**www.nataliejulian.com**
**Natalie Julian**

# Clustering con K-means

i) El Clustering es una técnica muy utilizada en Machine Learning que consiste en agrupar puntos n-dimensionales en torno a una clasificación o target de interés. Un clustering eficiente entrega grupos o clústers heterogéneos entre ellos, pero homogéneos dentro de sí mismos.

ii) El algoritmo de K-means es una metodología no paramétrica, no necesita supuestos distribucionales, ventaja respecto a los métodos parámetricos usuales.

Pasos en un algoritmo de K-means:

1. Como input, es necesario entregar la cantidad de clústers K y el set de puntos $x_1, x_2, ..., x_n$. Si bien, es posible entregarle los K centroides para comenzar el algoritmo, en `R` si no se especifican, se parte con centros aleatorios (se toman K registros aleatorios de la data). También es posible añadir como especificación `iter.max` que corresponde al número máximo de iteraciones a realizar.
3. Al tener los centroides, para cada punto se calcula la distancia euclideana y se le asigna como clúster aquél cuyo centroide es más cercano a él.
4. Una vez asignado a cada punto su clúster, se toma como nuevo centroide el promedio de los puntos asociados en cada clúster, y se itera nuevamente.
5. El proceso culmina cuando en las iteraciones los puntos ya no cambian de clúster o cuando se alcanza el máximo de iteraciones fijada.

# Problema 1

Una empresa vitivinícola muy exitosa en la venta de vinos ofrece 177 vinos distintos a sus clientes. La industria ha realizado análisis de seguimiento a sus clientes y ha podido concluir que en general, sus clientes compran vinos muy similares entre sí, por lo que le gustaría poder determinar una agrupación para sugerir de manera personalizada a cada cliente un vino similar al último que compró, y así mejorar la experiencia de sus valiosos clientes.
La base de datos `segmentacion` contiene información de la carta de 177 vinos que vende

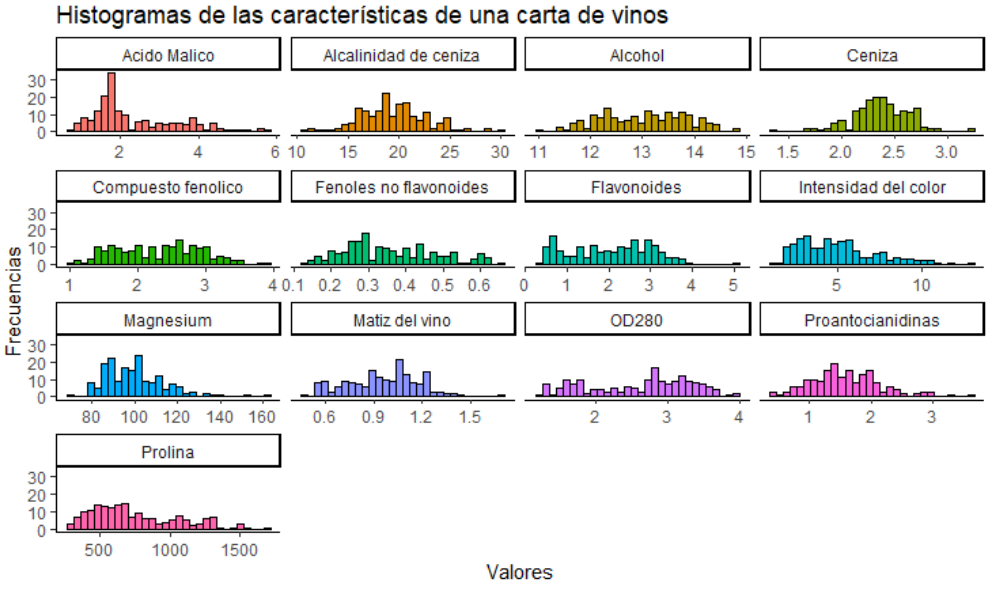
la empresa. Realice clustering con el fin de apoyar a la empresa vitivinícola en su estrategia de promoción y así, aumentar el sentimiento de pertenecencia del cliente a la empresa.

## Análisis de la data

```
#Cargar la data
library(readxl)
segmentacion <- read_excel(file.choose())


#Vista de la data

names(segmentacion)
 [1] "Codigo"                  "Alcohol"
 [3] "Acido Malico"            "Ceniza"
 [5] "Alcalinidad de ceniza"   "Magnesium"
 [7] "Compuesto fenolico"      "Flavonoides"
 [9] "Fenoles no flavonoides"  "Proantocianidinas"
[11] "Intensidad del color"    "Matiz del vino"
[13] "OD280"                   "Prolina"


print(segmentacion)
# A tibble: 177 x 14
   Codigo Alcohol `Acido Malico` Ceniza `Alcalinidad de   Magnesium
   <chr>    <dbl>          <dbl>  <dbl>             <dbl>     <dbl>
 1 VIN177    14.2           1.71   2.43              15.6       127
 2 VIN176    13.2           1.78   2.14              11.2       100
 3 VIN175    13.2           2.36   2.67              18.6       101
 4 VIN174    14.4           1.95   2.5               16.8       113
 5 VIN173    13.2           2.59   2.87              21         118
 6 VIN172    14.2           1.76   2.45              15.2       112
 7 VIN171    14.4           1.87   2.45              14.6        96
 8 VIN170    14.1           2.15   2.61              17.6       121
 9 VIN169    14.8           1.64   2.17              14          97
10 VIN168    13.9           1.35   2.27              16          98
#     with 167 more rows, and 8 more variables: `Compuesto
#   fenolico` <dbl>, Flavonoides <dbl>, `Fenoles no flavonoides` <dbl>,
#   Proantocianidinas <dbl>, `Intensidad del color` <dbl>, `Matiz del
#   vino` <dbl>, OD280 <dbl>, Prolina <dbl>

#Nos aseguramos de que las variables se lean en el formato adecuado:

str(segmentacion)

#S lo la variable Codese lee como caracter, las demas variables
# son de tipo numeric

dim(segmentacion)    #Carta de 177 vinos, 1 ID y 13 variables
[1] 177  14

table(table(segmentacion$Codigo))
  1
177

#No hay repeticion de ID
```

## Análisis de las variables

```
#An lisis de las variables

library(ggplot2) #Para realizar graficos
library(dplyr) #Para utilizar %>%
library(tidyverse)  #Para utilizar gather()


##Histogramas

segmentacion %>%
  gather(Attributes, value, 2:14) %>%
  ggplot(aes(x=value, fill=Attributes)) +
  geom_histogram(colour="black", show.legend=FALSE, bins=30) +
  facet_wrap(~Attributes, scales="free_x") +
  labs(x="Valores", y="Frecuencias",
       title="Histogramas de las caracteristicas de una carta de vinos") +
  theme_classic()
```




Es posible observar que las variables se encuentran en escalas bastante diferentes, `Magnesium` y `Prolina` presentan escalas muy altas respecto a las demás.

## Reescalar variables

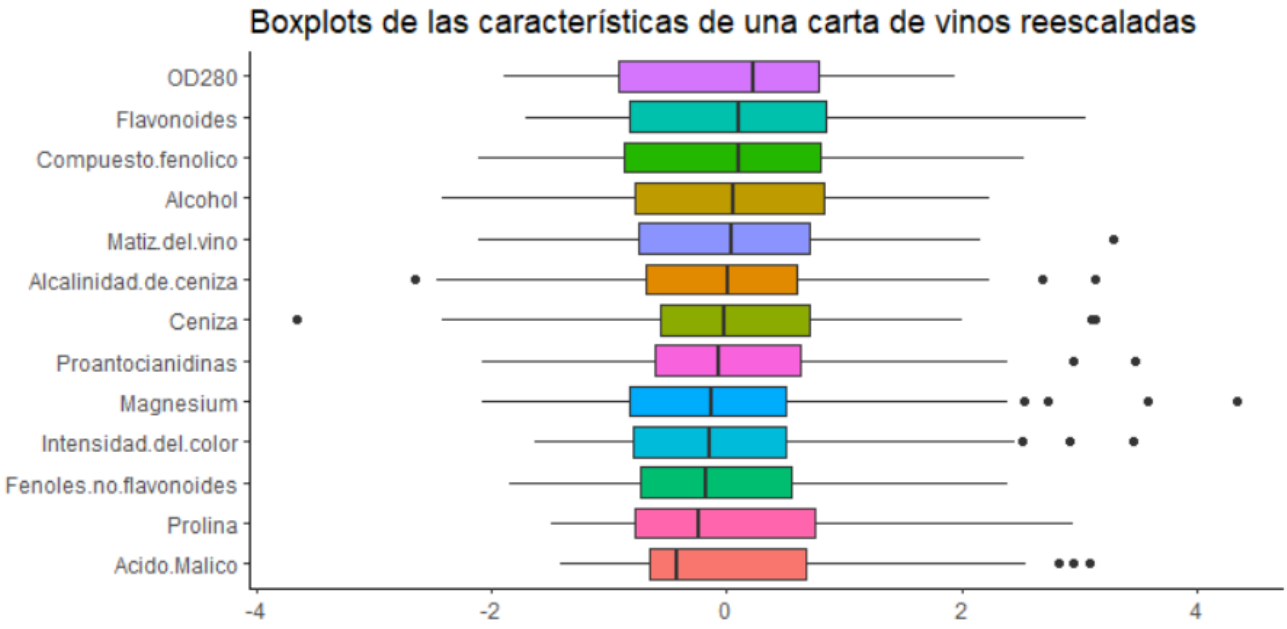
Cuando las variables están en distintas escalas, es necesario reescalarlas de modo que llevarlas a una misma escala. Incluso, cuando estén en las mismas escalas pero presenten variabilidades muy distintas, se debe estandarizar la data. K-means en cada iteración

calcula una distancia de cada punto a los centroides, si existen diferencias en las varianzas de las variables el algoritmo puede asignarle más peso a unas variables por sobre otras.

```
#Variabilidad de las variables:

diag(var(segmentacion[,-1]))

#Existen diferencias importantes en las variabilidades. Ademas, las variables
#estan en distintas escalas. Es necesario reescalar

stand<-scale(segmentacion[-1])

diag(var(stand))    #Variabilidades unitarias
```

## Deteccion de Outliers

```
#Boxplots

stand<-data.frame(stand)

stand %>%
  gather(Attributes, values) %>%
  ggplot(aes(x=reorder(Attributes,values,FUN=median),y=values,fill=Attributes
  )) +    geom_boxplot(show.legend=FALSE) +
  labs(title="Boxplots de las caracteristicas de una carta de vinos
  reescaladas") +    theme_classic() +
  theme(axis.title.y=element_blank(),
        axis.title.x=element_blank())  +
  coord_flip()
```



Podemos ver que las variables que tienen outliers son:

- Matiz del vino
- Alcalinidad de ceniza
- Ceniza
- Proantocianidinas
- Magnesium
- Intesidad del color
- Acido Malico

Como el algoritmo de K-means obtiene promedios en cada iteración, naturalmente, observaciones outliers podrían generar ruido para establecer los centroides. Es útil conocer cuál es la tasa de observaciones outliers en la data.

```
#Se identifican los outliers:

(outliers<-unique(c(out1,out2,out3,out4,out5,out6,out7)))  #Lista de outliers
 [1] 116  60  74 122 128  26  96 111  70  79 152 159 160 124 138 173

length(outliers)/nrow(segmentacion)    #Tasa de observaciones outliers
[1] 0.09039548
```

Dado que la tasa de observaciones outliers es baja y además estos outliers no se alejan extremadamente del centro de masa de los datos, una opción es realizar K-means con todas las observaciones, luego, ajustar K-means sin los outliers y ver si existen grandes diferencias en la clusterización.

## K-means

```
#Ejemplo, utilizando k=2

set.seed(2020)  #Semilla de aleatoriedad para los centroides iniciales

clusterk2<-kmeans(stand, centers=2)

#Podemos extraer:

names(clusterk2)
[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[7] "betweenss"    "size"         "iter"         "ifault"

#Clusters

clusterk2$cluster
```

```
#Centroides finales

clusterk2$centers


#Distribucion de observaciones en clusters

clusterk2$size     #En el cluster 1 se agruparon 64 vinos
[1]  64 113        #En el cluster 2 se agruparon 113 vinos

# withinss, dentro de cada cluster la suma cuadratica
#importante considerar que si el numero de observaciones
#en un cluster es mayor, withinss aumenta


clusterk2$withinss
[1]  506.9246 1139.1413


#tot.whithinss

clusterk2$tot.withinss   #La suma cuadratica intra cluster total
[1] 1646.066

#Lo ideal es que sea menor (clusters homogeneos dentro de si)


# betweenss

clusterk2$betweenss      #La suma cuadratica entre cluster

#Lo ideal es que sea mayor (clusters heterogeneos entre si)
```

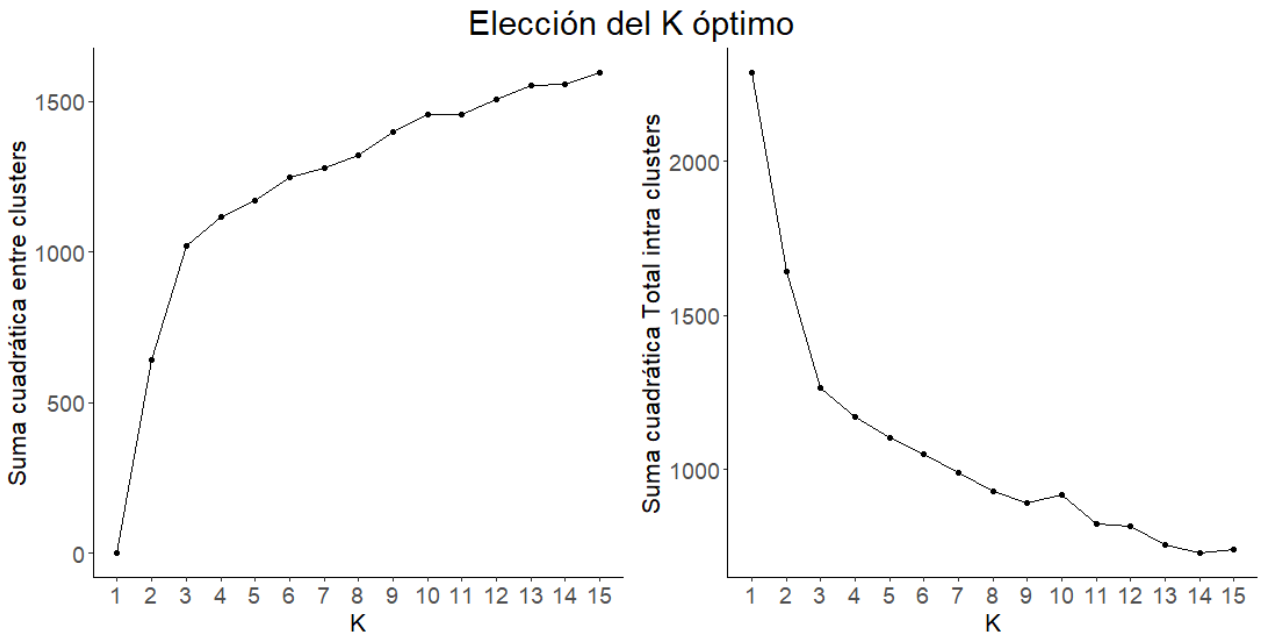
Sin embargo, utilizar k=2 es muy arbitrario. ¿Cuál es el $k$ óptimo?

## Elección del k óptimo

Realizaremos distintas clusterizaciones con $k = 1, ...., 15$ (en este ejercicio, es posible hacerlo) y compararemos las cantidades `betweenss` y `whithinss` por iteración.

```
###Eleccion del k optimo


n<-15         #Cantidad de valores k a probar

bss <- rep(NA,n)
wss <- rep(NA,n)

set.seed(2020)

for(i in 1:n){
  bss[i] <- kmeans(stand, centers=i)$betweenss
  wss[i] <- kmeans(stand, centers=i)$tot.withinss

}

#Graficas
```

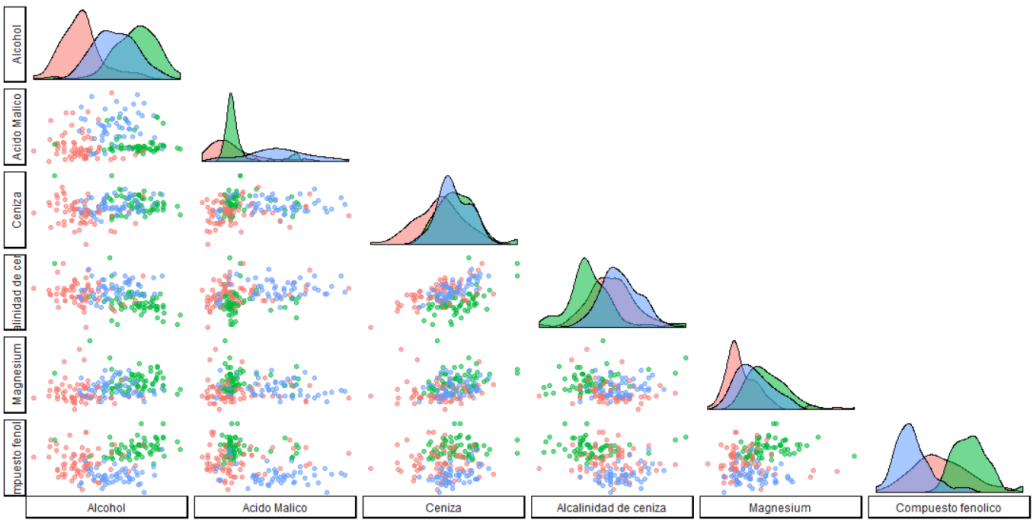
```
betweenplot <- qplot(1:n, bss, geom=c("point", "line"),
          xlab="K", ylab="Suma cuadr tica entre clusters") +
  scale_x_continuous(breaks=seq(0, n, 1)) +
  theme_classic()+theme(axis.text.y = element_text(size=14),
  axis.title.y = element_text(size=16),
                        axis.text.x = element_text(size=14),
                        axis.title.x = element_text(size=16))


whithinplot <- qplot(1:n, wss, geom=c("point", "line"),
           xlab="K", ylab="Suma cuadr tica Total intra clusters") +
  scale_x_continuous(breaks=seq(0, n, 1)) +
  theme_classic()+theme(axis.text.y = element_text(size=14),
  axis.title.y = element_text(size=16),
                        axis.text.x = element_text(size=14),
                        axis.title.x = element_text(size=16))


library(ggpubr)
plot<-ggarrange(betweenplot, whithinplot, ncol=2)

annotate_figure(plot,top=text_grob("Eleccion del K optimo",size=22))
```




La segunda gráfica corresponde a una gráfica de sedimentación. Si bien, el criterio de selección puede variar, en general, se selecciona aquél valor de K tal que en ese valor se observa un cambio de pendiente notable. Por ejemplo, en ambos gráficos podemos ver que en $K = 3$ ser observa la diferencia más notable, por lo tanto, realizaremos clusterizacion con 3 clusters.

## Utilizando el k óptimo

```
###Eleccion del k optimo
#3 clusters

set.seed(2020)

clusterk3<-kmeans(stand, centers=3)

clusterk3$centers  #Centroides

clusterk3$size  #El tama o de los grupos es bastante similar
[1] 65 62 50

library(GGally) #Para realizar grafico

ggpairs(cbind(segmentacion[-1], Cluster=as.factor(clusterk3$cluster)),
        columns=1:6, aes(colour=Cluster, alpha=0.5),
        lower=list(continuous="points"),
        upper=list(continuous="blank"),
        axisLabels="none", switch="both") +
  theme_classic()
```




## Variantes

Muchas veces al realizar clustering, es necesario disminuir la dimensionalidad (en este caso, utilizamos un número de variables bastante trabajable, pero no siempre es así). Y también, en algunos casos es útil otorgarle mayor peso a algunas variables más que a otras, muchas veces por conocimiento a priori del negocio. Además, es posible utilizar distintas distancias, por ejemplo, al clusterizar imágenes con información en escala de grises, es necesario utilizar distancias con propiedades distintas a la distancia euclideana. Un ejemplo: *Hausdor-based distance*.