20)
k$cluster <- as.factor(k$cluster)

library(cluster)
clusplot(df, k$cluster, main = 'Cusplot')
```

```
k
## K-means clustering with 2 clusters of sizes 238, 260

##

## Cluster means:

##        Sexo    Edad

## 1 0.5336134 58.84034

## 2 0.5538462 30.52308

##
```

```
## Clustering vector:
##[1] 1 1 2 1 2 1 2 1 1 2 2 2 2 1 1 1 1 2 2 2 1 2 1 1 1 2 1 1 2 2 2 2 2
##[36] 2 1 2 1 2 1 2 2 2 2 2 2 2 2 1 2 1 1 1 1 2 2 1 2 2 1 2 1 1 1 1 1 1 1
##[71] 2 1 2 2 2 2 2 2 2 1 2 2 1 2 1 1 1 2 2 2 2 2 2 1 1 2 2 2 1 1 2 2 2 1 1
##[106] 1 1 1 2 1 1 2 2 2 1 1 2 2 1 2 1 2 1 2 2 2 2 1 1 1 2 1 1 1 2 1 1 1 2 2
##[141] 2 2 2 2 1 2 1 1 1 2 1 1 1 2 1 1 2 1 1 1 2 2 1 2 1 2 2 2 1 1 2 2 1 2 2
##[176] 1 2 2 2 2 1 1 2 2 1 1 2 2 1 2 2 2 1 2 2 2 1 2 1 2 2 1 2 2 2 2 1 1 1 1
##[211] 2 2 2 1 2 2 2 2 1 2 2 1 2 2 2 1 1 1 2 2 2 2 2 2 2 1 1 2 1 1 2 2 1 1 1
##[246] 1 1 2 1 2 2 2 1 1 1 1 1 2 1 1 2 2 1 1 2 2 2 2 2 2 1 1 2 2 2 1 1 2 2 2
##[281] 1 2 2 2 1 1 2 2 1 2 2 2 2 1 1 1 1 2 1 1 2 1 2 2 1 2 2 1 2 2 1 2 1 2 2
##[316] 2 2 1 1 1 2 2 2 1 1 2 2 1 1 2 2 1 2 2 1 2 2 1 2 2 1 2 2 1 1 1 1 1 1 2
##[351] 2 1 1 2 1 1 1 1 1 1 2 2 1 2 2 2 1 1 2 2 1 1 1 2 2 1 1 2 1 1 1 1 2 1 2
##[386] 2 1 1 2 2 1 1 2 1 1 1 2 1 1 1 2 1 1 1 1 2 1 1 1 1 2 2 2 2 1 1 1 1 1 2
##[421] 1 1 1 1 2 2 2 2 2 1 2 2 2 1 2 1 1 1 1 2 2 2 2 2 1 2 1 1 1 1 1 1 2 2 2
##[456] 1 2 2 2 2 2 1 2 2 1 2 2 2 2 2 1 2 2 2 1 1 1 1 2 2 1 1 2 1 1 1 1 2 2 1
##[491] 2 2 1 2 2 1 1 2
## Levels: 1 2
##
## Within cluster sum of squares by cluster:
## [1] 15701.16 14489.11
##  (between_SS / total_SS =  76.7 %)
##
## Available components:
##
## [1] "cluster"        "centers"   "totss"     "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Un primer intento ha sido crear un cluster con 2 grupos (específicamente Sexo y Edad).

