Clase 094 — MDS, Isomap, t-SNE, UMAP, LDA

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 8 + UMAP docs. Duración estimada: 80 min.

Clase 094 — MDS, Isomap, t-SNE, UMAP, LDA

> *Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 8 + UMAP docs. Duración estimada: 80 min.*

Objetivo

Conocer y aplicar técnicas de reducción de dimensionalidad más allá de PCA: MDS, Isomap, t-SNE, UMAP y LDA. Entender qué preserva cada una (distancias, geodésicas, vecindarios locales, estructura global, separación entre clases) y cuándo elegir cada método según el problema (visualización 2D, preprocesamiento para clasificación, datos en variedades no lineales).

Resultados de aprendizaje

1. Distinguís entre métodos lineales (PCA, LDA) y no lineales (Isomap, t-SNE, UMAP) y sabés cuándo usar cada uno.
2. Aplicás MDS, Isomap, TSNE, umap.UMAP y LinearDiscriminantAnalysis de scikit-learn / umap-learn sobre datasets reales.
3. Configurás los hiperparámetros clave: perplexity (t-SNE), n_neighbors y min_dist (UMAP), n_neighbors (Isomap).
4. Interpretás correctamente embeddings 2D: qué significan los clusters, qué NO significan las distancias entre clusters en t-SNE.
5. Justificás por qué UMAP suele ser preferible a t-SNE: más rápido, preserva mejor la estructura global y es determinista con random_state.

Temas

Método	Tipo	Preserva	Uso típico	Hiperparámetros clave
MDS	No lineal	Distancias pairwise	Visualización de matrices d	n_components, metric
Isomap	No lineal	Distancias geodésicas (sob	Datos en manifold curvo (S	n_neighbors
t-SNE	No lineal	Vecindarios locales (proba	Visualización 2D/3D de clu	perplexity, learning_rate
UMAP	No lineal	Estructura local y global	Visualización + preprocesa	n_neighbors, min_dist
LDA	Lineal supervisado	Separación entre clases	Reducción previa a clasific	n_components (≤ n_clases−1)

Definiciones y características

- MDS (Multidimensional Scaling): proyecta a baja dimensión intentando que las distancias entre puntos en el espacio reducido sean lo más parecidas posible a las distancias originales. No asume linealidad pero es costoso: $O(n^2)$ en memoria.
- Isomap: variante de MDS que reemplaza la distancia euclídea por la distancia geodésica (camino más corto sobre el grafo de k vecinos más cercanos). Ideal cuando los datos viven en una variedad curva (ej: Swiss roll).
- t-SNE (t-distributed Stochastic Neighbor Embedding): convierte similitudes en distribuciones de probabilidad y minimiza la divergencia KL entre el espacio original y el reducido. Excelente para

visualizar clusters; no sirve como preprocesamiento general porque es estocástico y no preserva distancias globales.
- Perplexity (t-SNE): controla cuántos vecinos efectivos considera cada punto. Valores típicos: 5–50. Datasets grandes → perplexity mayor.
- UMAP (Uniform Manifold Approximation and Projection): alternativa moderna a t-SNE basada en topología algebraica. Más rápido, escalable, preserva mejor la estructura global y soporta transform sobre datos nuevos.
- n_neighbors (UMAP): balance entre estructura local (valores bajos, ~5–15) y global (valores altos, ~50–200). min_dist controla qué tan "apretados" se ven los clusters.
- LDA (Linear Discriminant Analysis): método supervisado que proyecta los datos maximizando la separación entre clases y minimizando la varianza dentro de cada clase. Limitado a n_clases − 1 dimensiones.
- Supervisado vs no supervisado: LDA usa las etiquetas y; PCA, MDS, Isomap, t-SNE y UMAP no. Esto hace que LDA sea óptimo como preprocesamiento de clasificación cuando hay etiquetas.

