Informaticae Abstracta
Año 2024, Vol. 2, No. 1, pp. 4-24 ENE | JUN 2024

Algoritmos de clustering para la detección de posibles paradas de autobús

Sánchez Carmona, M. A.

Información del Artículo

Recibido: 3-may-2023
Aceptado: 18-ago-2023

Palabras clave: Clustering, k-means, DBSCAN, HDBSCAN, OPTICS, Códigos postales, Paradas de autobús, Haversine

Resumen

Este artículo propone una comparativa entre cuatro algoritmos de agrupamiento para la identificación de posibles paradas de autobús. Los algoritmos seleccionados fueron *k-means*, el *Agrupamiento Espacial Basado en Densidad de Aplicaciones con Ruido* (DBSCAN, por sus siglas en inglés), *Agrupamiento Espacial Jerárquico Basado en Densidad de Aplicaciones con Ruido* (HDBSCAN, por sus siglas en inglés) y el algoritmo de *Ordenación de Puntos para Identificar la Estructura de Clusters* (OPTICS, por sus siglas en inglés). El agrupamiento se aplicó a un conjunto de datos de una matrícula escolar utilizando la métrica de distancia *Haversine*. Los resultados del conteo de *clusters* y de *elementos no agrupados*, así como de la aplicación de métricas de calidad de los clusters (referidas en la literatura), mostraron que HDBSCAN tiene mejor desempeño que DBSCAN y OPTICS al generar una cantidad adecuada de clusters; por otro lado k-means mostró un mejor rendimiento computacional y mejores resultados de entre todos los algoritmos; sin embargo la cantidad de clusters son menores y por consiguiente inviables para ser usados como paradas de autobús.

1. Introducción

El agrupamiento (clustering, en inglés) consiste en formar grupos (clusters) con elementos que presentan características similares y se utiliza principalmente para determinar patrones climáticos, análisis cartográficos, análisis de datos, segmentación de imágenes, agrupación de artículos por temas o para segmentar clientes, entre otros [1]. Es una técnica de *Machine Learning*, que tiene como objetivo separar datos en grupos *homogéneos* con características comunes [2, 3]. Su uso en esta disciplina se enfoca en el *aprendizaje automático*, en la categoría del *aprendizaje no supervisado*; donde los dos problemas más comunes son el análisis de mixturas (mezclas) o agrupaciones y el aprendizaje de variables [3].

En el área del transporte el clustering ha sido utilizado para determinar y establecer paradas vehiculares, Liu y Wang [4] proponen un estudio enfocado en un "transporte en situación de emergencia", cuando el transporte ferroviario presenta fallas y los usuarios quedan varados; establecen un proceso de agrupamiento de datos en tiempo real, con el uso de un algoritmo de clustering DBSCAN mejorado y adaptativo, logrando determinar los puntos adecuados para establecer las paradas vehiculares donde se concentra la mayor cantidad de usuarios, con lo cual establecen rutas de transporte adecuadas para su modelo matemático.

Algunos algoritmos de clustering necesitan ser configurados para realizar el proceso de agrupamiento mediante un radio (distancia) de búsqueda para encontrar el elemento a añadirse al cluster. Sciortino et al. [5] describen las características

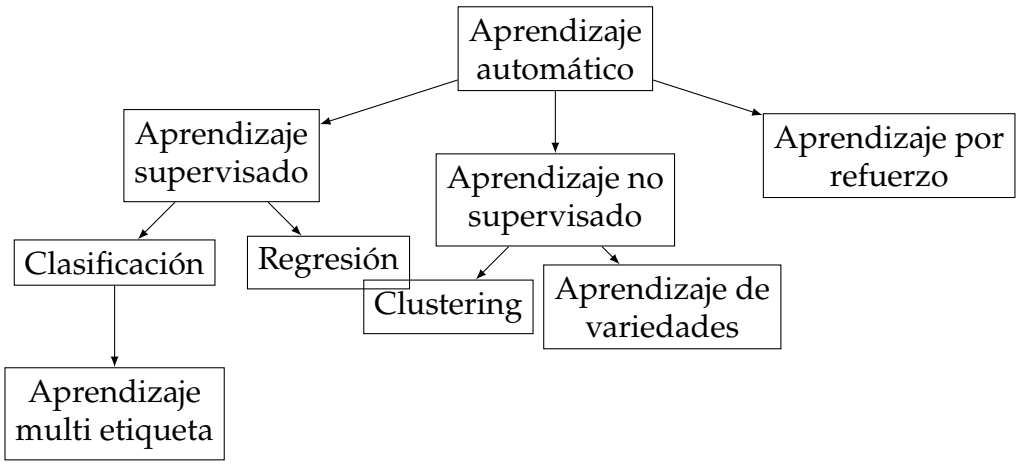


Figura 1: Esquema de las ramas principales del aprendizaje automático

de un Problema de Enrutamiento de Autobuses Escolares (SBRP, por su siglas en inglés), que es un problema del mundo real compuesto por varios subproblemas (selección de paradas de autobús, generación de rutas de autobús, entre otros), establecen mediante estudios estadísticos, que los alumnos que utilizan un servicio de transporte escolar puede recorrer una distancia de 1,6 km para trasladarse a la parada de autobús. Con este tipo de estudios es factible determinar parámetros para los algoritmos de clustering que requieren ser configurados con una distancia.

El objetivo de este trabajo es comparar cuatro algoritmos de clustering para la generación de grupos con datos espaciales, utilizando una métrica de distancia adecuada que permita relacionar cada elemento o elementos para determinar si pertenecen o no a un grupo en particular y finalmente establecer puntos de parada para un servicio de autobuses. La métrica más utilizada es la distancia euclidiana, pero existen otras métricas como la distancia Manhattan, la distancia sobre una esfera con fórmula Haversine, entre otras [6]. Los algoritmos de clustering propuestos, bastante conocidos en el área del aprendizaje no supervisado, son k-means y DBSCAN, así como variaciones de este último como son HDBSCAN y OPTICS. En la siguientes secciones se describen los algoritmos de clustering utilizados, métricas de distancia y valoración de la calidad de los clusters, así como del preprocesamiento de datos realizados y los resultados obtenidos.

