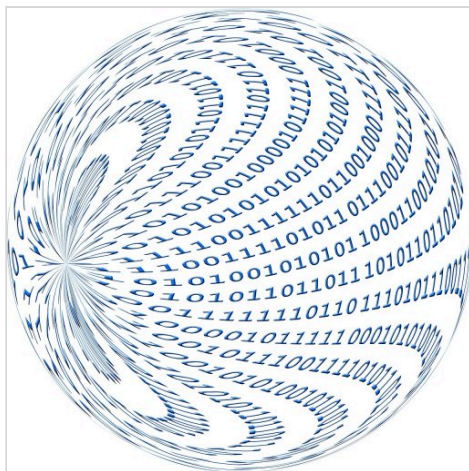


L U M B R E



Arrays de NumPy y broadcasting

Grado: 2

Materia: Programación con Python para Datos · Programación

03/10/2026

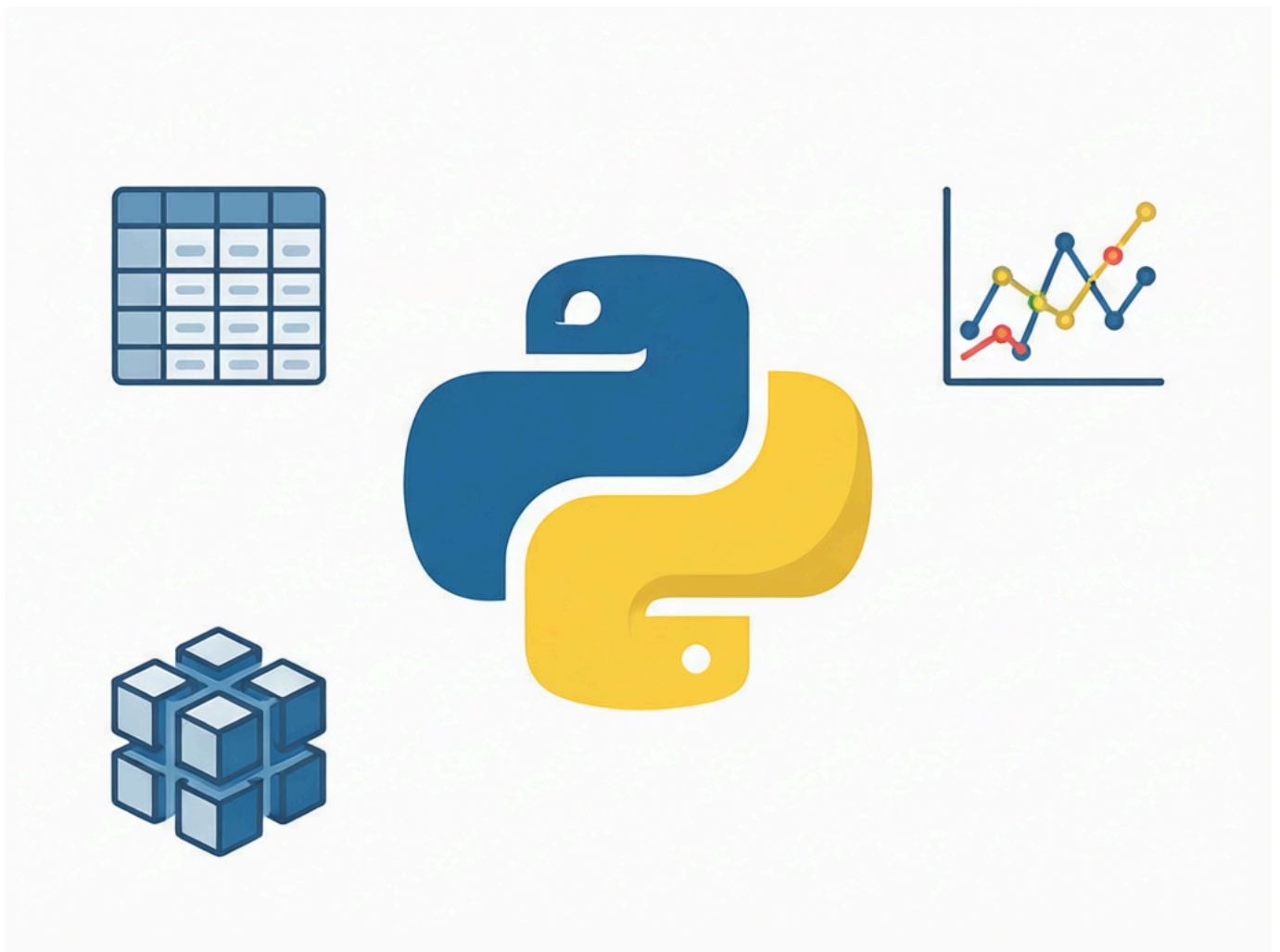
Antes de empezar

NumPy es el cimiento computacional de la ciencia de datos en Python: un array es un bloque de datos homogéneo sobre el que se opera entero, sin bucles. El broadcasting extiende esa capacidad a formas distintas.

QUÉ APRENDERÁS

- Crear arrays de NumPy, vectorizar operaciones y aplicar el broadcasting correctamente para operar con matrices de distinto tamaño.

Arrays de NumPy y broadcasting



Python para ciencia de datos: NumPy, pandas y manipulación de datos.

Mapa conceptual

- **Python para Datos**
 - Entorno
 - pip

- venv
- Reproducibilidad
- Funcional
 - Comprensiones
 - Lambda
 - map filter
- Calidad
 - try except
 - pytest
 - Asserts
- NumPy
 - Arrays
 - Broadcasting
 - Indexacion
- Pandas
 - DataFrame
 - Groupby
 - Merge Join

El mapa mental interactivo es interactivo en la versión online.



El cimiento de todo

Casi toda la ciencia de datos en Python se apoya en NumPy. Pandas, scikit-learn y los tensores de aprendizaje automático son, por debajo, arrays. Aprender a pensar en operaciones sobre arrays enteros, y no en bucles elemento a elemento, cambia tu forma de programar.

¿Qué es un array?

Un array de NumPy es una cuadrícula de valores del mismo tipo, indexada por posiciones. A diferencia de una lista de Python, guarda los datos de forma contigua en memoria, lo que permite operaciones vectorizadas mucho más rápidas. Puede tener una, dos o más dimensiones.

Vectorización

Vectorizar es aplicar una operación a todos los elementos a la vez, sin escribir el bucle. Cuando sumas dos arrays con `a + b`, NumPy recorre los elementos internamente en código compilado. El resultado es el mismo que un bucle, pero órdenes de magnitud más rápido y mucho más legible.

Crear arrays

1. `np.array([1, 2, 3])` desde una lista; `np.array([[1,2],[3,4]])` para una matriz.
2. `np.zeros(5)`, `np.ones((2,3))`, `np.arange(0, 10, 2)` generan estructuras al instante.
3. `np.linspace(0, 1, 11)` reparte un rango en un número fijo de puntos, ideal para mallas.

dtype y forma

- `dtype`: el tipo de dato que guardan todas las casillas (`float64`, `int64`, `bool`). Homogéneo por diseño.
- `shape`: las dimensiones del array como tupla; `(2, 3)` significa 2 filas y 3 columnas.
- `size`: número total de elementos; `ndim`: cuántas dimensiones tiene.

Inspeccionar estos atributos es el primer paso antes de operar.

Atributo	Qué informa	Ejemplo para una matriz 2x3
<code>shape</code>	dimensiones	<code>(2, 3)</code>
<code>ndim</code>	número de ejes	2
<code>size</code>	total de elementos	6
<code>dtype</code>	tipo homogéneo	<code>float64</code>

1. ¿Qué ventaja principal ofrece vectorizar con NumPy frente a un bucle Python elemento a elemento?

- ☐ Ocupa menos disco
- ☐ Ejecuta el recorrido en código compilado, mucho más rápido y con código más claro
- ☐ Permite guardar tipos distintos en el mismo array
- ☐ Elimina la necesidad de memoria

Broadcasting: la idea

El broadcasting es la regla que permite operar arrays de formas distintas sin copiar datos. NumPy «extiende» virtualmente el array más pequeño alineando sus ejes con el mayor, de derecha a izquierda, para que las formas sean compatibles. No duplica memoria: repite el valor sobre la marcha.

Reglas de compatibilidad

- Se comparan las formas eje a eje empezando por la derecha.
- Dos ejes son compatibles si son iguales, o si uno de ellos vale 1.
- Un eje ausente (array con menos dimensiones) se trata como si valiera 1.

Ejemplo de broadcasting

Una matriz de forma (3, 4) y un vector de forma (4,) son compatibles: el vector se aplica a cada una de las tres filas. Una columna de forma (3, 1) se propaga a las cuatro columnas. Entender esto evita escribir bucles anidados para restar una media por columna, por ejemplo.

Broadcasting en la práctica

Estandarizar una matriz restando la media de cada columna y dividiendo por su desviación se resuelve en dos líneas gracias al broadcasting: las estadísticas por columna tienen la forma adecuada para propagarse sobre las filas. Es la operación detrás de innumerables preprocesamientos.

Errores frecuentes

- Forzar formas incompatibles y recibir un error de broadcast que no se sabe leer.
- Confundir una copia con una vista: modificar el resultado de un rebanado puede alterar el original.
- Guardar datos mixtos en un array y descubrir que NumPy los promueve todos a texto.

2. El broadcasting copia físicamente el array pequeño tantas veces como sea necesario para igualar formas.

☐ Verdadero

☐ Falso

Cuándo NO usar arrays

Si tus filas tienen longitudes distintas o estructuras heterogéneas, un array rectangular no es la herramienta: ahí suelen encajar mejor una lista de objetos o un DataFrame de Pandas. El array brilla cuando los datos son una tabla numérica regular.

3. Clasifica cada operación:

Elementos:

- $a * b$ con dos arrays de igual forma
- sumar un vector de forma (4,) a una matriz (3, 4)
- `np.zeros((2, 3))`
- elevar al cuadrado todos los elementos con $a ** 2$

Vectorizacion	. . .
Broadcasting	. . .
Creacion	. . .

NumPy Docs <https://numpy.org/doc/stable/user/basics.broadcasting.html> Reglas de broadcasting explicadas con diagramas.

VIDEO

Video complementario

Recurso audiovisual para reforzar los conceptos.



Escanea para verlo en Lumbre

<https://www.youtube.com/watch?v=rfscVS0vtbw>

4. Tienes una matriz de temperaturas de forma (7, 24) (una semana por horas) y quieres restar la media de cada hora. Explica con tus palabras qué forma debe tener el vector de medias para que el broadcasting funcione y por qué.

Documentacion pandas <https://pandas.pydata.org/docs/> Guia de referencia para manipulacion de datos en Python.

Referencia curricular

Grado:	2
Materia:	Programación con Python para Datos · Programación

Versión interactiva completa (con actividades, videos y autoevaluación):



<https://lumbre.lat/object/cdia-pyth-004-arrays-de-numpy-y-broadcasting>

Material educativo generado desde Lumbre · lumbre.lat