Dataset / recursos

- sklearn.datasets.make_swiss_roll — clásico para Isomap vs PCA.
- sklearn.datasets.load_digits — 8x8 = 64 dims, ideal para visualizar con t-SNE/UMAP.
- sklearn.datasets.fetch_openml('mnist_784') — 70k × 784, para comparar tiempos t-SNE vs UMAP.
- Librerías: scikit-learn (MDS, Isomap, TSNE, LDA) + umap-learn (pip install umap-learn).

Ejercicios

1. Generá un Swiss roll con make_swiss_roll(n_samples=1500) y reducí a 2D con PCA, MDS e Isomap. Graficá los tres y comentá cuál "desenrolla" la variedad.
2. Cargá load_digits y aplicá t-SNE con perplexity ∈ {5, 30, 50, 100}. Mostrá los 4 plots y explicá el efecto.
3. Sobre load_digits, compará t-SNE vs UMAP: medí tiempo de ejecución con time.perf_counter() y reportá ambos embeddings 2D.
4. Aplicá LDA a load_digits reduciendo a 2D y a 9D. Entrená un LogisticRegression sobre cada versión y compará accuracy con el original (64 dims).
5. Con UMAP sobre load_digits, probá n_neighbors ∈ {2, 15, 100} con min_dist=0.1. Graficá los tres y explicá el trade-off local vs global.

Homework verificable

Tomá load_digits y producí un script compare_dimred.py que:

1. Aplique PCA(2), Isomap(2), t-SNE(2) y UMAP(2).
2. Para cada embedding, entrene un KNeighborsClassifier(n_neighbors=5) con cross_val_score (cv=5) sobre el embedding 2D.
3. Imprima una tabla con método, tiempo de ajuste y accuracy media.

Criterio: UMAP debe terminar al menos 3× más rápido que t-SNE sobre los 1797 dígitos, y la accuracy 2D de UMAP debe superar 0.90.

Errores comunes

1. Interpretar las distancias entre clusters en t-SNE como reales. t-SNE preserva vecindarios locales, no distancias globales: dos clusters cercanos en el plot no son necesariamente similares.
2. Usar t-SNE como preprocesamiento de un modelo. No tiene transform confiable sobre datos nuevos; usá UMAP o PCA para eso.
3. No escalar antes de Isomap o t-SNE. Estos métodos dependen de distancias; sin StandardScaler las features con escala grande dominan.
4. Pedirle a LDA más componentes que n_clases − 1. scikit-learn lanza error; LDA está topeado por la cantidad de clases.
5. Olvidar random_state en t-SNE/UMAP. Son estocásticos: sin semilla el embedding cambia en cada corrida y los plots no son reproducibles.

Preguntas frecuentes

1. ¿t-SNE o UMAP? UMAP en casi todos los casos: más rápido, preserva estructura global, tiene transform, escala a millones de puntos. t-SNE sigue siendo válido para visualización pequeña cuando ya tenés pipelines hechos.
2. ¿Cuándo usar Isomap en vez de PCA? Cuando sospechás que los datos viven en una variedad curva (ej: imágenes de un objeto rotando). PCA proyecta linealmente y aplana mal la curvatura.
3. ¿LDA o PCA antes de clasificar? Si tenés etiquetas, LDA suele dar mejor separación con menos dimensiones. PCA es agnóstico al target y puede tirar info útil.
4. ¿Por qué MDS es tan lento? Calcula la matriz de distancias $O(n^2)$ y resuelve un problema de optimización. Para n > 5000 es impráctico; usá Isomap o UMAP.
5. ¿Puedo usar UMAP como preprocesamiento supervisado? Sí: UMAP acepta y en fit y aprende un embedding que respeta las etiquetas (semi-supervisado).

Referencias

- Géron, Hands-On ML, cap. 8 — "Other Dimensionality Reduction Techniques".
- McInnes, Healy & Melville (2018). UMAP: Uniform Manifold Approximation and Projection. arXiv:1802.03426.
- van der Maaten & Hinton (2008). Visualizing Data using t-SNE. JMLR.
- Documentación: <https://umap-learn.readthedocs.io/> · <https://scikit-learn.org/stable/modules/manifold.html>

Siguiente clase

Clase 095 — Clustering K-Means: selección de K, MiniBatch

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb