Universidad de Granada

# Minería de Datos: Aprendizaje no Supervisado y detección de Anomalías
## Propuesta de trabajo: Clustering

**Descripción.**

**El objetivo del trabajo es encontrar e identificar los grupos que pueden darse en el data set seleccionado. Para ello el proceso será siempre:**

1. **Seleccionar la distancia o combinación de distancias que se ajuste mejor a los tipos de variables que tenga el dataset.**
2. **Empezar realizando un agrupamiento jerárquico para determinar un número de grupos adecuado.**
3. **Probar otros métodos de agrupamiento con número de grupos conocidos como, k-medias, dbscan o k-medioides, k-medias difusas comparando resultados.**

**La medida de los resultados se hará:**

a) **Mediante el coeficiente de silueta y algunas otras medidas no supervisadas que ofrece R.**
b) **Si hay alguna clasificación de referencia mediante medidas no supervisadas.**

**Una vez obtenido el mejor agrupamiento posible se deberá intentar su interpretación, para ello se deberán realizar representaciones de los grupos en función de distintas variables intervinientes, intentando identificar los grupos encontrados.**

**En el caso de que se encuentre que el agrupamiento tiene un alto grado de dependencia de un conjunto menor de variables se intentará mejorar los resultados, volviendo a realizar todo el proceso para este conjunto.**

---

En primer lugar, y antes de comenzar cualquier análisis o procesamiento, se cargarán aquellas **librerías** necesarias para realizar tareas de agrupamiento. Dichas librerías son:

- **`Stats:`** Permite realizar agrupamiento jerárquico y tiene implementado el algoritmo de las k-medias y el cálculo de distancias.
- **`Cluster:`** Permite calcular el coeficiente de silueta, además de implementar otros tipos de agrupamiento como AGNES, DIANA, CLARA y k-medias difusas.
- **`Fpc:`** Pone a disposición del programador el algoritmo DBSCAN y otros métodos para análisis de bondad.

Para cargar dichas librerías, se ejecutará el siguiente fragmento de código que utiliza la función **library().**



*Julio Alberto López Gómez*
*Amparo Vila Miranda*

```
library(stats)
library(cluster)
library(fpc)
```

A continuación, el siguiente paso consistirá en cargar en la variable **German_Credit** el **dataset** con el mismo nombre y estudiar sus **dimensiones**, para saber si estamos ante un dataset largo, estrecho... Para ello, se ejecutará el siguiente código R:

```
setwd("C:/Users/MAT/Dropbox/Master Ciencia de Datos/MinDat NoSuper y Anomalias/Documentos generales/German Credit")
German_Credit <- read.csv("german_credit.csv")
#Estudiamos el tamaño del dataset
German_dim <- dim(German_Credit) #Se trata de una dataset relativamente largo (1000 registros) y ancho (21 variables o atributos)
German_rows <- German_dim[1]
German_cols <- German_dim[2]
view(German_Credit)
```

Estamos, dados los valores de las variables **German_rows y German_cols**, ante un dataset relativamente largo, pues tiene mil registros y ancho, pues tiene veintiuna variables o atributos. La función **view()** aplicada sobre un dataset, permite abrir una ventana de R que muestra en formato tabular a modo de hoja de cálculo, el contenido del dataset.

Una vez cargadas las librerías que darán soporte a las operaciones que se realizarán a continuación, así como el dataset sobre el que se realizarán los experimentos, se calcularán, siguiendo las indicaciones de los scripts propuestos por Amparo Vila, las **variables numéricas y sus respectivas distancias**, así como las **variables binarias y sus respectivas distancias**. Para ello, se calculará el **rango** de valores de cada una de las variables del dataset (valor máximo y mínimo de una variable) y se calculará el **estadístico y medida de dispersión rango**, que se define de la siguiente manera:

$$Rango = valor_{maximo} - valor_{minimo}$$

De manera que aquellas variables cuyo rango sea 1 serán variables binarias, mientras que todas aquellas variables cuyo rango sea mayor que uno serán numéricas. Este procesamiento de los datos se realiza ejecutando el siguiente código R.

```
German_rangos <- apply(German_Credit,2,range)
Rangos <- German_rangos[2,] - German_rangos[1,]
Numericas <- Rangos>1
Binarias <- Rangos == 1
#Data frame con las variables Numéricas
German_Numericas <- German_Credit[,Numericas]
#Data frame con las variables Binarias
German_Binarias <- German_Credit[,Binarias]
```

# Minería de Datos: Aprendizaje no Supervisado y detección de Anomalías
## Propuesta de trabajo: Clustering

Una vez obtenidos los data frame **German_Numericas** que contiene las variables numéricas de German_Credit y **German_Binarias**, que contiene las variables binarias del dataset, es posible calcular las distancias para cada una de estas variables, donde se utilizará la distancia euclídea para las variables numéricas y la distancia binaria para las variables dicotómicas o binarias. Una vez calculadas dichas distancias, se calculará una tercera distancia como la media de las dos anteriores, de manera que se obtenga para todo el dataset una distancia ponderada entre las dos calculadas anteriormente.

El siguiente código R, refleja el modo en que se han calculado las distancias.

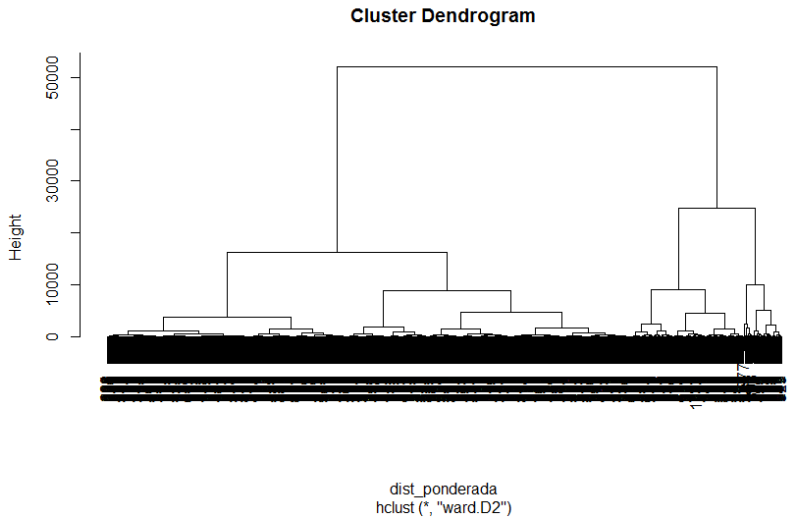
```
#Variables numéricas (Distancia Euclídea)
dist_numerica = dist(German_Numericas)
#Variables binarias (Distancia Numérica)
dist_binaria = dist(German_Binarias,method = "binary")
#4. Cálculo de la distancia ponderada como la media de las distancias numérica y
binaria.
dist_ponderada = (dist_numerica + dist_binaria)/2
```

Una vez calculadas las distancias, ya se dispone de todos los elementos necesarios para realizar el agrupamiento. Por tanto, el siguiente paso consistirá en aplicar un algoritmo de **clustering o agrupamiento jerárquico** para tener una idea del número de grupos óptimo aproximado, y aplicar diferentes algoritmos de clustering con este número de grupos obtenido.

De este modo, el siguiente fragmento de código R, **realiza el agrupamiento jerárquico utilizando la distancia ponderada y el método Ward**, el cual es un procedimiento jerárquico en el que en cada etapa se unen los dos clusters para los cuales se tenga el menor incremento en el valor total de la suma de los cuadrados de las diferencias, dentro de cada cluster, de cada individuo al centroide del cluster.

```
h = hclust(dist_ponderada, method = "ward.D2")
h
plot(h)
```

El **dendrograma** obtenido tras el agrupamiento jerárquico se puede ver en la siguiente imagen.



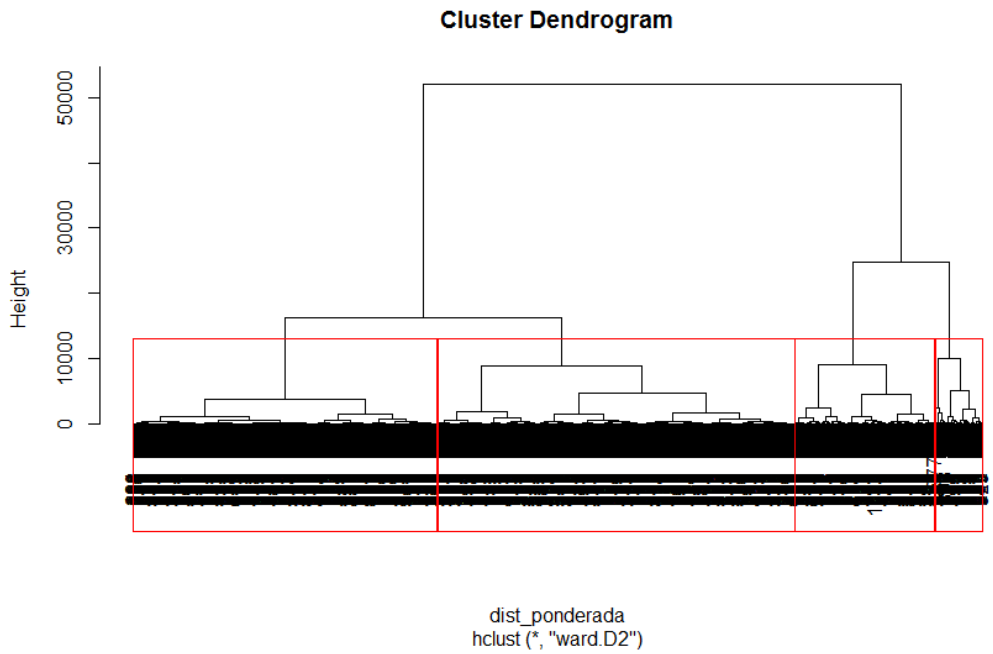



*Julio Alberto López Gómez*
*Amparo Vila Miranda*

A la vista del dendrograma, es posible deducir que un número de grupos que puede dar buenos resultados es **k = 4**. Estos grupos coinciden con las ramas de altura **h = 3** del dendrograma. El siguiente código R, permite seleccionar los k = 4 clusters detectados tras el análisis visual del dendrograma.

```
rect.hclust(h, k = 4)
groups_h=cutree(h,k=4)
```

Obteniendo como resultado, el siguiente dendrograma:




En el cual se pueden ver perfectamente los cuatro grupos en el tercer nivel del dendrograma.