Ahora veremos otra aproximación.

```
set.seed(10)
k2 <- kmeans(df[, 2:4], 3, nstart = 1)
k2$cluster <- as.factor(k2$cluster)
k2
```

```
## K-means clustering with 3 clusters of sizes 183, 139, 176
##
## Cluster means:
##        Sexo     Edad Anyos.Coche
## 1 0.5191257 62.00546    9.879781
## 2 0.5539568 42.87050   10.158273
## 3 0.5625000 26.32955    9.704545
##
## Clustering vector:
##[1] 2 1 3 2 2 1 3 1 1 3 2 3 3 1 1 1 1 2 3 3 3 1 3 2 1 2 2 3 1 1 3 2 3 3 3
##[36] 3 1 2 1 3 2 3 3 3 2 2 2 3 3 1 3 1 1 2 1 2 3 2 3 2 1 3 1 1 1 1 2 2 2 1
##[71] 3 1 2 3 3 3 3 2 2 1 2 3 1 3 2 1 1 3 2 3 3 3 3 2 2 3 2 3 1 1 3 3 3 1 1
##[106] 1 1 2 3 1 1 3 3 3 2 1 2 2 1 3 1 3 1 3 2 2 3 1 1 1 3 2 2 1 3 2 1 2 2 3
##[141] 3 3 2 2 1 3 1 2 1 3 1 1 1 3 1 1 2 1 1 1 3 3 1 3 1 3 3 2 2 1 3 2 1 3 3
##[176] 1 3 3 2 3 2 1 3 3 2 1 2 3 1 3 2 2 1 3 3 3 1 2 2 3 3 1 3 2 3 2 1 1 1 1
##[211] 3 2 3 1 3 2 2 3 1 3 3 1 3 3 2 1 1 1 3 3 2 3 3 3 3 2 1 2 1 1 2 2 2 1 2
##[246] 1 1 3 1 3 2 3 1 1 1 1 1 3 1 1 3 3 1 1 2 2 2 3 3 3 1 1 2 3 2 1 1 3 2 3
##[281] 1 3 2 3 1 1 3 2 1 3 3 2 3 1 2 1 1 3 1 2 3 1 2 3 1 3 3 2 3 3 1 3 1 3 3
##[316] 2 3 1 2 2 2 3 3 1 1 3 3 1 1 2 3 2 3 3 1 3 3 1 3 3 2 2 3 1 2 1 2 1 2 2
##[351] 3 1 1 2 1 1 1 1 1 1 2 3 1 2 2 3 1 1 3 3 1 1 2 3 3 1 1 3 1 1 1 1 2 1 3
##[386] 3 1 1 3 3 1 1 3 1 1 1 2 1 1 1 2 1 1 1 1 2 1 2 1 1 2 2 3 3 1 2 2 1 1 3
##[421] 1 1 1 1 3 2 3 3 2 2 3 3 3 1 3 2 2 1 1 2 2 3 2 3 1 3 1 2 1 2 1 1 3 2 2
##[456] 2 3 3 3 2 3 1 3 2 1 3 3 2 3 2 1 3 3 3 1 1 2 1 2 2 2 2 2 1 2 2 1 3 3 2
##[491] 2 3 1 2 2 1 1 2
## Levels: 1 2 3
##
## Within cluster sum of squares by cluster:
## [1] 13802.033  7902.532  9284.835
##  (between_SS / total_SS =  78.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
```

```
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

En el segundo intento hemos definido agrupación en 3 clases distintas, además de poner valor distinto para la seed.

d) La calidad del modelo

Cuando hablamos de la calidad del modelo, según el material de estudios de la UOC, el **criterio de evaluación** depende de la aplicación a la que queremos destinar el resultado del proceso. Se espera obtener grupos cohesionados y diferentes entre sí (más adelante, al mirarnos los resultados y conocimientos extraídos, haré consideraciones sobre este punto).

Es necesario medir:

1) Similitud intragrupo;
2) Similitud intergrupos;
3) Variantes

Por lo tanto, creo que la calidad del modelo que he alcanzado no está tan satisfactorio si observo, por ejemplo, los gráficos de las figuras 2 y 3 a seguir, queda claro que no solamente estas similitudes no están bien establecidas, sino que además se mezclan un poco entre grupos distintos.

e) Gráficos de los modelos

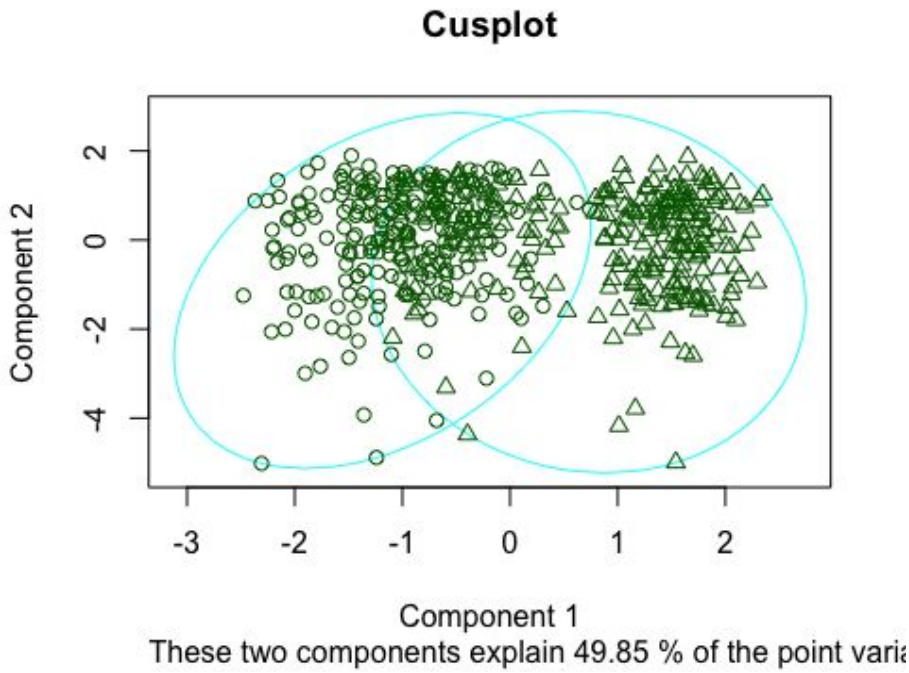

Figura 2: Cluster para 2 grupos, ejercicio 2.c

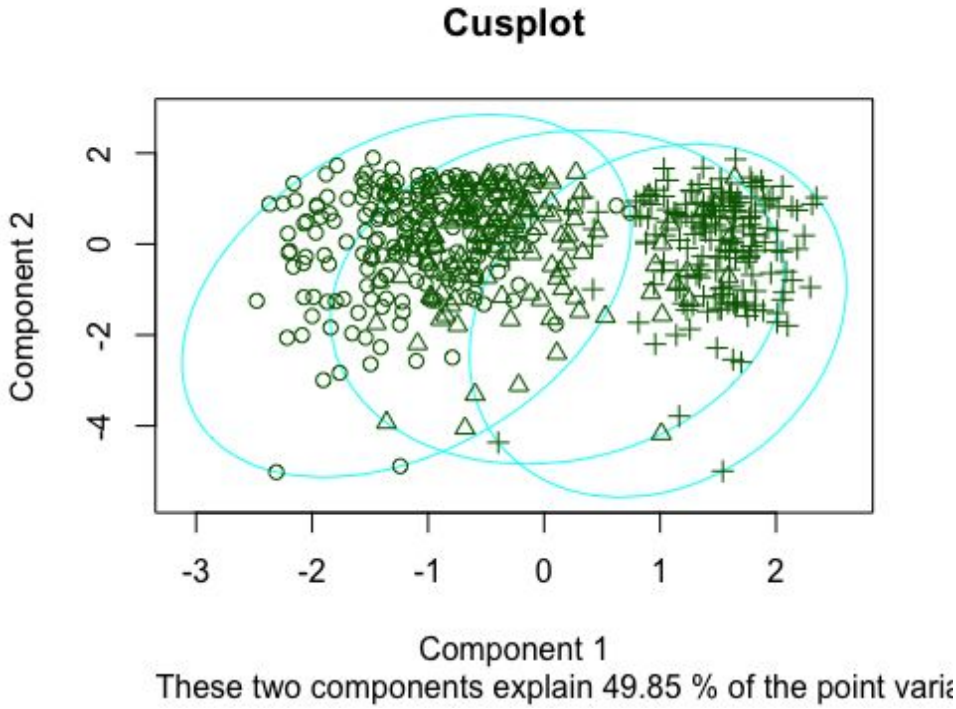

Figura 3: Cluster para 2 grupos, ejercicio 2.c

f) Conocimiento extraído

Conforme he dicho previamente, hay una posibilidad que no he logrado crear un modelo suficiente bueno para agrupar los registros de la manera adecuada o, en una hipótesis más lejana, los datos que son muy parecidos entre sí. En la realidad, creo que no es el caso de la última hipótesis.
Aparentemente, con 2 o con 3 clusters, los modelos no estaban tan buenos al punto de seren capazes de crear divisiones del conjunto original que sean bastante diferentes entre sí y que, además, los objetos de cada partición mantengan la alta similitud.

Ejercicio 3

Repetid el ejercicio 2 con otro conjunto de datos. Pueden ser datos reales de vuestro ámbito laboral o de algún repositorio de datos de Internet. Mirar: http://www.ics.uci.edu/~mlearn/MLSummary.html y los otros repositorios ya citados.
Seguir el guión propuesto en la pregunta anterior. Podéis añadir nuevos puntos y probar otras implementaciones del algoritmo. Por ejemplo, parece interesante que el número de particiones lo proponga el algoritmo. Recordad también que el ciclo de vida de los proyectos de minería contempla retroceder para volver a generar el modelo con datos modificados o parámetros del algoritmo variados, si el resultado no es lo suficientemente bueno.

RESPUESTA:

La primera cosa a ser hecha es escoger el dataset más adecuado para proceder una agrupación. Para ello, usando el conjunto de datasets sugerido, escogí el dataset Iris donado por Marshall [2] – http://archive.ics.uci.edu/ml/datasets/Iris.

El estudio de los datos

Observamos:

