METODO K-MEANS

Equipo: 09

Marco Antonio Oviedo Acevedo 1851698

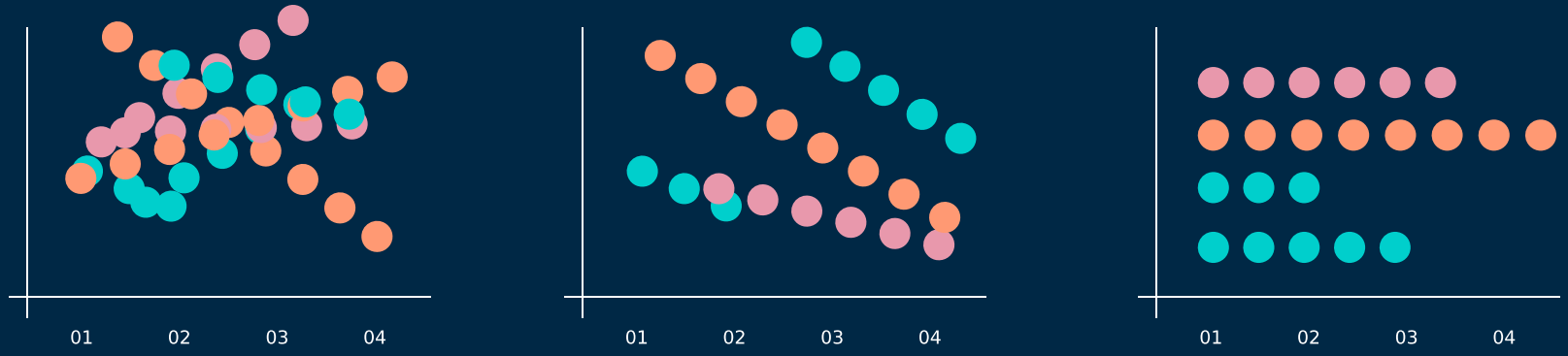
Nayelli Alondra Gaona Allende 1860995

Eliud Moreno Narvaez 1863955

Luis Adrian Navarro Garcia 1866581

APRENDIZAJE NO SUPERVISADO: K-MEANS CLUSTERING

Es un tipo de aprendizaje no supervisado, que se utiliza cuando tiene datos no etiquetados, es decir, datos sin categorías o grupos definidos. El objetivo de este algoritmo es encontrar grupos en los datos.
Los puntos de datos se agrupan según la similitud de características



K-MEANS CLUSTERING

PREDICCIÓN DE ANÁLISIS DE DATOS

ALGORITMO

VARIABLE INDEPENDIENTE

Variable independiente (caracteristicas): son las características que queremos usar para predecir grupos en los datos

APRENDIZAJE NO SUPERVISADO: K-MEANS CLUSTERING

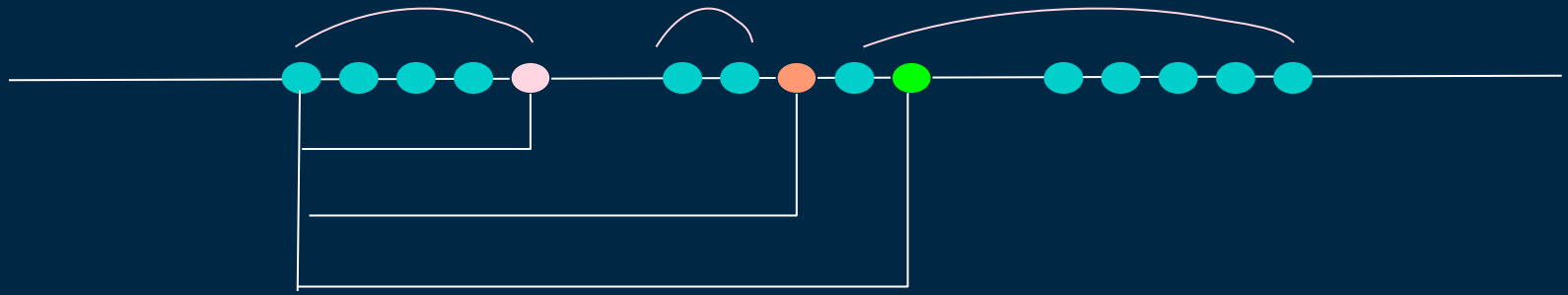
Paso 1: Seleccionar el número de cluster (K) qué deseas identificar en los datos. En este caso K=3

APRENDIZAJE NO SUPERVISADO: K-MEANS CLUSTERING

Paso 2: Seleccionar al azar K puntos. No necesariamente deben ser puntos de nuestros datos pueden ser puntos nuevos

APRENDIZAJE NO SUPERVISADO: K-MEANS CLUSTERING

Paso 3: Medimos la distancia entre cada uno de los datos y los puntos seleccionados, asignándole el punto que se encuentre más cerca



