Práctica DBSCAN

Agrupamiento Espacial Basado en Densidad

Machine Learning

Nombre completo: Fernando Gutierrez Delgado

Código: _____ **Fecha:** _____

Criterio de evaluación	Puntaje máximo	Puntaje obtenido
Resolución del ejercicio HackerRank (código)	5 pts	
Resolución de este cuadernillo (ejecución a mano)	15 pts	
Total	**20 pts**	

Instrucciones

En esta práctica ejecutarás **a mano** el algoritmo DBSCAN sobre un conjunto pequeño de datos. Los recuadros ☐ indican casilleros que debes completar. El objetivo es comprender cada fase del algoritmo: cálculo de distancias, determinación de vecindarios, clasificación de puntos y formación de clústeres.

1. Introducción al algoritmo DBSCAN

DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) es un algoritmo de agrupamiento basado en densidad propuesto por Ester, Kriegel, Sander y Xu en 1996. A diferencia de algoritmos como k-means, DBSCAN:

- No requiere especificar el número de clústeres de antemano.
- Puede descubrir clústeres de **forma arbitraria**.
- Identifica automáticamente los **puntos de ruido** (outliers).

Referencia al paper original

Ester, M., Kriegel, H.-P., Sander, J. y Xu, X. (1996). *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), pp. 226–231.

Enlace: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

1.1. Definiciones fundamentales

Dado un conjunto de datos D con n puntos, y dos parámetros $\varepsilon > 0$ (radio) y $MinPts$ (mínimo de puntos):

ε-vecindario. El conjunto de puntos a distancia menor o igual a ε de un punto p:

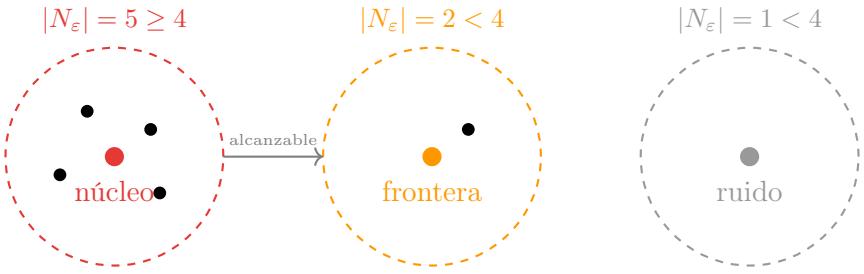
$$N_\varepsilon(p) = \{q \in D \mid dist(p, q) \leq \varepsilon\}$$

Distancia euclidiana.

$$dist(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

Punto núcleo (*core point*): p tal que $|N_\varepsilon(p)| \geq MinPts$.
Punto frontera (*border point*): p tal que $|N_\varepsilon(p)| < MinPts$, pero $p \in N_\varepsilon(q)$ para algún punto núcleo q.
Punto de ruido (*noise*): p que no es ni núcleo ni frontera.

La siguiente figura ilustra estos conceptos con $MinPts = 4$:



2. Pseudocódigo del algoritmo DBSCAN

Algoritmo 1: DBSCAN

Entrada: D: conjunto de puntos, ε: radio, $MinPts$: mínimo de puntos
Salida: Etiquetas de clúster para cada punto (o -1 si es ruido)

```
cluster_id ← 0;
para cada punto P_i en D (en orden i = 0, 1, ..., n − 1) hacer
    si P_i ya fue clasificado entonces
        continuar;
    vecinos ← regionQuery(P_i, ε);
    si |vecinos| < MinPts entonces
        Marcar P_i como RUIDO (−1);
    sino
        cluster_id ← cluster_id + 1;
        Asignar cluster_id a P_i y a todos los puntos en vecinos;
        seeds ← vecinos \ {P_i};
        para cada punto Q en seeds hacer
            resultado ← regionQuery(Q, ε);
            si |resultado| ≥ MinPts entonces
                para cada punto R en resultado hacer
                    si R es SIN_CLASIFICAR entonces
                        Agregar R a seeds;
                        Asignar cluster_id a R;
                    si R es RUIDO entonces
                        Asignar cluster_id a R;
```

Algoritmo 2: regionQuery

Entrada: P: punto de consulta, ε: radio
Salida: Conjunto de puntos dentro del ε-vecindario de P

```
vecinos ← ∅;
para cada punto Q en D hacer
    si dist(P, Q) ≤ ε entonces
        vecinos ← vecinos ∪ {Q};
retornar vecinos;
```

3. Problema: Detección de zonas sísmicas

Enunciado del problema (HackerRank)

Eres un científico de datos en un observatorio sismológico. Tu equipo ha recolectado las coordenadas (x, y) de epicentros de eventos sísmicos en una región. Para identificar **zonas sísmicas activas**, necesitas agrupar los eventos que están espacialmente concentrados, y al mismo tiempo detectar **eventos aislados** (ruido) que no pertenecen a ninguna zona.

