29/11/2025

Algoritmos de agrupación

Extracción de Conocimiento en Bases de Datos

Luis Eduardo Aguilar Sarabia

IDGS91N

Docente: Luis Enrique Mascote Cano

Introducción

El agrupamiento (clustering) y la reducción de dimensionalidad son técnicas fundamentales en la extracción de conocimiento. El clustering permite descubrir estructuras o grupos naturales en los datos sin supervisión, útil para segmentación, detección de anomalías y exploración. La reducción de dimensionalidad simplifica conjuntos de datos de alta dimensión conservando la información más relevante, facilitando visualización, acelerando algoritmos y mitigando el ruido y la maldición de la dimensionalidad. Ambos enfoques son complementarios: reducir dimensiones puede mejorar la eficiencia y la calidad del clustering; a su vez, el clustering puede guiar selecciones de características o interpretaciones.

Algoritmos de agrupación

A) K-Means

Principio de funcionamiento

K-means es un método centroidal que busca particionar n observaciones en k clusters minimizando la suma de distancias al cuadrado entre puntos y el centro (media) del cluster. Se suele usar el algoritmo de Lloyd: inicializar k centroides, asignar cada punto al centro más cercano, recalcular centroides como medias de los puntos asignados, repetir hasta convergencia.

Parámetros clave

- k: número de clusters (debe decidirse a priori).
- criterio de convergencia: cambio mínimo en centroides o número máximo de iteraciones.
- inicialización de centroides (aleatoria, k-means++).

Ventajas

- Simple, rápido en práctica con complejidad aproximadamente O(n · k · t) (t iteraciones).

- Buen desempeño cuando los clusters son esféricos y de tamaño similar.
- Fácil de implementar e interpretar.

Limitaciones

- Requiere k a priori.
- Sensible a inicialización (solucionable con k-means++).
- Solo captura clusters convexos/esféricos; vulnerable a outliers y escalas de atributos.
- No maneja bien densidades variables.

Ejemplo (pseudocódigo)

```
Inicializar k centroides c1..ck (p. ej. k-means++) repetir hasta convergencia: para cada punto x: asignar x al cluster i con centroid más cercano: argmin_i ||x - ci||^2 para cada cluster i: actualizar ci = media de puntos asignados a i Fin
```

B) DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Principio de funcionamiento

Algoritmo basado en densidad que agrupa puntos densamente conectados y marca como ruido los puntos en regiones de baja densidad. Define vecinos dentro de un radio ε y clasifica puntos como núcleo (core), frontera (border) o ruido. Clusters se forman expandiendo desde puntos core conectados.

Parámetros clave

- ε (eps): radio de vecindad.
- minPts: número mínimo de puntos en la vecindad para que un punto sea core.
- Distancia: normalmente Euclidiana pero puede usarse otra métrica.

Ventajas

- Detecta clusters de forma arbitraria (no solo esférica).
- No requiere número de clusters a priori.
- Identifica ruido/outliers explícitamente.
- Funciona bien con densidades relativamente homogéneas.

Limitaciones

- Sensible a elección de ε y minPts; difícil en datos con densidades muy variables.
- Escalabilidad: búsqueda de vecinos costosa sin estructuras de índice (KD-tree, ball-tree) en alta dimensión.
- En alta dimensión la noción de vecindad pierde significado (curse of dimensionality).

Ejemplo (diagrama de flujo simplificado)

1. Para cada punto no visitado:
2. Marcar como visitado y recuperar vecinos dentro de ε.
3. Si vecinos ≥ minPts → iniciar nuevo cluster y expandir agregando vecinos alcanzables por densidad.
4. Si no → marcar como ruido.
5. Continuar hasta cubrir todos los puntos.

Pseudocódigo (esquemático)

```
for cada punto p no visitado: marcar p como visitado N = vecinos(p, eps) if |N| < minPts: marcar p como ruido else: crear nuevo cluster C expandir C con p y vecinos N (recursivo/bucle)
```

C) Gaussian Mixture Models (GMM)

Principio de funcionamiento

Modela los datos como una mezcla de k distribuciones gaussianas (cada cluster corresponde a una componente gaussiana con su media y covarianza). Se estima mediante el algoritmo EM (Expectation-Maximization): E-step calcula probabilidades (responsabilidades) de pertenencia de cada punto a cada componente; M-step actualiza parámetros de las gaussianas ponderadas por esas responsabilidades.

Parámetros clave

- k: número de componentes gaussianas.
- tipo de covarianza: esférica, diagonal, completa (afecta forma de clusters).
- criterios de convergencia y regularización de covarianzas.
- inicialización (p. ej. usando K-means).

Ventajas

- Permite clusters elípticos (no necesariamente esféricos) y superposición (soft assignments).
- Provee probabilidades de pertenencia, útil cuando la asignación es incierta.
- Flexible por elección de covarianzas.

Limitaciones

- Requiere k a priori.
- Sensible a inicialización y a singularidades en covarianzas (requiere regularización).
- Supone que los datos aproximan una mezcla de gaussianas — si no, modelo puede fallar.
- Más costoso computacionalmente que K-means.

Ejemplo (esquema del EM)

Inicializar parámetros {π_j, μ_j, Σ_j} para j=1..k repetir hasta convergencia: E-step: para cada dato x_i y cada componente j: γ_{ij} = π_j * N(x_i | μ_j, Σ_j) / sum_l π_l * N(x_i | μ_l, Σ_l) M-step: para cada j: N_j = sum_i γ_{ij} μ_j = (1/N_j) sum_i γ_{ij} x_i Σ_j = (1/N_j) sum_i γ_{ij} (x_i - μ_j)(x_i - μ_j)^T π_j = N_j / n Fin

Algoritmos de reducción de dimensionalidad

A) Análisis de Componentes Principales (PCA)

Fundamento matemático / conceptual

PCA busca direcciones (componentes principales) que capturen la máxima varianza de los datos mediante una proyección lineal. Matemáticamente se centra en la descomposición en valores propios de la matriz de covarianza (o SVD de la matriz centrada). Las primeras componentes (autovectores asociados a mayores autovalores) retienen la mayor parte de la varianza.

Parámetros clave

- n_components: número de componentes retenidas.
- si se centra/estandariza: centrar restando la media; opcionalmente escalar por desviación estándar.
- método numérico (SVD o eigen-decomposition).

Ventajas

- Lineal, rápido y determinista.
- Conserva la varianza máxima y reduce ruido.
- Fácil interpretación matemática y computacionalmente eficiente.
- Buen preprocesamiento para muchos algoritmos (clasificación, clustering).

Limitaciones

- Solo captura relaciones lineales; no modela bien estructuras no lineales complejas.
- Componentes son combinaciones lineales globales (poca interpretabilidad en algunos dominios).
- Sensible a escala de variables (recomienda estandarizar).

Ejemplo (pseudocódigo)

```
Dado X (n x d): 1. Centrar X: Xc = X - mean(X) 2. Calcular matriz de covarianza C = (1/(n-1)) Xc^T Xc 3. Obtener autovalores λ y autovectores V de C 4. Ordenar autovectores por λ descendente y seleccionar k primeros 5. Proyectar: X_reducido = Xc · V_k
```

B) t-SNE (t-distributed Stochastic Neighbor Embedding)

Fundamento matemático / conceptual

t-SNE es un método no lineal de reducción para visualización que preserva la estructura local. Convierte distancias entre puntos en altas dimensiones en probabilidades de similitud (distribución gaussiana local), define analogamente probabilidades en el espacio reducido usando una distribución t de Student (colas más pesadas) y optimiza la representación en baja dimensión minimizando la divergencia Kullback-Leibler entre ambas distribuciones mediante gradiente descendente.

Parámetros clave

- perplexity: controla equilibrio entre estructura local y global (efectivo como tamaño de vecindad).
- n_components: dimensiones resultantes (normalmente 2 o 3).
- learning_rate (lr), número de iteraciones, inicialización (PCA suele usarse).
- parámetros de momento y exageración inicial (early exaggeration).

Ventajas

- Excelente para visualización (2D/3D) de estructuras locales y clústeres.
- Maneja datos con formas complejas y no lineales.

Limitaciones

- Costoso computacionalmente (aunque hay aproximaciones aceleradas).
- No preserva distancias globales: la distancia entre clusters no siempre es interpretable.
- Resultado estocástico y sensible a parámetros (perplexity) y escala de datos.
- No es adecuado como método de preprocesamiento para todas las tareas (mejor solo visualización exploratoria).

Ejemplo (pseudocódigo simplificado)

Calcular similitudes p_{j|i} en espacio alto (Gaussiana) con perplexity dada Simetrizar p_{ij} = (p_{j|i} + p_{i|j})/(2n) Inicializar Y (baja dimensión) (p. ej. PCA) Iterar: Calcular q_{ij} en Y usando Student t (1 DOF) Calcular gradiente de KL(p||q) Actualizar Y por gradiente descendente con momentum Fin

Comparativa

Aspecto	K-means	DBSCAN	GMM	PCA	t-SNE
Tipo	Particionamiento	Basado en densidad	Modelo probabilístico (mixture)	Proyección lineal	Proyección no lineal (visualización)
Requiere k	Sí	No	Sí	No (pero n_components)	No (pero perplexity)
Captura formas	Esféricas	Arbitrarias	Elípticas (según covarianza)	—	Arbitrarias (locales)
Maneja	No (sensible)	Sí (detecta	Parcialmente	—	Puede

Aspecto	K-means	DBSCAN	GMM	PCA	t-SNE
ruido		ruido)	(probabilístico)		separar ruido en visualización
Complejidad	Baja	Media–alta	Media–alta	Baja–media	Alta
Uso recomendado	Clusters rápidos y claros	Clusters con densidad, outliers	Cuando hay solapamiento y asignación probabilística	Reducción/visualización lineal, preprocessing	Visualización exploratoria de estructuras locales

Conclusiones

El clustering y la reducción de dimensionalidad son herramientas complementarias en el análisis exploratorio y la extracción de conocimiento. La elección del algoritmo depende fuertemente de la forma de los datos, la presencia de ruido, la necesidad de interpretabilidad y los objetivos (visualización vs. despliegue). K-means es rápido y útil en escenarios simples; DBSCAN resuelve problemas de ruido y formas arbitrarias; GMM aporta una perspectiva probabilística. PCA es la primera opción cuando se busca una transformación lineal eficiente que preserve varianza; t-SNE es excelente para revelar estructura local en visualizaciones aunque no para tareas que requieran estabilidad ni conservación de distancias globales. En la práctica, combinar técnicas —por ejemplo, PCA seguido de DBSCAN o t-SNE para visualización de resultados de GMM— suele ofrecer los mejores resultados.

Referencias

Betancourt, J. C. M. (2021, 16 diciembre). *8 algoritmos de agrupación en clústeres en el aprendizaje automático que todos los científicos de datos deben conocer*.

freeCodeCamp.org. https://www.freecodecamp.org/espanol/news/8-algoritmos-de-agrupacion-en-clusteres-en-el-aprendizaje-automatico-que-todos-los-cientificos-de-datos-deben-conocer/

Ibm. (2025b, febrero 18). Agrupación en clústeres. *IBM*. https://www.ibm.com/mx-es/think/topics/clustering

2.3. Clustering. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/clustering.html