AAI - Tema 8: Selección de variables y reducción de la dimensionalidad

- Maldición de la dimensionalidad
- Enfoques
  - Proyección
  - Manifold
- Técnicas
  - Análisis principal de componentes (PCA)
  - Incrustadores localmente lineales (LLE)
  - Isomap
  - t-NSE
  - Análisis discriminante lineal (LDA)

Ventajas de reducción de la dimensionalidad:

- Simplicación del conjunto de datos
- Mejorar la interpretabilidad
- Reduce el uso de memoria

Posibles incovenientes:

- Puede perderse información útil y empeorar el rendimiento posterior.

Maldición de la dimensionalidad

Concepto de maldición de la dimensionalidad.

Enfoques

En este apartado se revisan los distintos enfoques de las técnicas de reducción de dimensionalidad.

- Proyección
- Aprendizaje de variedadades (Manifold)

Proyección

La **proyección** asume que la mayoría de instancias de datos están en subespacios de menor dimension y permite reducir el número de dimensiones de los conjuntos de datos.

Aprendizaje de variedades (Manifold)

Un ***manifold*** es una parte de un espacio dimensional que localmente parece que es un objeto de una dimensión menor pero que el alto espacio dimensional aparece enrollado.

La **hipótesis manifold** o suposición manifold sostiene que la mayoría de objetos de alta dimensión puede simplificarse en un manifold de menor dimensión.

Suele ir acompañado de la suposición adicional de que los problemas son más fáciles de solucionar cuanto menor sea la dimensión.

Técnicas

- PCA
- Incrustadores localmente lineales (LLE)
- Escalado multidimensional (MDS)

Análisis de componentes principales (PCA)

Análisis de componentes principales o *principal component analysis* (PCA) es una técnica no supervisada. Es probablemente la más popular para reducción de dimensionalidad.

- Ficha
  - Supervisión: No supervisado
  - Preprocesado: centrado
  - Tareas: reducción de dimensionalidad
  - Vol. dimensiones: medio-bajo
- Características
  - Requiere escoger k. Se hace gracias a EVR
- Proceso
  - Se centran los datos
  - Se calculan los PCs con SVD
  - Se proyectan los PCs
  - El núm. dim. correcto se escoge en base a varianza explicada (%)
- Implementación sklearn: `PCA`, `IncrementalSVD`
  - Hiperparámetros
    - `n_components`: núm. dimensiones final
    - `svd_solver`: algoritmo de SVD a aplicar
- Versiones de SVD
  - Completo: complejidad $O(m \cdot d^2) + O(d^3)$ (mejor cuando d»n)
  - Aleatorizado
    - estocástico
    - complejidad $O(m \cdot d^2) + O(d^3)$ (mejor cuando n»d).
    - `svd_solver=randomized`
  - Incremental
    - Mini-batches
    - Clase `IncrementalSVD`
  - Con kernel
    - Para capturar relaciones no lineales

Un **componente principal** (PC) es cada uno de los ejes que contiene la mayor varianza. PCA lo identifica.

Además, PCA encuentra para cada PC un vector unitario centrado en cero que apunta a la misma dirección del PC. La dirección es inestable y puede variar.

Es importante recordar que PCA asume que los datos están centrados y por tanto **requiere el centrado** como preprocesamiento.

La técnica de **descomposición de valor singular** o *singular value decomposition* (SVD) se emplea para encontrar los PCs.

$X = U\,\Sigma\,V^\top$

donde:

- V: matriz de componentes principales, que contiene en orden los vectores unitarios que definen los PCs.

La matriz de componentes principales se define como:

$$V = \begin{pmatrix} | & | & & | \\ \mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_n \\ | & | & & | \end{pmatrix}$$

Está implementado en NumPy en la función `svd()`.

Finalmente se reduce la dimensionalidad a d dimensiones proyectando en el hiperplano definido en las primeras d PCs.

$$X_{d-proj} = X \cdot W_d$$

donde:

- X: conjunto de datos original
- W_d: matriz que contiene las primeras d columnas de V
- X_{d-proj}: conjunto de datos reducido de d dimensiones.

Se implementa en Scikit-learn en `PCA`, que usa SVD.

El **ratio de varianza explicada** o *explained variance ratio* (EVR) indica la proporción de varianza que se mantiene en cada PC tras la transformación

En Scikit-learn se establece con el atributo `explained_variance_ratio_` de `PCA`. Devuelve una matriz con el % para cada PC.

La **elección de dimensiones** se basa en un criterio aplicado:

- Para visualización: de 2 a 3 dimensiones.
- Resto de casos: mantener un % de la varianza (ej. 95%).

En Scikit-learn se establece con el atributo `n_components` de `PCA`.

Si se representa gráficamente el número de dimensiones contra la varianza explicada se puede escoger el número de dimensiones adecuado en el punto donde quede el codo.

Enfoques de SVD: - Completo - Aleatorizado - Incremental

Bibliografía:

- Básica
  - Capítulo 9 HOML. Págs. 243-251.
  - Decomposition
  - Unsupervised reduction
  - Decomposiciont CTA
  - IncrementalPCA
  - KernelPCA

Proyección aleatoria

Este concepto no es evaluado en la asignatura, por lo que se puede saltar su lectura.

Incrustadores localmente lineales (LLE)

Los inscrustadores localmente lineales o *locally linear embedding* (LLE) es una técnica **no lineal**.

- Ficha
  - Supervisión: No supervisado
  - Uso: reducción de la dimensionalidad
  - Aplicación: aplanar datos en forma brazo enrollado
  - Vol. instancias: bajo
  - Complejidad: $\sim O(m^2\, n\, k^3)$
  - No linear
- Pasos
  1. Modelar linearmente relaciones locales
  2. Reducir dim. preservando relaciones

Tienen como aplicación más ejemplar el aplanar datos almacenados con forma de un brazo enrollado (en inglés, *Swiss roll*).

Se divide en dos pasos

1. Modelar linealmente las relaciones locales
2. Reducir la dimensionalidad mientras se preservan las relaciones

Los pasos son:

#	Tarea	Complejidad
1	Aplica kNN como una función lineal con los vecinos.	$O(m \log(m)\, n\, \log(k))$
2	Optimiza los pesos	$O(m\, n\, k^3)$

#	Tarea	Complejidad
3	Construye la representación de menor dimensión	$O(m^2)$

La complejidad computacional de la técnica en conjunto es la peor de cada uno de los pasos.

El último paso lo hace poco escabable para un volumen alto de instancias.

Bibliografía:

- Capítulo 9 HOML. Págs. 254-256
- Comparison of Manifold Learning Methods

Escalado multidimensional (MDS)

Escalado multidimensional o *multidimensional scaling* (MDS) reduce la dimensionalidad mientras trata de preservar las distancias entre instancias.

La proyección aleatoria lo aplica bien para muchas dimensiones, pero funciona mal para pocas.

Bibliografía:

Básica: - Capítulo 9 HOML. Pág. 256 - https://scikit-learn.org/stable/modules/generated/sklearn.manifold.MDS.html#sklearn.manifold.MDS

Isomap

Isomap crea un **grafo** que conecta cada instancia con sus vecinos más próximos, reduce la dimensionalidad tratando de preservar la distancia geodésica entre isntancias.

La **distancia geodéstica** entre dos nodos en un gráfo es el número de nodos en la ruta más corta entre nodos.

Bibliografía:

- Básica
  - Capítulo 9 HOML. Pág. 256
  - https://scikit-learn.org/stable/modules/generated/sklearn.manifold.Isomap.html

t-SNE

t-distributed stochastic neighbor embedding (t-SNE) reduce la dimensionalidad mientras mantiene cerca las instancias similares y lejos las distintas.

Se usa principalmente para **visualización**.

Bibliografía:

- Básica
  - Capítulo 9 HOML. Pág. 257
  - https://scikit-learn.org/stable/modules/generated/sklearn.manifold.LocallyLinearEmbedding.html#sklearn.manifold.LocallyLinearEmbedding

Análisis discriminante lineal (LDA)

Análisis discriminante lineal o *linear discriminant analysis* (LDA) es un algoritmo **lineal** que aprende los ejes más discriminativos entre clases. Estas puede ser usadas para definir un hiperplano sobre el que proyectar.

La ventaja de LDA es que **mantiene las clases lo más lejos posible**, y por ello es adecuada como preprocesamiento antes de la clasificación.

Bibliografía:

- Básica
  - Capítulo 9 HOML. Pág. 257
  - https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html#sklearn.discriminant_analysis.LinearDiscriminantAnalysis