Tu tarea: Implementa el algoritmo DBSCAN **desde cero**. Dado un conjunto de n puntos en 2D y los parámetros ε y $MinPts$, asigna cada punto a un clúster o márcalo como ruido.

Formato de entrada:

- Primera línea: n (entero), ε (decimal), $MinPts$ (entero).
- Siguientes n líneas: coordenadas x_i, y_i de cada punto.

Formato de salida:

- n líneas, cada una con el ID del clúster (empezando desde 1) o -1 si es ruido.

Restricciones: $1 \leq n \leq 10000$, $0 < \varepsilon \leq 100$, $1 \leq MinPts \leq n$.

4. Caso de prueba: datos de entrada

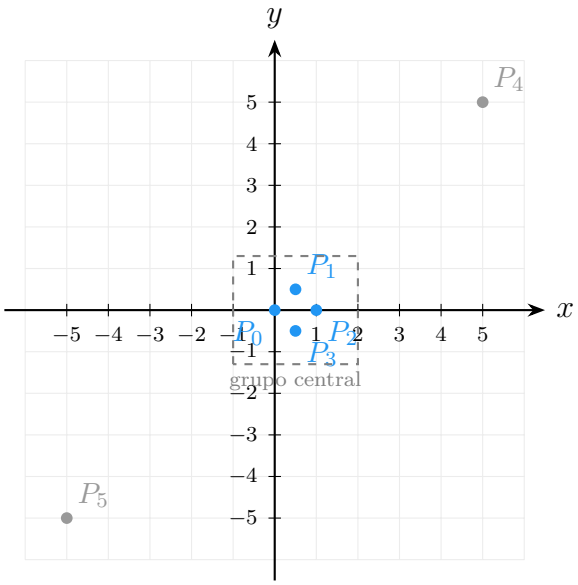
El siguiente caso de prueba corresponde al archivo `input_1.txt`:

```
6 1.5 3
0.0 0.0
0.5 0.5
1.0 0.0
0.5 -0.5
5.0 5.0
-5.0 -5.0
```

Parámetros: $n = 6$, $\varepsilon = 1.5$, $MinPts = 3$.

Punto	x	y
P_0	0.0	0.0
P_1	0.5	0.5
P_2	1.0	0.0
P_3	0.5	-0.5
P_4	5.0	5.0
P_5	-5.0	-5.0

4.1. Visualización de los puntos



5. Paso 1: Cálculo de distancias euclidianas

Calculamos las distancias entre todos los pares de puntos usando la fórmula:

$$dist(P_i, P_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Ejemplo resuelto:

$$dist(P_0, P_1) = \sqrt{(0.0 - 0.5)^2 + (0.0 - 0.5)^2} = \sqrt{0.25 + 0.25} = \sqrt{0.50} \approx 0.707$$

$$dist(P_0, P_2) = \sqrt{(0.0 - 1.0)^2 + (0.0 - 0.0)^2} = \sqrt{1.00 + 0.00} = \sqrt{1.00} = 1.000$$

5.1. Distancias entre los puntos del grupo central (P_0–P_3)

Completa las distancias faltantes. Recuerda que $\varepsilon = 1.5$.

	P_0	P_1	P_2	P_3
P_0	0	0.707	1.000	0.707
P_1		0	0.707	1
P_2			0	0.707
P_3				0

Pregunta: ¿Todas estas distancias son $\leq \varepsilon = 1.5$? Si

5.2. Distancias hacia los puntos aislados (P_4 y P_5)

Par	**Distancia**	$\leq \varepsilon$?
$dist(P_0, P_4) = \sqrt{25+25}$	≈ 7.071	No
$dist(P_0, P_5) = \sqrt{25+25}$	≈ 7.071	No
$dist(P_4, P_5) = \sqrt{100+100}$	≈ 14.142	No
$dist(P_1, P_4)$	6.3639	No
$dist(P_2, P_4)$	6.4031	No
$dist(P_3, P_5)$	7.1063	No

Conclusión: Los puntos P_4 y P_5 están **muy lejos** de todos los demás puntos. Ninguna distancia intergrupo es $\leq \varepsilon$.

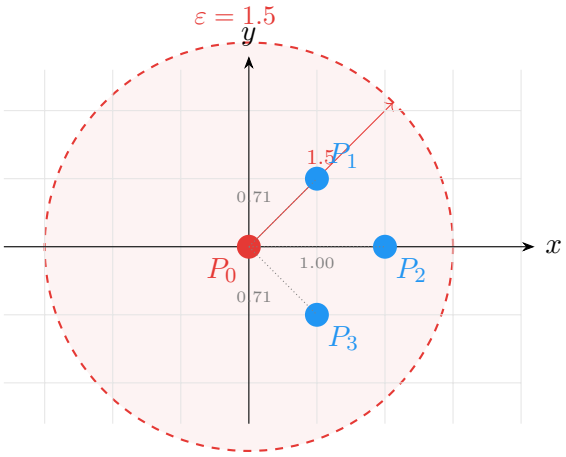
6. Paso 2: Determinación de ε-vecindarios

Para cada punto P_i, determinamos $N_\varepsilon(P_i)$ — el conjunto de **todos** los puntos (incluido él mismo) cuya distancia a P_i es $\leq \varepsilon = 1.5$.

Punto	$N_\varepsilon(P_i)$	$\lvert N_\varepsilon \rvert$	$\geq MinPts$?	**Tipo**
P_0	$\{P_0, P_1, P_2, P_3\}$	4	Sí	**Núcleo**
P_1	{P0, P1, P2, P3}	4	si	Nucleo
P_2	{P0, P1, P2, P3}	4	si	Nucleo
P_3	{P0, P1, P2, P3}	4	si	Nucleo
P_4	$\{P_4\}$	1	No	**Ruido**
P_5	{P5}	1	No	Ruido

6.1. Visualización de ε-vecindarios

El siguiente diagrama muestra el ε-vecindario de P_0 (círculo de radio $\varepsilon = 1.5$). Todos los puntos P_0–P_3 caen dentro del círculo.



7. Paso 3: Ejecución del algoritmo DBSCAN

Ahora ejecutamos el Algoritmo 1 paso a paso. Cada punto se procesa en orden ($i = 0, 1, \ldots, 5$).

Estado inicial:

	P_0	P_1	P_2	P_3	P_4	P_5
labels	0	0	0	0	0	0

(0 = sin clasificar, -1 = ruido, > 0 = ID del clúster)

$cluster\_id = 0$

7.1. Iteración $i = 0$ (punto P_0)

1. **Verificación:** $labels[0] = 0$ (sin clasificar) → **procesar**.
2. **regionQuery**$(P_0, 1.5)$: Calculamos la distancia de P_0 a todos los puntos.

	Distancia	≤ 1.5?	¿Vecino?
P_0	0.000	Sí	✓
P_1	0.707	Sí	✓
P_2	1.000	Sí	✓
P_3	0.707	Sí	✓
P_4	7.071	No	
P_5	7.071	No	

   Resultado: $vecinos = \{P_0, P_1, P_2, P_3\}$, $|vecinos| = 4$.
3. **Decisión:** $|vecinos| = 4 \geq MinPts = 3$ → **¡Nuevo clúster!**
4. $cluster\_id = 0 + 1 = \mathbf{1}$
5. Asignar $cluster\_id = 1$ a todos los vecinos: $labels[0] = labels[1] = labels[2] = labels[3] = 1$.
6. $seeds = \{P_1, P_2, P_3\}$ (vecinos sin P_0).

7. **Expansión de semillas:**

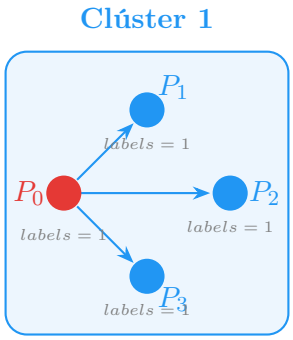
   - **Semilla P_1:** $regionQuery(P_1, 1.5) = \{P_0, P_1, P_2, P_3\}$, $|result| = 4 \geq 3$ (es núcleo). Pero todos ya tienen $labels = 1 \rightarrow$ **sin cambios**.
   - **Semilla P_2:** $regionQuery(P_2, 1.5) = \{P_0, P_1, P_2, P_3\}$, $|result| = 4 \geq 3$ (es núcleo). Todos ya clasificados $\rightarrow$ **sin cambios**.
   - **Semilla P_3:** $regionQuery(P_3, 1.5) = \{P_0, P_1, P_2, P_3\}$, $|result| = 4 \geq 3$ (es núcleo). Todos ya clasificados $\rightarrow$ **sin cambios**.

8. Semillas agotadas. **Clúster 1 completo**: $\{P_0, P_1, P_2, P_3\}$.

Estado de labels después de $i = 0$:

	P_0	P_1	P_2	P_3	P_4	P_5
labels	**1**	**1**	**1**	**1**	0	0

La siguiente figura muestra cómo se propaga el clúster 1 desde el punto P_0:



7.2. Iteración $i = 1$ (punto P_1)

1. **Verificación:** $labels[1] = 1 \neq 0$ (ya clasificado) $\rightarrow$ **saltar**.

No se realizan más acciones. El estado de *labels* no cambia.

7.3. Iteración $i = 2$ (punto P_2) — Completar

1. **Verificación:** $labels[2] =$ 1 $\rightarrow$ ya clasificado $\rightarrow$ saltar

7.4. Iteración $i = 3$ (punto P_3) — Completar

1. **Verificación:** $labels[3] =$ 1 $\rightarrow$ ya clasificado $\rightarrow$ saltar

7.5. Iteración $i = 4$ (punto P_4) — Completar

1. **Verificación:** $labels[4] = 0$ (sin clasificar) $\rightarrow$ **procesar**.
2. **regionQuery**$(P_4, 1.5)$: El punto $P_4 = (5.0, 5.0)$ está lejos de todos los demás.

	Distancia	≤ 1.5?	¿Vecino?
P_0	7.071	No	
P_1	6.364	No	
P_2	6.4031	No	
P_3	7.106	No	
P_4	0.000	Sí	✓
P_5	14.142	No	

Resultado: $vecinos =$ 1, $|vecinos| =$ 1.

3. **Decisión:** $|vecinos| =$ 1 $< MinPts = 3 \rightarrow$ -1

4. $labels[4] =$ -1

Estado de labels después de $i = 4$:

	P_0	P_1	P_2	P_3	P_4	P_5
labels	1	1	1	1	-1	0

7.6. Iteración $i = 5$ (punto P_5) — Completar

1. **Verificación:** $labels[5] =$ 0 $\rightarrow$ procesar

2. **regionQuery**$(P_5, 1.5)$:

   Resultado: $vecinos =$ 1, $|vecinos| =$ 1.

3. **Decisión:** -1

4. $labels[5] =$ -1

7.7. Estado final de *labels*

Completa la tabla final:

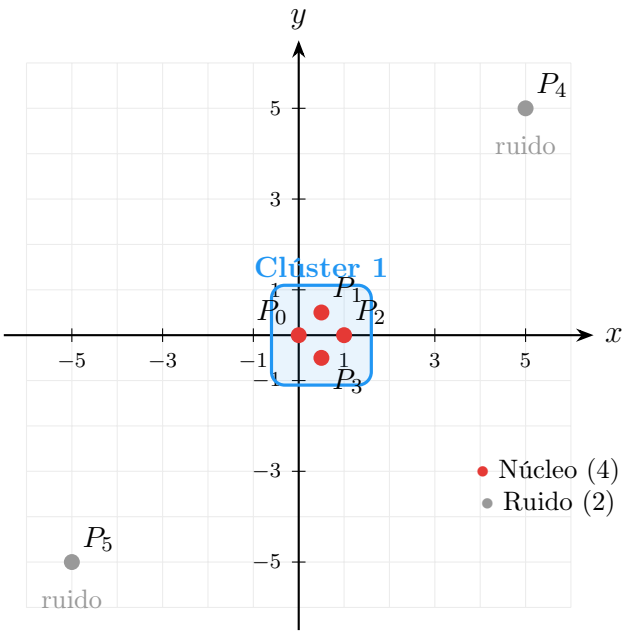
	P_0	P_1	P_2	P_3	P_4	P_5
labels	1	1	1	1	-1	-1

8. Paso 4: Resultado y visualización

8.1. Resumen de la ejecución

Punto	Coordenadas	Tipo	Clúster	Razón
P_0	$(0.0, 0.0)$	Núcleo	1	$\lvert N_\varepsilon \rvert = 4 \geq 3$
P_1	$(0.5, 0.5)$	Núcleo	1	$\lvert N_\varepsilon \rvert = 4 \geq 3$
P_2	$(1.0, 0.0)$	Núcleo	1	$\lvert N_\varepsilon \rvert = 4 \geq 3$
P_3	$(0.5, -0.5)$	Núcleo	1	$\lvert N_\varepsilon \rvert = 4 \geq 3$
P_4	$(5.0, 5.0)$	Ruido	-1	$\lvert N_\varepsilon \rvert = 1 < 3$, sin núcleos cercanos
P_5	$(-5.0, -5.0)$	Ruido	-1	$\lvert N_\varepsilon \rvert = 1 < 3$, sin núcleos cercanos

8.2. Visualización final



9. Interpretación de la salida esperada

La salida esperada (`output_1.txt`) es:

```
1
1
1
1
-1
-1
```

Cada línea i indica la etiqueta asignada al punto P_i:

- **Líneas 1–4** (valor `1`): Los puntos P_0, P_1, P_2, P_3 pertenecen al **Clúster 1**. Forman un grupo denso donde cada punto tiene al menos $MinPts = 3$ vecinos dentro de $\varepsilon = 1.5$, es decir, **todos son puntos núcleo**.

- **Líneas 5–6** (valor `-1`): Los puntos P_4 y P_5 son **ruido**. Cada uno está completamente aislado: su único vecino dentro de ε es él mismo ($|N_\varepsilon| = 1 < 3$), y ningún punto núcleo los alcanza.

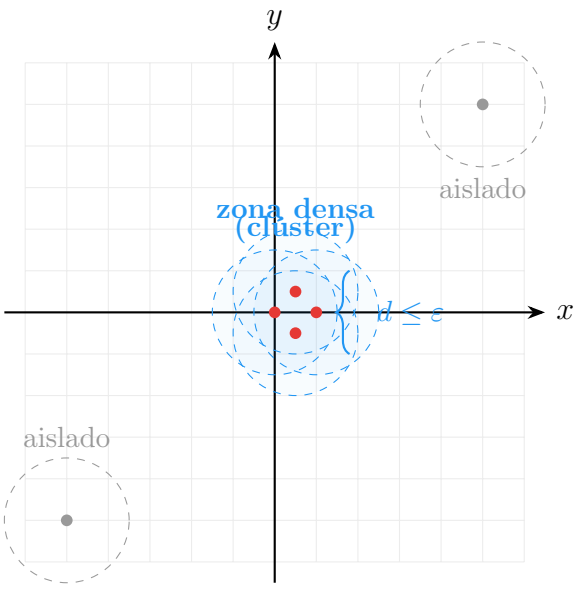
1. **Zona sísmica activa (Clúster 1):** Los puntos P_0–P_3 están todos dentro de un radio de ≈ 1 unidad entre sí (mucho menor que $\varepsilon = 1.5$). Esto los convierte en una región **densa**: cada punto alcanza a los otros tres, formando un grupo de 4 puntos núcleo mutuamente conectados por densidad.

   La estructura geométrica es un **rombo** centrado en $(0.5, 0.0)$:

   $$P_0 = (0, 0), \quad P_1 = (0.5, 0.5), \quad P_2 = (1, 0), \quad P_3 = (0.5, -0.5)$$

2. **Eventos aislados (Ruido):** El punto $P_4 = (5, 5)$ está a distancia ≈ 7.07 del punto más cercano del clúster. El punto $P_5 = (-5, -5)$ está a distancia ≈ 7.07 del punto más cercano. Ambos están a más de **4 veces** ε de cualquier otro punto, por lo que son claramente ruido.

3. **No hay puntos frontera** en este ejemplo porque no hay puntos que cumplan $|N_\varepsilon| < MinPts$ y estén dentro del vecindario de un punto núcleo.



Conclusión

El algoritmo DBSCAN identificó correctamente:

- **1 clúster** (zona sísmica activa) con 4 puntos núcleo mutuamente conectados.
- **2 puntos de ruido** (eventos sísmicos aislados) que no pertenecen a ninguna zona activa.

La clave es que DBSCAN no necesitó que le dijéramos cuántos clústeres buscar: los descubrió automáticamente basándose únicamente en la **densidad local** de los puntos.