Optativa IV. Machine Learning

Alumno: Gerardo Hernández Nava
Correo electrónico: ghernandez100@alumnos.uaq.mx

Algoritmos de agrupamiento de datos. K-means++ y EM-Clustering.

Tarea 04

I. Introducción.

Los problemas de agrupamiento probablemente sean uno de los temas más estudiados dentro del aprendizaje máquina y la minería de datos. El agrupamiento de datos tiene múltiples dominios de aplicación como en datos multimedia, de texto, redes sociales, datos biológicos y más [1].

En este trabajo se aplica dicha técnica de agrupamiento a la base de datos Iris, la cual es una de las bases de datos más conocidas y además, se encuentra en la literatura de reconocimiento de patrones.

La base de datos Iris, o también conocida como Iris de Fisher o Flor Iris, fue presentada por primera vez por Ronald Fisher en [2]. Dicha base de datos presenta 3 especies de iris -iris setosa, iris virginica e iris versicolor- cada una de ellas con 50 muestras.

El agrupamiento de la ya mencionada base de datos se realizará con dos algoritmos distintos, K-means ++ y el algoritmo de Expectation-Maximization (EM-Clustering).

II. Objetivo

El objetivo principal de esta práctica es implementar y comparar dos algoritmos de agrupamiento diferentes, los cuales serán K-means++ y EM-Clustering. Utilizando como datos a clasificar la base de datos Iris.

III. Marco teórico.

A. Clustering

El agrupamiento o clustering (en inglés) es una herramienta, específica del aprendizaje no supervisado, encargada de agrupar un conjunto de objetos sin etiqueta en clases o clusters que son altamente similares entre sí, es decir, comparten propiedades y/o características, mientras que son escasamente diferente al resto de objetos, dado que no tienen propiedades y/o características similares [3][4]. Básicamente, se desconoce si los datos ocultan patrones, así que el algoritmo se encarga de encontrarlos si es que los hay[5].

El agrupamiento tiene una gran cantidad de aplicaciones en distintos ámbitos de la vida, desde segmentación de mercado hasta en imágenes médicas, así como en análisis de redes sociales, agrupación de resultados de búsqueda, segmentación de imágenes, motores de búsqueda y más [3].

Existen distintos tipos de datos lo cual conlleva a que existan diferentes tipos de algoritmos, por ejemplo, aquellos basados en la densidad de los datos, en la distribución o los basados en centroides y jerarquías[5].

B. Algoritmo K-means ++

Este tipo de agrupamiento es el más popular y se basa en centroides. Sin embargo, presenta una desventaja, depende en gran medida de la inicialización de los centroides[6]. Es decir, si el centroide se inicializa en un punto alejado podría terminar sin puntos asociados a él o en caso contrario podría terminar con más de un cluster asociado, esto y otros factores pueden ocasionar un clustering deficiente (ver Figura 1 y Figura 2) [6].

Figura 1. Clustering deficiente.

Figura 2. Clustering ideal.

Debido a la desventaja que presenta la técnica K.means, surge K-means++, que intenta solucionar la situación asegurando que los centroides se inicialicen de mejor manera, de tal forma que se mejore la calidad del clustering; esencialmente esta es la única diferencia entre k-means y K-means ++, el resto del método es idéntico[6].

Principalmente, la inicialización consiste en seleccionar de forma aleatoria el primer centroide los puntos de datos, para posteriormente, calcular la distancia de cada punto de datos al centroide más cercano elegido en el paso uno. Después, se selecciona el siguiente centroide de los puntos de datos, de tal manera que el punto que se desea elegir como centroide tiene la distancia máxima medida desde el centroide más cercano y tiene la mayor probabilidad de ser elegido, y así, sucesivamente se repiten los dos pasos anteriores hasta que se cumplan para todos los k-ésimos centroides[7].

$$J = \sum_{k \in K} \sum_{i} z_k^i |x_i - \mu_k|^2$$

Donde:

xi: representa al objeto.

μk: media del cluster

C. Algoritmo EM

El algoritmo de maximización de expectativas o EM (Expectation Maximization), es también otra técnica de agrupación popular, de hecho, es la base de muchas otras, tal como el algoritmo de Markov oculto (HMM) [7].

