Técnicas de agrupación de datos mediante aprendizaje no supervisado K-Means, Spectral Clustering y Mezcla de Gaussianas

Ana Toxqui, *MMOP, CIMAT,*

***Abstract*—El trabajo presenta un análisis de tres técnicas de clustering de datos: el algoritmo K-Means, Spectral Clustering y la Mezcla de Gaussianas. Estas técnicas se aplicaron a diferentes tipos de datos, incluyendo la clasificación de dígitos, cuantificación de imágenes y segmentación de imágenes.**

***Index Terms*—Clustering de datos, K-Means, Spectral Clustering, Mezcla de Gaussianas, Cuantificación de imágenes, Segmentación de imágenes, Aprendizaje no supervisado, Minería de datosAlgoritmos de clustering, Evaluación de algoritmos, Reducción de colores**

I. Introducción

ACTUALMENTE el análisis e interpretación de datos se ha convertido en un elemento esencial para la toma de decisiones en proyectos de investigación, empresas y en múltiples disciplinas. Una parte importante del análisis es encontrar patrones dentro de un conjunto de datos, esto se puede hacer de muchas formas, en particular se puede realizar mediante clustering o agrupamiento de datos, ya que permite organizar la información no etiquetadas en grupos que poseen características similares, permitiendo así comprender patrones ocultos en los datos.

En este documento se exploran tres técnicas para hacer clustering de datos, las cuales son:el algoritmo K-Means, Spectral Clustering y la Mezcla de Gaussianas. Se probaron para diferentes tipos de datos y se aplicaron al problema de clasificación de dígitos, cuantificación de imágenes y segmentación de imágenes. En la siguiente sección se desarrollá la teoria de cada una de las técnicas de clustering, posteriormente se explica como se implementaron estas funciones en python, más adelante se discuten los resultados para finalmente dar las recomendaciones y conclusiones alcanzadas en este trabajo.

II. Marco Teórico

El clustering, también conocido como agrupamiento, es una técnica de aprendizaje no supervisado que tiene como objetivo dividir un conjunto de datos en grupos o clústeres basados en la similitud entre sus elementos. Este enfoque se utiliza para descubrir estructuras ocultas y patrones intrínsecos en los datos. Los algoritmos de clustering se pueden aplicar en diversas áreas, como la minería de datos, la biología, la visión por computadora y más.

A. Algoritmo de K-means

K-Means es uno de los algoritmos de clustering más populares y ampliamente utilizados. Este método se basa en la partición de los datos en K clústeres, donde K es un valor predefinido. El algoritmo comienza asignando aleatoriamente K centroides y luego asigna cada punto de datos al centroide más cercano. Los centroides se actualizan iterativamente hasta que se alcanza la convergencia. K-Means busca minimizar la suma de las distancias cuadradas entre los puntos y sus centroides asignados.

Algorithm 1 Algoritmo K-Means

1: Inicializar los centroides aleatoriamente
2: **repeat**
3: Asignar cada punto de datos al centroide más cercano
4: Calcular nuevos centroides como el promedio de los puntos asignados a cada centroide
5: **until** No hay cambios significativos en la asignación de puntos

B. Algoritmo de Spectral Clustering

Spectral Clustering es un enfoque que se basa en la representación espectral de los datos. Este método se utiliza para encontrar clústeres en datos que pueden no ser linealmente separables. Implica la construcción de una matriz de afinidad que mide las relaciones entre los puntos de datos y luego la descomposición espectral de esta matriz. La descomposición espectral permite identificar los clústeres en función de los eigenvectores asociados a los eigenvectores más pequeños.

Algorithm 2 Algoritmo Spectral Clustering

1: Construir la matriz de afinidad
2: Calcular los eigenvectores y eigenvalores de la matriz de afinidad
3: Seleccionar los primeros k eigenvectores
4: Normalizar las filas de la matriz de eigenvectores
5: Aplicar K-Means a las filas de la matriz de eigenvectores normalizada para clusterizar

C. Mezcla de Gaussianas

La Mezcla de Gaussianas (Gaussian Mixture Model, GMM) es un modelo probabilístico que asume que los datos provienen de una combinación de varias distribuciones gaussianas. Cada componente gaussiano representa un clúster en los datos. El proceso de ajuste de un GMM implica estimar los parámetros de cada gaussiana y luego asignar cada punto de datos a la gaussiana con la probabilidad más alta.