Los resultados obtenidos por los distintos algoritmos de agrupamiento serán almacenados en una matriz como la siguiente, que en primera instancia es inicializada a cero.

```
resul_k4 <- matrix(data = 0, nr = 2, nc = 3)
rownames(resul_k4) <- c("Sin Normalizar","Normalizado")
colnames(resul_k4) <- c("Jerárquico","k-medias","k-medioides")
resul_k4
```

La siguiente imagen muestra la estructura de dicha matriz:

```
               Jerárquico k-medias k-medioides
Sin Normalizar          0        0           0
Normalizado             0        0           0
>
```



*Julio Alberto López Gómez*
*Amparo Vila Miranda*

Una vez que se dispone de los datos preparados y de la estructura de datos, en este caso una matriz, en la que se almacenarán los resultados, se está en condiciones de comenzar a utilizar las distintas técnicas de agrupamiento. En este caso, para **k = 4** grupos, se han realizado agrupamientos con los datos ***normalizados y sin normalizar***, utilizando los siguientes algoritmos:

- **Clustering Jerárquico**
- **K-medias**
- **K-medioides**

Los resultados para estos métodos bajo estas circunstancias, se comentan a continuación:

Clustering para k = 4 grupos SIN Normalización.

Una vez que se ha realizado el clustering jerárquico para los k = 4 grupos que se vio anteriormente, es posible calcular el **coeficiente de silueta** para evaluar la bondad de este agrupamiento. Para ello, se han utilizado los scripts de agrupamiento proporcionados por Amparo Vila y se han encapsulado las sentencias de cálculo del coeficiente de silueta y visualización de resultados en la función **Analisis_Bondad,** que se muestra a continuación.

```
##############################################################################
# Parámetros                                                                 ##
  # Dataset --> Conjunto de datos                                            ##
  # x --> Data frame con las variables numéricas del dataset                 ##
  # y --> Data frame con las variables binarias del dataset                  ##
  # grupos --> Agrupamiento realizado con algún algoritmo de clustering      ##
  # k --> Número de Grupos                                                   ##
  # distancia --> función de distancia utilizada en el agrupamiento          ##
# Salida                                                                     ##
  # Shi --> Coeficiente Silhouette del Agrupamiento                          ##
                                                                             ##
Analisis_Bondad <- function(dataset,x,y,grupos,k,distancia){                 ##
  idx=sample(1:dim(dataset)[1],200)                                          ##
  plotcluster(x[idx,],grupos[idx])                                           ##
  plotcluster(y[idx,],grupos[idx])                                           ##
  d1=dist(x[idx,])                                                           ##
  d2=dist(y[idx,])                                                           ##
  d3=(d1+d2)/2                                                               ##
  shi= silhouette(grupos[idx],d3)                                            ##
  plot(shi,col=1:k)                                                          ##
  cluster.stats(distancia,grupos)                                            ##
                                                                             ##
  return(shi)                                                                ##
                                                                             ##
}                                                                            ##
##############################################################################
```



*Julio Alberto López Gómez*
*Amparo Vila Miranda*

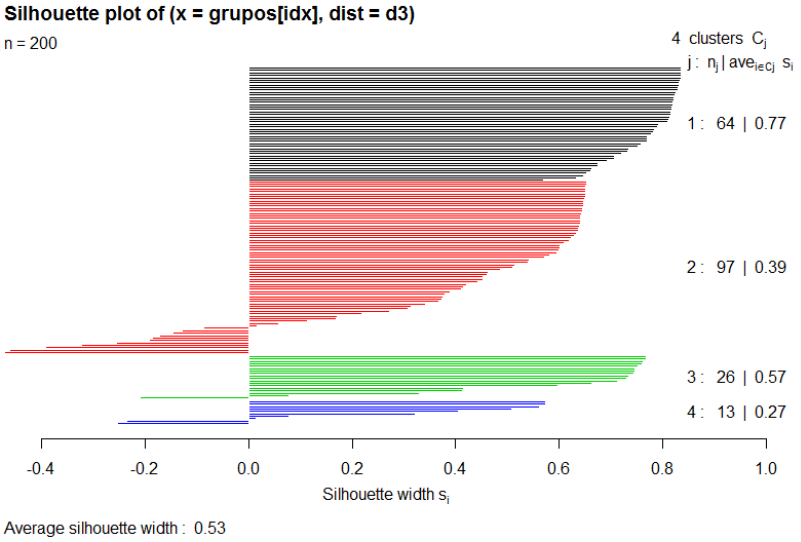
Por otra parte, la function **Average_Silhouette**, permite calcular el coeficiente de Silueta medio para un agrupamiento dado. El siguiente código R, implementa dicha función.

```
##############################################################################
#Parámetros                                                                  ##
  #Silhouette -> Tipo de dato Silhouette para un agramiento                  ##
#Salida                                                                      ##
  #average_silhouette <- Coeficiente de silueta medio para un agrupamiento   ##
Average_Silhouette <- function(Silhouette){                                  ##
  media_cluster <- 0                                                         ##
  min <- 1                                                                   ##
  max <- range(Silhouette[,1])                                               ##
  max <- max[2]                                                              ##
  silhouette_cluster = matrix(data = 0, nr = 1, nc = max)                    ##
  for (i in 1:max){                                                          ##
    c <- Silhouette[Silhouette[,1]==i,3]                                     ##
    media_cluster <- mean(c)                                                 ##
    silhouette_cluster[i] <- media_cluster                                   ##
  }                                                                          ##
average_silhouette <- mean(silhouette_cluster)                               ##
return (average_silhouette)                                                  ##
}                                                                            ##
##############################################################################
```

Una vez definidas estas funciones se realizará el análisis de bondad para el agrupamiento seleccionado y se calculará su coeficiente de silueta medio.

```
Sil_H <-
Analisis_Bondad(German_Credit,German_Numericas,German_Binarias,groups_h,k,dist_p
onderada)
avg_sil_h <- Average_Silhouette(Sil_H)
```

Los resultados de esta operación, pueden resumirse en la siguiente imagen.






*Julio Alberto López Gómez*
*Amparo Vila Miranda*

Donde, como se puede ver en el gráfico, el **índice de silueta medio para este agrupamiento es de 0.53,** siendo **el primer grupo el que mayor coeficiente presenta respecto del resto (un 0.77).**

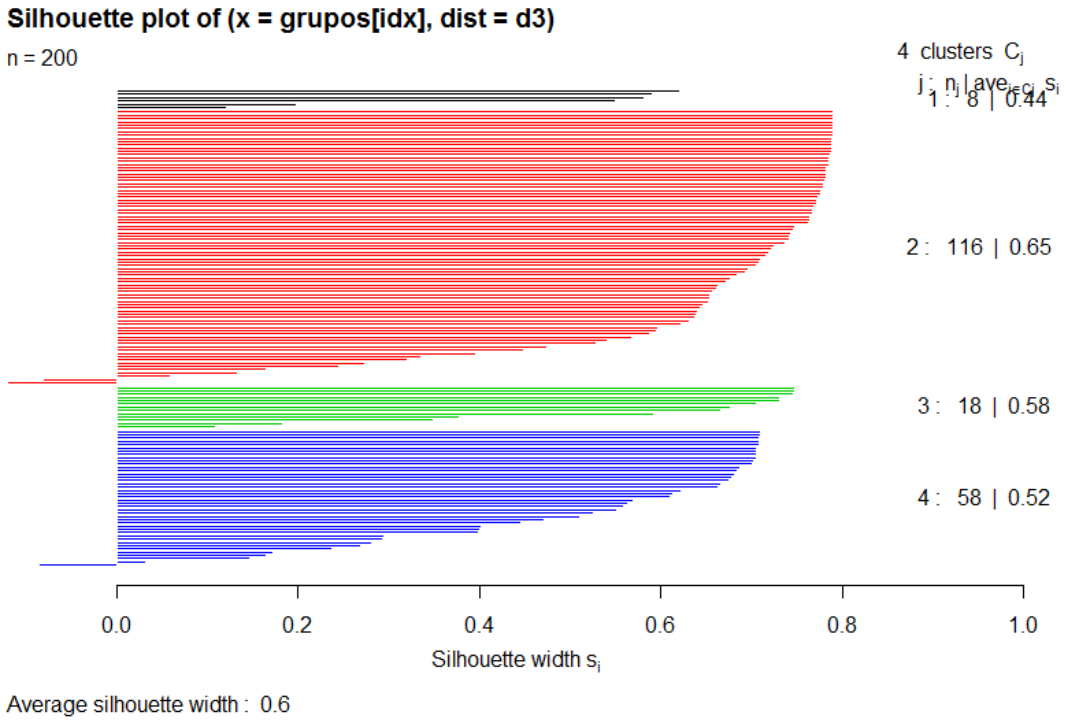
Para contrastar la eficiencia de este agrupamiento, a continuación se aplicarán otros métodos de clustering, tales como el algoritmo de las k-medias y el algoritmo de k-medioides.