Existen ocasiones en las que los modelos probabilísticos contienen tanto variables observadas como ocultas. Si todas las variables del modelo son observadas, dependiendo de los datos, el modelo puede estimarse mediante métodos de estimación, ya sea de máxima verosimilitud o bayesiano. Por otro lado, el algoritmo de maximización de expectativas toma lugar cuando el modelo contiene ambos tipos de variables, los modelos no se pueden estimar mediante los mismos métodos[7][8].

Esencialmente el algoritmo EM consiste en dos pasos:

1. Paso E: es el paso Expectativa, consiste en encontrar una función de límite inferior en la probabilidad original a través de la estimación actual de los parámetros estadísticos[8][9].

$$Q(\theta, \theta^{(t)}) = \sum_{i=1}^{n} \sum_{y=1}^{K} p(y|x_i, \theta^{(t)}) log \frac{p(x_i, y|\theta)}{p(y|x_i, \theta^{(t)})}$$

2. Paso M: es el paso de Maximización, ahora el algoritmo encontrará nuevas estimaciones de los parámetros estadísticos del pase E, de forma tal que maximice la función de límite inferior[8][9].

$$\theta^{(t+1)} = \arg\max_{\theta} Q(\theta, \theta^{(t)})$$

Después de estos pasos el algoritmo produce estimaciones cuya probabilidad es mayor que la anterior, hasta que convergen a un máximo.

IV. Metodología

A. K-Means ++

1. Se importaron las librerías necesarias para poder implementar el algoritmo, como numpy, matplot y random
2. Se comienzan a declarar las funciones en las cuales se va a basar nuestro algoritmo, cabe mencionar que algunas de ellas también servirán para el segundo algoritmo de agrupamiento.

   La primera de ellas fue la de normalización, en donde se ajustan los valores de la base de datos entre 0 y 1.

   La segunda, denominada *datos_entrenamiento_prueba*, nos ayudará a dividir la base de datos en los conjuntos de entrenamiento y de prueba. La función tiene como valor predefinido de división 80 por ciento de entrenamiento y 20 por ciento de prueba, pero puede ser modificado.

Posteriormente se definió la función con la que inicializamos los centroides dada la teoría de K-means ++.

En seguida se realizó una función para calcular la distancia euclidiana, la cual será utilizada en la siguiente función, mandándola a llamar de forma iterativa para obtener las distancias de cada uno de los centroides hacia todos los puntos.

Posteriormente se realizó una función para seleccionar los centroides a los cuales se va a asociar cada unos de los puntos, la asociación está dada por la distancia mínima calculada.

Después se implementó la función para poder actualizar los centroides, mediante la media de los datos agrupados en el cluster de cada centroide.

Para poder saber la precisión con la cual agrupó los datos nuestro algoritmo, se realizó una función en la cual se obtienen los grupos a los cuales se asignaron cada una de las clases y de igual manera esta función nos arroja una matriz de confusión en la cual podemos ver el número de aciertos y categorizaciones erróneas.

Para finalizar se generó una función para graficar los agrupamientos finales y poder tener un apoyo visual.

3. Se mandó a llamar la librería de sci-kit learn, en especial el módulo de dataset de donde se descarga la base de datos utilizada. Asignando el valor de los datos 'X' y de las etiquetas.
4. Se normaliza y se divide la base de datos en los conjuntos para entrenar y para probar el algoritmo implementado.
5. Se determinan el número de iteraciones; una variable que nos ayudará a saber si no ha habido cambios en los centroides durante ciertas iteraciones, el cual será uno de nuestros criterios de paro; y los centroides iniciales.
6. Se comienza con el entrenamiento de nuestro algoritmo. En un bucle limitado por el número de iteraciones asignado se calculan las distancias, se selecciona el centroide para cada uno de los datos, se actualizan los centroides, y se compara si hubo cambio en los centroides, en caso de que no se presente un cambio durante tres iteraciones se interrumpe el bucle.

7. Una vez terminado el entrenamiento, se probó el algoritmo calculando las distancias de cada uno de los datos hacia los centroides calculados y asignándoles el más cercano.
8. Para finalizar se calculó la precisión, matriz de confusión y se graficó la solución.

B. EM-Clustering.

1. Aparte de las librerías ya importadas para K-means ++ se mandaron llamar dos funciones (det e inv) de numpy.linalg para la implementación del algoritmo.
2. Ya que se van a ocupar varias funciones de las que se implementaron para el algoritmo anterior solo fue necesario implementar tres funciones más.

   La primera de ellas, es la función de densidad de la distribución normal. La senda fue la que corresponde al paso E o de cálculo de esperanza, que nos indica la probabilidad de que un dato pertenezca a un cluster.

   Por último, se implementó la función en la cual se llevará a cabo el paso M o de maximización, que es donde ajustamos la media, la covarianza y el valor pi, con los cuales tratamos de interpretar la distribución de cada una de nuestras clases.

3. Como ya se realizó la importación de la base de datos, su normalización y la división de la misma en la sección de K-means ++, se procede al entrenamiento del algoritmo.

   Primero se inicializan el número de iteraciones, los valores de pi y el número de clases. De igual forma se divide el conjunto de entrenamiento en tres secciones, que son el número de clases, para poder calcular una media inicial y una covarianza inicial.

   Una vez calculados y declarados los valores iniciales inicia un bucle que durará el número de iteraciones que hayamos seleccionado. Dentro de él llamamos primero a nuestra función del paso E y posteriormente realizamos el paso M.

   Ya que se cumplieron el número de iteraciones se le asigna un grupo a cada dato dada la probabilidad calculada.

4. Ya que se terminó de entrenar se pasa a la prueba del algoritmo, ingresando la x de prueba, junto con el número de clústeres, el valor pi, medias y covarianzas obtenidas en el entrenamiento, a la función de paso_E. Una vez que arroje la probabilidad se asigna un grupo a cada uno de los datos.
5. Se procede a calcular la precisión del algoritmo y a graficar la solución final.

V. Resultados

A continuación, se muestran los resultados obtenidos a partir de la modelación de los algoritmos de clustering.

Clases reales

Se presentan los datos (clases reales) divididos en el conjunto de entrenamiento y conjunto de prueba.

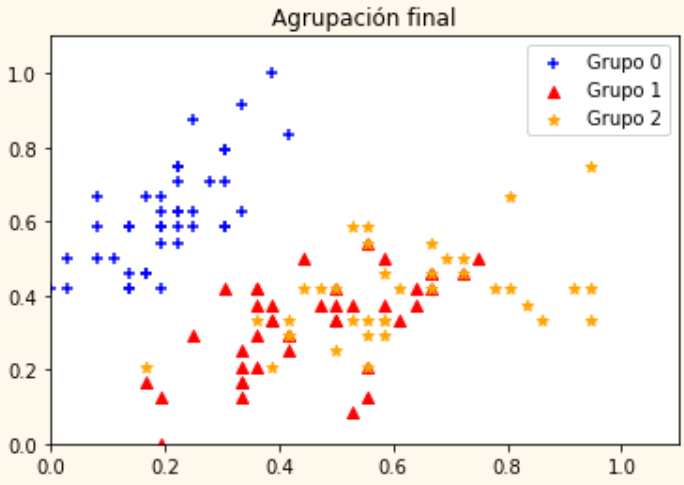


Figura 3. Clases reales para el entrenamiento.

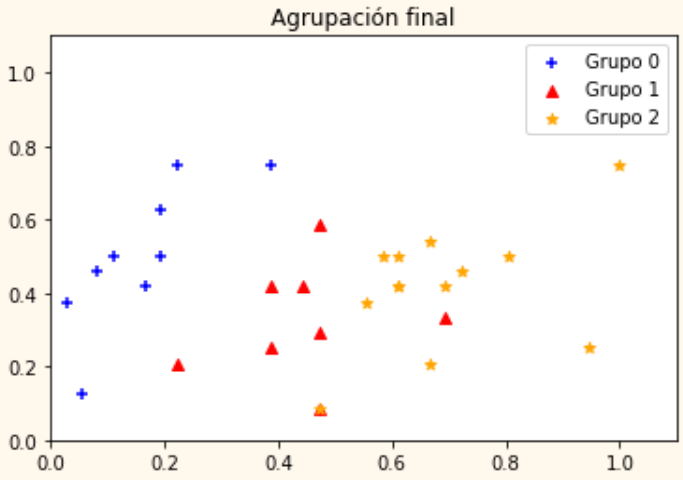


Figura 4. Clases reales para la etapa de prueba.