Algorithm 3 Algoritmo Mezcla de Gaussianas

1: Inicializar los parámetros de las gaussianas (medias, covarianzas y pesos)
2: **repeat**
3: Calcular las probabilidades de pertenencia a cada gaussiana para cada punto
4: Asignar cada punto al cluster de probabilidad más alta
5: Actualizar los parámetros de las gaussianas basados en las probabilidades
6: **until** Convergencia

D. Cuantificación y segmentación de imágenes

La cuantificación de imágenes y la segmentación de imágenes son aplicaciones importantes del clustering en el campo de la visión por computadora. La cuantificación de imágenes se refiere a la reducción de la cantidad de colores o niveles de gris en una imagen, lo que permite una representación más compacta. La segmentación de imágenes implica la división de una imagen en regiones o segmentos significativos con características similares, lo que facilita el análisis y la interpretación de la imagen.

III. Implementación en Python

En esta sección se explica como se realizó la implementación de cada algoritmo en python.

Para el algoritmo de k-means se tomaron tres parámetros: datos (los datos que deseas agrupar), K (el número de clústeres que deseas formar) y iteraciones (el número de veces que se realizarán las iteraciones para ajustar los clústeres). Primero se inicializan los centros de los clústeres de manera aleatoria seleccionando K puntos aleatorios del conjunto de datos datos. Luego, se realiza un bucle a lo largo de iteraciones, donde se realizará la asignación de datos a clústeres y la actualización de los centros. En cada iteración, se crea una lista de clusters vacía que contendrá los puntos de datos asignados a cada clúster.

A continuación, se calcula la distancia entre cada punto de datos y todos los centros, y se asigna el punto de datos al clúster con el centro más cercano. Después de asignar todos los puntos de datos a los clústeres, se actualizan los centros de cada clúster como el promedio de todos los puntos de datos en ese clúster. El proceso se repite durante el número especificado de iteraciones. Finalmente, la función devuelve los centros finales de los clústeres y una lista de clústeres, donde cada clúster es una lista de puntos de datos que pertenecen a ese clúster.

Para el algoritmo de Spectral Clustering se programo la función SpecClus la cual tiene dos parámetros: datos (los datos que deseas agrupar) y K (el número de clústeres que deseas formar). Se comienza por calcular la matriz de similitud utilizando el kernel gaussiano. Esta matriz W mide la similitud entre todos los pares de puntos de datos en datos. La similitud se calcula utilizando la distancia euclidiana entre los puntos de datos y se transforma con una función gaussiana. Posteriormente se calcula la matriz diagonal D, que contiene la suma de las similitudes de cada punto de datos con todos los demás puntos. Con estas dos se calcula la matriz Laplaciana L, que es la diferencia entre D y W.

Más adelante se calcula los eigenvectores y eigenvalores de L. Luego, selecciona los primeros k eigenvectores correspondientes a los k eigenvalores más grandes (distintos de cero). El valor de k es el número de clústeres que deseas formar. Finalmente se aplica el algoritmo K-Means a los eigenvectores seleccionados para agrupar los datos en k clústeres. La función devuelve las etiquetas de los clústeres a los que se asigna cada punto de datos y los centroides de los clústeres.

La implementación del algoritmo de mezcla de Gaussianas se realizó a través de una función llamada MG la cual toma tres parámetros: datos (los datos que deseas agrupar), k (el número de componentes Gaussianas o clústeres que deseas ajustar) y iteraciones (el número de iteraciones que realizará el algoritmo para ajustar los parámetros del modelo). Se inicializa las medias de las componentes Gaussianas de manera aleatoria seleccionando k puntos aleatorios del conjunto de datos. Además, inicializa las varianzas de las componentes Gaussianas con matrices de identidad y los pesos de cada componente Gaussiana de manera uniforme.