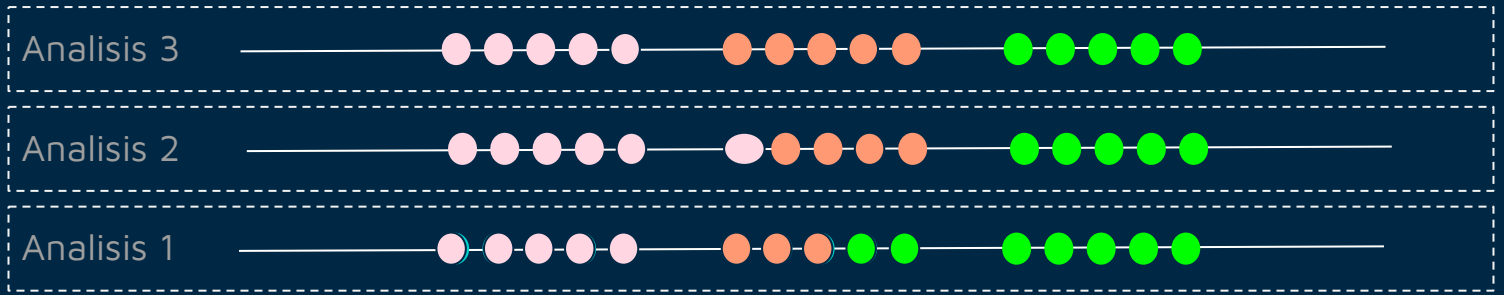
APRENDIZAJE NO SUPERVISADO: K-MEANS CLUSTERING

Paso 3: Medimos la distancia entre cada uno de los datos y los puntos seleccionados, asignándole el punto que se encuentre más cerca

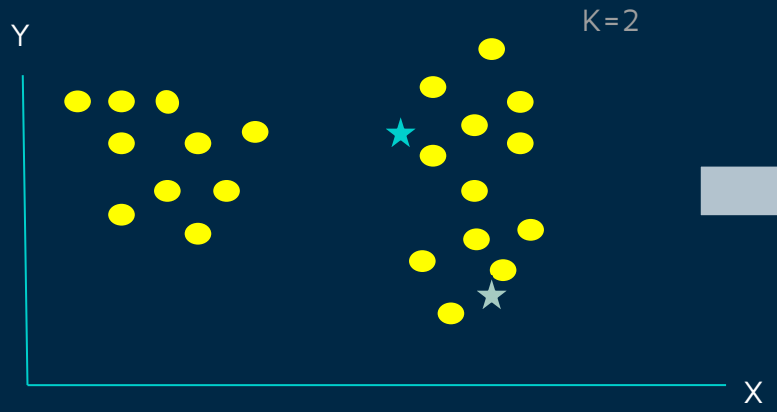
Verificamos la distribución de nuestros datos con cada uno de nuestros centroides

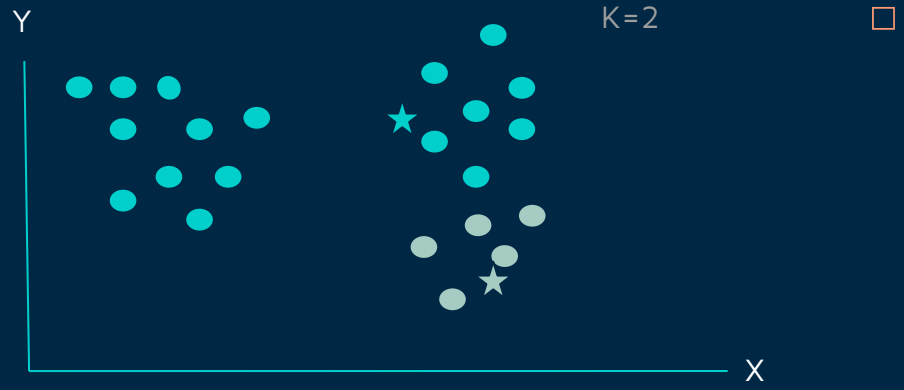
APRENDIZAJE NO SUPERVISADO: K-MEANS CLUSTERING

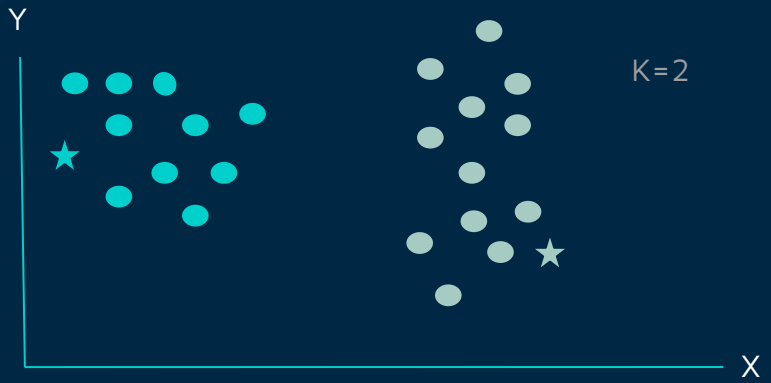
Paso 4: Colocamos nuevos K puntos y repetimos el procedimiento.



graficas







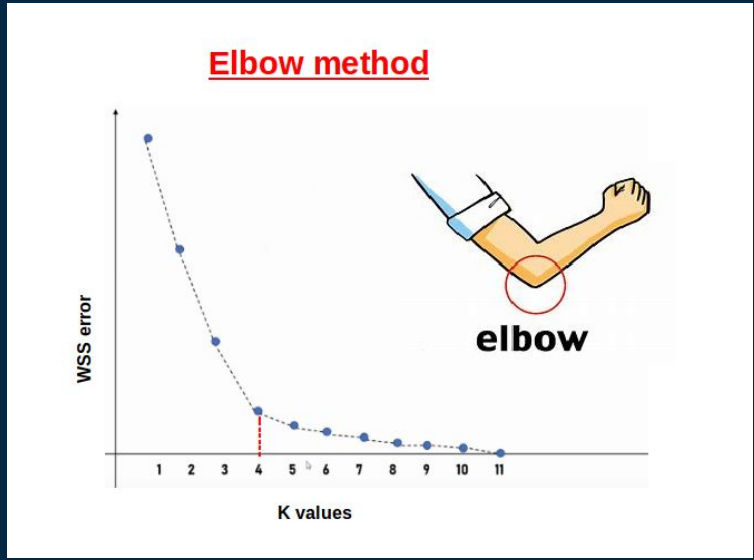
APRENDIZAJE NO SUPERVISADO: K-MEANS CLUSTERING

1	Es un algoritmo rápido. Robusto y simple que proporciona resultados confiables cuando los conjuntos de datos son distintos o bien separados entre sí de forma lineal.
2	Debemos especificar el número de clusters, de antemano, y los resultados finales son sensibles a la inicialización
3	La agrupación puede no funcionar bien si contiene datos muy superpuestos o si los datos son ruidosos o están llenos de valores atípicos

El Método Del Codo

La idea básica de los algoritmos de clustering es la minimización de la varianza intra-cluster y la maximización de la varianza inter-cluster. Es decir, queremos que cada observación se encuentre muy cerca a las de su mismo grupo y los grupos lo más lejos posible entre ellos.

El método del codo utiliza la distancia media de las observaciones a su centroide. Es decir, se fija en las distancias intra-cluster. Cuanto más grande es el número de clusters k, la varianza intra-cluster tiende a disminuir. Cuanto menor es la distancia intra-cluster mejor, ya que significa que los clusters son más compactos. El método del codo busca el valor k que satisfaga que un incremento de k, no mejore sustancialmente la distancia media intra-cluster.



El método del codo es a veces ambiguo, una alternativa es el análisis de la silueta, que es más objetivo que el método del codo.

EJEMPLO

Vamos a utilizar un [conjunto de datos de créditos] (https://github.com/OviedoMarco/Mineria_de_datos/blob/DataSets/datos_creditos.tx) para agrupar diferentes características de los solicitantes.
Este conjunto de datos contiene las características (numéricas y categóricas) de los solicitantes de créditos de una financiera.

Primero necesitamos cargar algunas bibliotecas y leer el conjunto de datos.

Y tengamos una idea de con qué estamos trabajando.

	Edad	Sexo	Trabajo	Vivienda	Cuenta_Ahorro	Cuenta_Corriente	Monto_Credito	Periodo_Meses	Tipo	Resultado
1	67	0	calificados	propia	NA	bajo	1169	6	radio/TV	1
2	22	1	calificados	propia	bajo	medio	5951	48	radio/TV	0
3	49	0	no_calificados_residentes	propia	bajo	NA	2096	12	educacion	1
4	45	0	calificados	familiar	bajo	bajo	7882	42	mobiliario	1
5	53	0	calificados	familiar	bajo	bajo	4870	24	automotriz	0
6	35	0	no_calificados_residentes	familiar	NA	NA	9055	36	educacion	1
7	53	0	calificados	propia	muy_alto	NA	2835	24	mobiliario	1
8	35	0	altamente_calificados	rentada	bajo	medio	6948	36	automotriz	1
9	61	0	no_calificados_residentes	propia	alto	NA	3059	12	radio/TV	1
10	28	0	altamente_calificados	propia	bajo	medio	5234	30	automotriz	0
11	25	1	calificados	rentada	bajo	medio	1295	12	automotriz	0

```
# Cargar bibliotecas
biblioteca (tidyverse)
biblioteca (corrplot)
biblioteca (gridExtra)
biblioteca (GGally)
biblioteca (knitr)
```

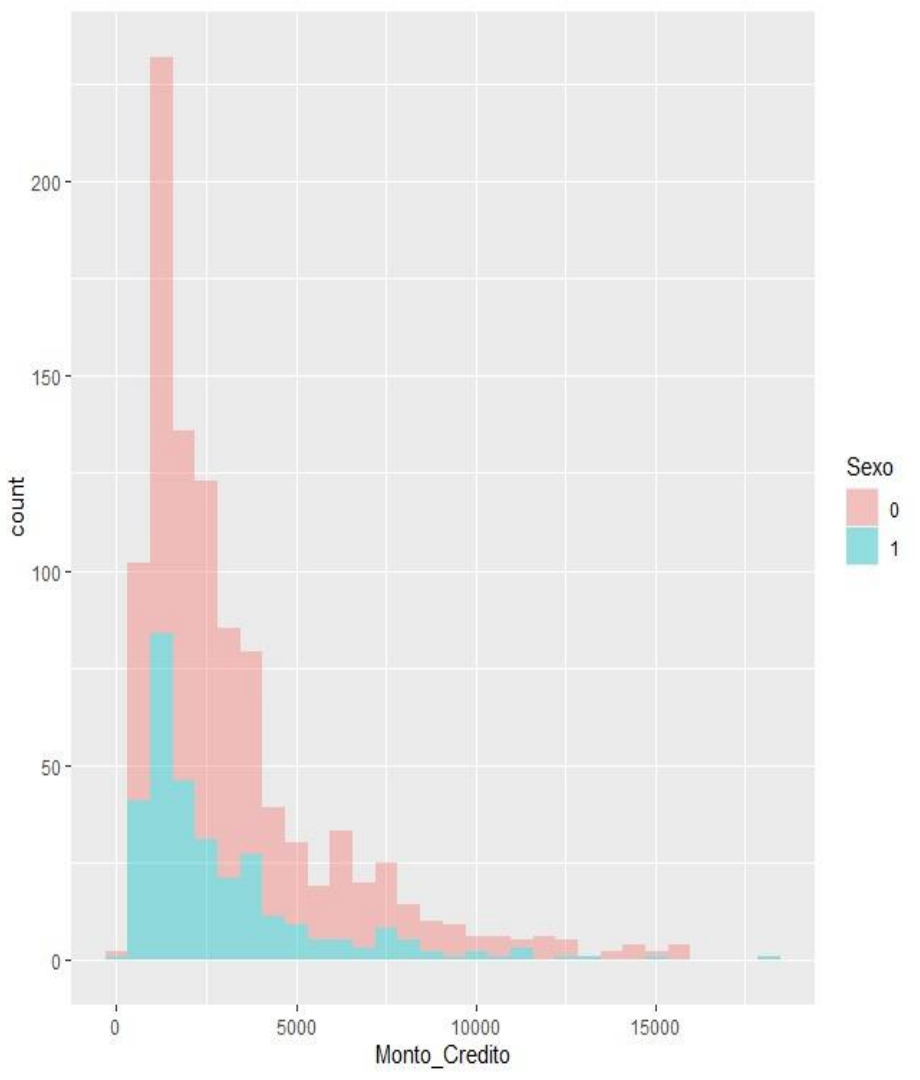
Librerías que se usaron para el algoritmo k-means

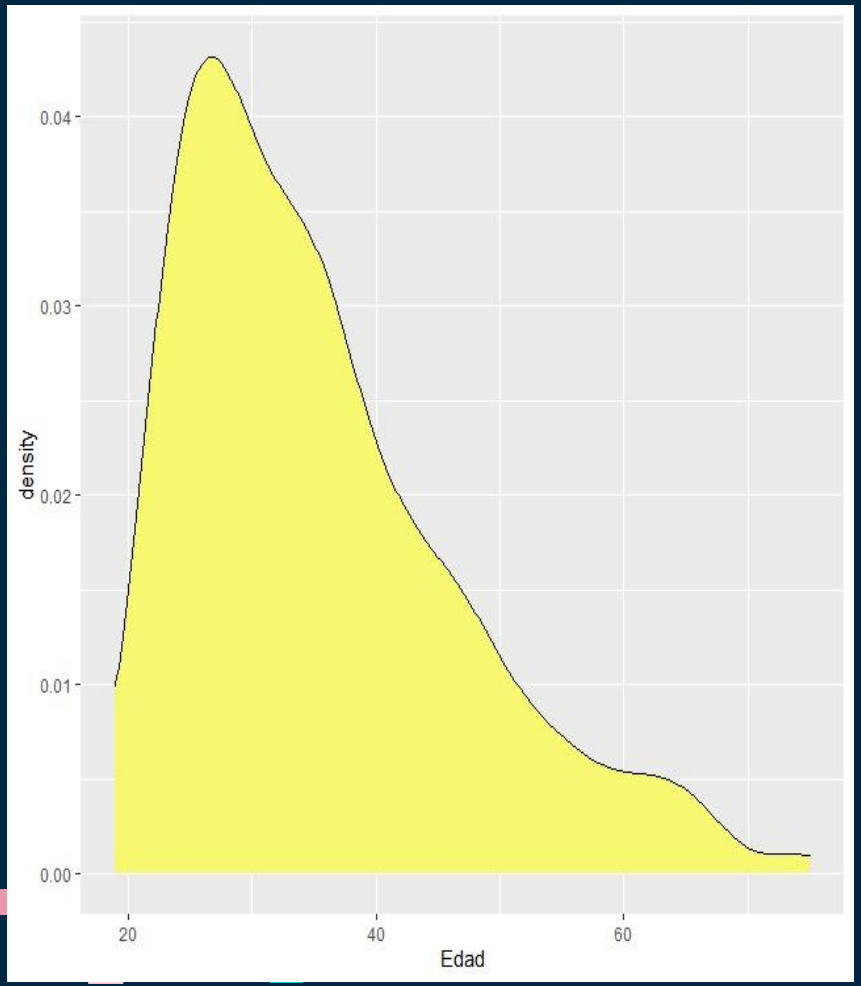
Análisis de los datos

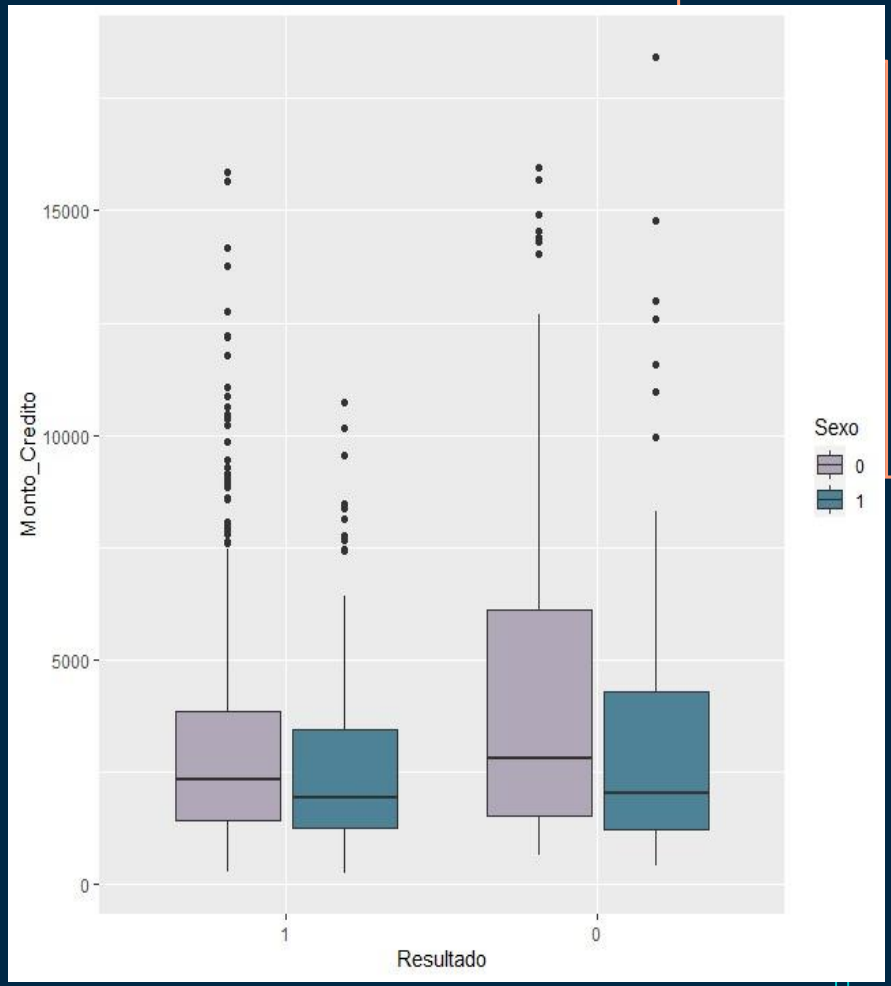
Primero tenemos que explorar y

visualizar los datos.

Librerías que se usaron para el algoritmo k-means




0.04
0.03
0.02
0.01
0.00
density
20
40
60
Edad


15000
10000
5000
0
Monto_Credito
1
0
Resultado
Sexo
0
1

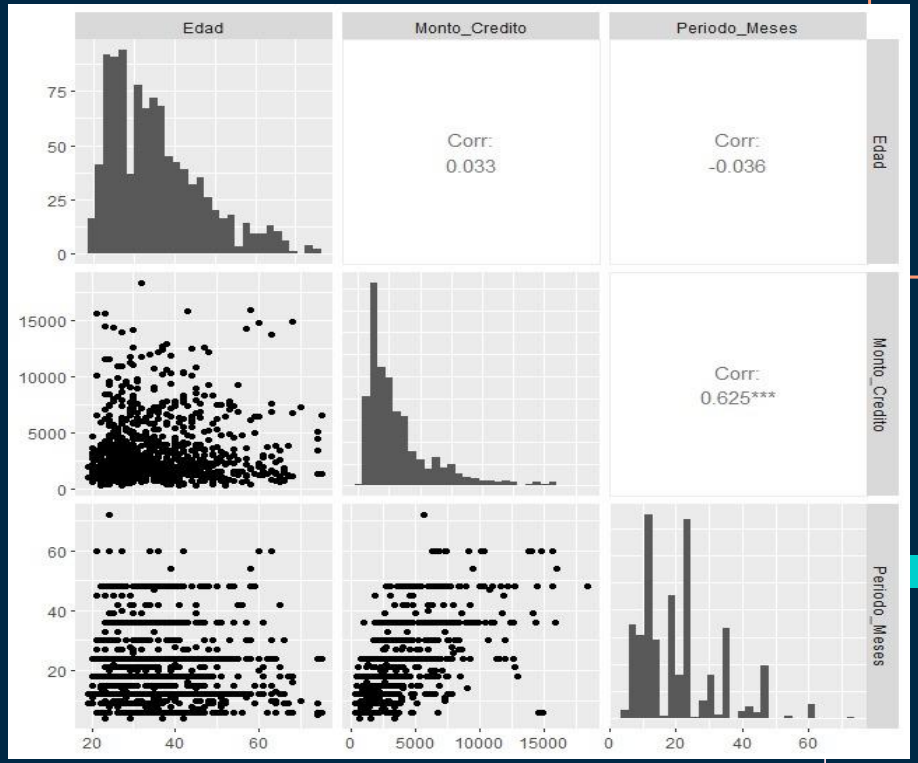
Ahora vamos a segmentar nuestras variables categóricas y numéricas. Como dijimos antes, k-means es un algoritmo de aprendizaje automático no supervisado y funciona con datos sin etiquetar esto en resumen quiere decir que debemos trabajar con variables numéricas.

```
#CATEGORICA
i = 0
var_cat<- c() # Vector vacío
for(i in colnames(base)){
if( class(data.frame(base)[,i]) %in% c("factor", "character") ){
var_cat<- c(var_cat, i)
}
}
var_cat
```


```
#NUMERICA
var_num<- c() # Vector vacío
i = 0
for(i in colnames(base)){
if( (class(data.frame(base)[,i]) %in% c("numeric", "integer")) ){
var_num<- c(var_num, i)
}
}
var_num
```

¿Cuál es la relación entre los diferentes atributos? Podemos usar la función `corrplot ()` para crear una visualización gráfica de una matriz de correlación.

```
#El análisis bivariado analiza dos conjuntos de  #datos emparejados,
estudiando si existe una relación entre ellos library(GGally)
#grafica de correlaciones de las variables numéricas
base %>%
    select(var_num) %>%
    ggpairs(lower = list(continuous= "points", combo= "facehist"),
        diag= list(continuous= "bar", comobo= "facehist") )
```



Preparación de datos

Tenemos que normalizar las variables para expresarlas en el mismo rango de valores. En otras palabras, la normalización significa ajustar los valores medidos en diferentes escalas a una escala común.

```
# Normalizacion
MontoNormalizado <-base %>%
  mutate(edad_media= mean(Edad), edad_desv= sd(Edad), edad= (Edadedad_media)/edad_desv) %>%
 mutate(Monto_Credito_media= mean(Monto_Credito), Monto_Credito_desv= sd(Monto_Credito), Monto_Credito=
(Monto_Credito-Monto_Credito_media)/Monto_Credito_desv)
```

```
# Datos originales
p1 <- ggplot(base, aes(x=Monto_Credito, y=Edad)) +
    geom_point() +
    labs(title="Data originial") +
    theme_bw()
# Datos normalizados
p2 <- ggplot(MontoNormalizado, aes(x=Monto_Credito, y=Edad)) +
   geom_point() +
   labs(title="Data normalizada") +
   theme_bw()
```

La función `kmeans ()` devuelve un objeto de la clase "` kmeans` "con información sobre la partición:

`cluster`.	Un vector de números enteros que indica el grupo al que se asigna cada punto.
`centers`.	Una matriz de centros de conglomerados.
`size`.	El número de puntos en cada grupo.

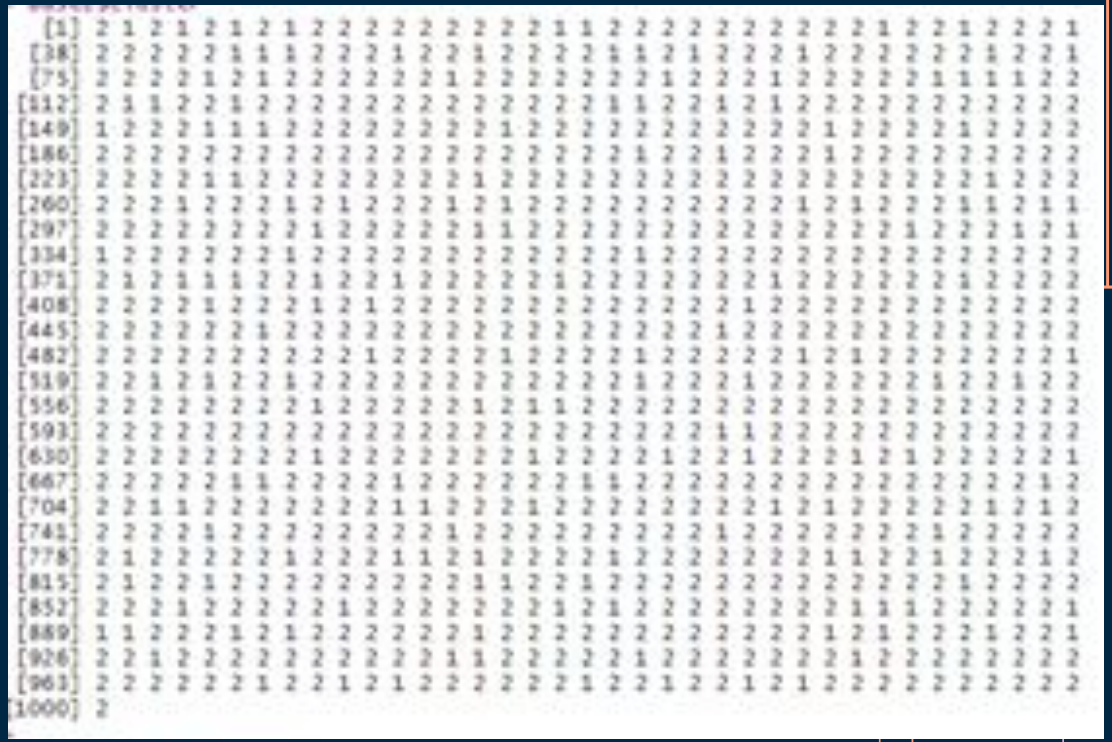
k-means ejecución

En esta sección vamos a ejecutar el algoritmo k-means y analizar los componentes principales que devuelve la función.

```
# Ejecución de k-medias con k = 2
  set.seed(1234)
  Base2 <- kmeans(MontoNormalizado, centers=2)
```

k-means ejecución

Cluster que se le asigna a cada punto
Base2$cluster



k-means ejecución

```
# Centros de clusters
Base2$centers
```

```
          V1
1  1.8128684
2 -0.3845478
```

```
# Tamaño de cluster
Base2$size
```

```
[1] 175 825
```

Además, la función `kmeans ()` devuelve algunas proporciones que nos permiten saber qué tan compacto es un clúster y qué tan diferentes son varios clústeres entre sí.

k-means ejecución

```
# Suma de cuadrados entre grupos
Base2$betweenss
```

```
[1] 697.1347
```

```
# Suma de cuadradados entre grupos
Base2$withinss
```

```
[1] 160.6488 141.2166
```

```
# Suma total de cuadrados dentro del
conglomerado
 Base2$tot.withinss
```

```
[1] 301.8653
```

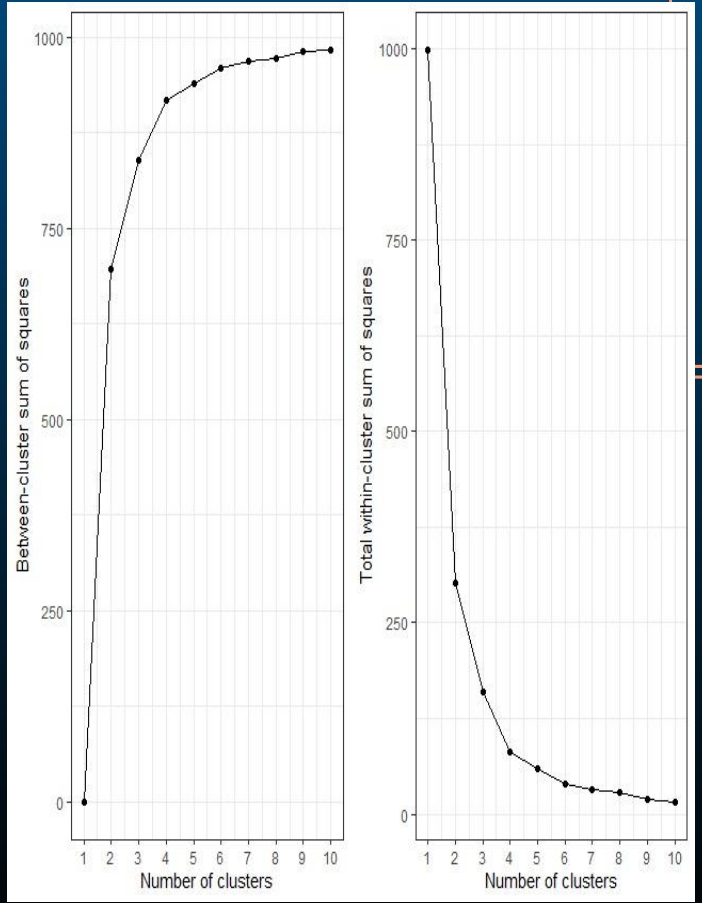
```
# Suma total de cuadrados
Base2$totss
```

```
[1] 999
```

¿Cuántos clústeres?

Para estudiar gráficamente qué valor de `k` nos da la mejor partición, podemos trazar` between` y `tot.withinss` frente a la elección de` k`.

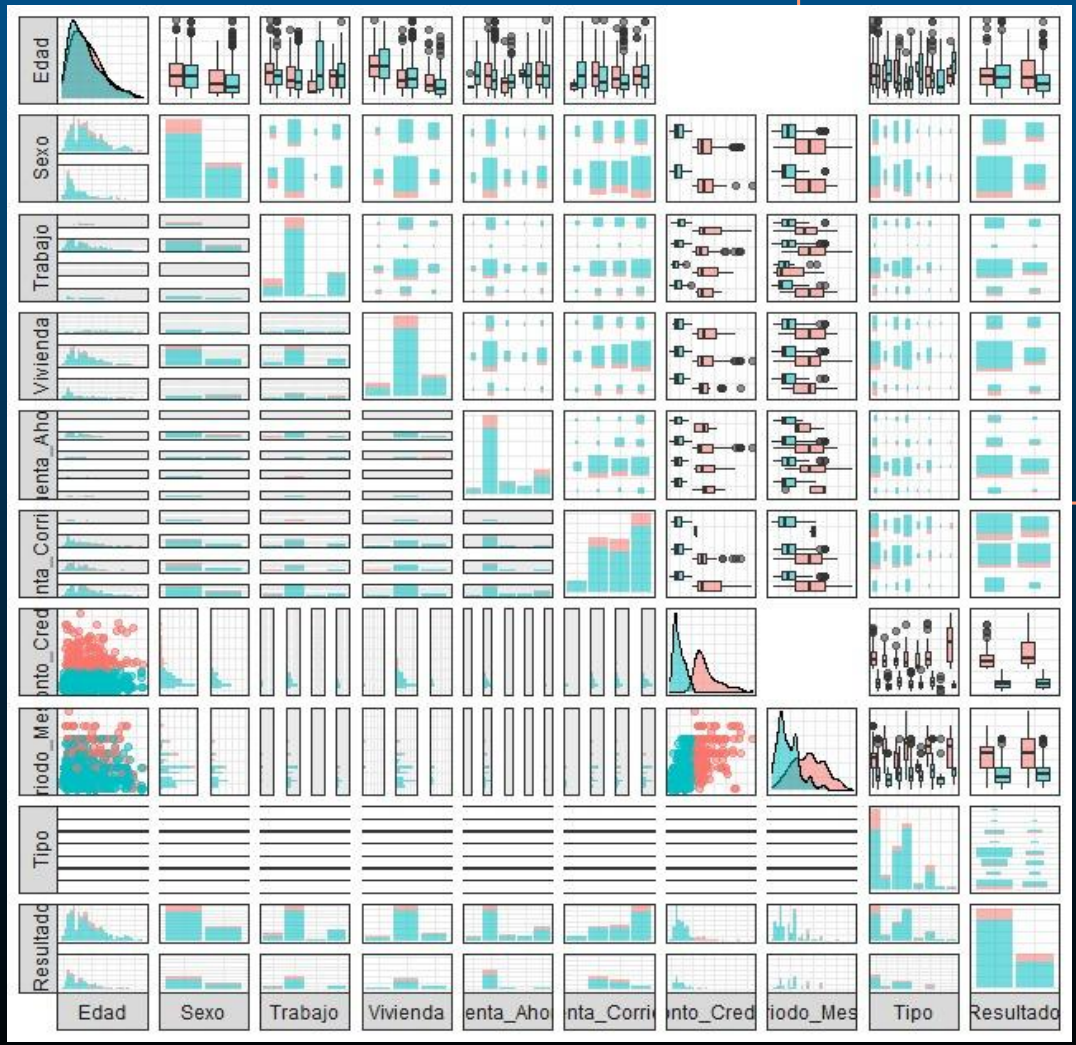
```
bss <- numérico ()
wss <- numérico ()
# Ejecutar el algoritmo para diferente valores de k
set.seed(1234)
for(i in 1:10){
      # Para cada k calcule entre tot.withins
      bss[i] <- kmeans(MontoNormalizado, centers=i)$betweenss
      wss[i] <- kmeans(MontoNormalizado, centers=i)$tot.withinss
}
# Suma de cuadrados entre grupos vs Elección de K
p3 <- qplot(1:10, bss, geom=c("point", "line"),
          xlab="Number of clusters", ylab="Between-cluster sum of squares") +
scale_x_continuous(breaks=seq(0, 10, 1)) +
theme_bw()
# Suma de cuadrados total dentro del conglomerado vs Elección de K
p4 <- qplot(1:10, wss, geom=c("point", "line"),
       xlab="Number of clusters", ylab="Total within-cluster sum of squares") +
scale_x_continuous(breaks=seq(0, 10, 1)) +
  theme_bw()
# grafica
 grid.arrange(p3, p4, ncol=2)
```



¿Cuál es el valor óptimo para `k`?

Uno debe elegir varios grupos para que agregar otro grupo no proporcione una partición mucho mejor de los datos. En algún momento, la ganancia caerá, dando un ángulo en el gráfico (criterio del codo). En este punto, se elige el número de conglomerados. En nuestro caso, está claro que 4 es el valor apropiado para `k`.

```
# Agrupación
ggpairs (cbind (base, Clúster = as.factor (base2 $ clúster)),
        columnas = 1: 6, aes (color = Cluster, alpha = 0.5),
        inferior = lista (continua = "puntos"),
        superior = lista (continua = "en blanco"),
        axisLabels = "ninguno", switch = "ambos") +
        theme_bw ()
```



Bibliografias

- AprendeIA con Ligdi Gonzalez. (2018, 9 de abril). *APRENDIZAJE NO SUPERVISADO: K. MEANS CLUSTERING | #15 Curso de Introducción a Machine Learning* [Video]. YouTube. https://www.youtube.com/watch?v=EZOab1vkFmI
- *K-Means: Agrupamiento con Minería de datos [Introducción]*. (s. f.). ESTRATEGIAS DE TRADING. https://estrategiastrading.com/k-means/#K-Means