K-Means

Beltrán Medrano Hector Alonso
Departamento de Sistemas y Computación
Instituto Tecnológico de Tijuana
Tijuana Baja California, México
hector.beltran@tectijuana.edu.mx

Resumen - Elaboración de una práctica utilizando el método de k means para determinar a qué grupos o clusters pertenece cada flor que pertenecen a un dataset de flores llamado iris.

I. INTRODUCCIÓN

Uno de los aspectos más relevantes de la Estadística es el análisis de grupos o clusters entre variables. Frecuentemente resulta de interés conocer el efecto que una o varias variables pueden tener con respecto a las otras, e incluso poder definir dependiendo sus características a qué grupo pertenecen.

El clustering es una técnica para encontrar y clasificar K grupos de datos (clusters). Así, los elementos que comparten características semejantes estarán juntos en un mismo grupo, separados de los otros grupos con los que no comparten características.

Para saber si los datos son parecidos o diferentes el algoritmo K-medias utiliza la distancia entre los datos. Las observaciones que se parecen tendrán una menor distancia entre ellas. En general, como medida se utiliza la distancia euclidiana aunque también se pueden utilizar otras funciones.

II. MARCO TEÓRICO

Para entender el funcionamiento de la práctica, primeramente se deben explicar ciertos conceptos de clustering, así como el conjunto de datos a utilizar.

A. Clustering

Básicamente es un tipo de método de aprendizaje no supervisado. Un método de aprendizaje no supervisado es un método en el que extraemos referencias de conjuntos de datos que consisten en datos de entrada sin respuestas etiquetadas. Generalmente, se utiliza como un proceso para encontrar una estructura significativa, procesos subyacentes explicativos, características generativas y agrupaciones inherentes a un conjunto de ejemplos.

La agrupación es muy importante ya que determina la agrupación intrínseca entre los datos no etiquetados presentes. No hay criterios para un buen agrupamiento. Depende del usuario, cuáles son los criterios que pueden usar para satisfacer sus necesidades. Por ejemplo, podríamos estar interesados en encontrar representantes para grupos homogéneos (reducción de datos), en encontrar 'grupos naturales' y describir sus propiedades desconocidas (tipos de datos 'naturales'), en encontrar agrupaciones útiles y adecuadas (clases de datos 'útiles') o en la búsqueda de objetos de datos inusuales (detección de valores atípicos). Este algoritmo debe hacer algunas suposiciones que constituyen la similitud de puntos y cada suposición crea grupos diferentes e igualmente válidos.

B. Métodos de agrupamiento:

Métodos basados en la densidad: estos métodos consideran los grupos como la región densa que tiene cierta similitud y es diferente de la región de menor densidad del espacio. Estos métodos tienen una buena precisión y capacidad para fusionar dos agrupaciones. Ejemplo DBSCAN (Agrupación espacial basada en densidad de aplicaciones con ruido), ÓPTICA (Puntos de pedido para identificar la estructura de agrupación), etc.

Métodos basados en la jerarquía: los grupos formados en este método forman una estructura de tipo árbol basada en la jerarquía. Se forman nuevos grupos utilizando el previamente formado. Se divide en dos categorías. Aglomerativo (enfoque ascendente), Divisivo (enfoque de arriba hacia abajo) ejemplos CURE (agrupamiento utilizando representantes), BIRCH (agrupamiento reductor iterativo equilibrado y uso de jerarquías), etc.

Métodos de partición: estos métodos dividen los objetos en k grupos y cada partición forma un grupo. Este método se utiliza para optimizar una función de similitud de criterio objetivo, como cuando la distancia es un parámetro importante, por ejemplo, K-means, CLARANS (Agrupación de aplicaciones grandes basadas en búsqueda aleatoria), etc.

Métodos basados en cuadrícula: en este método, el espacio de datos se formula en un número finito de celdas que forman una estructura similar a una cuadrícula. Todas las operaciones de agrupación realizadas en estas cuadrículas son rápidas e independientes del número de objetos de datos, por

ejemplo, STING (Cuadrícula de información estadística), grupo de ondas, CLIQUE (CLustering In Quest), etc.