2. Aprendizaje no supervisado

En el área de la inteligencia artificial (IA) es común mencionar el término aprendizaje automático, considerado como una rama de la IA, la cual trata temas relacionados a la predicción y a la toma automática de decisiones, así como técnicas de representación, descripción y exploración de datos [3]. El aprendizaje automático cuenta con varias ramas, una clasificación breve se puede apreciar en la Figura 1. En la rama del *aprendizaje no supervisado* se encuentra la categoría de *clustering*, en dicha área radica una variedad de algoritmos cuya tarea es formar grupos de datos [1].

k-means

Es uno de los algoritmos de clustering más populares y ampliamente utilizados en el campo del aprendizaje automático no supervisado [6]. La letra k en k-means se refiere al número de clusters que el algoritmo intenta encontrar en un conjunto de datos. Su objetivo principal es particionar un conjunto de datos en k clusters, donde cada punto de datos pertenece al cluster cuyo *centroide* está más cerca [7].

Características y funcionamiento

- Se puede utilizar para segmentar clientes en diferentes grupos basados en sus características, como comportamientos de compra o preferencias.
- En minería de textos se puede agrupar documentos similares para tareas como la clasificación de textos o la recomendación de contenido.
- Puede utilizarse para reducir el número de colores en una imagen, lo que resulta en una representación comprimida de la misma.

Algoritmo

El algoritmo k-means opera de la siguiente manera:

1. Se comienza seleccionando aleatoriamente k puntos como centroides iniciales.
2. Cada punto de datos se asigna al cluster cuyo centroide está más cerca según alguna medida de distancia (comúnmente la distancia euclidiana).
3. Una vez que todos los puntos han sido asignados a un cluster, los centroides se recalculan como el promedio de todos los puntos asignados a ese cluster.
4. Los pasos 2 y 3 se repiten hasta que los centroides no cambien significativamente entre iteraciones o hasta que se alcance un número máximo de iteraciones.
5. El algoritmo converge cuando los centroides no cambian significativamente entre iteraciones o cuando se alcanza un número máximo de iteraciones predefinido.

El pseudocódigo del algoritmo k-means se muestra en el Algoritmo 1

Algorithm 1: Algoritmo k-means

kmeansdatos, k, max_iter centros ← inicializar_centros(datos, k) iteraciones ← 0 **while** iteraciones < max_iter **do**-_

clusters ← asignar_datos_a_clusters(datos, centros)
centros_nuevos ← calcular_nuevos_centros(datos, clusters) **if** centros_nuevos ≈ centros
_**then**

break centros ← centros_nuevos iteraciones ← iteraciones + 1 **return** clusters

La complejidad computacional promedio del algoritmo k-means esta dada por $O(k \cdot n \cdot T)$, donde k es el número de clusters, n es el número de muestras y T el

número de iteraciones; para el peor de los casos la complejidad del algoritmo es de $O(n^{\frac{k+2}{p}})$ con p siendo el número de características [8].

Agrupamiento Espacial Basado en Densidad de Aplicaciones con Ruido

Este algoritmo de agrupamiento de datos, conocido simplemente como DBSCAN, propuesto por Martin Ester et al. [9], se le denomina algoritmo de agrupamiento basado en densidad, porque encuentra un número de clusters comenzando por una estimación de la distribución de densidad de los nodos correspondientes. Es uno de los algoritmos de agrupamiento más usados y citados en la literatura científica [9].

Características y funcionamiento

DBSCAN presentan varias elementos que le permiten su eficiencia en el agrupamiento de datos espaciales, las características principales son:

- Para cada observación se mira el número de puntos a una distancia máxima eps de ella (esta zona se denomina eps-vecindad de la observación).
- Si una observación tiene al menos un cierto número de vecinos, incluida ella misma, se considera una observación central.
- Todas las observaciones en la vecindad de una observación central pertenecen al mismo grupo.
- Cualquier observación que no sea una observación central y que no tenga ninguna observación central en su vecindad se considera una *anomalía*.

Algoritmo

El proceso general que realiza DBSCAN es el siguiente:

1. Encontrar los puntos en la eps-vecindad de cada punto e identificar los puntos centrales con la mayor cantidad de $minPts$ vecinos.
2. Encontrar los componentes conectados de los puntos centrales en el gráfico vecino, ignorando todos los puntos no centrales.
3. Asignar cada punto no central a un grupo cercano si el grupo es un vecino eps; de lo contrario, asignarlo como un dato no etiquetado comúnmente denominado ruido.

Este proceso continúa hasta construir completamente un grupo densamente conectado. Entonces, un nuevo punto no visitado se visita y procesa con el objetivo de descubrir otro grupo o ruido.

El algoritmo original DBSCAN, retomado de [9], se compone de tres elementos: algoritmo principal 2, la función *ExpandCluster* 3 y el método *regionQuery* 4.

Algorithm 2: DBSCAN

```
setOfPoints, eps, minPts n ← size(setOfPoints) clusterId ← UNCLASSIFIED
visited[n] ← FALSE clusterAssignment ← nelementos, inicializadosa0 for i ← 1 TO n
do-_
visited[i] ← TRUE continue visited[i] = TRUE
_neighbors = regionQuery(setOfPoints, i, eps) if size(neighbors) < minPts then
_clusterAssignment[i] ← NOISE else
clusterId ← clusterId + 1 ExpandCluster(setOfPoints, i, neighbors, clusterId, eps, minPts,
            visited, clusterAssignment)
return clusterAssignment
```

Algorithm 3: ExpandCluster

```
setOfPoints, point, neighbors, clusterId, eps, minPts, visited,
            clusterAssignment clusterAssignment[point] ← clusterId i ← 1 while
i ← size(neighbors) do-_
_p ← neighbors[i] if !visited[p] then
visited[p] ← TRUE newNeighbors ← regionQuery(setOfPoionts, p, eps) if
_size(newNeighbors) ⩾ minPts then
_append(neighbors, newNeighbors) if clusterAssignment[p] == 0 then
clusterAssignment[p] ← clusterId i = i + 1
```

Algorithm 4: RegionQuery

```
setOfPoints, eps neighbors[] n ← size(setOfPoints) for j ← 1 TO n do-_
lat1 ← setOfPoints[i][1] lon1 ← setOfPoints[i][2] lat2 ← setOfPoints[j][1]
_lon2 ← setOfPoints[j][2] if distanceHaversine(lat1, lon1, lat2, lon2) ⩽ eps then
neighbors.push(j)
return neighbors
```

Complejidad

DBSCAN visita cada punto de la base de datos, posiblemente varias veces (e.g., como candidatos a diferentes clusters). Para la práctica, sin embargo, la complejidad temporal es mayormente gobernada por el número de llamados a regionQuery. DBSCAN ejecuta exactamente una consulta por cada punto, y si se utiliza una estructura de índice que ejecute la consulta a los vecinos (neighborhood query) en $O(\log n)$, la complejidad temporal total sería $O(n \log n)$. Sin usar la estructura de índice, la complejidad temporal sería $O(n^2)$. A menudo, la matriz de distancias de tamaño $(n^2 - n)/2$ se construye para evitar recalcular las distancias. Esto necesita $O(n^2)$ de memoria, mientras que con una implementación que no se base en matrices solo se necesita $O(n)$ de memoria [9, 10].

Agrupamiento Espacial Jerárquico Basado en Densidad de Aplicaciones con Ruido

Conocido como HDBSCAN, es un algoritmo de clustering avanzado que se basa en densidad y jerarquía, es una extensión del algoritmo DBSCAN, debido a que éste puede tener dificultades para capturar con éxito grupos con diferentes densidades, mientras que HDBSCAN explora todas las escalas de densidad posibles mediante

la construcción de una representación alternativa del problema de agrupación [11]. Se utiliza principalmente para identificar patrones y estructuras en grandes conjuntos de datos, especialmente cuando la forma y la densidad de los clusters no son uniformes [11, 12].

Características principales y funcionamiento

- Agrupa puntos de datos que están densamente conectados, al igual que DBSCAN, lo que significa que hay un número suficiente de puntos en su vecindad.
- Crea una jerarquía de clusters a diferencia de DBSCAN, esta jerarquía permite al algoritmo identificar clusters a diferentes niveles de granularidad.
- Etiqueta los puntos que no pertenecen a ningún cluster como dato no etiquetado, lo que ayuda a identificar outliers (valores atípicos) en los datos.
- Es más robusto en comparación con DBSCAN en términos de la selección de parámetros. En lugar de depender de un solo valor de densidad mínima, construye una jerarquía y selecciona clusters basados en la estabilidad de estos.

Funcionamiento del Algoritmo

1. Transforma el espacio de datos en función de su densidad. Esto ayuda a identificar áreas con mayores concentraciones de puntos de datos.
2. Se crea un gráfico que conecta puntos de datos dentro de un cierto umbral de distancia, esto forma la base para identificar grupos.
3. Se identifican los componentes conectados en la gráfico, formando una estructura jerárquica de clusters potenciales.
4. Según el parámetro de tamaño mínimo del cluster, los clusters más pequeños se fusionan en otros más grandes y estables.
5. La solución de agrupación final se extrae de la jerarquía condensada, lo que garantiza que los grupos identificados sean resistentes a las variaciones de parámetros.

La complejidad algorítmica total de HDBSCAN es de aproximadamente O(N log N), lo que lo hace más eficiente que otros algoritmos jerárquicos que pueden tener complejidades cuadráticas o peores, especialmente para conjuntos de datos grandes [11, 12].

Ordenación de Puntos para Identificar la Estructura de Clusters

El algoritmo OPTICS es una técnica avanzada de clustering basada en densidad que se utiliza para descubrir estructuras de clusters en conjuntos de datos complejos. Es una extensión del algoritmo DBSCAN y aborda algunas de sus limitaciones, particularmente en la identificación de clusters con densidades variables [13]. Se utiliza en una amplia gama de aplicaciones similares a DBSCAN, incluyendo:

- Análisis de datos geoespaciales: Identificación de regiones con alta densidad de eventos geográficos.
- Bioinformática: Agrupación de genes o proteínas con características similares.
- Análisis de mercado: Segmentación de clientes basado en comportamientos de compra.
- Procesamiento de imágenes: Agrupación de características de imágenes.

Características principales y funcionamiento

- Agrupa puntos de datos que están densamente conectados al igual que DBSCAN.
- No produce directamente una asignación de clusters, en su lugar genera una ordenación lineal de puntos de datos que representa la estructura de densidad subyacente.
- La salida principal es un gráfico de "reachability", que visualiza las distancias de accesibilidad de los puntos en el orden procesado, ayudando a identificar clusters de diferentes densidades.
- Maneja mejor los clusters con diferentes densidades en comparación con DBSCAN.

Los parámetros principales que utiliza OPTICS son:

- $min\_samples$: Número mínimo de puntos requeridos para que un punto sea considerado núcleo.
- epsilon (ϵ): Radio máximo de búsqueda de vecinos. En OPTICS, este parámetro es menos crítico ya que el algoritmo explora un rango de valores de ϵ.

Funcionamiento del Algoritmo

1. Se calcula para cada punto una "distancia de accesibilidad" que indica la distancia mínima necesaria para alcanzar este punto desde un punto núcleo.
2. Se ordenan los puntos en función de sus distancias de accesibilidad, generando una representación ordenada de la estructura de densidad del conjunto de datos.
3. A partir de la ordenación de distancias de accesibilidad, se pueden extraer clusters utilizando diferentes técnicas de post-procesamiento, como la identificación de valles en el gráfico de reachability.

La complejidad algorítmica total de OPTICS, en el peor de los casos, es de $O(N^2)$, cuando el algoritmo debe considerar todos los puntos como vecinos en un conjunto de datos denso [13, 14].

3. Comparativa de los algoritmos de clustering

En el Cuadro 1 se resumen las principales ventajas y desventajas de cada uno de los algoritmos revisados en este trabajo.

Tabla 1: Ventajas y desventajas de k-means, DBSCAN, HDBSCAN y OPTICS

Algoritmo	**Ventajas**	**Desventajas**
k-means	Es fácil de comprender y de implementar. Es relativamente rápido y eficiente en términos de tiempo de ejecución. Converge rápidamente a una solución, lo que lo hace adecuado para aplicaciones en tiempo real. Puede manejar grandes volúmenes de datos y puede ser fácilmente paralelizado.	Requiere que el número de clusters k sea especificado previamente. Los resultados del algoritmo pueden variar significativamente dependiendo de la inicialización de los centroides. Asume que los clusters son esféricos y de tamaño similar, lo cual puede no ser apropiado para datos con clusters de formas arbitrarias o densidades variadas.
DBSCAN	No requiere que se especifique el número de clusters en los datos. Posee una noción de ruido y es resistente a los valores atípicos. Requiere solo dos parámetros y es prácticamente insensible al orden de los puntos en los datos usados. Los parámetros `minPt` y *eps* pueden ser establecidos por un experto en el dominio, si los datos se comprenden bien.	Los puntos fronterizos a los que se puede llegar desde más de un grupo pueden formar parte de cualquiera de ellos (dependiendo del orden en que se procesen los datos). La calidad depende de la métrica de distancia utilizada. No puede agrupar bien conjuntos de datos con grandes diferencias en densidades.

Algoritmo	Ventajas	Desventajas
HDBSCAN	Puede identificar clusters de formas arbitrarias. Es eficaz en la identificación de puntos de ruido, lo cual mejora la calidad del clustering al excluir datos ruidosos que podrían distorsionar los resultados. No se necesita especificar el número de clusters. Produce una jerarquía de clusters, proporcionando una visión más detallada de la estructura de los datos a diferentes niveles de granularidad.	Requiere la selección adecuada de parámetros como $min\_samples$ (minPts) y $min\_cluster\_size$, lo que puede ser complejo sin conocimiento previo de los datos. Aunque es escalable, para conjuntos de datos extremadamente grandes puede requerir un tiempo de computación significativo y recursos de memoria elevados. El rendimiento y los resultados del algoritmo dependen en gran medida de la métrica de distancia utilizada.
OPTICS	No requiere que se especifique el número de grupos en los datos. Puede manejar conjuntos de datos con clusters que tienen diferentes densidades. El gráfico de reachability proporciona una visualización intuitiva de la estructura de densidad, facilitando la identificación de clusters y outliers.	Es más complejo computacionalmente en comparación con DBSCAN debido a la necesidad de calcular y ordenar las distancias de accesibilidad. La salida requiere pasos adicionales de post-procesamiento para la extracción de clusters, lo que puede ser menos intuitivo.

Métricas para la determinación de la distancia

Para los algoritmos de clustering basados en densidad, el definir una métrica de distancia es un factor determinante para la cantidad de clusters que se pueden formar [6], en DBSCAN es más notorio este factor debido al parámetro eps, por otro lado HDBSCAN y OPTICS tienen la particularidad de establecer dicho parámetro mediante cálculos [11, 13]. La distancia puede ser dada bajo una métrica en particular dependiendo del contexto de los datos, es decir puede ser usada una distancia euclidiana, Haversine, entre otras. Las principales métricas de distancia utilizadas son:

Distancia Euclidiana

Es un número positivo que indica la separación de dos puntos en un espacio donde se cumplen los axiomas y teoremas de la geometría de Euclides [6]. La distancia entre dos puntos $p = (x_1, y_1)$ y $q = (x_2, y_2)$ de un espacio euclidiano es la longitud del vector pq perteneciente a la única recta que pasa por dichos puntos. Para un espacio de dos dimensiones, la fórmula de la distancia euclidiana se muestra en la Ecuación 1.

$$d(p, q) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

En un espacio n-dimensional, la distancia Euclideana entre los puntos $P = (p_1, p_2, \ldots, p_n)$ y $Q = (q_1, q_2, \ldots, q_n)$ se puede definir en la Ecuación 2.

$$d_E(P, Q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)} \tag{2}$$

Distancia Manhattan

También conocida como distancia de la ciudad o distancia L_1, es una métrica de distancia utilizada en el análisis de datos y aprendizaje automático. Se llama así porque representa la distancia que recorrería un taxi en una cuadrícula de calles rectangulares, similar al sistema de calles de Manhattan [6]. La distancia Manhattan entre dos puntos $P_1 = (x_1, y_1)$ y $P_2 = (x_2, y_2)$ en un espacio bidimensional se calcula como:

$$Distancia\ Manhattan = |x_1 - x_2| + |y_1 - y_2| \tag{3}$$

Si se trata de un espacio de más dimensiones entre dos puntos $p = (p_1, p_2, \ldots, p_n)$ y $q = (q_1, q_2, \ldots, q_n)$ en un espacio de n-dimensional.

$$Distancia\ Manhattan = \sum_{i=1}^{n} |p_i - q_i| \tag{4}$$

Fórmula Haversine

Es una fórmula utilizada para calcular la distancia entre dos puntos de una esfera dadas sus coordenadas de longitud y latitud. En términos más sencillos, calcula la distancia más corta entre dos puntos de la superficie de un objeto esférico, como la Tierra. La fórmula de Haversine, ver Ecuación 5, se utiliza para calcular rutas de vuelo, distancias en mapas y navegación marítima [15]. También se utiliza en meteorología para calcular las trayectorias de las tormentas, para posicionar equipos de telecomunicaciones y para trazar la trayectoria de los satélites en el cielo [16, 17].

$$d = 2 \cdot R \cdot \arcsin \sqrt{\sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)} \tag{5}$$

donde:

- φ_1, φ_2 son las latitudes del punto 1 y 2 respectivamente.

- λ_1, λ_2 son las longitudes del punto 1 y 2 respectivamente.
- R es el radio de la esfera (en este caso el radio de la Tierra, que equivale a $6,371$ km).

La fórmula Haversine no puede ser usada por más de dos dimensiones, por lo que en el contexto de clustering no permite realizar un análisis de diversas características numéricas a la vez, en esas situaciones es ideal utilizar distancia Euclideana o Manhattan.

Métricas de evaluación para algoritmos de clustering

Para determinar qué algoritmos de clustering son mejores para un conjunto de datos, es necesario el uso de métricas de evaluación. Algunas de las métricas más comunes son:

Silhouette Score

Mide cuán similares son los puntos dentro de un cluster en comparación con los puntos de otros clusters. Su valor oscila entre -1 y 1, donde 1 indica clusters bien definidos [18]. El proceso del cálculo de Silhouette Score es el siguiente:

1. Se calcula la distancia promedio $a(i)$ entre el punto i y todos los demás puntos en el mismo clúster mediante la Ecuacion 6, siendo C_i el conjunto de puntos y $\|x_i - x_j\|$ la distancia entre los puntos i y j.

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, j \neq i} \|x_i - x_j\| \tag{6}$$

2. Se calcula de la distancia promedio $b(i)$ entre el punto i y todos los puntos en el cluster más cercano (que no sea el propio cluster de i) mediante la Ecuación 7, donde C_k es un cluster diferente al cluster C_i.

$$b(i) = \min_{k \neq C_i} \frac{1}{|C_k|} \sum_{j \in C_k} \|x_i - x_j\| \tag{7}$$

3. Se realiza el cálculo del índice de Silhouette para el punto i con la Ecuación 8.

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{8}$$

4. Finalmente se calcula el Silhouette score para el conjunto completo con la Ecuación 9, siendo esta el promedio del índice de Silhouette de todos los puntos, donde N es el número total de puntos en el conjunto de datos.

$$S = \frac{1}{N} \sum_{i=1}^{N} s(i) \tag{9}$$

Índice Davies-Bouldin

Evalúa la dispersión de los clusters y la distancia entre ellos. Un valor más bajo indica clusters mejor formados [19]. El procedimiento para obtener el índice Davies-Bouldin es el siguiente:

1. Se obtienen la dispersión de cada cluster utilizando la Ecuación 10, siendo C_i es el conjunto de puntos, c_i es el centroide del cluster, $\|x - c_i\|$ la distancia entre el punto x y el centroide, y $|C_i|$ es el número de puntos en el cluster todo esto para el punto i.

$$S_i = \frac{1}{|C_i|} \sum_{x \in C_i} \|x - c_i\| \tag{10}$$

2. El cálculo de la separación entre clusters se consigue aplicando la Ecuación 11, donde c_i y c_j son los centroides de los clusters i y j respectivamente y $\|c_i - c_j\|$ siendo la distancia entre los centroides de los clusters.

$$M_{ij} = \|c_i - c_j\| \tag{11}$$

3. La medida de Davies-Bouldin para cada par de clusters se obtienen con la Ecuación 12.

$$R_{ij} = \frac{S_i + S_j}{M_{ij}} \tag{12}$$

4. Se calcula el máximo de R_{ij} para cluster i sobre todos los clusters $j \neq i$ utilizando la Ecuación 13.

$$D_i = \max_{j \neq i} R_{ij} \tag{13}$$

5. Finalmente se obtienen el índice de Davies-Bouldin con la Ecuación 14, donde N es el número total de clusters.

$$DBI = \frac{1}{N} \sum_{i=1}^{N} R_i \tag{14}$$

Índice Calinski-Harabasz

También conocido como el criterio de varianza de la relación, evalúa la dispersión entre clusters y la cohesión dentro de los clusters. Un valor más alto indica clústeres mejor formados [20].

1. Se obtienen la matriz de dispersión entre clústeres (B_k) mediante la Ecuación 15, siendo k el número de clusters, C_i es el conjunto de puntos en el cluster i, $|C_i|$ es el número de puntos en el cluster i, c_i es el centroide del cluster iy c es el centroide global de todos los datos.

$$B_k = \sum_{i=1}^{k} |C_i|(c_i - c)(c_i - c)^T \tag{15}$$

2. Se calcula la matriz de dispersión dentro de clusters W_k con la Ecuación 16

$$W_k = \sum_{i=1}^{k} \sum_{x \in C_i} (x - c_i)(x - c_i)^T \tag{16}$$

3. El cálculo del índice de Calinski-Harabasz (CH), se obtienen finalmente aplicando la Ecuación 17, donde $trace(B_k)$ es la traza de la matriz de dispersión entre clusters, $trace(W_k)$ es la traza de la matriz de dispersión dentro de clusters y n es el número total de puntos en el conjunto de datos.

$$CH = \frac{trace(B_k)/(k-1)}{trace(W_k)/(n-k)} \tag{17}$$

4. Procesamiento de los datos

Los datos utilizados en los algoritmos de clustering, se conforman de la matrícula de alumnos, compuesto por 96,604 registros, cada registro contienen la siguiente información: Nombre del espacio académico (EA), calle, población, estado, municipio, código postal (CP). Después de una limpieza a los datos, el conteo de registros pasó a ser de 96,113. Para poder realizar el proceso de clustering con coordenadas geográficas, se tomó el CP para realizar una identificación aproximada de la zona donde cada alumno radica, en términos de longitud y latitud; para ello se realizó el siguiente proceso:

1. Identificar el rango de códigos postales para establecer la zona de servicio de transporte, donde se requieren identificar posibles paradas de autobús, para este trabajo se tomaron los códigos postales en el rango de 50000 a 52077 de México.

2. Cada código postal, en el rango descrito, debe ser mapeado a una coordenada geográfica en formato (longitud, latitud), por lo cual se utilizó la API (Application Programming Interface) que proporciona el servicio de OpenStreetMap [21], el cual incluye un método de consulta sobre la información de códigos postales en un determinado país, su uso para el presente trabajo es la conversión de códigos postales a coordenadas geográficas, mediante un solicitud *GET*. Con este proceso, se construyó un archivo con la asignación CP a Longitud, Latitud.

3. Al realizar un filtrado aplicado a los datos de la matrícula de alumnos y obtener aquellos registros con el rango de CP, determinado anteriormente, se obtuvo un total de 41,298 registros para realizar el proceso de clustering.

4. Debido a que la matrícula de alumnos contenía registros con CP iguales, se realizó un proceso para obtener los CP "únicos", es decir, CP donde radican los alumnos sin repetición de los mismos, este proceso permite eficientar los algoritmos de clustering reduciendo la cantidad de datos a analizar para cada algoritmo.

Los estructura de los datos obtenidos después del procesamiento se muestran en el Cuadro 2, estos datos son los utilizados para realizar el clustering con cada uno de los algoritmos.

Tabla 2: Muestra de los datos finales para el proceso de clustering

CP	Cantidad de alumnos	Longitud	Latitud
50000	93	-99.657693	19.291105
50010	1432	-99.646928	19.315436
50016	156	-99.605140	19.314034
50017	115	-99.614347	19.308198
...	...	...	...

5. Implementación de los algoritmos de clustering

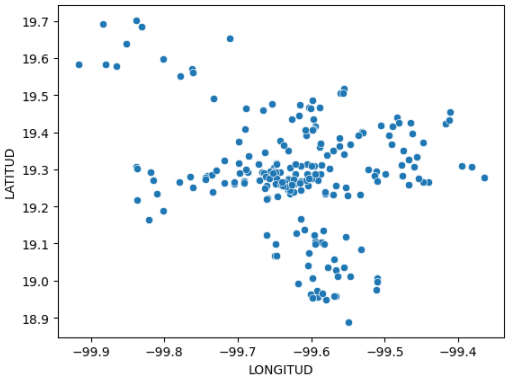
Dentro de los módulos que contienen la biblioteca *scikit-learn* se encuentra la definición de algunos algoritmos de agrupamiento para el lenguaje python, en los que se encuentran: *k-means, K-medoids, DBSCAN, HDBSCAN, OPTICS*, entre otros; además de métodos para evaluarlos [22].

Para el uso de los algoritmos de clustering en el lenguaje python con scikit-learn, se deben establecer algunos parámetros de configuración; en el caso particular de k-means se debe especificar la cantidad de clusters a formar mediante el parámetro $n\_clusters$. Por otro lado, en DBSCAN se deben especificar dos parámetros necesarios para el funcionamiento del algoritmo, estos son eps y $min\_samples$, el primero para establecer el radio de distancia, probado con valores de 1 km a 3 km, y el segundo para definir la cantidad de puntos mínimos, dejándolo en un valor fijo de 2 (dos puntos mínimos para se considerado cluster). En cuanto a HDBSCAN y OPTICS soló se estableció $min\_samples = 2$. Para los tres algoritmos de clustering basados en densidad se estableció la métrica Haversine.

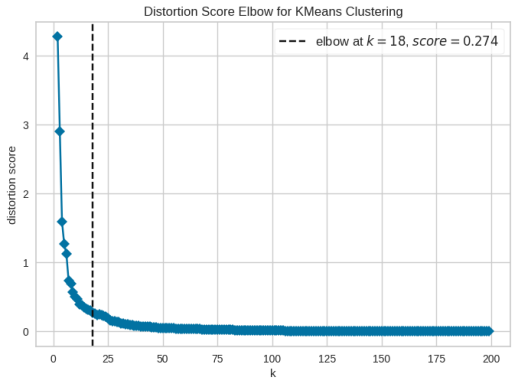
6. Resultados

Con el preprocesamiento descrito previamente, la gráfica de los datos de los códigos postales únicos con sus respectivas coordenadas se muestran en la Figura 2a. Mediante el uso de un gráfico de codo, ver Figura 2b, se pudo determinar el valor adecuado de clusters para configurar k-means, con una valor $k = 18$. Los resultados del agrupamiento por k-means se puede ver en la Figura 2c. Tomando estos resultados podemos determinar un máximo de 18 posibles paradas de autobús para el ámbito de transporte.

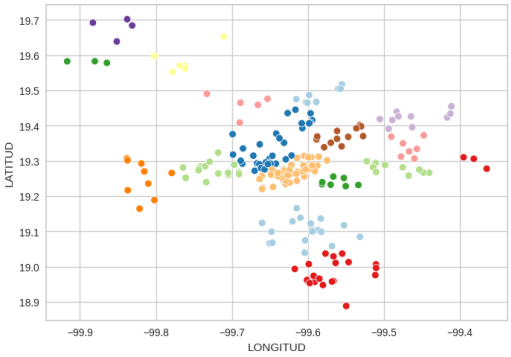
Los resultados obtenidos con la implementación de DBSCAN se muestran en la Tabla 3, donde se hizo variaciones al parámetro eps, con valores de 1,0 km, 1,5 km, 2,0 km, 2,5 km y 3,0 km; mientras que el parámetro $minPTS$ se mantuvo constante con un valor de 2, esto con el objetivo de contabilizar la cantidad de grupos que se generan, así como la cantidad de datos no agrupados, en la Figura 3 se puede apreciar los cambios en el conteo de clusters y datos no agrupados. Para la comparativa entre los otros algoritmos de densidad se tomó DBSCAN con los resultados obtenidos de $eps = 1{,}5$ km.



(a) Códigos postales en un plano cartesiano



(b) Gráfica de codo que muestra el mejor valor de k = 18



(c) 18 clusters formados con k-means

Figura 2: Proceso de clustering con k-means

Tabla 3: Resultados de las pruebas de agrupamiento mediante DBSCAN

Resultados del clustering con DBSCAN			
Eps	**minPts**	**Cantidad de clusters**	**Datos no etiquetados**
1	2	29	126
1.5	2	28	84
2	2	32	54
2.5	2	26	30
3	2	21	22

El proceso de clustering con DBSCAN, HDBSCAN y OPTICS se muestran en las Figuras 4a, 4b y 4c respectivamente, se puede observar que OPTICS tiende a formar una mayor cantidad de grupos en comparación de los otros algoritmos basados en densidad, lo cual es un resultado contrario a lo que determina k-means. Para el área de transporte de pasajeros, es un punto importante ya que se tienen mayor cantidad de paradas, pero adecuadas ya que no son de un área muy extensa y por ende los pasajeros pueden trasladarse de sus residencias al punto de parada determinado.

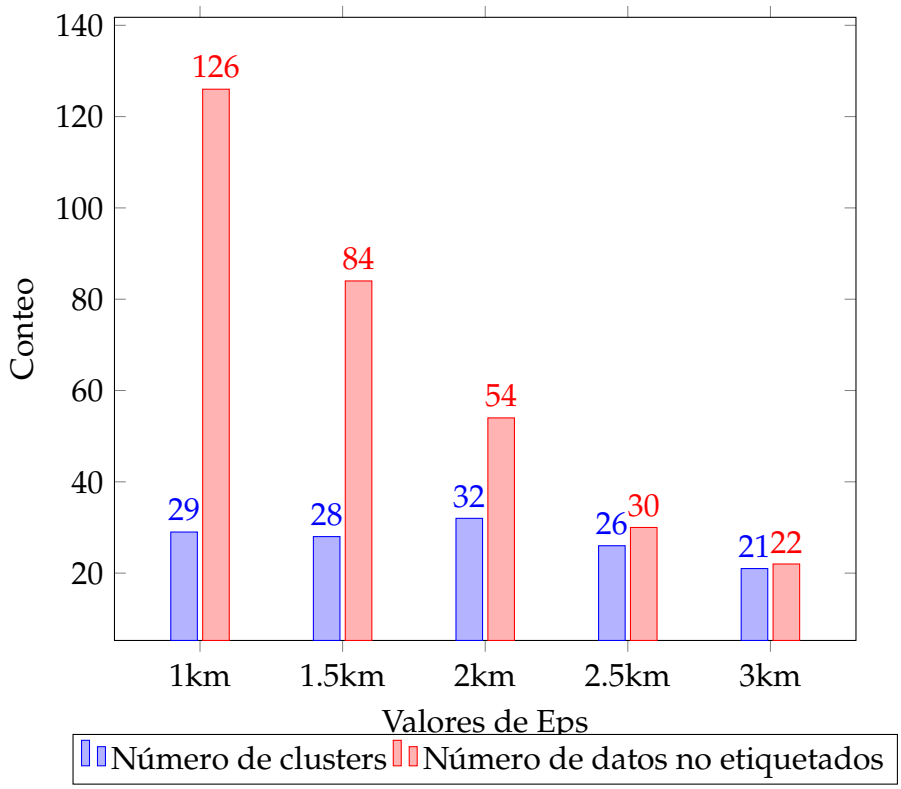
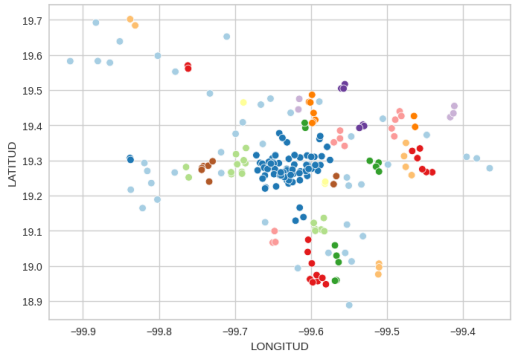
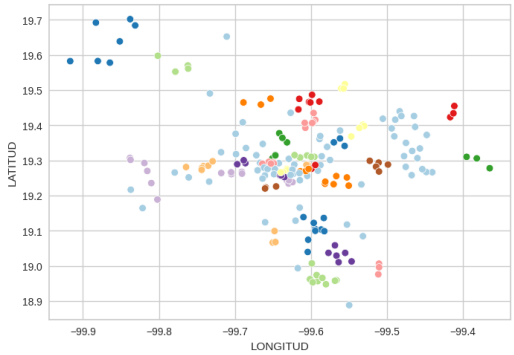


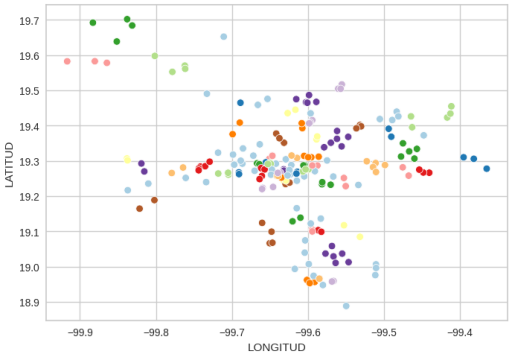
Figura 3: Gráfico de barras de los valores obtenidos al variar Eps



(a) 28 clusters con DBSCAN



(b) 33 clusters con HDBSCAN



(c) 64 clusters con OPTICS

Figura 4: Comparativa de proceso de clustering con los algoritmos basados en densidad

Con el uso de las métricas Silhouette score, índice Davies-Bouldin y el índice Calinski-Harabasz para realizar una comparativa de los cuatro algoritmos de clus-

tering se obtuvieron los siguientes resultados, mostrados en el Cuadro 4, dónde se puede determinar que el mejor algoritmo para establecer clusters, bien formados e identificables, es k-means; pero en un contexto de determinación de paradas vehiculares, no es adecuado por el tamaño y lejanía de los elementos en la frontera del cluster hasta su centroide. Comparando los algoritmos de clustering basados en densidad, el que tienen una mejor valoración para determinar los clusters, es HDBSCAN teniendo mejores valores en cuanto a la calidad de los clusters formados.

Tabla 4: Resultados de las métricas para valorar la calidad de los clusters

Algoritmo	**Silhouette score**	**Índice Davies-Bouldin**	**Índice Calinski-Harabasz**
K-means	0.3894	0.7579	255.6820
DBSCAN	-0.2044	3.6188	2.0485
HDBSCAN	0.2091	1.7195	36.2838
OPTICS	0.3063	2.5802	14.1846