```
dfi <- read.csv("iris.csv")
head(dfi)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width     Species
## 1          5.1         3.5          1.4         0.2 Iris-setosa
## 2          4.9         3.0          1.4         0.2 Iris-setosa
## 3          4.7         3.2          1.3         0.2 Iris-setosa
## 4          4.6         3.1          1.5         0.2 Iris-setosa
## 5          5.0         3.6          1.4         0.2 Iris-setosa
## 6          5.4         3.9          1.7         0.4 Iris-setosa
```

```
summary(dfi)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.054   Mean   :3.759   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##             Species
```

```
##  Iris-setosa    :50
##  Iris-versicolor:50
##  Iris-virginica :50
##
##
##
```

```
nrow(dfi)
```

```
## [1] 150
```

```
ncol(dfi)
```

```
## [1] 5
```

```
colnames(dfi)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"
## [5] "Species"
```

```
res <- sapply(dfi, class)
kable(data.frame(variables=names(res),clase=as.vector(res)))
```

variables	clase
Sepal.Length	numeric
Sepal.Width	numeric
Petal.Length	numeric
Petal.Width	numeric
Species	factor

A primera vista, puede parecer como un intento de hacer trampa en el ejercicio (una vez que tenemos una columna de clasificación, esto es, sería adecuado para un modelo supervisado). Sin embargo, no la usaré en ningún momento que no sea para validar que nuestro clustering ha funcionado como esperado.

Los otros comentarios son que el muestreo presenta una cantidad limpia de datos (lo que nos facilita la vida para la próxima etapa de preproceso de datos.

g) Preparar los datos

Como comentado en el apartado anterior, los atributos están con el tipo adecuado, los formatos adecuados y sin la existencia de atributos no informados (NA). Por lo tanto, seguimos para el próximo apartado.

Es importante resaltar que, como dicho por Kodali, (2015) [3] los atributos *Petal.Length* y *Petal.Width* tienen similitudes entre las mismas especies, pero varianza considerable entre diferente especies, o sea, son buenos indicadores para agrupar.

Generando el modelo de agregación

```
set.seed(20)
irisCluster <- kmeans(dfi[, 3:4], 3, nstart = 20)
irisCluster
```

```
## K-means clustering with 3 clusters of sizes 50, 52, 48
##
## Cluster means:
##   Petal.Length Petal.Width
## 1     1.464000    0.244000
## 2     4.269231    1.342308
## 3     5.595833    2.037500
##
## Clustering vector:
## [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [71] 2 2 2 2 2 2 2 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3
##[106] 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##[141] 3 3 3 3 3 3 3 3 3 3
##
## Within cluster sum of squares by cluster:
## [1]  2.03840 13.05769 16.29167
##  (between_SS / total_SS =  94.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Un primer intento ha sido crear un cluster con 3 grupos usando los atributos que he comentado en el apartado anterior y, aparte, también he incluido `nStart = 20` que significa iniciar 20 asignaciones randómicas.

Ahora veremos la primera salida del modelo.

```
table(irisCluster$cluster, dfi$Species)

##
##     Iris-setosa Iris-versicolor Iris-virginica
##   1          50               0              0
##   2           0              48              4
##   3           0               2             46
```

P.D. Importante comentar que el modelo está inspirado por el que ha sido creado por Kodali, (2015) [3].

Conforme mencionado en el apartado 3.1, aunque estamos usando un *dataset* que contiene labels, nos ha sido útil solo para mostrar el comparativo de salida de nuestro modelo y, en una primera análisis, la agrupación parece ter sido bastante eficiente.

Sin embargo, para probar nuevas aproximaciones, voy a cambiar un poco los parámetros y ver cómo queda la salida:

```
#Sepal.Lenght and Sepal.Width en vez de Petal (Pasamos un grupo a más que
existe a ver que ocurre)
irisCluster2 <- kmeans(dfi[, 1:2], 4, nstart = 30)
irisCluster2

## K-means clustering with 4 clusters of sizes 28, 41, 28, 53
##
## Cluster means:
##   Sepal.Length Sepal.Width
## 1     5.232143    3.667857
## 2     6.880488    3.097561
## 3     4.782143    2.950000
## 4     5.924528    2.750943
##
## Clustering vector:
##   [1] 1 3 3 3 1 1 3 1 3 3 1 3 3 3 1 1 1 1 1 1 1 1 1 3 3 1 1 1 3 3 1 1 1
3
##  [36] 3 1 3 3 1 1 3 3 1 1 3 1 3 1 3 2 2 2 4 2 4 2 3 2 3 3 4 4 4 4 2 4 4 4
4
##  [71] 4 4 4 4 4 2 2 2 4 4 4 4 4 4 4 4 2 4 4 4 4 4 4 3 4 4 4 4 3 4 2 4 2 4
2
## [106] 2 3 2 2 2 2 4 2 4 4 2 2 2 2 4 2 4 2 4 2 2 4 4 4 2 2 2 4 4 4 2 2 2 4
2
## [141] 2 2 4 2 2 2 4 2 4 4
##
## Within cluster sum of squares by cluster:
```

```
## [1]  3.902143 10.634146  5.151071  8.250566
##  (between_SS / total_SS =  78.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
table(irisCluster2$cluster, dfi$Species)
```

```
##
##     Iris-setosa Iris-versicolor Iris-virginica
##   1          28               0              0
##   2           0              11             30
##   3          22               5              1
##   4           0              34             19
```

He añadido un nuevo cluster (ahora son 4) - aunque sepamos que no hay más que 3 porque tenemos acceso a los grupos previamente definidos. Además, no estoy utilizando los atributos de Petal, sino de Sepal - para mirar si ocurre una variación en los resultados observables. Ahora la calidad ha disminuido un poco, o sea, mantendremos la primera aproximación como nuestra salida final.

La calidad del modelo

La calidad del modelo es bastante alta (>90%), sin embargo hay que considerar que el muestreo es uno de los más sencillos disponibles.

Gráfico del modelo

```
clusplot(dfi, irisCluster$cluster, color=TRUE, shade = TRUE, lines=0)
```