C. K means

Los algoritmos de clustering son considerados de aprendizaje no supervisado. Este tipo de algoritmos de aprendizaje no supervisado busca patrones en los datos sin tener una predicción específica como objetivo (no hay variable dependiente). En lugar de tener una salida, los datos sólo tienen una entrada que serían las múltiples variables que describen los datos.

K-means necesita como dato de entrada el número de grupos en los que vamos a segmentar la población. A partir de este número k de clusters, el algoritmo coloca primero k puntos aleatorios (centroides). Luego asigna a cualquiera de esos puntos todas las muestras con las distancias más pequeñas.

El algoritmo K Means es un algoritmo iterativo que intenta dividir el conjunto de datos en subgrupos (grupos) no superpuestos distintos K-predefinidos donde cada punto de datos pertenece a un solo grupo. Intenta hacer que los puntos de datos dentro del clúster sean lo más similares posible, al tiempo que mantiene los clústeres lo más diferentes (lo más lejos posible). Asigna puntos de datos a un grupo de modo que la suma de la distancia al cuadrado entre los puntos de datos y el centroide del grupo (media aritmética de todos los puntos de datos que pertenecen a ese grupo) es mínima. Cuanta menos variación tengamos dentro de los grupos, más homogéneos (similares) serán los puntos de datos dentro del mismo grupo.

La forma en que funciona el algoritmo k means es la siguiente:

Especifique el número de grupos K.

Inicialice los centroides primero barajando el conjunto de datos y luego seleccionando aleatoriamente K puntos de datos para los centroides sin reemplazo.

Siga iterando hasta que no haya cambios en los centroides. es decir, la asignación de puntos de datos a grupos no está cambiando.

Calcule la suma de la distancia al cuadrado entre los puntos de datos y todos los centroides.

Asigne cada punto de datos al grupo más cercano (centroide).

Calcule los centroides para los grupos tomando el promedio de todos los puntos de datos que pertenecen a cada grupo.

El enfoque que kmeans sigue para resolver el problema se llama Expectation-Maximization. El E-step es asignar los puntos de datos al clúster más cercano. El paso M está calculando el centroide de cada grupo. A continuación se muestra un desglose de cómo podemos resolverlo matemáticamente (no dude en omitir).

La función objetivo es:

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} w_{ik} \|x^i - \mu_k\|^2$$

Donde wik = 1 para el punto de datos xi si pertenece al clúster k; de lo contrario, wik = 0. Además, μk es el centroide del grupo de xi.

Es un problema de minimización de dos partes. Primero minimizamos J w.r.t. wik y treat μk fijo. Luego minimizamos J w.r.t. μk y treat wik fijo. Técnicamente hablando, diferenciamos a J w.r.t. primero wik y actualice las asignaciones de clúster (E-step). Luego diferenciamos J w.r.t. μk y recalcule los centroides después de las asignaciones de clúster del paso anterior (paso M). Por lo tanto, E-step es:

$$\frac{\partial J}{\partial w_{ik}} = \sum_{i=1}^{m} \sum_{k=1}^{K} \|x^i - \mu_k\|^2$$

$$\Rightarrow w_{ik} = \begin{cases} 1 & \text{if } k = argmin_j \|x^i - \mu_j\|^2 \\ 0 & \text{otherwise.} \end{cases}$$

En otras palabras, asigne el punto de datos xi al grupo más cercano juzgado por su suma de distancia al cuadrado del centroide del grupo.

$$\frac{\partial J}{\partial \mu_k} = 2 \sum_{i=1}^{m} w_{ik}(x^i - \mu_k) = 0$$

$$\Rightarrow \mu_k = \frac{\sum_{i=1}^{m} w_{ik} x^i}{\sum_{i=1}^{m} w_{ik}}$$

Lo que se traduce en volver a calcular el centroide de cada grupo para reflejar las nuevas asignaciones.

D. Aplicaciones

El algoritmo k means es muy popular y se utiliza en una variedad de aplicaciones, como la segmentación del mercado, la agrupación de documentos, la segmentación de imágenes y la compresión de imágenes, etc. El objetivo

generalmente cuando nos sometemos a un análisis de conglomerados es:

Obtenga una intuición significativa de la estructura de los datos con los que estamos tratando.