**-El Algoritmo de las K-medias**

A continuación, el siguiente fragmento de código R muestra la ejecución del algoritmo de las k-medias para k = 4 grupos.

```
kmeans.result=kmeans(dist_ponderada,k)
kmeans.result~centers
groups_kmeans=kmeans.result$cluster
Sil_kmeans <-
Analisis_Bondad(German_Credit,German_Numericas,German_Binarias,groups_kmeans,k,d
ist_ponderada)
Sil_kmeans
avg_sil_kmeans <- Average_Silhouette(Sil_kmeans)
avg_sil_kmeans
```

En este caso, el resultado del agrupamiento viene dado por la siguiente imagen.




Tal y como se puede comprobar, para la ejecución actual del algoritmo de las k-medias con los datos escogidos de ejemplo, el coeficiente de silueta medio es de **0.6**, mejorando el resultado obtenido por el

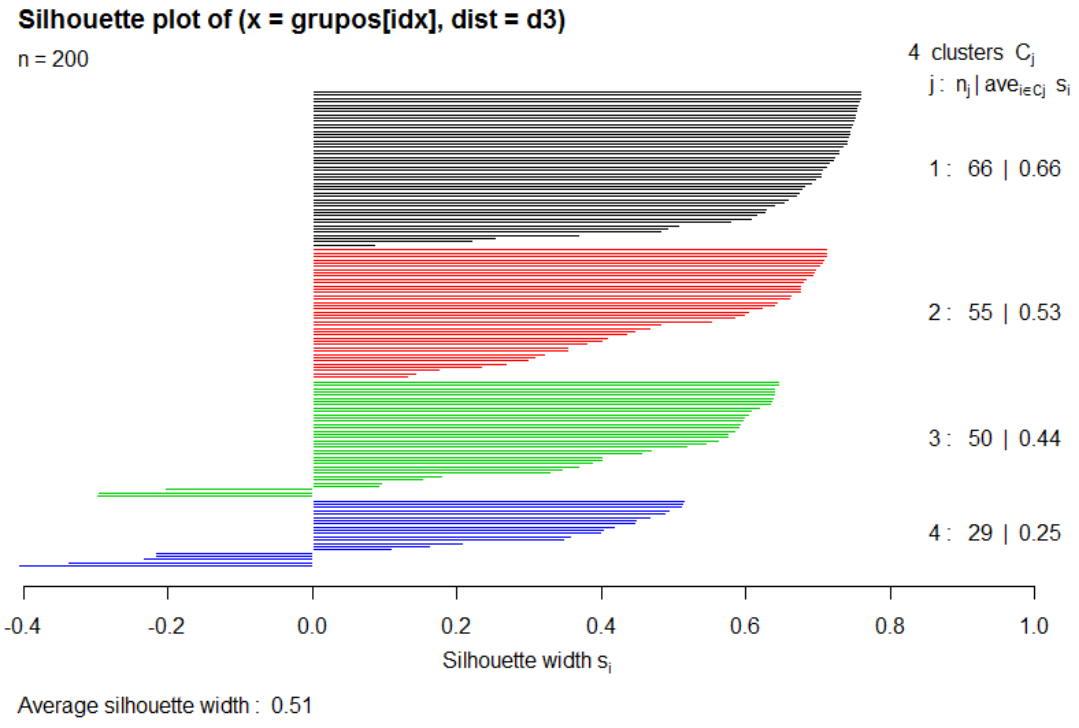
algoritmo de clustering jerárquico. Sin embargo, el cluster que presenta mayor coeficiente es el segundo con un coeficiente de **0.65**, al contrario de lo que sucedía en el anterior algoritmo, cuyo cluster con mayor coeficiente era el primero. Dicho coeficiente es peor que el índice que presentaba el mejor cluster en el agrupamiento jerárquico.

**-El algoritmo de los K-medioides**

Otro algoritmo de agrupamiento muy utilizado y que puede ser útil para contrastar la bondad de un agrupamiento es el algoritmo de los k-medioides, cuya ejecución se propone en el siguiente código R.

```
pam.result=pam(dist_ponderada,k)
groups_k_medioids <- pam.result$clustering
sil_kmedioids <- Analisis_Bondad(German_Credit,German_Numericas,German_Binarias,
groups_k_medioids, k, dist_ponderada)
sil_kmedioids
avg_sil_kmedioids <- Average_Silhouette(sil_kmedioids)
avg_sil_kmedioids
```

Los resultados obtenidos de la ejecución, vienen dados por la siguiente imagen.




Los resultados del algoritmo de los k-medioides son más similares a los obtenidos por el algoritmo de clustering jerárquico, donde el coeficiente de silueta medio para el agrupamiento es de un **0.51**, mientras que el cluster que presenta un mayor coeficiente es el primero, que presenta un **0.65**.

Clustering Para k = 4 grupos CON Normalización.

Una vez que se han comentado los resultados obtenidos para el agrupamiento con k = 4 sin normalización, cabe preguntarse si la **normalización de los datos influirá en la bondad del agrupamiento**. Por este motivo, los experimentos realizados anteriormente serán repetidos, previa normalización de las variables del dataset.

En primer lugar, se extraerán cien registros del dataset para visualizar mejor los datos obtenidos por el algoritmo de agrupamiento. Para ello:

```
idx=sample(1:dim(German_Credit)[1],100)
German_sample <- German_Credit[idx,]
```

A continuación, los datos obtenidos del dataset serán normalizados para posteriormente, ejecutar de nuevo los diferentes algoritmos de agrupamiento. El siguiente código en R, realiza dicha normalización.

```
for (j in dim(German_sample)[2]) {x=German_sample[,j] ; v=(x-
        mean(x))/sqrt(var(x)); German_sample[,j]=v}
```

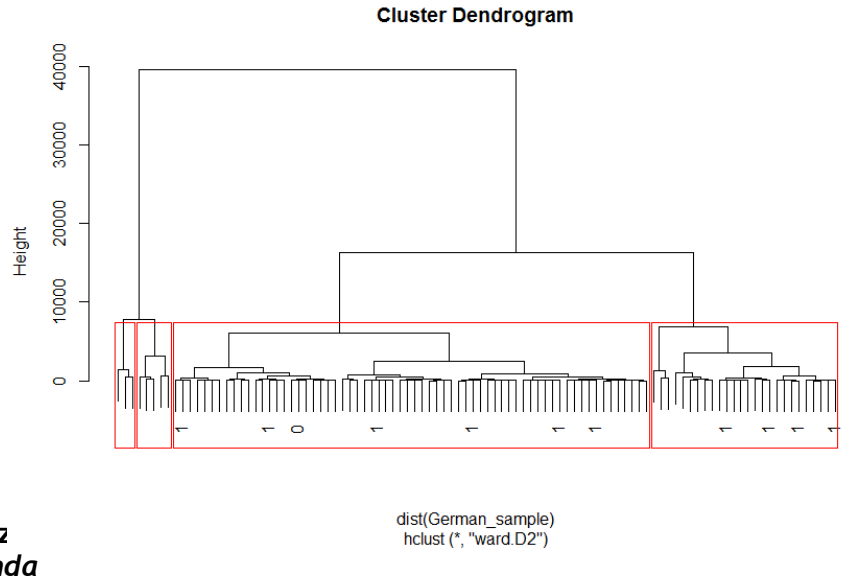
Una vez normalizadas las variables, se aplicarán los diferentes algoritmos de clustering.

**-Clustering Jerárquico.**

El siguiente código R muestra la ejecución del algoritmo de clustering jerárquico sobre los datos de ejemplo normalizados.

```
hc=hclust(dist(German_sample),method="ward.D2")
hc
plot(hc,labels=German_sample$Creditability[idx])
rect.hclust(hc,k)
group_hc=cutree(hc,k)
group_hc
```

El dendrograma resultante de la aplicación del algoritmo con k = 4 grupos, es el siguiente.





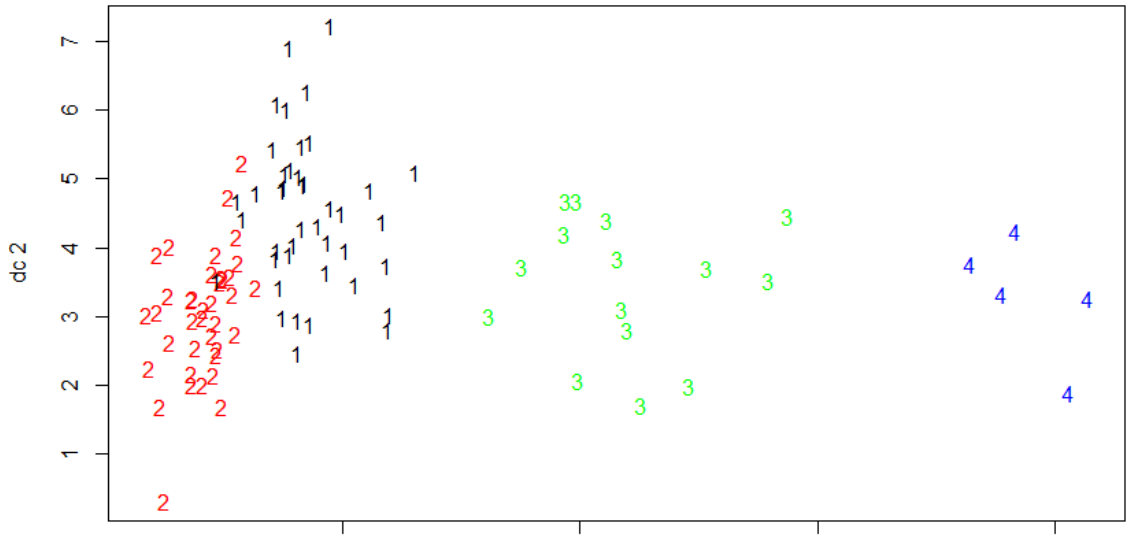

*Julio Alberto López*
*Amparo Vila Miranda*

Llama la atención que la distribución del dendrograma con las variables normalizadas es justamente inversa a la que se obtenía con el dendrograma en el que se tenían en cuenta las variables sin normalizar. A continuación, y con la ejecución de los diferentes algoritmos de clustering, se comprobará si éste hecho también repercute en los grupos.