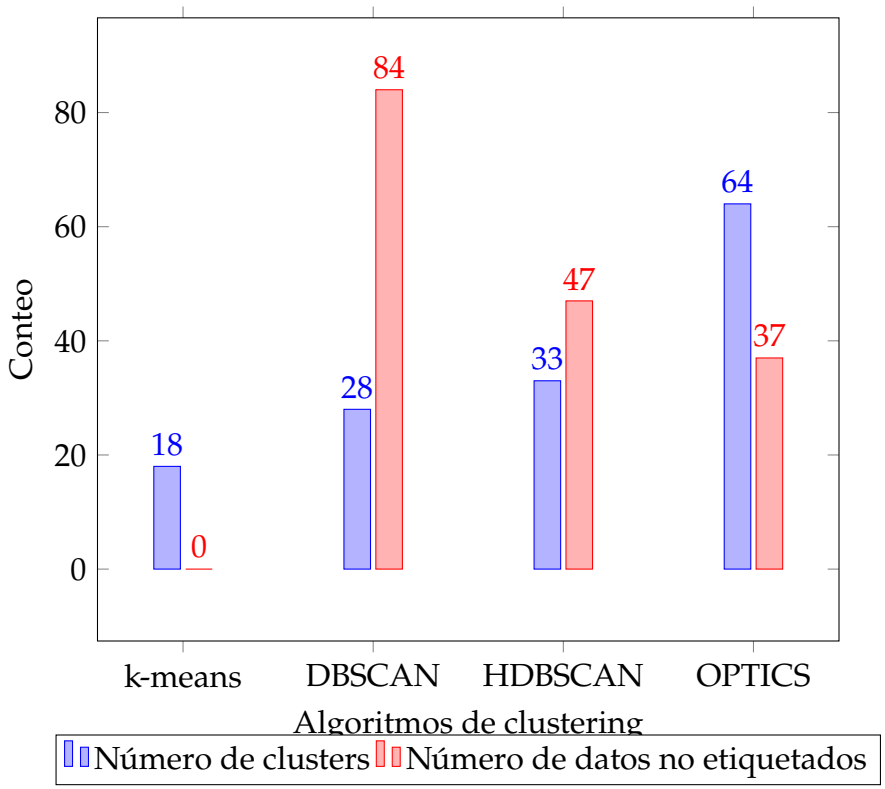


Figura 5: Conteo de clusters y datos no etiquetados por los algoritmos de clustering

En la gráfica de la Figura 5, se muestra la cantidad de clusters y de datos no agrupadas determinandos por cada algoritmo de clustering, de lo cual se puede obtener el total de paradas de autobús de acuerdo a la cantidad de clusters que cada algoritmo identificó.

7. Conclusiones

Los algoritmos de clustering basados en densidad para identificar zonas adecuadas y establecer paradas de autobús utilizando códigos postales son factibles en comparación a algoritmos más recurridos de clustering como k-means. Los algoritmos DBSCAN, HDBSCAN y OPTICS son similares en cuanto a su procesamiento, pero los últimos dos solo requieren un parámetro para su funcionamiento y tienen la característica de calcular el valor adecuado *eps*; esto se demuestra al comparar la cantidad de clusters que pueden identificar, siendo HDBSCAN y OPTICS capaces de determinan una mayor cantidad de clusters de los que puede determinar DBSCAN, sin embargo OPTICS crea una cantidad de clusters mucho mayor pero con problemáticas, en el sentido del contexto de transporte, debido a que los clusters determinados, al ser considerados paradas de autobús, son demasiados y la calidad de los clusters no fue bien valorada por las métricas usadas. El mejor algoritmo basado en densidad es HDBSCAN que permite determinar una cantidad adecuada de clusters y los clusters formados obtienen una buena valoración de calidad en comparación a los otros, consiguiendo así determinar 33 puntos de parada de autobús del conjunto de datos trabajado.

Agradecimientos

Se reconoce el apoyo de la dirección de transporte y a la dirección de control escolar de la Universidad Autónoma del Estado de México, así como los encargados de cada una de las áreas mencionadas por proporcionar los datos necesarios para la realización de las pruebas con el algoritmo de clustering, así como las pruebas necesarias para este trabajo.

Bibliografía

[1] Datascientest. Machine Learning y Clustering: el algoritmo DBSCAN. *Online*, 2023.

[2] A. Géron. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 2019.

[3] Departamento de Matemáticas e Informática de la Universidad de Barcelona. El poder de los datos: Del big data al aprendizaje profundo. *National Geographic*, 2017.

[4] R. Liu, N. Wang. Data-Driven Bus Route Optimization Algorithm Under Sudden Interruption of Public Transport. *IEEE Access*, 10:5250-5263, 2022.

[5] M. Sciortino, R. Lewis, J. Thompson. A School Bus Routing Heuristic Algorithm Allowing Heterogeneous Fleets and Bus Stop Selection. *SN Computer Science*, 4(1):74, 2022.

[6] P. Grabusts. The Choice of Metrics for Clustering Algorithms. *Environment. Technology. Resources. Proceedings of the International Scientific and Practical Conference*, 2:70, 2015.

[7] A. Nagpal, A. Jatain, D. Gaur. Review based on data clustering algorithms. *2013 IEEE CONFERENCE ON INFORMATION AND COMMUNICATION TECHNOLOGIES*, 298–303, 2013.

[8] D. Arthur, S. Vassilvitskii. How slow is the k -means method?. *Proceedings of the twenty-second annual symposium on Computational geometry*, 144–153, 2006.

[9] M. Ester, H. P. Hriegel, J. Sander, X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatian Databases with Npise. *in Proc. AAAI*, 23(1):226–231, 1996.

[10] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, X. Xu. DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN. *ACM Transactions on Database Systems*, 42(3):1–21, 2017.

[11] R. J. G. B. Campello, D. Moulavi, A. Zimek, J. Sander. Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection. *ACM Transactions on Knowledge Discovery from Data*, 10(1):1–51, 2015.

[12] scikit-learn developers (BSD License). HDBSCAN — scikit-learn.org. *Online*, 2023.

[13] M. Ankerst, M. M. Breunig, H. P. Kriegel. OPTICS: Ordering Points To Identify the Clustering Structure. *ACM SIGMOD Record*, 28(2):49–60, 1999.

[14] scikit-learn developers. OPTICS — scikit-learn.org. *Online*, 2023.

[15] P. Dauni, M. D. Firdaus, R. Asfariani, M. I. N. Saputra, A. A. Hidayat, W. B. Zulfikar. Implementation of Haversine formula for school location tracking. *Journal of Physics: Conference Series*, 1402(7):077028, 2019.

[16] W. Zhang. Applications of the Haversine formula in the evaluation of the great-circle distance between two points on the earth's surface. *Journal of Applied Mathematics and Physics*, 3(7):812–819, 2009.

[17] T. S. Rappaport. Wireless Communications: Principles and Practice. *IEEE Press*, 1996.

[18] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[19] D. L. Davies, D. W. Bouldin. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[20] T. Calinski, J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics - Theory and Methods*, 3(1):1–27, 1974.

[21] Openstreetmap.org. OpenStreetMap. *Online*, 2023.

[22] scikit-learn developers. scikit-learn developers. *Online*, 2023.

Direcciones de correo de los autores

Sánchez Carmona, M. A.: msanchezc048@alumno.uaemex.mx