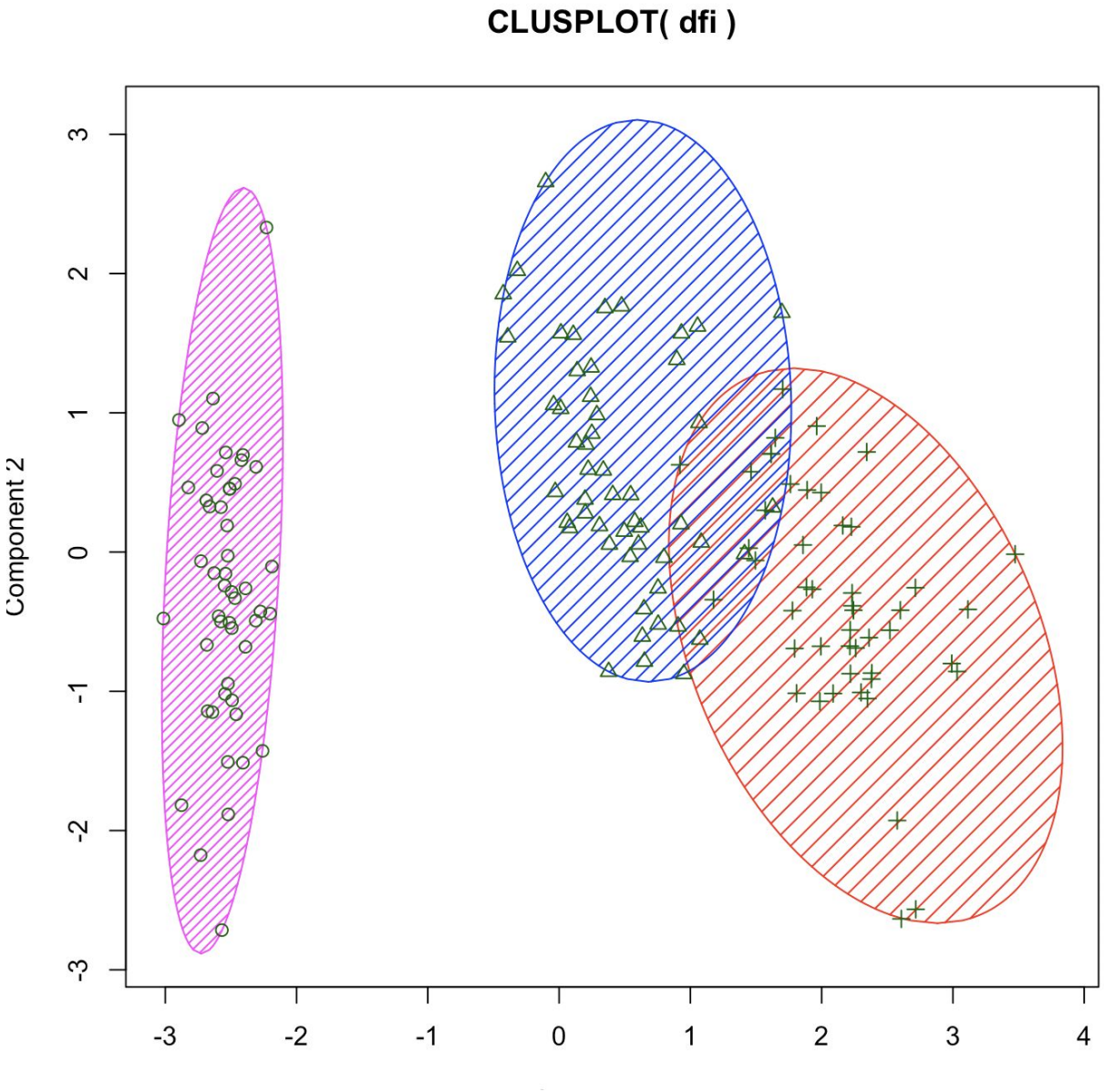

Figura 4: Cluster para 3 grupos del dataset Iris

Conocimiento extraído

Con un *dataset* más sencillo, seguramente es posible obtener una agrupación de una calidad bastante superior (como comentado, arriba de los 90%) donde las divisiones son

bastante diferentes entre sí y las particiones mantienen similitud. Aunque el conjunto de datos Iris sea demasiado sencillo para una aplicación real, me ha aportado bastante para ver funcionando y posiblemente entender mejor en los puntos que había fallado en el ejercicio anterior. Supongo que habría que hacer otra interacción en la búsqueda de mejorar el ejercicio 2, por ejemplo – al menos, en un escenario de mundo real, tendría que hacerlo de esta manera (por lo tanto, creo que aporta reflexionar en el conocimiento).

Bibliografía

[1] **Manglick, A. (2017).** "*K-Means Clustering*" [artículo en línea]. [Fecha de consulta: 22 de abril del 2018].
<http://arun-aiml.blogspot.com.es/2017/07/k-means-clustering.html>

[2] **Marshall, M. (1988)**. UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science. [Fecha de consulta: 23 de abril del 2018].
<http://archive.ics.uci.edu/ml/datasets/Iris>

[3] **Kodali, T. (2015)**. "*K-Means Clustering*" [artículo en línea]. [Fecha de consulta: 22 de abril del 2018].
<https://www.r-bloggers.com/k-means-clustering-in-r/>

Tiempo de dedicación

La Universidad está evaluando el tiempo de dedicación que implica la resolución de las PEC. A continuación, por favor, te pedimos que nos indiques cuánto tiempo (horas) te ha supuesto hacer esta actividad y la distribución de este tiempo en función de los tres tipos de tareas que te presentamos. Muchas gracias por tu colaboración.

	HORAS DE DEDICACIÓN
¿Cuántas horas has dedicado a hacer esta PEC? *Para calcular estas horas, debes tener en cuenta los siguientes tipos de tareas que te presentamos a continuación y debes indicar el porcentaje que has dedicado para hacer cada una de ellas.* (aproximadamente)	30
	PORCENTAJE DEDICACIÓN
Lectura, comprensión y reflexión sobre el enunciado de la actividad (comprensión de la actividad, evaluación de la demanda, planificación del proceso y / o otras consideraciones previas)	10%
Búsqueda, lectura y comprensión de los recursos necesarios para responder a la actividad (módulos, vídeos, artículos y / o otros recursos de consulta)	60%
Elaboración de las respuestas (hacer ejercicios, cálculos, redacción de preguntas cortas o largas, participación en debates, responder preguntas tipo test y / u otras formas de respuesta)	30%
Total	**100%**