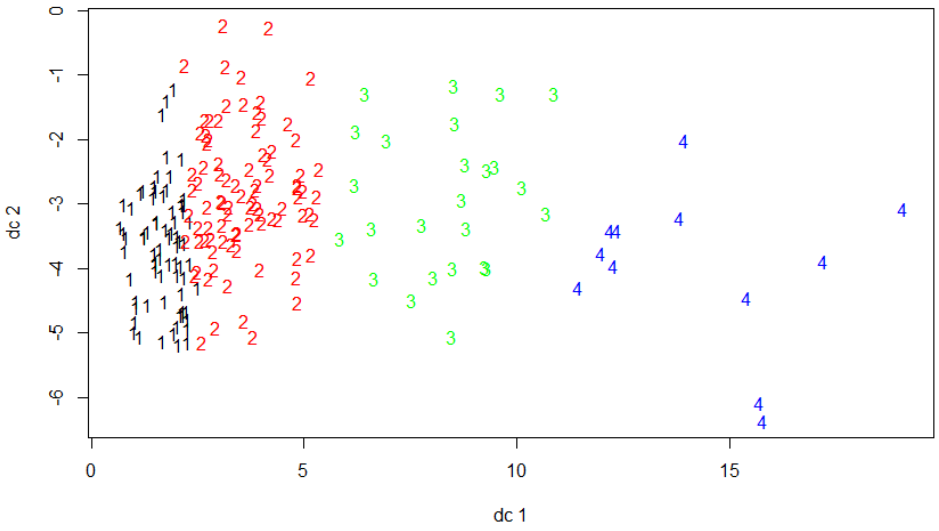
El siguiente código R muestra la realización del análisis de bondad sobre este agrupamiento:

```
plotcluster(German_sample,group_hc)
Sil_Hnorm <- silhouette(group_hc,dist(German_sample))
avg_sil_hnorm <- Average_Silhouette(Sil_Hnorm)
avg_sil_hnorm
```

Los grupos obtenidos por este algoritmo, pueden verse a continuación.




Mientras que los grupos obtenidos para los datos sin normalizar, vienen dados por la siguiente imagen. En esta imagen se puede comprobar como los grupos uno y dos están intercambiados en las dos ejecuciones.




Por otra parte, el



*Julio Alberto López Gómez*
*Amparo Vila Miranda*

coeficiente de silueta medio para este agrupamiento es de: **0.523,** prácticamente igual que el obtenido sin normalización.

**-El algoritmo de las K-Medias**

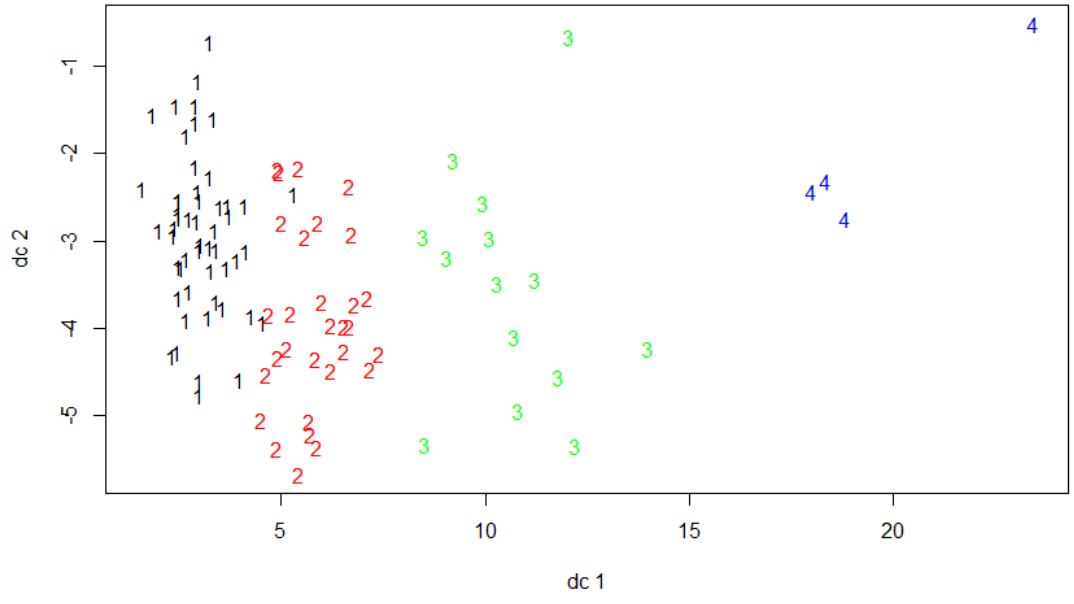
A continuación, se muestra la ejecución del algoritmo de las k-medias para los datos normalizados.

```
kmeans.result=kmeans(German_sample,k)
kmeans.result
table(German_sample$Creditability,kmeans.result$cluster)
#Análisis de Bondad y Gráficas
plot(German_sample[,1:5], col=kmeans.result$cluster)
points(kmeans.result$centers[,],col=1:3,pch=8,cex=2)
x=kmeans.result$cluster
plotcluster(German_sample,x)
sil_kmeans_norm= silhouette(kmeans.result$cluster,dist(German_sample))
avg_sil_kmeans_norm = Average_Silhouette(sil_kmeans_norm)
avg_sil_kmeans_norm
```

Donde la table calculada muestra el número de elementos clasificados en cada uno de los cuatro clusters según su categoría. Dicha tabla se muestra a continuación.

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 0 | 14 | 10 | 3 | 2 |
| 1 | 38 | 20 | 11 | 2 |

Por otra parte, el coeficiente de silueta medio para este agrupamiento es de **0.54** frente al **0.6** de las k-medias sin normalización. El resultado de los grupos obtenido por el algoritmo, se muestra a continuación:





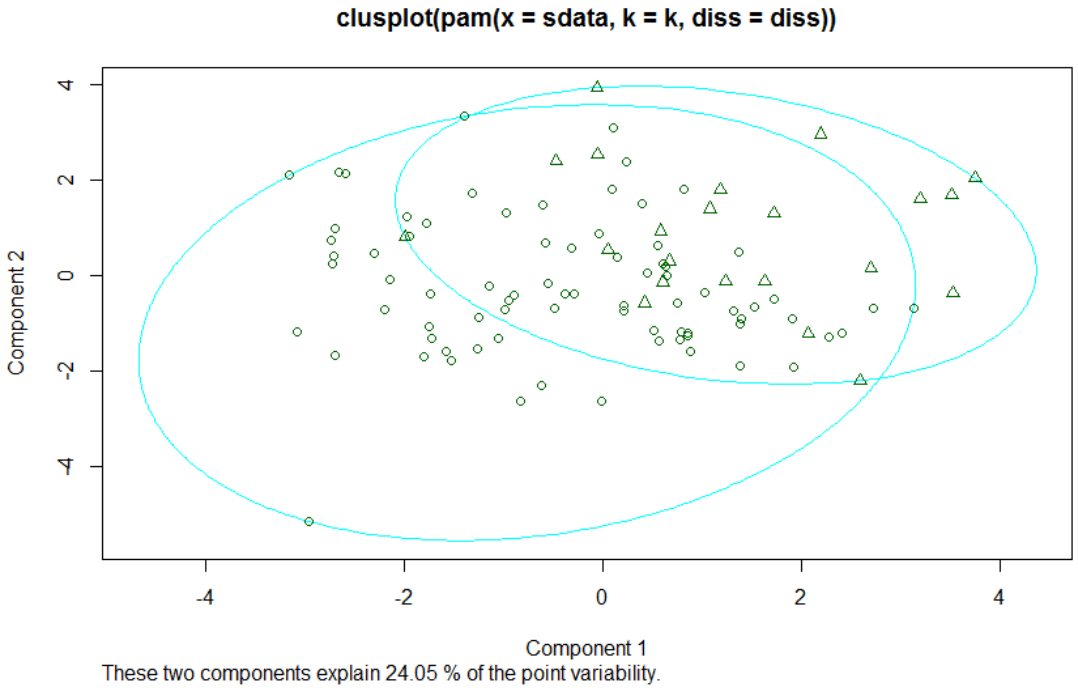

*Julio Alberto López Gómez*
*Amparo Vila Miranda*

**-El algoritmo de los K-medioides**

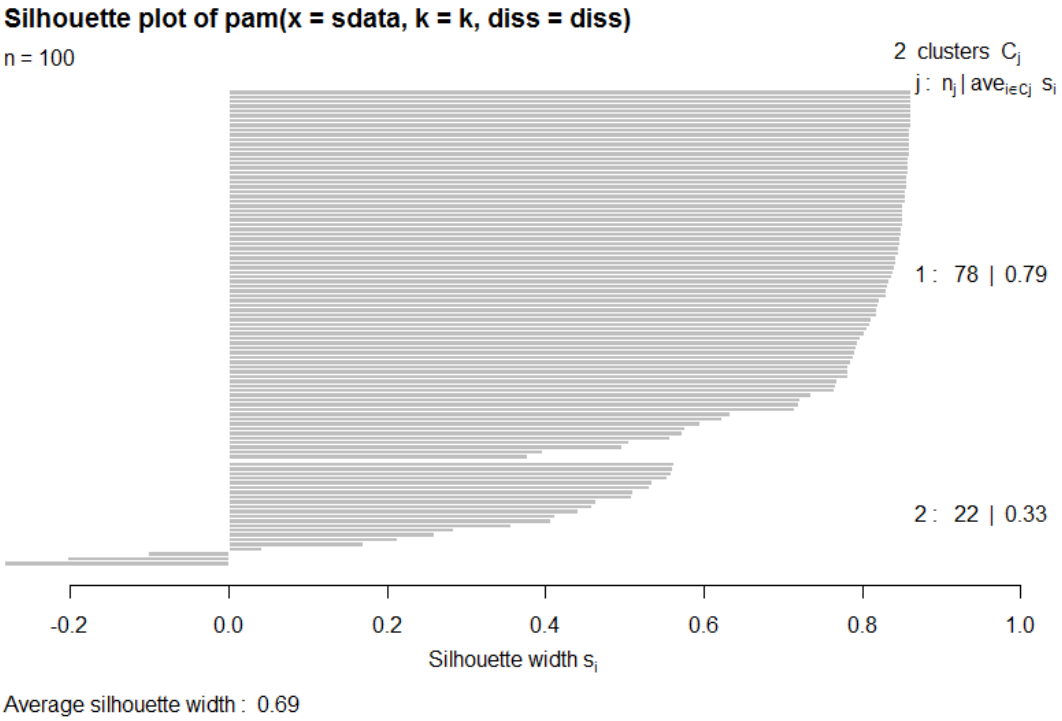
La ejecución del algoritmo de los k-medioides con el número óptimo de grupos elegidos por R sobre los datos normalizados se muestra en el siguiente fragmento de código R.

```
pamk.result=pamk(German_sample)
pamk.result
pamk.result$pamobject$nc
#table(pamk.result$pambject$clustering,German_sample$Creditability)
plot(pamk.result$pamobject)
#(Ver descripcion de la funcion)
group=pamk.result$pamobject$clustering
sil_kmedioids_norm <-
silhouette(pamk.result$pamobject$clustering,dist(German_sample))
avg_sil_kmedioids_norm <- Average_Silhouette(sil_kmedioids_norm)
avg_sil_kmedioids_norm
cluster.stats(dist(German_sample),group)
```

El resultado de la variabilidad explicada por la intersección de los dos clusters es del **24.05%,** tal y como se muestra en el siguiente gráfico.




Por su parte, el número de grupos designados por R es de dos, y el gráfico que muestra el coeficiente de silueta medio del **0.69** (el más alto hasta el momento), se muestra a continuación:




Finalmente, los resultados recogidos para el agrupamiento de k = 4 clusters, con los datos normalizados y sin normalizar se muestran a continuación.

| | Jerárquico | k-medias | k-medioides |
|---|---|---|---|
| Sin Normalizar | 0.5278494 | 0.6027964 | 0.4692196 |
| Normalizado | 0.5239216 | 0.5459269 | 0.5593024 |

Clustering para k = 3 grupos SIN Normalización.

A la vista de los resultados provistos por el dendrograma del clustering con **k = 4** grupos sin normalización – el primero de los mostrados – es posible llegar a la conclusión, de que en realidad, no existen cuatro grupos sino tres. Estos tres grupos vendrían dados por las dos ramas de altura **h = 3**, mientras que el tercer grupo, vendría dado por las dos ramas del dendrograma de altura **h = 2**, que en el anterior ejemplo constituían dos grupos independientes.

De esta forma, existen solo **k = 3 grupos**, donde es posible que el tercero esté desglosado en los grupos 3.1 y 3.2, que si bien son muy similares entre sí, poseen también ciertas diferencias características a tener en cuenta. De manera práctica, este agrupamiento podría determinar clientes morosos, normales, y clientes buenos de grados 1 y 2, en relación a los grupos 3.1 y 3.2 comentados anteriormente.

Así, a continuación se realizarán los mismos experimentos llevados a cabo en el anterior apartado para k = 4 grupos, de manera que se puede llegar a la conclusión de cuál de los agrupamientos realizados es mejor, atendiendo a las diferentes medidas de bondad estudiadas.

En primer lugar, es necesario señalar que para obtener k = 3 grupos con la distribución definida anteriormente, no es posible realizar el agrupamiento jerárquico con k = 3, ya que R reajusta los grupos y agrupa según diferentes criterios.

Para hacer esto, será necesario llevar a cabo un **post-procesamiento** de los datos, de manera que dados los datos de los grupos a los que pertenece cada registro para k = 4 grupos, se condensarán los grupos tres y cuatro, de manera que todos aquellos elementos que pertenecían al grupo cuatro, pertenecerán al grupo tres. Este post-proceso es llevado a cabo ejecutando el siguiente código R:

```
dim(groups_h) <- c(1000,1)
#Los grupos 3 y 4 se condensan en el grupo 3. POST-Procesamiento
groups_h[groups_h == 4] <- 3
k <- 3
```

Una vez llevado a cabo este proceso y re-asignado el número de clusters a tres, puesto que ahora ya solo existen tres grupos y no cuatro, es posible ejecutar los métodos de agrupamiento realizados para k = 4 grupos.

No obstante, antes de esto, se creará del mismo modo que se hizo anteriormente, una matriz para almacenar los resultados de los diferentes métodos de agrupamiento normalizado y sin normalizar para k = 3 grupos. Esta operación es llevada a cabo por el siguiente fragmento de código R.

```
resul_k3 <- matrix(data = 0, nr = 2, nc = 3)
rownames(resul_k3) <- c("Sin Normalizar","Normalizado")
colnames(resul_k3) <- c("Jerárquico","k-medias","k-medioides")
resul_k3
```

Dicha matriz, inicializada en primera a instancia a cero, se muestra en la siguiente imagen:

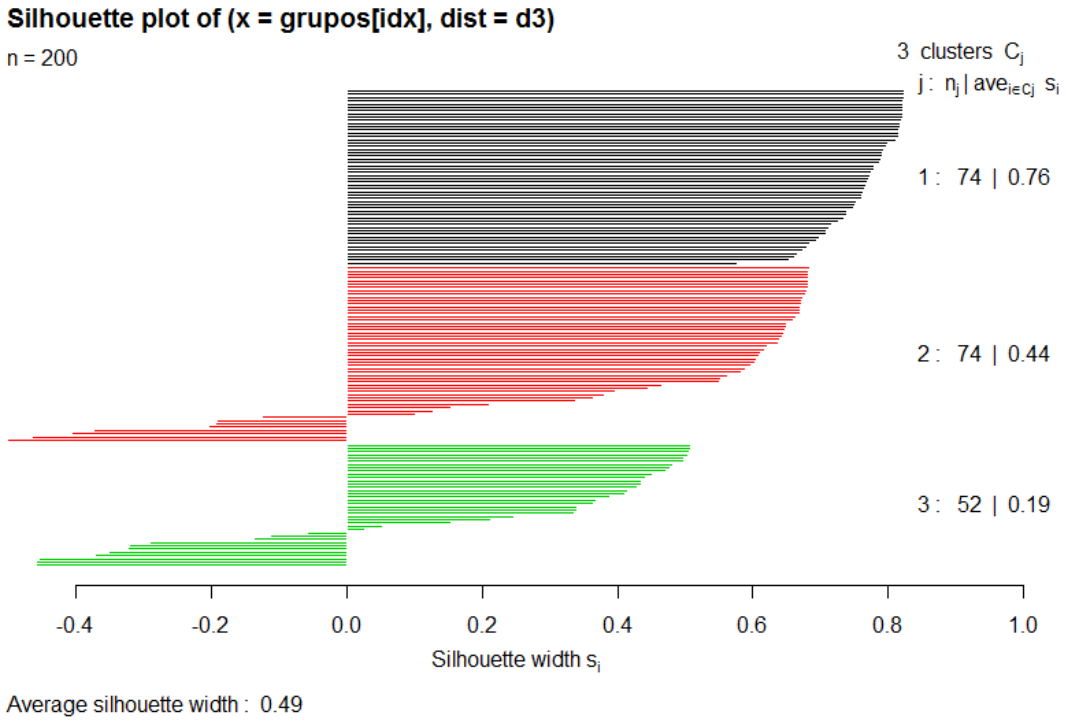
```
                Jerárquico k-medias k-medioides
Sin Normalizar          0        0           0
Normalizado             0        0           0
>
```

**-Clustering Jerárquico.**

El siguiente código R, muestra la ejecución del algoritmo de clustering jerárquico sobre los datos post-procesados.

```
Shi_H_3 <-
Analisis_Bondad(German_Credit,German_Numericas,German_Binarias,groups_h,k,dist_p
onderada)
avg_sil_h_3 <- Average_Silhouette(Shi_H_3)
```

Como se puede observar en el fragmento anterior, no es necesario volver a aplicar el algoritmo de clustering, sino que únicamente se debe calcular el coeficiente de Silueta en base a los nuevos grupos, que han sido post-procesados de manera que solo existan tres grupos, con la configuración que se definió al inicio del apartado. Gráficamente, la siguiente figura muestra el resultado de la ejecución de este algoritmo.



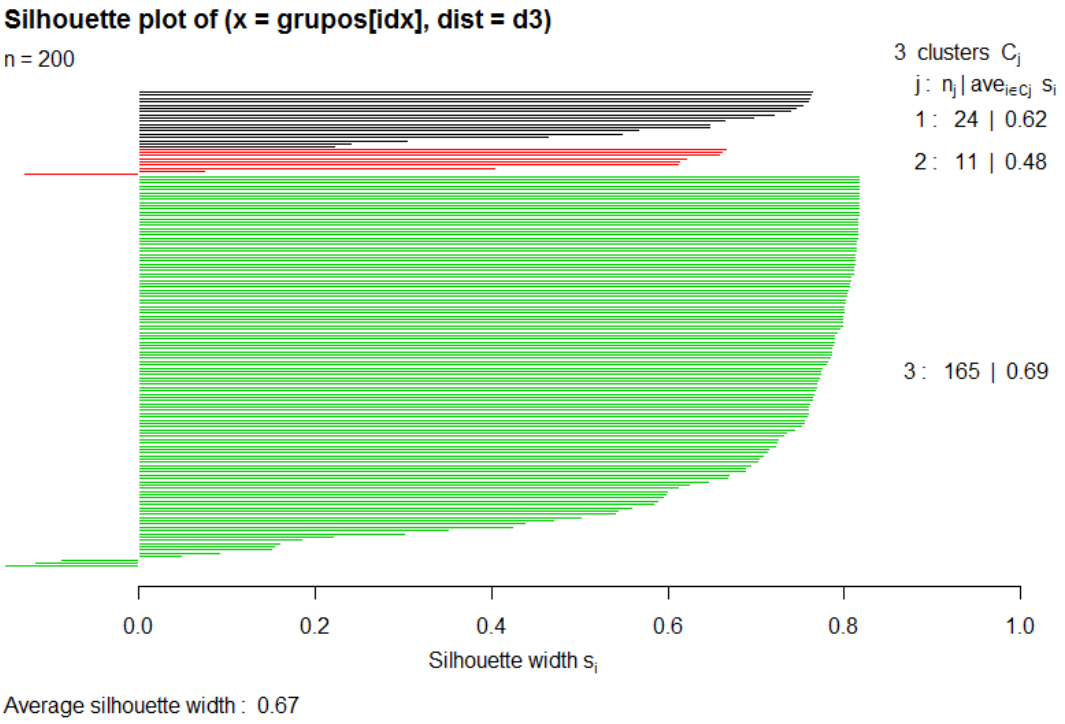

Donde tal y como se puede comprobar, el **coeficiente de silueta medio es de 0.49**, mientras que el grupo que presenta un mayor índice es el primero, con un **0.76** de coeficiente.