***K-means* ++**

Enseguida se muestran los resultados de aplicar el algoritmo K-means ++ a los conjuntos de datos de entrenamiento (120 datos) y prueba (30 datos).

- *Entrenamiento*

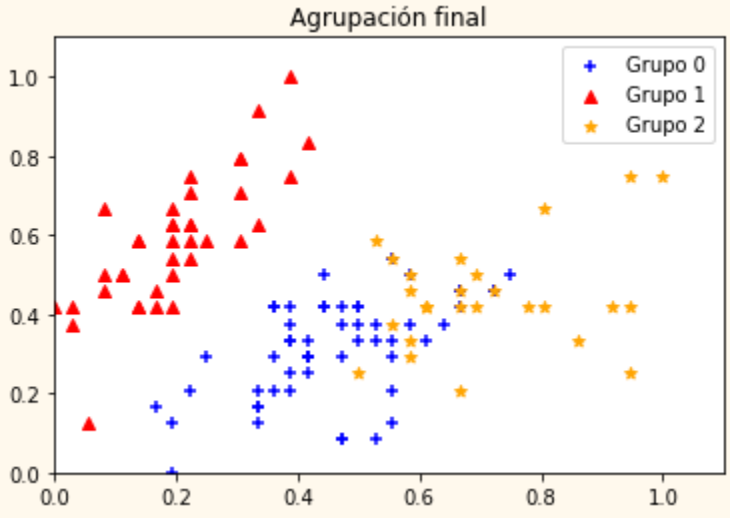


Figura 5. Resultado K-means++ con datos de entrenamiento.

Tabla 1. Matriz de confusión de agrupamientos de resultado K-means++ con datos de entrenamiento.

Clase \ Grupo asignado	0	1	2
0	0	38	0
1	43	0	0
2	12	0	27

En la matriz de confusión, las celdas de color verde indican aquellas clases que fueron agrupadas correctamente, es decir, los aciertos del algoritmo.

Los valores de precisión para cada clase son las siguientes:

Precisión clase 0 = 100%

Precisión clase 1 = 100%

Precisión clase 2 = 69%

- *Prueba*

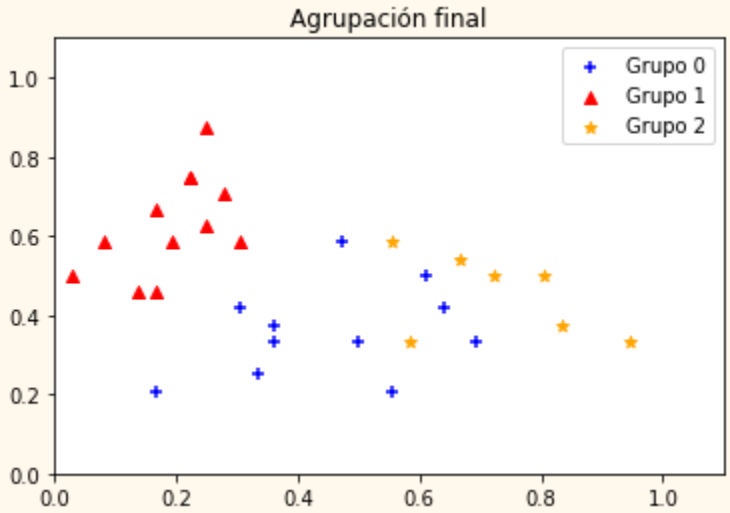


Figura 6. Resultado K-means++ con datos de prueba.

Tabla 2. Matriz de confusión de agrupamientos de resultado K-means++ con datos de prueba.

Clase \ Grupo asignado	0	1	2
0	0	12	0
1	7	0	0
2	4	0	7

En la matriz de confusión, las celdas de color verde indican aquellas clases que fueron agrupadas correctamente, es decir, los aciertos del algoritmo.

Los valores de precisión para cada clase son las siguientes:

Precisión clase 0 = 100%

Precisión clase 1 = 100%

Precisión clase 2 = 69%

EM Clustering

Enseguida se muestran los resultados de aplicar el algoritmo EM Clustering a los conjuntos de datos de entrenamiento y prueba.

- *Entrenamiento*

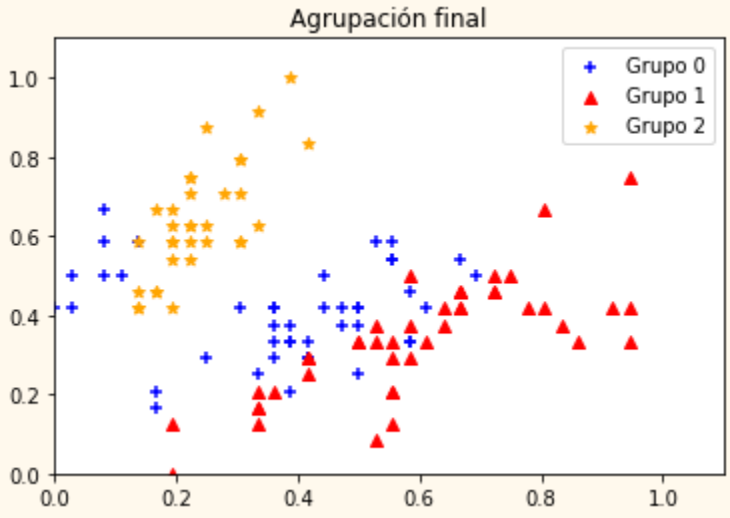


Figura 7. Resultado del algoritmo EM con datos de entrenamiento.

Tabla 3. Matriz de confusión de agrupamientos de resultado del algoritmo EM con datos de entrenamiento

Clase \ Grupo asignado	0	1	2
0	38	0	0
1	0	19	24
2	0	25	14

En la matriz de confusión, las celdas de color verde indican aquellas clases que fueron agrupadas correctamente, es decir, los aciertos del algoritmo.

Los valores de precisión para cada clase son las siguientes:

Precisión clase 0 = 100%

Precisión clase 1 = 55%

Precisión clase 2 = 64%

- *Prueba*

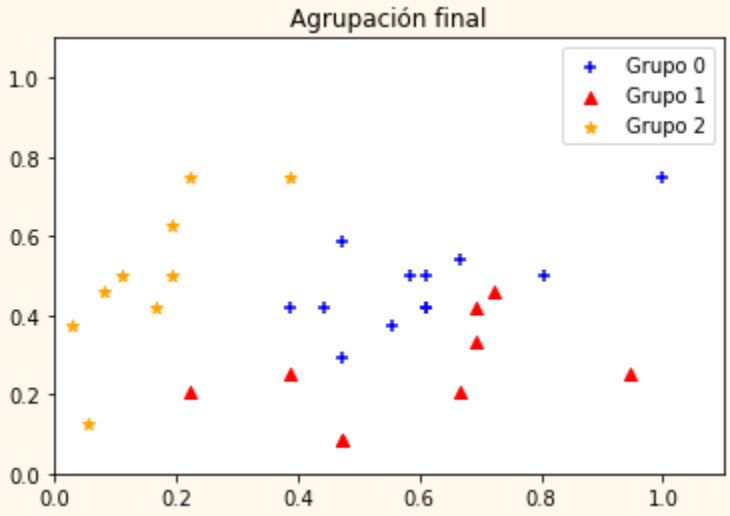


Figura 8. Resultado del algoritmo EM con datos de prueba.

Tabla 4. Matriz de confusión de agrupamientos de resultado del algoritmo EM con datos de prueba.

Clase \ Grupo asignado	0	1	2
0	12	0	0
1	0	3	4
2	0	8	3

En la matriz de confusión, las celdas de color verde indican aquellas clases que fueron agrupadas correctamente, es decir, los aciertos del algoritmo.

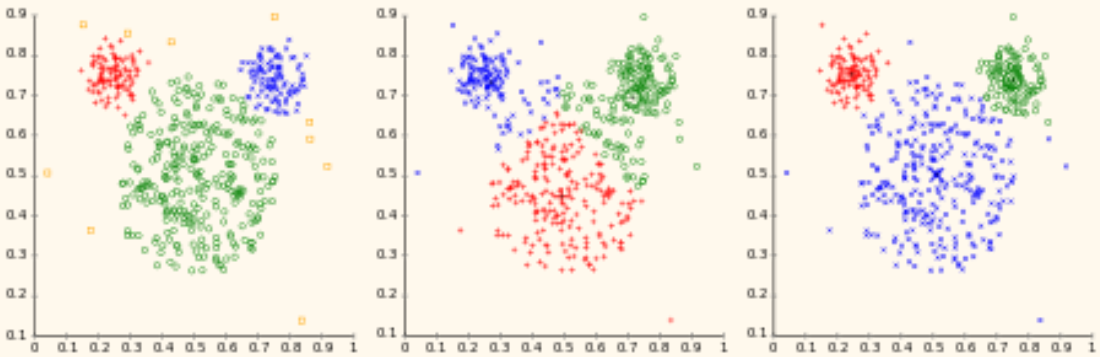
Los valores de precisión para cada clase son las siguientes:

Precisión clase 0 = 100%

Precisión clase 1 = 57%

Precisión clase 2 = 72%

VI. Conclusiones



Finalmente, se puede concluir que la mayor diferencia entre los algoritmos de clustering K-means y EM es la flexibilidad de este último, pues contrario a K-means, no se limita en formas esféricas. Sin embargo, en ambos métodos, se debe realizar un preprocesamiento de los datos, dado que, ninguno de ellos tiene la capacidad de detectar aquellos valores que sean atípicos.

Referencias.

[1] Gan, G., Ma, C., & Wu, J. (2020). Data clustering: theory, algorithms, and applications. Society for Industrial and Applied Mathematics.

[2] Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. Annals of eugenics, 7(2), 179-188.

[3]“Cluster Analysis Definition | DeepAI.” [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/cluster-analysis. [Accessed: 28-Mar-2022].

[4]“Algoritmos de Agrupamiento - 🤖 Aprende IA.” [Online]. Available: https://aprendeia.com/algoritmos-de-clustering-agrupamiento-aprendizaje-no-supervisado/. [Accessed: 28-Mar-2022].

[5]“8 algoritmos de agrupación en clústeres en el aprendizaje automático que todos los científicos de datos deben conocer.” [Online]. Available: https://www.freecodecamp.org/espanol/news/8-algoritmos-de-agrupacion-en-clusteres-en-el-aprendizaje-automatico-que-todos-los-cientificos-de-datos-deben-conocer/. [Accessed: 28-Mar-2022].

[6]“ML | K-means++ Algorithm - GeeksforGeeks.” [Online]. Available: https://www.geeksforgeeks.org/ml-k-means-algorithm/. [Accessed: 28-Mar-2022].

[7]“A Comparison Between K-Means & EM For Clustering Multispectral LiDAR Data | by Faizaan Naveed | Towards Data Science.” [Online]. Available: https://towardsdatascience.com/a-comparison-between-k-means-clustering-and-expectation-maximization-estimation-for-clustering-8c75a1193eb7. [Accessed: 28-Mar-2022].

[8]“Introducción al algoritmo de maximización de expectativas - programador clic.” [Online]. Available: https://programmerclick.com/article/84951279974/. [Accessed: 28-Mar-2022].

[9]“Expectation-Maximization Algorithm on Python | by PRATEEK KUMAR | Medium.” [Online]. Available: https://medium.com/@prateek.shubham.94/expectation-maximization-algorithm-7a4d1b65ca55. [Accessed: 28-Mar-2022].