



OBLIGATORIO

Machine Learning No Supervisado

Lic. En Gerencia y Administración

Docente: Ec. Damian Coltzau

Machado Cecilia – N° 213640

Fecha de entrega: 18 de Noviembre de 2020

Índice

1 – Introducción	3
- Descripción de la base de datos.	3
2 – Objetivo	3
3 – Análisis	3
3.1 - Análisis descriptivo.	3
3.1.1 Medidas de tendencia y separación de las variables:	3
3.2 - Representación gráfica de la estructura	4
3.3 - Selección de variables.	5
3.4. Validación de Clustering.	8
4 – Clusterización	10
4.1 Modelo K-Means.	10
4.2 Clasificación mediante modelo KNN (K-Nearest Neighbor).	12
5 - Anexos	13

1 – Introducción

1.1 - Descripción de la base de datos.

La base de datos utilizada en la elaboración del siguiente informe se denomina “*pandemia.csv*”. Dicha muestra contiene información sobre factores médicos de 2000 individuos. Dichos factores se encuentran representados en 6 dimensiones. Estas son:

- *Hemoglobina*: proteína encargada de transportar el oxígeno a través de la sangre.
- *Glúcidos*: representa la concentración de azúcar en sangre.
- *Temperatura*: temperatura corporal del individuo.
- *LDL*: indica el nivel de una lipoproteína (low density lipoprotein) encargada de transportar el colesterol en sangre.
- *Ondas*: actividad eléctrica producida por el cerebro
- *Defensas*: conjunto de elementos y procesos biológicos en el organismo que permite mantener la homeóstasis frente a agresiones externas.

2 – Objetivo

El objetivo del presente informe será detectar potenciales portadores de un virus para aislarlos y efectuar sobre los mismos mediciones adicionales, utilizando esta herramienta como clasificador para nuevos individuos, contenidos en la base “*nuevos.csv*” .

3 – Análisis

3.1 - Análisis descriptivo.

3.1.1 Medidas de tendencia y separación de las variables:

```
> summary(p)
hemoglobina      glucidos      temperatura
Min.   :-2.99658  Min.   :-3.063539  Min.   :-2.72951
1st Qu.: -1.56521  1st Qu.: -0.626107  1st Qu.: -0.85385
Median : -0.08857  Median : -0.000158  Median :  0.09788
Mean   : -0.03694  Mean   :  0.000000  Mean   :  0.00000
3rd Qu.:  1.58631  3rd Qu.:  0.606732  3rd Qu.:  0.78608
Max.   :  2.99918  Max.   :  4.754172  Max.   :  2.71040

ldl              ondas              defensas
Min.   :-2.99637  Min.   :-3.1575    Min.   :-3.9357
1st Qu.: -1.55462  1st Qu.: -0.4027    1st Qu.: -0.3022
Median : -0.10725  Median :  0.2499    Median :  0.3575
Mean   : -0.02952  Mean   :  0.0000    Mean   :  0.0000
3rd Qu.:  1.52569  3rd Qu.:  0.6914    3rd Qu.:  0.6271
Max.   :  2.99958  Max.   :  1.9858    Max.   :  1.5662
```

3.1.2 - Verificación de normalización de la base de datos

```
> colMeans(p)
hemoglobina      glucidos  temperatura          ldl          ondas      defensas
-3.694214e-02  1.657355e-17 -6.784634e-17 -2.951730e-02  3.882311e-17 -4.042123e-17

> apply(p, 2, var)
hemoglobina      glucidos  temperatura          ldl          ondas      defensas
  3.149679      1.000000      1.000000      3.078947      1.000000      1.000000
```

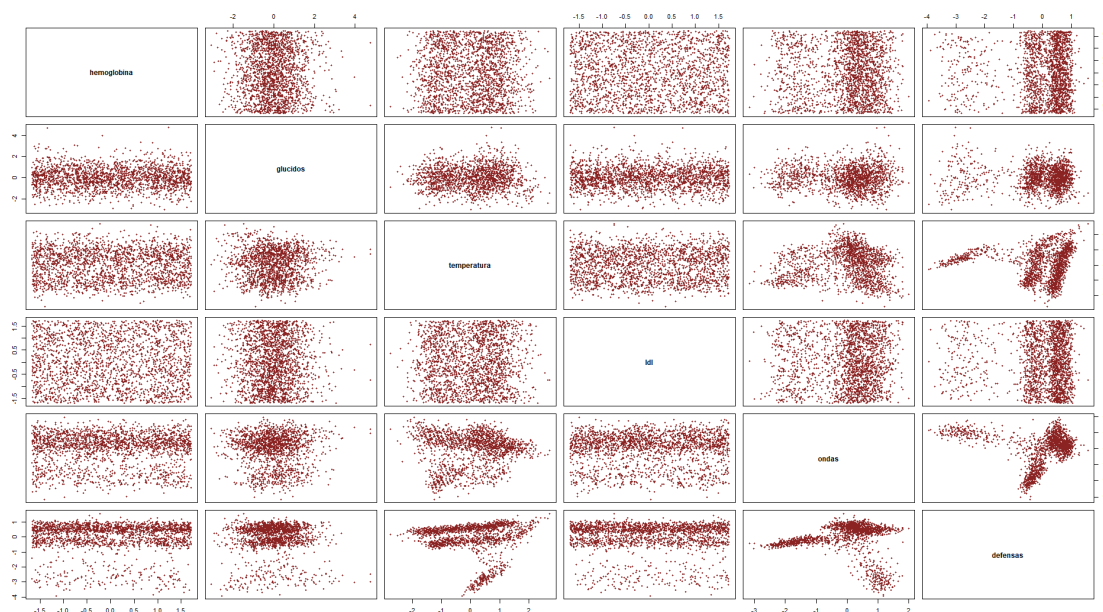
De los datos obtenidos podemos observar que existen dimensiones sin normalizar (hemoglobina y ldl), por lo que procedemos a escalarlos para poder proceder con nuestro análisis. Los siguientes datos confirman la normalización:

```
> colMeans(pscaled)
hemoglobina      glucidos  temperatura          ldl          ondas      defensas
1.383832e-17  1.158275e-17 -1.473778e-17  1.385762e-17  2.113761e-17 -1.831608e-17

> apply(pscaled, 2, var)
hemoglobina      glucidos  temperatura          ldl          ondas      defensas
           1           1           1           1           1           1
```

3.2 - Representación gráfica de la estructura

Una vez normalizados los datos, procedemos a realizar un análisis visual de manera de verificar la existencia de relación y estructura entre las dimensiones:



Observamos cierta estructura entre algunas de las dimensiones. Las que muestran una correlación más evidente son: “ondas” y “defensas” con respecto a las demás.

Asimismo se pueden apreciar dimensiones que no presentan estructura, tendiendo incluso a la uniformidad por lo que consideramos pertinente eliminar algunas de ellas mediante la aplicación del Modelo de Hopkins.

3.3 - Selección de variables.

Comenzamos analizando el valor del **estadístico de Hopkins** incluyendo todas las dimensiones de la base de datos; su valor es: 0.7289176. Se muestra a continuación su representación gráfica donde se puede apreciar la falta de estructura:

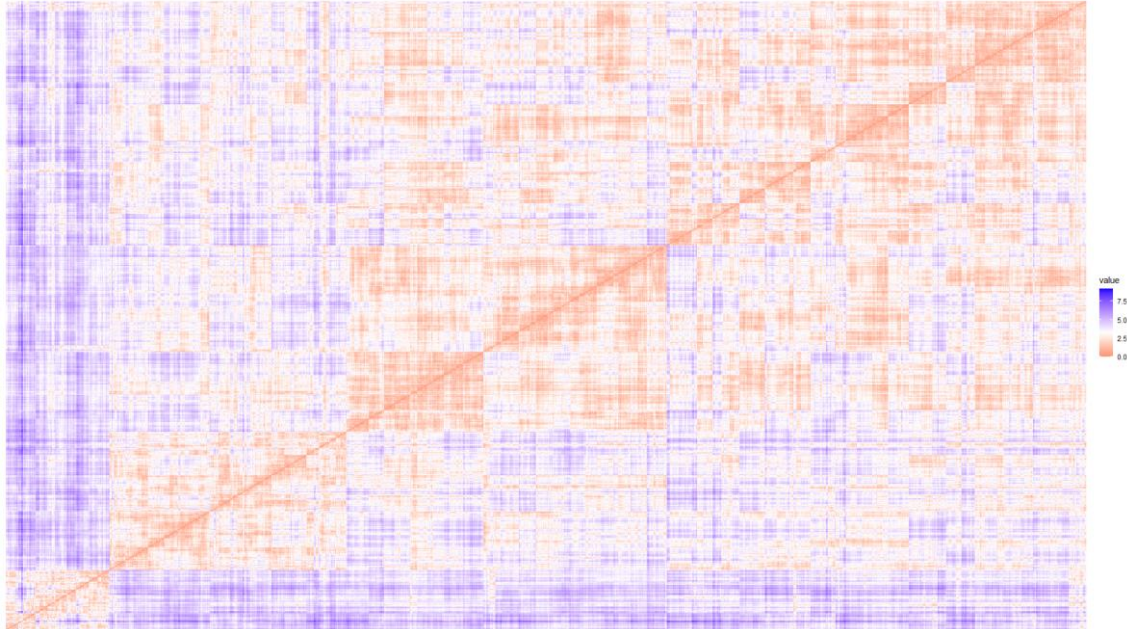


Gráfico: VAT de la base de datos con todas sus dimensiones.

Mediante un proceso iterativo obtenemos el valor del estadístico extrayendo de a una dimensión a la vez. De esta forma conservaremos las dimensiones que maximicen dicho coeficiente, lo cual garantiza eliminar las que no aporten valor a la estructura.

Tras la primera iteración observamos que eliminando la dimensión 4 (Ldl) el estadístico de Hopkins asciende a: 0.8003483. Obteniendo así el siguiente gráfico:

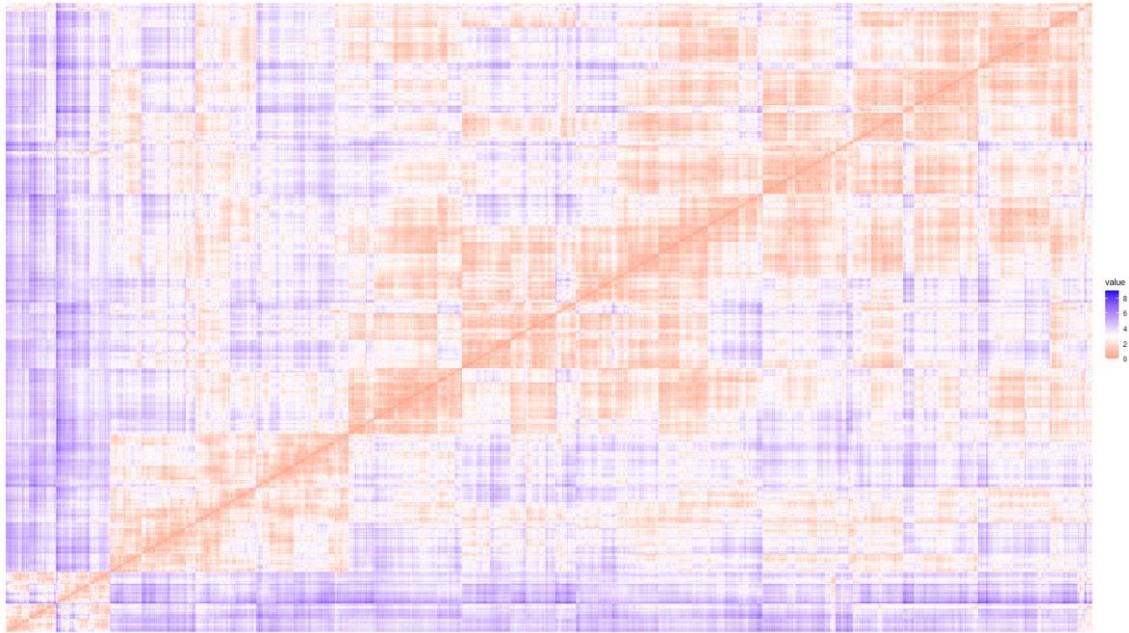


Gráfico: VAT de la base de datos quitando la dimensión ldl (4).

A continuación removemos la dimensión 1 (Hemoglobina) arrojando un resultado de: 0.8503175.



Gráfico: VAT de la base de datos quitando la dimensión ldl (4) y hemoglobina (1).

Posteriormente pasamos a remover la dimensión 2 (Glúcidos) obteniendo un estadístico de: 0.9166725.

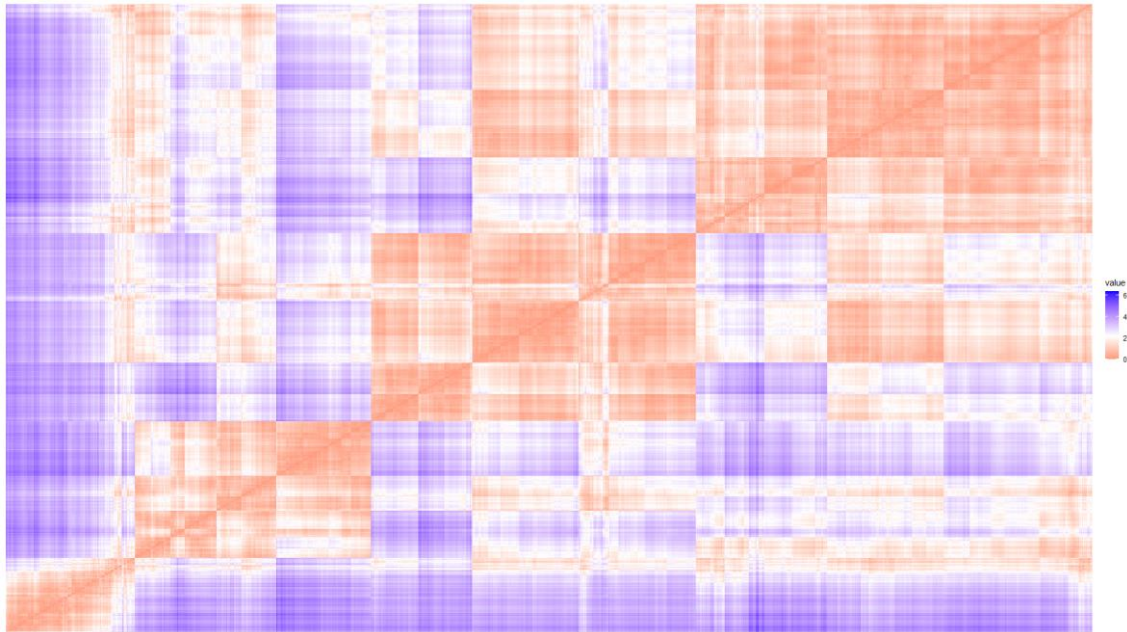


Gráfico: VAT de la base de datos quitando la dimensión ldl (4), hemoglobina (1) y glúcidos (2).

Finalmente quitamos la dimensión 3 (temperatura) lo que nos da un resultado del estadístico de Hopkins de: 0.9566463. De esta manera las dimensiones resultantes son la 5 y la 6 correspondientes a “Ondas” y “Defensas” respectivamente.

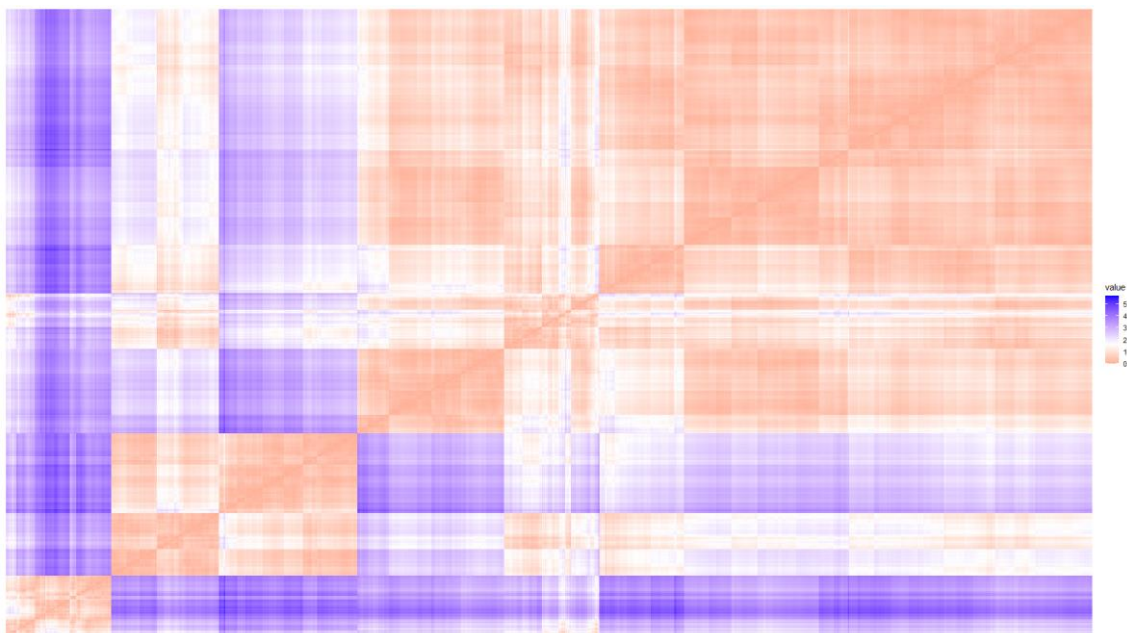
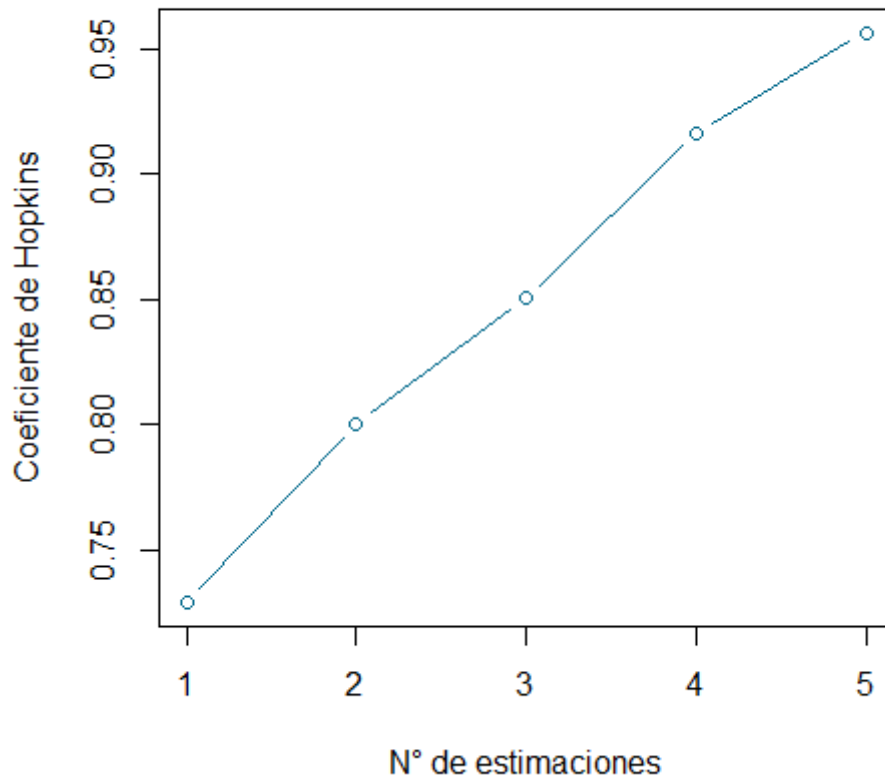


Gráfico: VAT de la base de datos quitando la dimensión ldl (4), hemoglobina (1), glúcidos (2), y temperatura (3).

Podemos observar en el siguiente gráfico la evolución favorable del estadístico de Hopkins a medida que se van removiendo dimensiones:

Evolución de estadístico de Hopkins

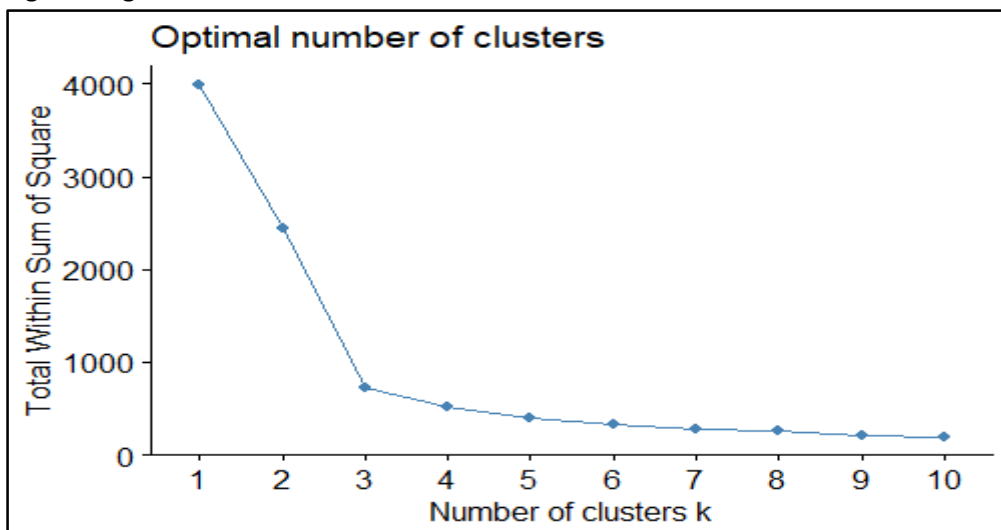


3.4. Validación de Clustering.

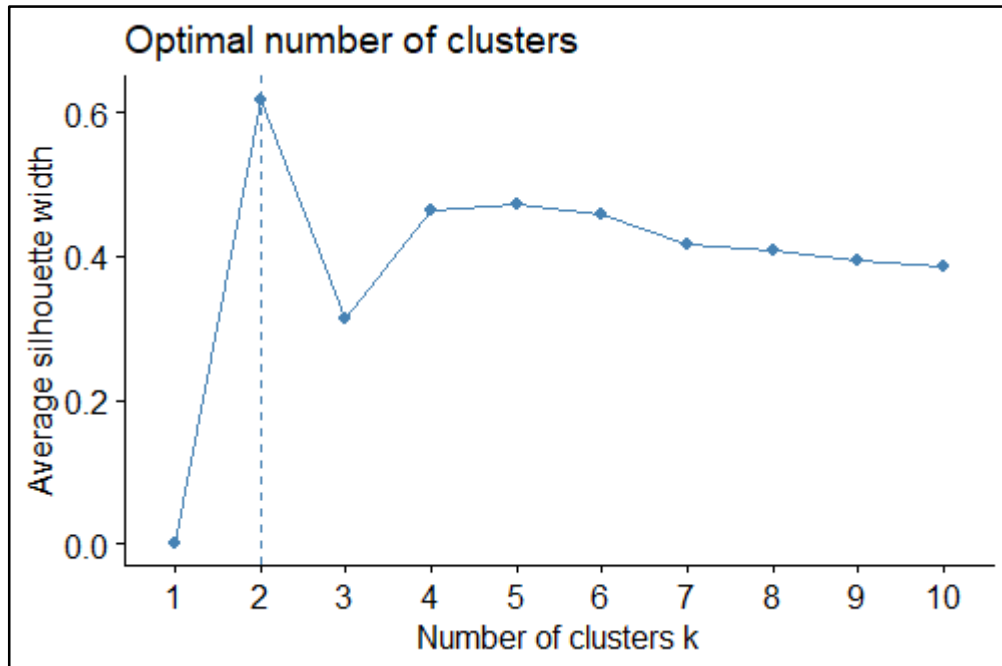
Una vez que encontramos las dimensiones que maximizan el estadístico de Hopkins procedemos a definir el número óptimo de clusters. Para ello recurrimos a tres técnicas:

Método de codo, silueta y GAP.

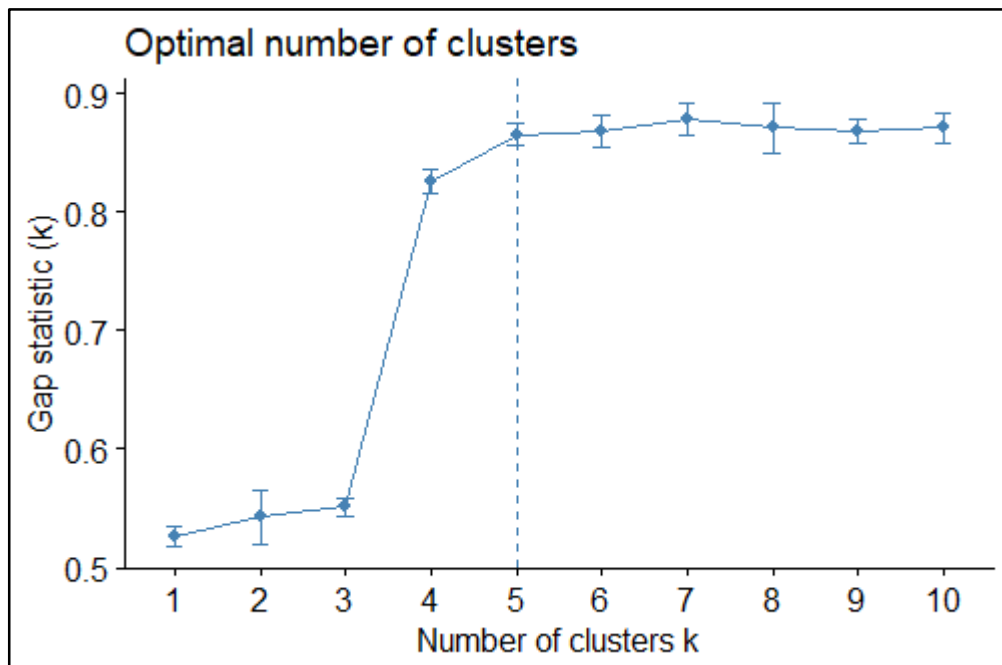
Mediante el método de **Codo** el modelo arrojó un óptimo de **tres clusters**, lo que podemos ver en el siguiente gráfico:



Aplicamos luego el método **Silueta**, obteniendo la siguiente representación gráfica, la cual indica un óptimo de **dos clusters**:



Por último recurrimos al método de **GAP**, el cual sugiere un óptimo de **cinco clusters**, lo que podemos interpretar en el gráfico mostrado a continuación:



Tras la interpretación de las salidas de los tres métodos y aplicando el conocimiento específico, que sugiere clasificar las observaciones en “infectados” y “no infectados” **concluimos que el número óptimo de cluster para la ejecución de K-means es de dos**, lo cual coincide además con el método **Silueta**.

4 – Clusterización

4.1 Modelo K-Means.

De manera de obtener el “Guess” inicial óptimo ejecutamos K-Means múltiples veces fijando distintas semillas.

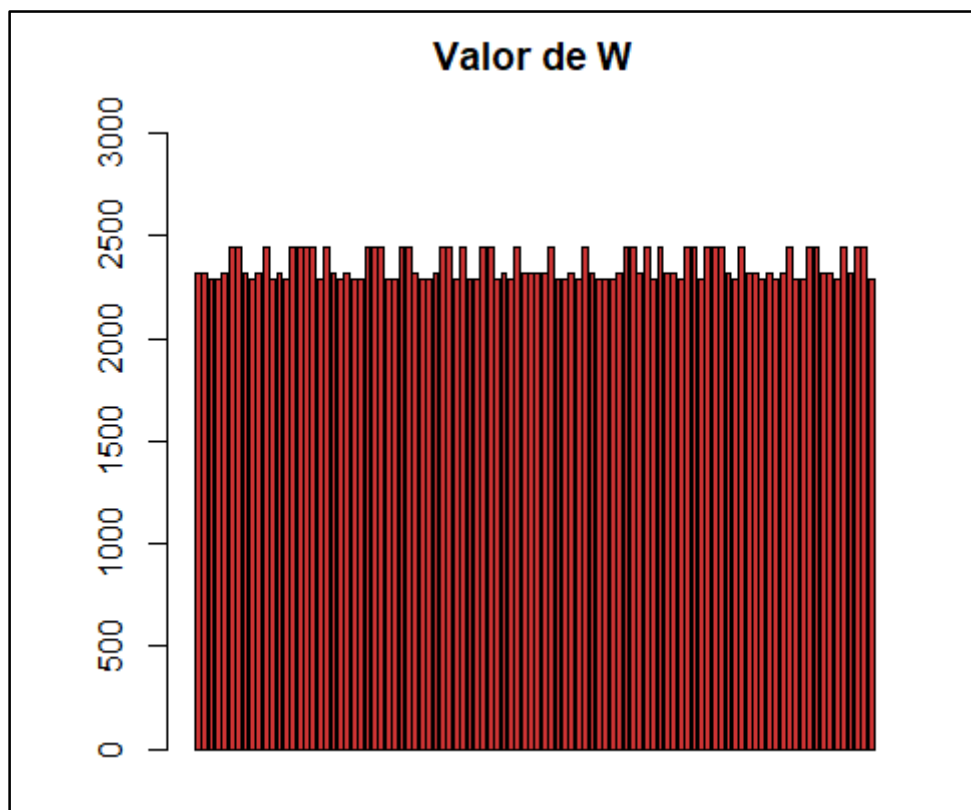
Seleccionamos a continuación aquel “Guess” inicial que brinde el menor valor de W (cuadrados intra cluster) , que a su vez repercutirá en un mayor valor de B (cuadrados inter cluster). Esto nos garantizará cluster más compactos y distantes entre sí.

Se muestra a continuación la iteración utilizada para este procedimiento:

```
n = 100
k = 2
w = integer(n)

for (i in 1:n) {
  set.seed(i)
  q <- kmeans(pdepurada, 2)
  w[i] <- q$tot.withinss
}
```

Una vez ejecutada la misma, analizamos los valores almacenados en nuestro vector llamado “w” procurando encontrar el menor W de todos:



Obtenemos como resultado, que el menor valor de W obtenido es aquel obtenido al ejecutar la semilla #3:

```
> ubicacion_menor_w = which.min(w)
> ubicacion_menor_w
[1] 3
>
> valor_menor_w = min(w)
> valor_menor_w
[1] 2292.332
```

Tras los resultados encontrados, podemos replicar el “Guess inicial” brindado por la semilla mencionada con anterioridad en el modelo K-Means, obteniendo así la mejor clasificación dentro de los 100 ensayos realizados.

Aplicamos K-Means con el mejor modelo encontrado, tras lo cual obtenemos la siguiente distribución de observaciones:

```
> ubicacion_centros=q$centers
> ubicacion_centros
      ondas  defensas
1 -0.1070666  0.2867828
2  0.9744145 -2.6100131
```

- El cluster 1 está caracterizado por aquellas observaciones que tienen valores de ondas y defensas próximas a la media (-0,1070 y 0,2867 respectivamente).
- El cluster 2 agrupa a las observaciones con valores más extremos: ondas por encima de la media (0.9744) y defensas muy por debajo de la media (-2.6100).

El segundo clúster por ser el que contiene observaciones anómalas (#2) podría ser identificado como el de sujetos “**infectados**” mientras que el otro concentra las observaciones con valores normales asociadas a los individuos “**no infectados**”.

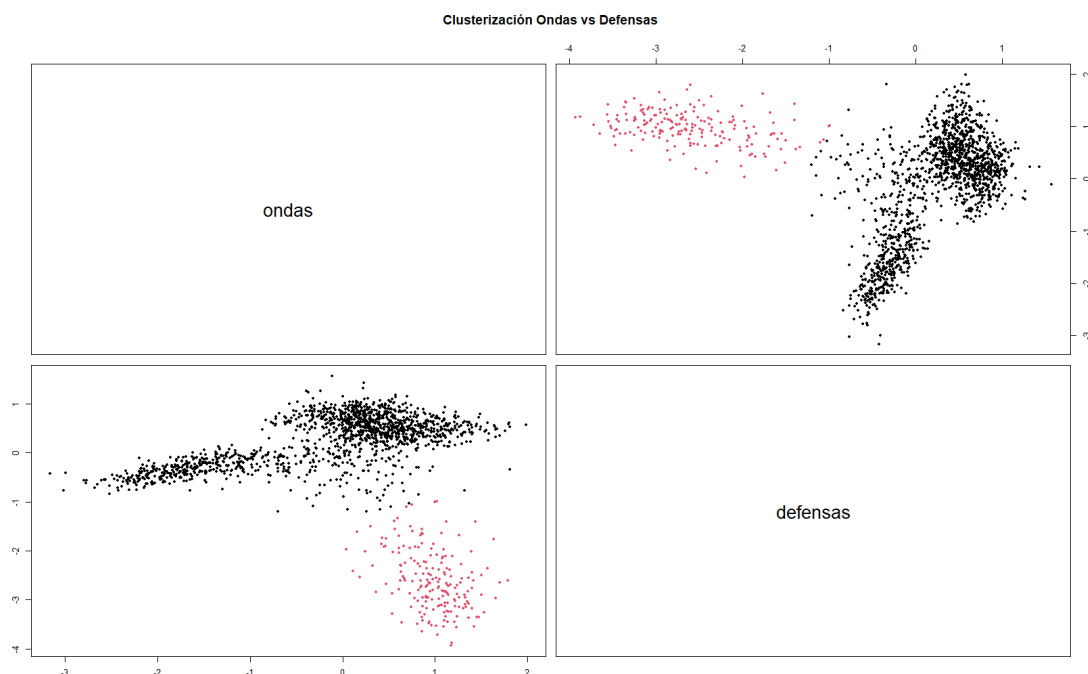


Gráfico: datos originales de pandemia (escalados) Clusterizados con K-Means

El cluster 1 “no infectados” cuenta con un total de 1802 observaciones mientras que el cluster 2 “infectados” agrupa a 198 observaciones.

Los resultados de esta clasificación se encuentran en el vector “clasificacion_clusters” donde es posible identificar la pertenencia de cada observación a su respectivo clúster .

4.2 Clasificación mediante modelo KNN (K-Nearest Neighbor).

Usando el modelo obtenido mediante K-Means, procederemos a clasificar los nuevos casos de la base “nuevos.csv” dentro de los clusters anteriormente definidos.

```
nuevos_casos = knn(q$centers, nscaled, 1:k, k = 1)
nuevos_casos
```

Esto nos permitirá identificar posibles “infectados” y “no infectados” en base al conocimiento generado.

Como resultado, de un total de 4435 observaciones, detectamos que 4023 se agrupan en el cluster 1 “no infectados” mientras que las 412 restantes lo hacen en el cluster 2 “infectados”.

```
> summary(nuevos_casos)
  1    2
4023 412
```

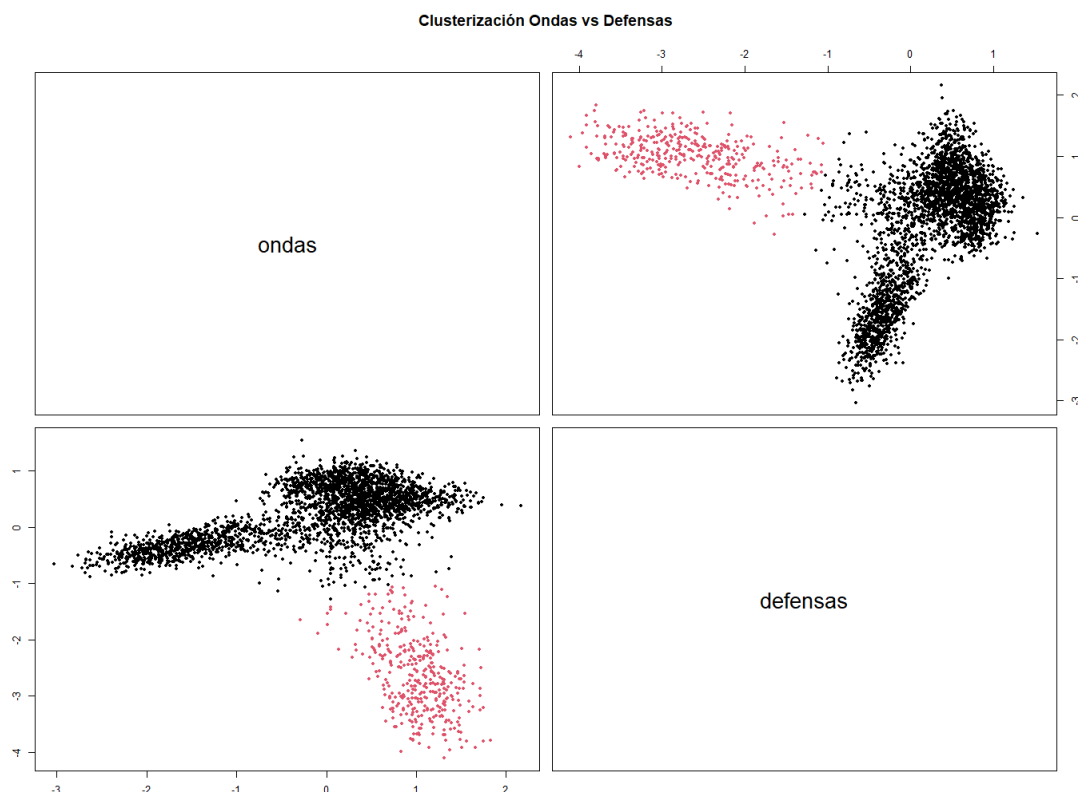


Gráfico: Nuevos casos escalados y clusterizados con el modelo obtenido

Los casos identificados en la nueva base pueden ser encontrados para futuros análisis en la matriz “ncasos” del script adjunto en anexos.

5 - Anexos

```
#####  
#####-----Universidad ORT Uruguay-----#####  
#####-----Obligatorio Machine Learning NO Supervisado-----#####  
#####-----Machado | Pioli | Silveyra-----#####  
#####-----Prof. Damian Coltzau-----#####  
#####
```

```
#=====
```

```
#*****
```

```
# Preámbulo
```

```
#*****
```

```
# Borramos datos de la memoria
```

```
rm(list = ls())
```

```
# Establecemos directorio de trabajo
```

```
setwd("C:/Users/Cecilia Machado/Desktop/Machine")
```

```
# Cargamos librerias a utilizar
```

```
library(factoextra)
```

```
library(class)
```

```
# Cargamos el archivo "Pandemia.csv"
```

```
p = read.csv('pandemia.csv')
```

```
# Visualizamos los datos
```

```
View(p)
```

```
head(p)
```

```
# Resumen de datos más importantes de todas las dimensiones
```

```
summary(p)
```

```
# Verificamos si los datos están escalados
```

```
colMeans(p)
```

```
apply(p, 2, var)
```

```
# Hay dimensiones sin escalar. Se normalizan
```

```
pscaled = scale(p)
```

pscaled

```
# Confirmamos normalización
colMeans(pscaled)
apply(pscaled, 2, var)
```

```
# Se confirma normalización ya que todas las desviaciones son = 1
```

```
# Fin del preambulo
```

```
#=====
```

```
# *****
```

```
# Introducción
```

```
# *****
```

```
# Se aplicarán métodos de selección de variables para depurar la base de datos de aquellas
dimensiones que puedan
# ocasionar distorsiones en la futura clusterización.
```

```
# *****
```

```
# Selección de variables
```

```
# *****
```

```
# Visualizamos los datos para observar si tienen algún tipo de estructura a simple vista
```

```
pairs(pscaled, col = "brown3", main = "Estructura de Pandemia.csv", pch = 20)
```

```
# Se observa que hay estructura en algunas dimensiones
# pero es necesario eliminar aquellas que presentan una estructura
# Uniforme y pueden ocasionar clusters distorsionados
```

```
#####
```

```
# Aplicación de Modelo de Hopkins#
```

```
#####
```

```
# Verificamos el coeficiente con todas las dimensiones
```

```
n = 50 # cantidad total de dimensiones de la base de datos
```

```
Hop_Total = get_clust_tendency(pscaled, n, graph = FALSE)
Hop_Total
```

```
# Verificamos el modelo sacando dimensiones de a una, de esta manera vemos si el coeficiente
mejora
```

```
v = integer(6)
```

```
for(i in 1:6) {  
  ni = -1*i  
  qi = get_clust_tendency(pscaled[,ni ], n, graph = FALSE)  
  v[i]=qi  
}  
v
```

```
# Sancando la dimensión 4 (Idl) se obtiene mejor coeficiente que Hop_Total
```

```
Hop_sin_4 = get_clust_tendency(pscaled[, -4], n, graph = FALSE)  
Hop_sin_4
```

```
# Verificamos el modelo sacando la dimension 4 y otra más
```

```
v_4 = integer(6)  
for(i in 1:6){  
  if(i != 4) {  
    ni = -1*i  
    q_4 = get_clust_tendency(pscaled[,c(-4,ni)], n, graph = FALSE)  
    v_4[i]=q_4  
  }  
}  
v_4
```

```
# Creamos tabla comparando resultados
```

```
tabla1= cbind(v, v_4)  
tabla1
```

```
# Sancando la dimensión 4 (Idl) y 1 (hemoglobina) se obtiene mejor coeficiente que Hop_sin_4
```

```
Hop_sin_4_1 = get_clust_tendency(pscaled[,c(-4,-1)], n, graph = FALSE)  
Hop_sin_4_1
```

```
# Verificamos el modelo sacando la dimension 4, 1 y otra más
```

```
v_4_1 = integer(6)  
for(i in 1:6){  
  ni = -1*i  
  if( (i != 1) & (i != 4)) {  
    q_4_1 = get_clust_tendency(pscaled[,c(-4,-1,ni)], n, graph = FALSE)
```

```
v_4_1[i]=q_4_1
}
}
v_4_1
```

Creamos tabla comparando resultados

```
tabla2= cbind(v, v_4, v_4_1)
tabla2
```

Sancando la dimensión 4 (ldl), 1 (hemoglobina) y 2 (glucidos) se obtiene mejor coeficiente que Hop_sin_4_1

```
Hop_sin_4_1_2 = get_clust_tendency(pscaled[,c(-4,-1,-2)], n, graph = FALSE)
Hop_sin_4_1_2
```

Verificamos el modelo sacando la dimension 4, 1, 2 y otra más

```
v_4_1_2 = integer(6)
for(i in 1:6){
  ni = -1*i
  if( (i != 1) & (i != 4) & (i != 2)) { # no es la columna 1 ni la 2 ni la 4 porque las estoy sacando
    q_4_1_2= get_clust_tendency(pscaled[,c(-4,-1, -2,ni)], n, graph = FALSE)
    v_4_1_2[i]=q_4_1_2
  }
}
v_4_1_2
```

Creamos tabla comparando resultados

```
tabla3= cbind(v, v_4, v_4_1, v_4_1_2)
tabla3
```

```
hist()
```

Sancando la dimensión 4 (ldl), 1 (hemoglobina), 2 (glucidos) y 3 (temperatura) se obtiene mejor coeficiente que Hop_sin_4_1_2

```
Hop_sin_4_1_2_3 = get_clust_tendency(pscaled[,c(-4,-1,-2,-3)], n, graph = FALSE)
Hop_sin_4_1_2_3
```

Visualizamos el aumento del estadístico Hopkins a medida que fuimos depurando la base de datos

```
tabla4 = c(Hop_Total$hopkins_stat, Hop_sin_4$hopkins_stat, Hop_sin_4_1$hopkins_stat,
```



```
Hop_sin_4_1_2$hopkins_stat, Hop_sin_4_1_2_3$hopkins_stat)
tabla4

plot(tabla4, xlab = "N° de estimaciones" , ylab = "Coeficiente de Hopkins",
      main = "Evolución de estadístico de Hopkins", type = "b", col="deepskyblue4")

# *****
# Validación de clustering
# *****

# Base de datos depurada

pdepurada = pscaled[,c(-4,-1,-2,-3)]

#Visualizamos que haya quedado correctamente depurada
head(pdepurada)

# Buscamos cantidad de clusters óptima por método de Codo
validCodo = fviz_nbclust(pdepurada, FUNcluster = kmeans, method = 'wss', k.max = 1e1, nboot
= 5e1)
validCodo

# El método del codo parece mostrarnos que el número óptimo de clusters es 3, lo cual resulta
contraintuitivo
# con el hecho de que únicamente buscamos infectados y no infectados (2 clusters)

# Aplicación de Silueta para validar la cantidad óptima de K
silueta = fviz_nbclust(pdepurada, FUNcluster = kmeans, method = 'silhouette', k.max = 1e1)
silueta

# El método Silueta nos muestra que el número óptimo de clusters es 2, lo cual coincide con el
conocimiento
# específico mencionado anteriormente

# Aplicación de estadístico de GAP

gap = fviz_nbclust(pdepurada, FUNcluster = kmeans, method = 'gap', nboot = 1e1)
gap

# Misma situación anterior: GAP statistic nos indica que K* es 5, sin embargo esto
# no concide con el conocimiento específico

#*****
#Aplicación de Modelo K-Means
#*****
```

```
# Plantamos semilla para fijar resultados

n = 100
k = 2
w = integer(n)

for (i in 1:n) {
  set.seed(i)
  q <- kmeans(pdepurada, 2)
  w[i] <- q$tot.withinss
}

#Visualizamos los W de cada uno de los Guess iniciales

valores_w = w
valores_w

#Graficamos dichos Ws

barplot(w, main = "Valor de W" , ylim=c(0,3000), col = "brown3")

#Buscamos la ubicación y valor del menor W

ubicacion_menor_w = which.min(w)
ubicacion_menor_w

valor_menor_w = min(w)
valor_menor_w

# Nos quedamos la semilla que crea un Guess inicial que minimiza W
# Con este dato aplicamos K-Means

set.seed(3)
q <- kmeans(pdepurada, 2)
q
q$tot.withinss

# Calculamos la proporcion del error bueno sobre T (cuadrados totales)

q$betweenss/q$tot.withinss

# Resultados
```

```
clasificacion_clusters = q$cluster
clasificacion_clusters

ubicacion_centros=q$centers
ubicacion_centros

tamaño_clusters = q$size
tamaño_clusters

# El primer cluster muestra observaciones con ondas y defensas cercanas a la media
# mientras que el cluster número 2, muestra observaciones con ondas por encima de la media
# al igual que las defensas, lo cual podría indicar personas infectadas con el virus

#*****
#Aplicación de Modelo KNN
#*****

# Trabajaremos con otra base de datos que contiene nuevos individuos.
# Intentaremos evaluar su estado(infectado/no infectados) comparandolos con los individuos
ya
# estudiados

# Cargamos el archivo "nuevos.csv"
nuevos = read.csv("nuevos.csv")

# Verificamos si los datos de "nuevos.csv" se encuentran escalados

colMeans(nuevos)
apply(nuevos, 2, var)

# Normalizamos

nscaled = scale(nuevos[,5:6])
nscaled

colMeans(nscaled)
apply(nscaled, 2, var)

# Utilizamos el modelo de KNN para integrar la base "nuevos.csv"

nuevos_casos = knn(q$centers, nscaled, 1:k, k = 1)
nuevos_casos
```

```
# Identificamos las observaciones que presentan características similares
# a las de los clusters 1 y 2 de la base "pandemia.csv"

# Visualizamos los datos con la clusterización obtenida

pairs(pdepurada, col = q$cluster, pch = 20, main= "Clusterización Ondas vs Defensas")

pairs(nscaled, col = nuevos_casos, pch = 20, main= "Clusterización Ondas vs Defensas")

# Identificamos las observaciones reclasificadas en no infectados (1) e infectados (2)
# datos
ncasos = matrix(t(nuevos_casos))

summary(nuevos_casos)

#Fin del análisis
#=====S
```