**-K-Means**

Por su parte, la ejecución del algoritmo de las k-medias sobre los datos post-procesados para k = 3 grupos se muestra en el siguiente fragmento de código:

```
kmeans.result=kmeans(dist_ponderada,k)
kmeans.result~centers
groups_kmeans_3=kmeans.result$cluster
Sil_kmeans_3 <-
Analisis_Bondad(German_Credit,German_Numericas,German_Binarias,groups_kmeans_3,k
,dist_ponderada)
Sil_kmeans_3
avg_sil_kmeans_3 <- Average_Silhouette(Sil_kmeans_3)
avg_sil_kmeans_3
```

Mientras que gráficamente, el resultado de aplicar este algoritmo puede verse en la siguiente imagen.



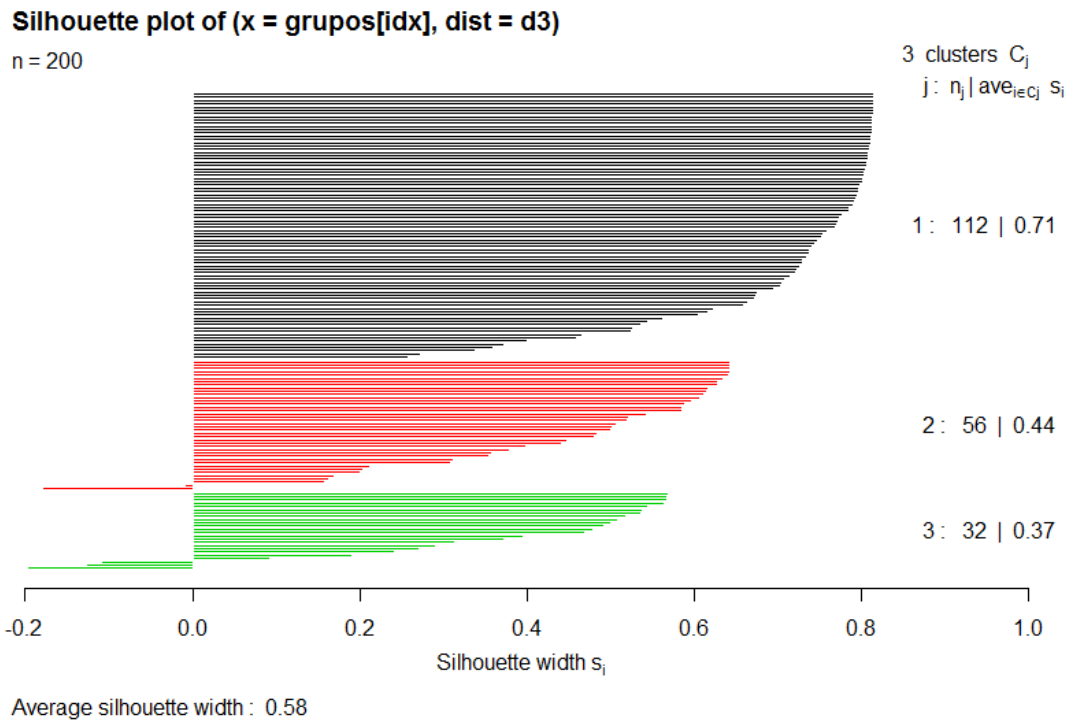

Donde se ha obtenido un coeficiente de silueta medio de **0.67**, mientras que en este caso, el grupo que presenta un mayor coeficiente es el tercero, con **0.69**, seguido muy de cerca por el primer cluster, que presenta un coeficiente de **0.62.**

**-K-Medioides**

Por último, se ejecutará el algoritmo de los k-medioides para k = 3 grupos con los datos post-procesados y sin normalizar. Dicha ejecución, se presenta en el siguiente fragmento de código R:

```
pam.result=pam(dist_ponderada,k)
groups_k_medioids_3 <- pam.result$clustering
sil_kmedioids_3 <-
Analisis_Bondad(German_Credit,German_Numericas,German_Binarias,
groups_k_medioids_3, k, dist_ponderada)
sil_kmedioids_3
avg_sil_kmedioids_3 <- Average_Silhouette(sil_kmedioids_3)
avg_sil_kmedioids_3
```

Mientras que los resultados de la aplicación del algoritmo pueden verse en la siguiente imagen.




Tal y como se puede ver, el coeficiente de silueta obtenido es de **0.58**, mientras que el grupo que obtiene un mayor coeficiente es el primero, que presenta un coeficiente de **0.71**.

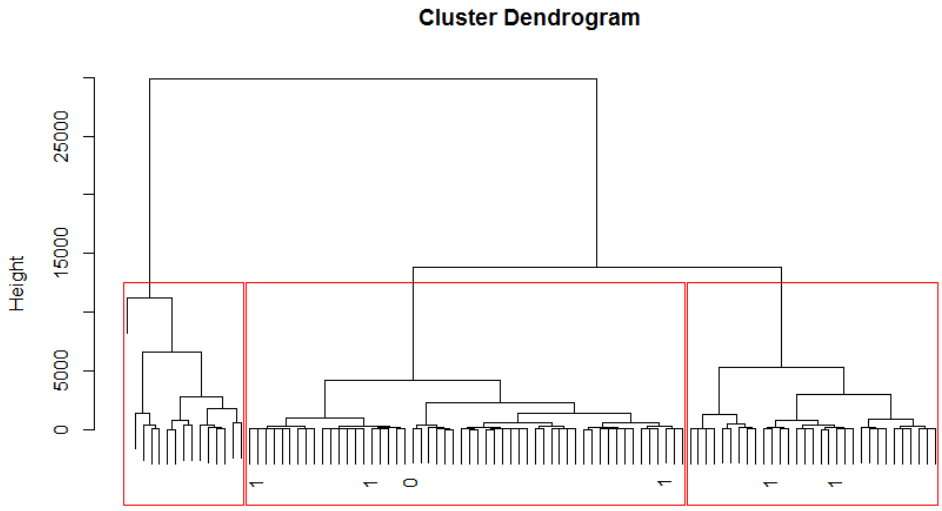
Clustering para k = 3 grupos CON Normalización.

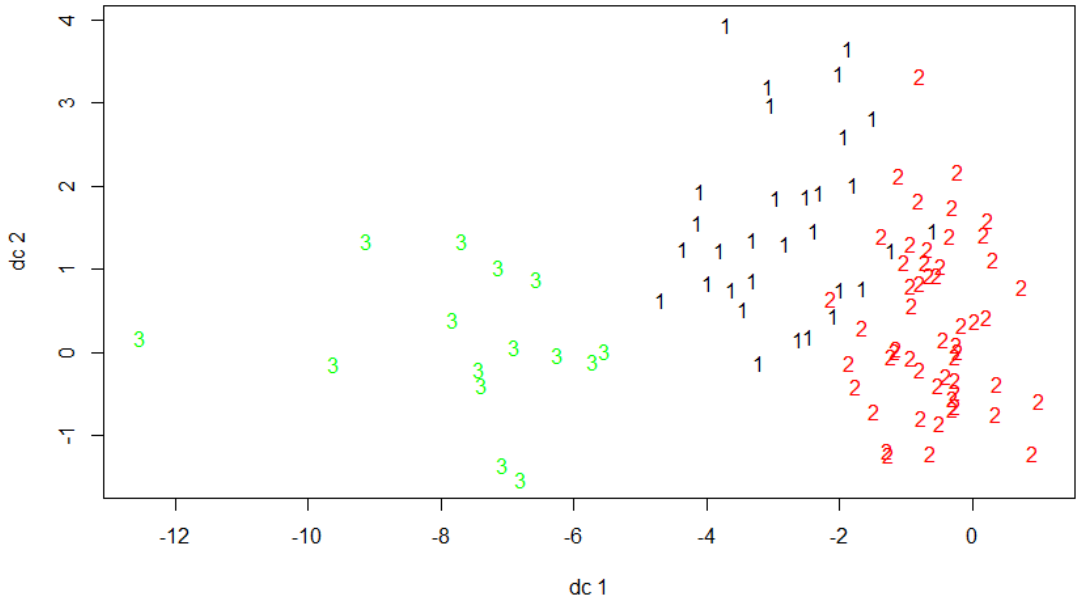
**-Clustering Jerárquico**

Por otra parte, a continuación se muestra el código R para aplicar el algoritmo de clustering jerárquico utilizando distancia euclídea, sobre los datos normalizados.

```
hc=hclust(dist(German_sample),method="ward.D2")
hc
plot(hc,labels=German_sample$Creditability[idx])
rect.hclust(hc,k)
group_hc=cutree(hc,k)
group_hc
#Análisis de Bondad
plotcluster(German_sample,group_hc)
Sil_Hnorm_3 <- silhouette(group_hc,dist(German_sample))
avg_sil_hnorm_3 <- Average_Silhouette(Sil_Hnorm_3)
avg_sil_hnorm_3
```

Donde el dendrograma obtenido así como la distribución de los clusters, se puede ver en las siguientes imágenes adjuntas:




Donde el coeficiente de silueta medio obtenido es de **0.53.**



*Julio Alberto López Gómez*
*Amparo Vila Miranda*

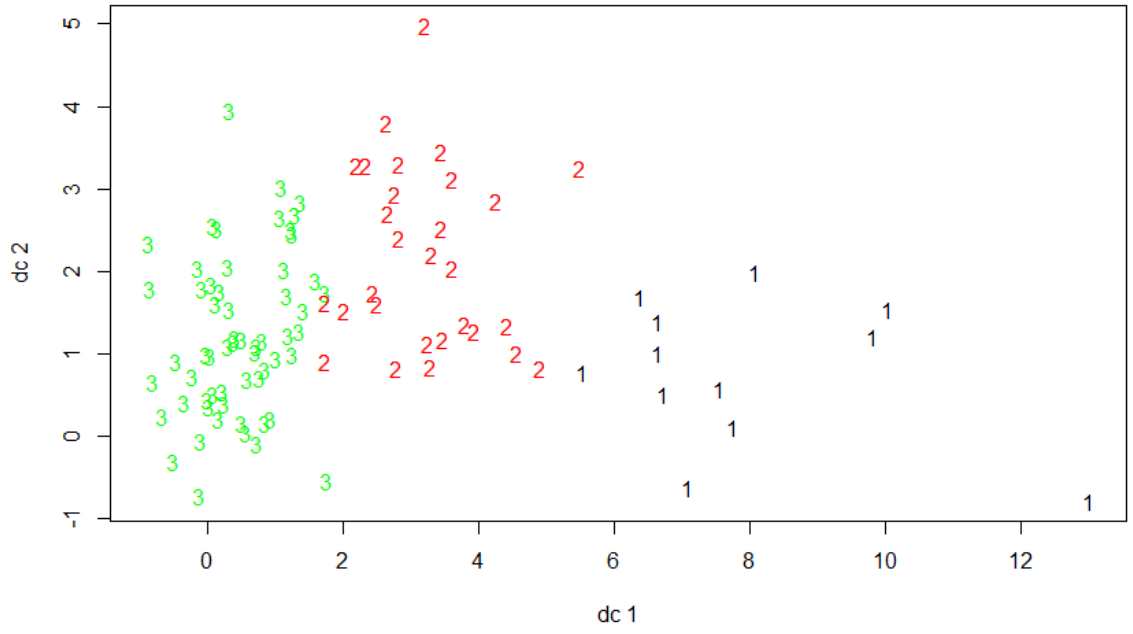
**-K-Medias**

La ejecución del algoritmo de las k-medias puede verse en el siguiente fragmento de código R.

```
kmeans.result=kmeans(German_sample,k)
kmeans.result
table(German_sample$Creditability,kmeans.result$cluster)
#Análisis de Bondad y Gráficas
plot(German_sample[,1:5], col=kmeans.result$cluster)
points(kmeans.result$centers[,],col=1:3,pch=8,cex=2)
x=kmeans.result$cluster
plotcluster(German_sample,x)
sil_kmeans_norm_3= silhouette(kmeans.result$cluster,dist(German_sample))
avg_sil_kmeans_norm_3 = Average_Silhouette(sil_kmeans_norm_3)

avg_sil_kmeans_norm_3
```

Donde la siguiente imagen refleja la distribución de los clusters y la matriz que se adjunta a continuación el total de elementos agrupados en cada cluster, atendiendo a la clase (Creditability) de cada elemento.




Siendo la matriz:

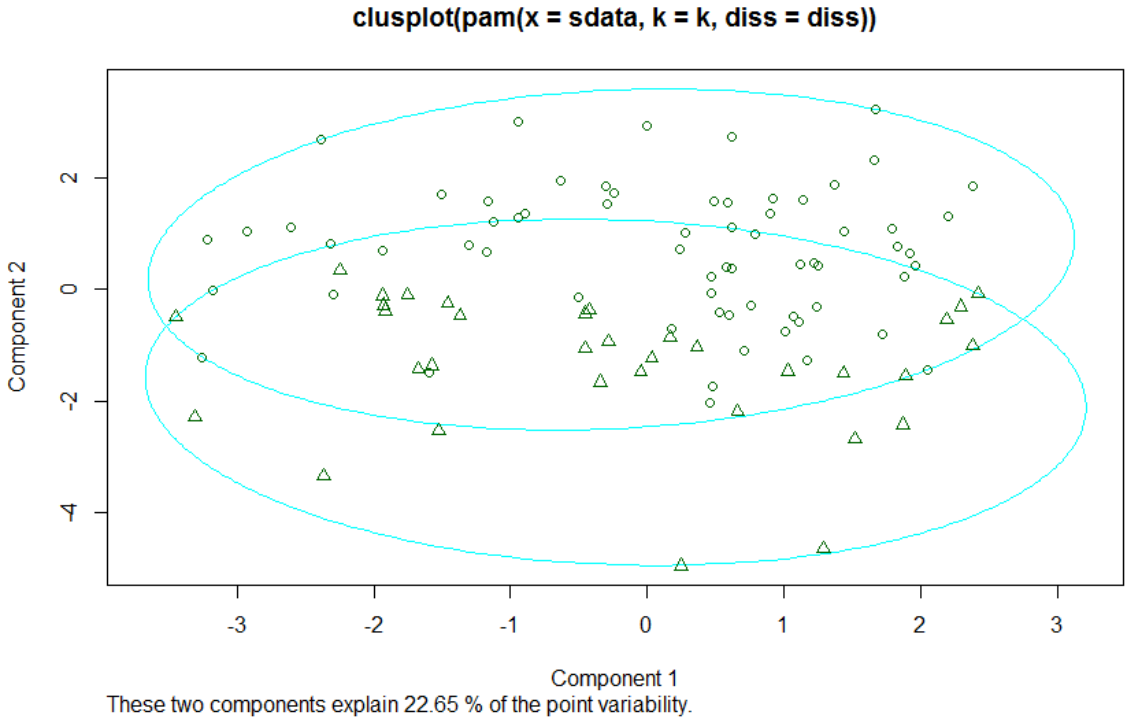
| | 1 | 2 | 3 |
|---|---|---|---|
| 0 | 4 | 7 | 17 |
| 1 | 8 | 22 | 42 |

Y el coeficiente de silueta medio obtenido **0.53**

**K-Medioides**

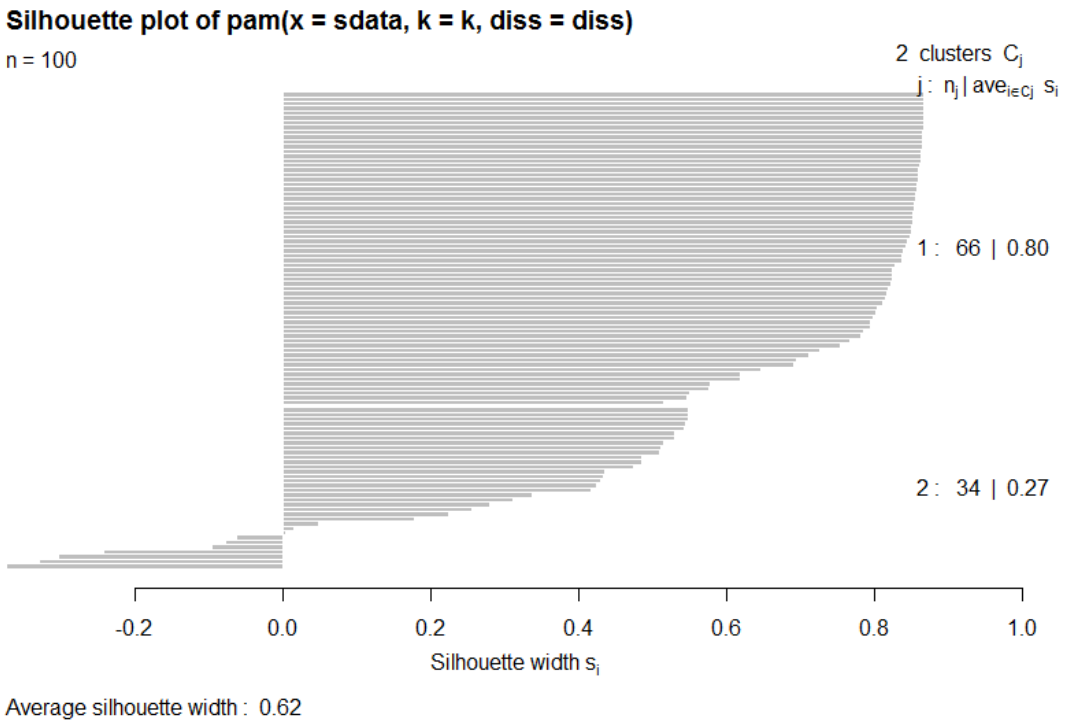
Por último, la ejecución del algoritmo de k-medioides con número de grupos óptimo se muestra a continuación.

```
#Se prueba con número óptimo de grupos
pamk.result=pamk(German_sample)
pamk.result
pamk.result$pamobject$nc
table(pamk.result$pambject$clustering,German_sample$Creditability)
plot(pamk.result$pamobject)
#(Ver descripcion de la funcion)
group=pamk.result$pamobject$clustering
sil_kmedioids_norm_3 <-
silhouette(pamk.result$pamobject$clustering,dist(German_sample))
avg_sil_kmedioids_norm_3 <- Average_Silhouette(sil_kmedioids_norm_3)
avg_sil_kmedioids_norm_3
cluster.stats(dist(German_sample),group)
```






*Julio Alberto López Gómez*
*Amparo Vila Miranda*




Con el número óptimo de grupos **k = 2**, el índice de silueta medio obtenido es de **0.62**, mientras que el coeficiente obtenido por el grupo uno, mejor que el del grupo dos, es de **0.80**.

## Comparativa entre los distintos Agrupamientos.

En este apartado se va a realizar una comparativa estudiando las estructuras de datos donde se han guardado los resultados obtenidos por los distintos algoritmos de agrupamiento para los casos de k = 4 grupos y k = 3 grupos.

Los resultados de los diferentes algoritmos de **clustering**, con y sin normalización para **k=4** grupos, vienen dados por la estructura resul_k4, que se muestra a continuación.

```
resul_k4
               Jerárquico  k-medias k-medioides
Sin Normalizar  0.5196961 0.6136426   0.4644265
Normalizado     0.4285196 0.5079904   0.5330022
```

Por otra parte, los diferentes algoritmos de clustering, con y sin normalización para k=3 grupos, vienen dados por la estructura resul_k4, que se muestra a continuación.

```
resul_k3
              Jerárquico  k-medias k-medioides
Sin Normalizar  0.4702127 0.6810257   0.5063367
Normalizado     0.6330381 0.5585805   0.5193429
```

Por lo que se puede deducir de las dos matrices, el algoritmo que presenta mejores resultados es aquel que define tres grupos y utiliza los datos normalizados, el cual aparece señalado en la matriz correspondiente.

## Interpretación del Agrupamiento

Una vez que se ha elegido de los dos agrupamientos propuestos, el que presentaba mejores resultados a la hora de calcular las medidas de bondad, en este caso, el agrupamiento en tres clusters con los datos de las variables normalizadas.

No obstante, una vez seleccionado este agrupamiento, debe darse una **interpretación** al mismo. Esto es, **¿En base a qué razones o criterios se ha hecho el agrupamiento?, Atendiendo a éstas razones, ¿Es posible asignar una etiqueta a estos grupos de manera que existan k grupos etiquetados?**

Para ello, el siguiente código R muestra cómo se han agrupado los elementos de los diferentes clusters en las variables **cluster_1, cluster_2 y cluster_3.** Para ello, en primer lugar se añadió un atributo más al dataset German_Credit que identifica el grupo al que pertenece cada elemento, atendiendo al agrupamiento normalizado con k = 3 clusters.

```
German_Credit <- cbind(German_Credit,group_hc)
cluster_1 <- German_Credit[which(German_Credit$group_hc==1),]
cluster_2 <- German_Credit[which(German_Credit$group_hc==2),]
cluster_3 <- German_Credit[which(German_Credit$group_hc==3),]
```

Una vez que se tienen los elementos que forman parte de cada cluster, se calculará la media de cada atributo para todos los elementos de un cluster. Este cálculo se muestra en el siguiente fragmento de código R.

```
m_cluster_1 <- apply(cluster_1,2,mean)
m_cluster_2 <- apply(cluster_2,2,mean)
m_cluster_3 <- apply(cluster_3,2,mean)
```

Una vez calculadas las medias, es posible comprobar que el principal argumento donde varían las medias de los atributos para cada grupo es **Credit Amount,** de hecho, es fácil comprobar como los dos primeros grupos tienen un valor en este atributo de aproximadamente 3.300 y 3.500, mientras que el tercer grupo, que presenta menor similitud con los otros dos, tal y como se aprecia en el dendrograma, presenta un valor de 3000.

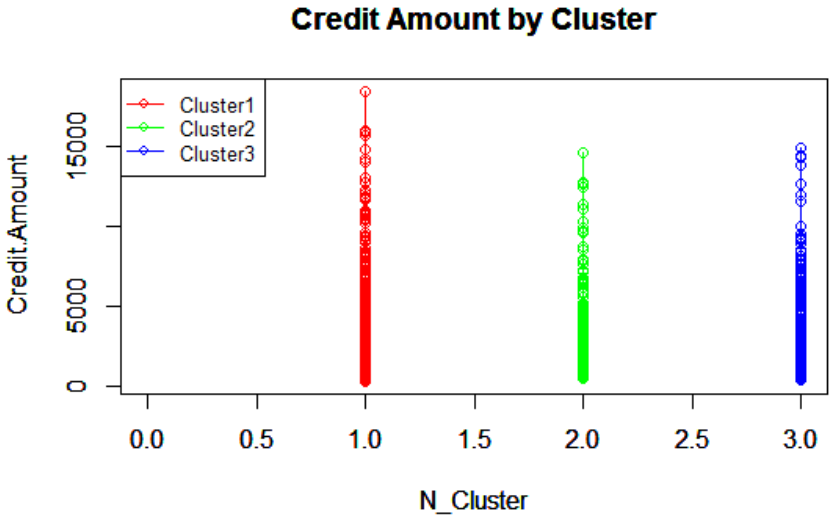
El siguiente atributo cuya media presenta diferencias significativas en los tres grupos es **Age (years).** Donde los valores para cada grupo, respectivamente, son de aproximadamente 34, 35 y 36 años.

Otra alternativa para averiguar cuáles son los atributos por los cuáles se realiza el agrupamiento, es construir un **árbol de decisión** que permita predecir para una nueva entrada, cuál será el procesamiento de la misma para agruparla dentro de uno u otro cluster. Por motivos de tiempo y simplicidad, se ha elegido la del cálculo de la media en lugar de esta opción que se contempla como alternativa posible.

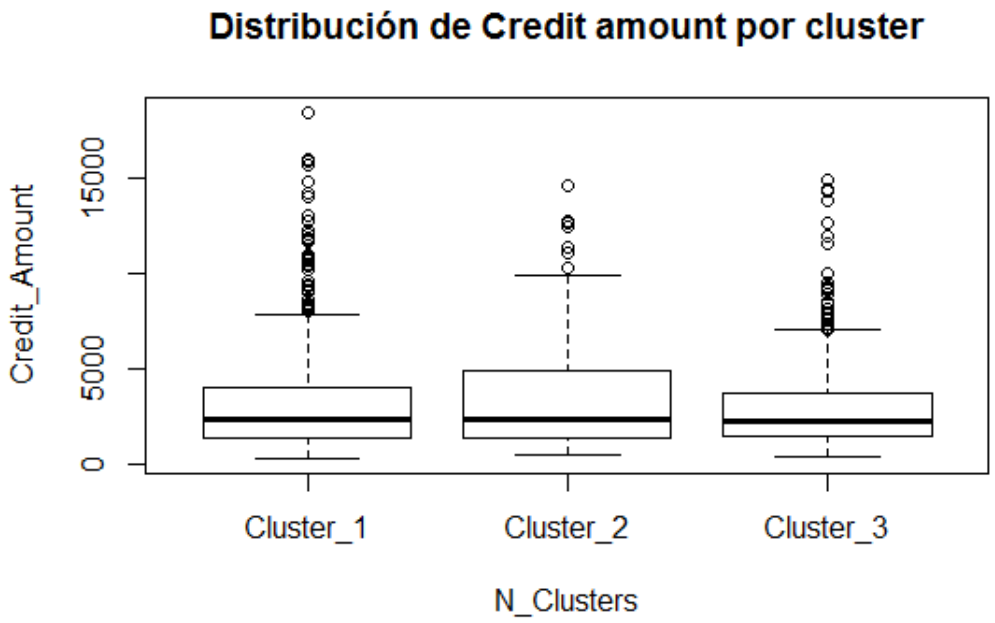
Por último, y para hacer más notable esta diferencia, se han generado dos gráficos que representan **la distribución de Credit.Amount para cada uno de los clusters**, de manera que gráficamente se pueden apreciar las diferencias que se notaron anteriormente con el cálculo de la media. El primero de ello es un gráfico usual, mientras que el segundo es un diagrama de cajas, que permite visualizar más claramente la distribución de una variable, y que a buen seguro, clarificará las dudas que se puedan tener de la interpretación del gráfico anterior.

El código R para el gráfico usual que representa la distribución de Credit.Amount para cada cluster, así como el resultado del mismo, pueden verse a continuación.

```
rango <- range(German_Credit$Credit.Amount)
plot(cluster_1$group_hc,cluster_1$Credit.Amount, type = "o", xlim = c(0,3),ylim
= rango, col = "red", main = "Credit Amount by Cluster", xlab = "N_Cluster",
ylab = "Credit.Amount")
par(new=TRUE)
plot(cluster_2$group_hc,cluster_2$Credit.Amount, type = "o", xlim = c(0,3),ylim
= rango, col = "green", main = "Credit Amount by Cluster", xlab = "N_Cluster",
ylab = "Credit.Amount")
par(new = TRUE)
plot(cluster_3$group_hc,cluster_3$Credit.Amount, type = "o", xlim = c(0,3),ylim
= rango, col = "blue", main = "Credit Amount by Cluster", xlab = "N_Cluster",
ylab = "Credit.Amount")
legend("topleft",legend = c("Cluster1","Cluster2","Cluster3"), cex = 0.8, col =
c("red","green","blue"),pch = 21, lty = 1)
```



Mientras que el código R que realiza el diagrama de cajas para la distribución de estas tres variables, así como el diagrama en sí mismo, se muestran a posteriori.




**En conclusión, el agrupamiento seleccionado es influido principalmente por una única variable (Credit.Amount). Es por esto, que a continuación, debería realizarse de nuevo todo el procedimiento visto hasta aquí para encontrar un agrupamiento mejor. Sin embargo, esto no se ha realizado debido a las limitaciones de tiempo, mostrándose a lo largo de este documento y del script entregado el procedimiento completo de clustering, evaluación e interpretación, aunque los resultados no hayan sido óptimos.**