Luego, el algoritmo entra en un bucle que consta de iteraciones. En cada iteración, se realiza el proceso de Expectation-Maximization (E-M) para ajustar los parámetros del modelo de mezcla de gaussianas.

En la etapa de Expectation (E), se calculan las probabilidades de pertenencia de cada punto de datos a cada componente Gaussiana utilizando la función de densidad de probabilidad de una distribución Gaussiana la cual también se programó. Estas probabilidades se utilizan para asignar cada punto de datos al clúster más probable.

En la etapa de Maximization (M), se actualizan las medias, varianzas y pesos de cada componente Gaussiana en función de las asignaciones realizadas en la etapa de Expectation. Esto implica el cálculo de las medias ponderadas de los puntos de datos asignados a cada componente, la estimación de las nuevas varianzas y el ajuste de los pesos.

El proceso de Expectation-Maximization se repite durante el número especificado de iteraciones. La función devuelve

las asignaciones de clústeres para cada punto de datos y las medias finales de las componentes Gaussianas las cuales son usadas para graficar los puntos.

IV. Resultados

En esta sección se muestran los resultados alcanzados en la experimentación de cada uno de los algoritmos.

A. Agrupación de datos

Se probaron los tres algoritmos de clusterización para cuatro conjuntos diferentes de datos, el primero consta de 10000 puntos agrupados en 40 clusters diferentes, el segundo son puntos que forman dos medias lunas y que se consideran más difíciles de clusterizar, el tercer conjunto son tres poblaciones que a simple vista se pueden separar en tres secciones, finalmente se tienen datos que forman dos círculos concéntricos.

Para el primer conjunto de datos , se realizó de manera correcta la división en los 40 clusters, debido a que no existe una solución única para la división se considera que los tres algoritmos realizaron la tarea de forma adecuada. El tercer conjunto de datos se clasifico de manera similar en cada algoritmo y de manera eficiente.

Por otro lado en el segundo conjunto ningún algoritmo pudo realizar la división en dos clustes de manera que seleccionaran cada uno una media luna, el algoritmo de mezcla de gaussianas muestra un mejor acomodo de los clusters sobre los otros dos. En el cuarto conjunto de puntos ningún algoritmo pudo realizar la división en círculos, los resultados se pueden apreciar en la figura 1.

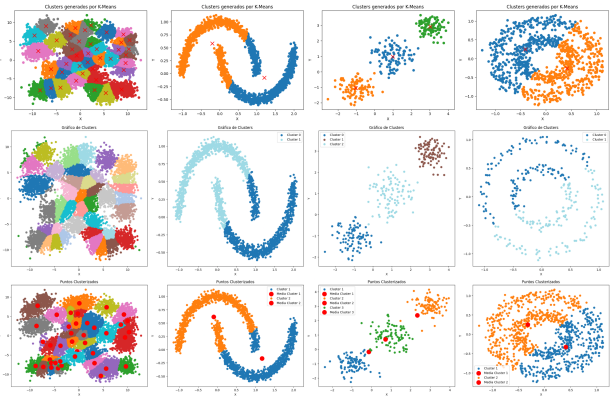

Fig. 1. Resultados para la agrupación de datos, primera fila K-means, segunda fila Spectral Clustering, tercer fila Mezcla de Gaussianas

B. Clasificación de dígitos

El problema de clasificación de dígitos es un tipo de tarea de aprendizaje automático en la que se intenta asignar una etiqueta o categoría a una imagen que contiene un dígito retomado de un conjunto de datos. Comúnmente, se utiliza un conjunto de datos que contiene imágenes de dígitos, donde cada imagen se asocia con la etiqueta correcta que representa el dígito correspondiente en este caso del 0 al 9.

Para este trabajo se realizó la clasificación con los tres algoritmos y con k-means de la paquetería sklearn de python, como se puede ver en la tabla 1, la clasificación con Spectral Clusterin es mucho mejor que la de K-Means pues se acerca más a la de sklearn, sin embargo debido a que los datos que representan a los dígitos tienen valores 0,1, no es posible llevar la implementación de Mezcla de medias pues la matriz de varianzas es singular.

	K-means	S-C	M-G	sklearn
Cluster 0	181	169	SM	182
Cluster 1	259	177	SM	156
Cluster 2	230	195	SM	197
Cluster 3	202	138	SM	179
Cluster 4	178	183	SM	180
Cluster 5	177	181	SM	176
Cluster 6	29	191	SM	166
Cluster 7	150	169	SM	242
Cluster 8	152	222	SM	93
Cluster 9	239	183	SM	226

TABLE I
Resultados para clasificación de dígitos

C. Cuantificación de imágenes

Para el problema de cuantificación de imágenes se tomo una imagen inicial y se clusterizó en treinta colores, esto se realizó con cada algoritmo. Por un lado se tiene que el algoritmo de k-means es rápido pero es el menos eficiente, mientras que Spectral Clustering es mucho mejor pero tarda mucho tiempo. Finalmente la mezcla de gaussianas tarda más que K-means pero mucho menos que Spectral Clustering yda resultados buenos.

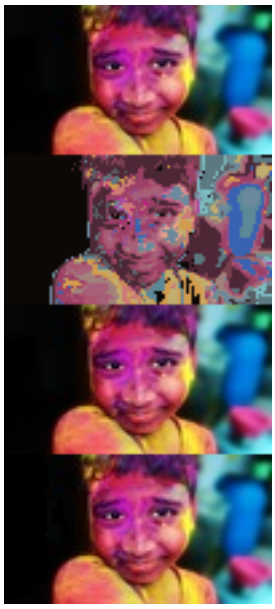

Fig. 2. Resultados para la cuantificación de imágenes: primera fila es la imagen original, segunda fila imagen generada por k-means, tercer fila imagen generada por Spectral Clustering, cuarta fila imagen generada por mezcla de gaussianas

D. Segmentación de imágenes

En el caso de la segmentación los resultados no fueron buenos para ninguno de los algoritmos ya que al iniciar con una imagen en escala de grises, las agrupaciones en los clusters no permiten extraer de manera eficiente las secciones que componen la imagen inicial.

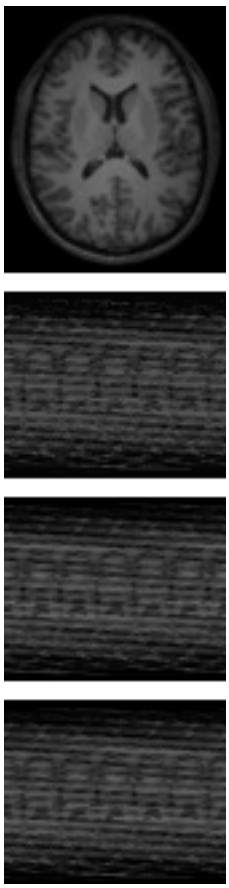

Fig. 3. Resultados para la segmentación de imágenes: primera fila es la imagen original, segunda fila imagen generada por k-means, tercer fila imagen generada por Spectral Clustering, cuarta fila imagen generada por mezcla de gaussianas

V. Análisis y Conclusión

Las implementaciones en python no son las mejores, sin embargo se muestra ventaja del algoritmo de mezcla de gaussianas sobre los otros algoritmos para los problemas de segmentación y cuantificación de imágenes, para el problema de clasificación de dígitos se hizo notar una debilidad del algoritmo que es, para ciertos datos la matriz de varianzas es singular por lo que no es posible realizar el algoritmo.

Los resultados obtenidos no reflejan el poder y las ventajas y desventajas que tiene cada algoritmo, esto se debe a que las implementaciones se consideran básicas, para mejorar los resultados es necesario considerar la naturaleza de los datos y mejorar algunas partes del código como los son la inicialización de parámetros de manera más avanzada o la detección de convergencia de manera más precisa. Para futuros trabajos se recomienda revisar que el algoritmo sea el óptimo para las estructuras de datos que se desee clusterizar.

VI. Referencias

- Shalev-Shwartz, S. (2014). Shai Ben-David - Understanding Machine Learning- From Theory To Algorithms. Cambridge University Press.
- kmeans. (s/f). Unioviedo.es. Recuperado el 12 de septiembre de 2023, de https://www.unioviedo.es/compnum/laboratorios-py/kmeans/kmeans.html