Agrupe y luego prediga dónde se construirán diferentes modelos para diferentes subgrupos si creemos que existe una amplia variación en los comportamientos de diferentes subgrupos. Un ejemplo de esto es agrupar a los pacientes en diferentes subgrupos y construir un modelo para cada subgrupo para predecir la probabilidad del riesgo de sufrir un ataque cardíaco.

E. Dataset iris

Esta es quizás la base de datos más conocida que se encuentra en la literatura de reconocimiento de patrones. El artículo de Fisher es un clásico en el campo y se hace referencia con frecuencia a este día. (Ver Duda & Hart, por ejemplo). El conjunto de datos contiene 3 clases de 50 instancias cada una, donde cada clase se refiere a un tipo de planta de iris. Una clase es linealmente separable de las otras 2; estos últimos NO son linealmente separables entre sí.

III. DESARROLLO DEL PROGRAMA

Se cuenta con un conjunto de datos de 150 registros de los cuales se dividirán en 3 clusters diferentes. Se utilizó R como lenguaje de programación, dplyr y ggplot2 para manipular el dataset y graficar el diagrama de dispersión.

Para realizar el método de k means, solamente se pueden utilizar dos variables para analizar el conjunto de datos, en este caso se van a utilizar los campos de “SepalLengthCm” el cual representa el largo del cepal en centímetros, y “SepalWidthCm” que representa el ancho del cepal en centímetros.

A. Procedimiento

Estos son los pasos utilizados para realizar el método de regresión lineal simple:

1. Descargar y convertir el conjunto de datos en formato csv.
2. Importar el conjunto de datos en R.
3. Identificar los dos tipos de variables a utilizar (‘*x*’ y ‘*y*’).
4. sacar distancias de cada uno de los clusters iniciales.
5. Calcular la media de ‘*x*’ y ‘*y*’.
6. Crear la columna para la clasificación.
7. Comparación de distancias para asignar un cluster.
8. Cálculos para los nuevos valores de centroides.
9. Comparación de distancias para asignar un cluster.
10. Visualización final de los clusters.

B. Código del programa

```
# importar librerias
library(ggplot2)
library(dplyr)

# importar dataset iris
df <-
read.csv("C:\\Users\\hecto\\Downloads\\data
sets_19_420_Iris.csv")

# visualizacion de los primeros datos del
dataset y entendiendo la data
str(df)
summary(iris)
head(df$Id)

# Definimos numero de clusters
K <- c(1,2,3)

# Visualizacion de los datos
view <- df %>%
  ggplot(aes(x = SepalLengthCm, y =
SepalWidthCm)) + geom_point()
view

# primer paso sacar distancias de cada uno
de los clusters iniciales
distancia <- list(1,2,3)
for (var in K) {
  for (row in 1:nrow(df)) {
        len <- df[row, "SepalLengthCm"]
        wid <- df[row, "SepalWidthCm"]
        distancia[[var]][row] <- ((df[var,
"SepalLengthCm"] - len)^2+(df[var,
"SepalWidthCm"] - wid)^2)^0.5
  }
}
distancia

# creamos la columna para la clasificacion
data <- df %>%
  mutate(cluster = "")
data

# Comparacion de distancias para asignar un
cluster
for (row in 1:nrow(df)) {
  if (distancia[[1]][row] <
distancia[[2]][row]) {
        if (distancia[[1]][row] <
distancia[[3]][row]) {
        # pernetence al closter 1
        data$cluster[row] <- '1'
```

```
        }else{
        # pertenece al cluster 3
        data$cluster[row] <- '3'
        }
  }else{
        if (distancia[[2]][row] <
distancia[[3]][row]) {
        # pertenece al cluster 2
        data$cluster[row] <- '2'
        }else{
        # pertenece al cluster 3
        data$cluster[row] <-'3'
        }
  }
}

# calculos para los nueevos valores de
centroides
c1xa<- 0; c1ya<- 0; c2xa<- 0; c2ya<- 0;
c3xa<- 0; c3ya <- 0
repeat {

  c1x <- data %>%
        filter(cluster == "1") %>%
        select(SepalLengthCm)
  c1xS <- c1x %>%
        sum()
  c1xD <- c1x %>%
        summarize(count = n())
  c1x <- c1xS / c1xD

  c1y <- data %>%
        filter(cluster == "1") %>%
        select(SepalWidthCm)
  c1yS <- c1y %>%
        sum()
  c1yD <- c1y %>%
        summarize(count = n())
  c1y <- c1yS / c1yD

  c2x <- data %>%
        filter(cluster == "2") %>%
        select(SepalLengthCm)
  c2xS <- c2x %>%
        sum()
  c2xD <- c2x %>%
        summarize(count = n())
  c2x <- c2xS / c2xD

  c2y <- data %>%
        filter(cluster == "2") %>%
        select(SepalWidthCm)
  c2yS <- c2y %>%
        sum()
  c2yD <- c2y %>%
        summarize(count = n())
  c2y <- c2yS / c2yD

  c3x <- data %>%
        filter(cluster == "3") %>%
        select(SepalLengthCm)
  c3xS <- c3x %>%
        sum()
  c3xD <- c3x %>%
        summarize(count = n())
  c3x <- c3xS / c3xD

  c3y <- data %>%
        filter(cluster == "3") %>%
        select(SepalWidthCm)
  c3yS <- c3y %>%
        sum()
  c3yD <- c3y %>%
        summarize(count = n())
  c3y <- c3yS / c3yD

  distancia <- list(1,2,3)

  for (row in 1:nrow(df)) {
        len <- df[row, "SepalLengthCm"]
        wid <- df[row, "SepalWidthCm"]
        distancia[[1]][row] <- ((c1x -
len)^2+(c1y - wid)^2)^0.5
        distancia[[2]][row] <- ((c2x -
len)^2+(c2y - wid)^2)^0.5
        distancia[[3]][row] <- ((c3x -
len)^2+(c3y - wid)^2)^0.5
  }

  # Comparacion de distancias para asignar
un cluster
  for (row in 1:nrow(df)) {
        if (distancia[[1]][[row]][1] <
distancia[[2]][[row]][1]) {
        if (distancia[[1]][[row]][1] <
distancia[[3]][[row]][1]) {
        # pernetence al closter 1
        data$cluster[row] <- '1'

        }else{
        # pertenece al cluster 3
        data$cluster[row] <- '3'
        }
        }else{
        if (distancia[[2]][[row]][1] <
```

```
distancia[[3]][[row]][1]) {
        # pertenece al cluster 2
        data$cluster[row] <- '2'
        }else{
        # pertenece al cluster 3
        data$cluster[row] <-'3'
        }
        }
  }

  if (c1x == c1xa && c1y == c1ya && c2x ==
c2xa && c2y == c2ya && c3x == c3xa && c3y
== c3ya){
        break
        print("break")
  }

  c1xa <- c1x; c1ya <- c1y; c2xa <- c2x;
c2ya <- c2y; c3xa <- c3x; c3ya <- c3y

}

# visualizacion final de los clusters
lastview <- data %>%
  ggplot(aes(x = SepalLengthCm, y =
SepalWidthCm, col= cluster)) + geom_point()
lastview
```

RESULTADOS

En la Fig. 1 se muestra parte del conjunto de datos a manipular, para confirmar que la importación del dataset fue exitosa.

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
1	1	5.1	3.5	1.4	0.2	Iris-setosa
2	2	4.9	3.0	1.4	0.2	Iris-setosa
3	3	4.7	3.2	1.3	0.2	Iris-setosa
4	4	4.6	3.1	1.5	0.2	Iris-setosa
5	5	5.0	3.6	1.4	0.2	Iris-setosa
6	6	5.4	3.9	1.7	0.4	Iris-setosa

Fig. 1 Visualización del head del dataset

Como se mencionó en el tema III se utilizaron dos columnas para realizar el método de k means, “SepalLengthCm” el cual representa el largo del cepal en centímetros, y “SepalWidthCm” que representa el ancho del cepal en centímetros, aunque también se realizaron pruebas con las otras columnas para observar la diferencia de clusters formados, pero realmente no se identificó un cambio significativo y se decidió continuar utilizando la columna SepalLengthCm y SepalWidthCm.

Se utilizó una gráfica de dispersión de puntos para analizar el conjunto de datos. En la Fig. 2 se muestran los 150 datos, donde se puede observar a simple vista dos clusters distintos en este caso utilizaremos 3.

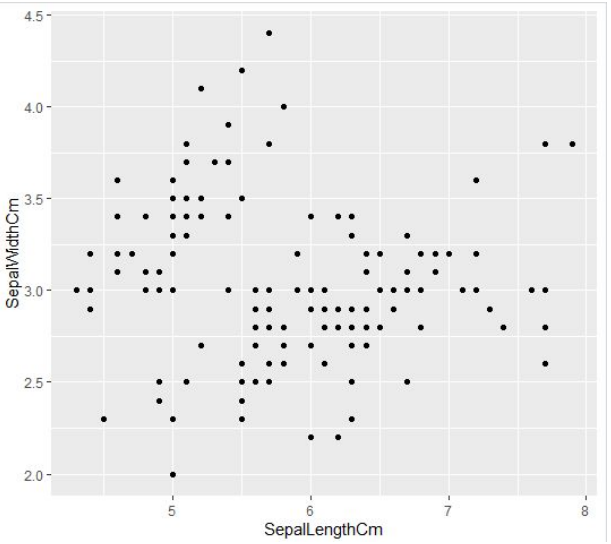

Fig.2 Visualización de datos por dispersión.

De acuerdo a los procedimientos realizados en el código, se obtuvieron los siguientes resultados del método de k means:

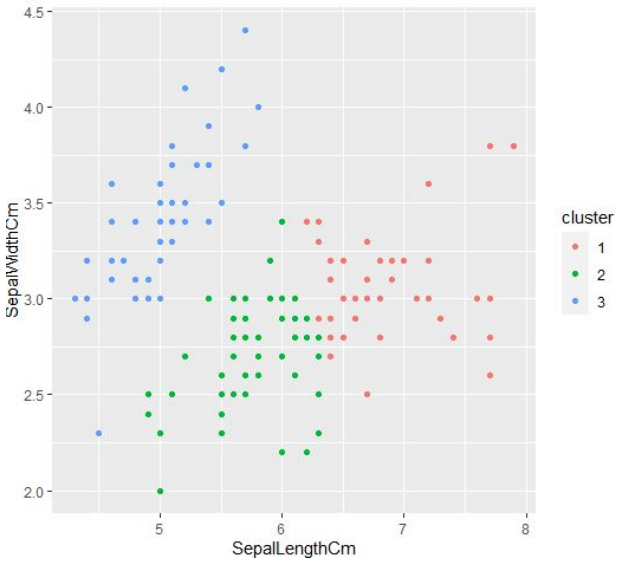

Fig. 3 Visualización por dispersión de los dato.

Realizando una análisis rápido de puede deducir que los tres clusters que se forman los cuales serían los tres tipos de flores de este dataset.

IV. CONCLUSIONES

Mediante k means se esperaba poder definir la tres clusters distintos los cuales se obtuvo en la totalidad y con los grupos bien definidos.

Aunque los datos que se analizaron se encontraban normalizados, el resultado fue el esperado ya que se obtuvo tres clusters bien definidos, lo cual quiere decir que el método de k means fue muy eficiente para analizar este dataset.

REFERENCIAS

[1] Surya Priy. (2016). Clustering in Machine Learning. Junio de 2020, de geeksforgeeks Sitio web: https://www.geeksforgeeks.org/clustering-in-machine-learning/

[2] Duk2. (2019). K-Means Clustering: Agrupamiento con Minería de datos. Junio de 2020, de estrategiastrading Sitio web: https://estrategiastrading.com/k-means/

[3] Imad Dabbura, (2018). K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks. Junio de 2020, de towardsdatascience Sitio web: https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a

[4] Unknown, (2008). k-means clustering algorithm. Junio de 2020, de sites.google.com Sitio web: https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm