Clase 016 — NumPy: agregaciones

Parte: 0 — Prerrequisitos · Fuente: VanderPlas, cap. 2 § 2.4 Aggregations: Min, Max, and Everything in Between. · Duración estimada: 60 min.

Clase 016 — NumPy: agregaciones

> *Parte: 0 — Prerrequisitos · Fuente: VanderPlas, cap. 2 § 2.4 Aggregations: Min, Max, and Everything in Between. · Duración estimada: 60 min.*

Objetivo

Que el alumno reduzca arrays a estadísticos (sum, mean, std, percentile, min, max) controlando el axis correcto — la fuente del 50% de los bugs de pandas/sklearn cuando alguien se confunde de eje. También: variantes nan* y reducciones acumulativas.

Resultados de aprendizaje

Al finalizar la clase, el alumno podrá:

1. Calcular sum, mean, std, var, median, percentile sobre arrays.
2. Controlar el eje con axis=0 (a lo largo de filas, da resultado por columna) y axis=1 (a lo largo de columnas, da por fila).
3. Usar variantes nan* (nansum, nanmean, etc.) cuando hay datos faltantes.
4. Reducciones acumulativas con cumsum y cumprod.
5. Encontrar índice del min/max con argmin/argmax.

Temas

#	Tema	Por qué importa
1	Reducciones básicas	sum, mean, std, var, min, max, median, per
2	Eje: el bug más común	axis=0 reduce filas (resultado por columna
3	Variantes NaN-aware	nansum, nanmean, nanmedian, nanstd.
4	Acumulativas	cumsum, cumprod — útiles para series tempo
5	argmin/argmax	Posición del extremo.
6	all y any	Reducciones booleanas.

Definiciones y características

Agregación / reducción

: Operación que colapsa un array a menos dimensiones: sum, mean, std, min, max, argmax. Sin axis, reduce a un escalar; con axis=N, elimina la dimensión N.

axis=0 vs axis=1

: Regla: el axis que pasas es el que desaparece. En matriz (filas, cols): axis=0 colapsa filas → un valor por columna; axis=1 colapsa cols → un valor por fila. Mnemónico inverso al que muchos esperan.

argmin / argmax

: Devuelven el índice del extremo (no el valor). arr.argmax() = posición del máximo; arr[arr.argmax()] = valor máximo. Con axis= devuelven array de índices por fila/columna.

Variantes nan*

: Versiones que ignoran NaN en vez de propagarlo: nansum, nanmean, nanstd, nanmin, nanmax, nanmedian, nanpercentile, nanargmax. Útiles cuando los datos tienen missing.

Acumulativas (cumsum, cumprod)

: No colapsan — devuelven array de igual shape con valores acumulados hasta cada posición. Útiles para series temporales (precio acumulado, drawdown).

percentile / quantile

: Valor por debajo del cual cae el N% de los datos. np.percentile(arr, 50) = mediana. np.percentile(arr, [25, 50, 75]) = cuartiles.

Dataset / recursos

Sintético (matriz aleatoria 100×10 simulando "10 features × 100 muestras") generado en el notebook. Sin descarga.

Ejercicios

1. Promedio por columna. Dada matriz 100×4 de ventas (filas=día, cols=tienda), calcula la media por tienda y por día.

2. Estadísticos completos. Para un array de 1000 normales, reporta mean, std, median, p25, p75, min, max.

3. Con NaN. Inserta 50 NaN aleatorios en el array anterior. Compara mean (propaga) vs nanmean.

4. Cumsum. Genera array de retornos diarios aleatorios. Calcula el precio acumulado con cumprod(1+r).

5. Mejor tienda. Con la matriz del ejercicio 1, usa argmax(axis=0) para encontrar el día de mayor venta de cada tienda.

Homework verificable

Notebook con matriz simulada 365 días × 5 tiendas de ventas, reportando: media/std por tienda, mejor y peor día de cada tienda, cumsum total anual, % de días con NaN simulados (20 aleatorios) usando variantes nan*.

Criterio de aceptación: Eje correcto en todas las agregaciones; valores reproducibles con seed.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
M.sum(axis=0) da resultado por columna y e	Confusión clásica. Regla: axis=0 reduce fi
arr.mean() da NaN aunque solo hay un par d	Cualquier NaN propaga. Fix: np.nanmean(arr
np.argmax(matriz) devuelve un solo número	Sin axis, aplana el array primero y devuel
np.percentile([1,2,3,4], 50) da 2.5 no 2	Interpolación lineal por default. Si quier
cumsum de floats acumula error de redondeo	Sumas múltiples introducen error numérico.

Preguntas frecuentes

¿arr.sum() o np.sum(arr)?

Equivalentes. Método del array (.sum()) es más legible en cadenas (arr.clip(0).sum()). Función (np.sum) acepta también listas (no solo ndarray).

¿Cómo recuerdo qué axis colapsa cuál?

El axis que pasas es el que desaparece. Si shape es (3, 4) y haces sum(axis=0), queda (4,) — desapareció dim 0. Si axis=1, queda (3,).

¿std usa N o N-1?

Default ddof=0 (divide por N — desviación poblacional). Para desviación muestral (N-1), arr.std(ddof=1). Pandas y scipy.stats usan N-1 por default — cuidado al comparar.

¿np.median ignora NaN?

No — usa np.nanmedian. Mismo patrón que mean/nanmean.

¿Hay una agregación 'top-3' built-in?

No directa. Usa np.partition(arr, -3)[-3:] para los 3 mayores (más rápido que sort completo). Si necesitas ordenados, .sort() después.

Referencias

- VanderPlas, cap. 2 § 2.4.
- NumPy statistics functions

Siguiente clase

Clase 017 — NumPy: broadcasting

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
import numpy as np
rng = np.random.default_rng(42)
```

Archivos complementarios

- notebook.ipynb