NumPy II – 2D tömbök (mátrixok), axis fogalma, aggregálás tengely mentén

Mi az a 2D tömb?

- 1D tömb = lista jellegű adatsor
- 2D tömb = sorok és oszlopok (táblázat)

Példa:

```
import numpy as np

matrix = np.array([
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
])

print(matrix)

#Dimenzió lekérdezése:
print(matrix.shape)
```

Indexelés 2D tömbben

Indexelés: [sor, oszlop]

- `print(matrix[0, 0])`
- `print(matrix[1, 2])`

Teljes sor:

- `print(matrix[1, :])`

Teljes oszlop:

- `print(matrix[:, 0])`

Feladat: Hozz létre egy 3×3-as tömböt 0–8 értékekkel. Írd ki a második sort. Írd ki a harmadik

oszlopot.

```
import numpy as np

arr = np.arange(9).reshape(3, 3)

print("Második sor:", arr[1, :])
print("Harmadik oszlop:", arr[:, 2])
```

Axis fogalma

- axis=0 → oszlopok mentén (sorokat “összevonjuk”)
- axis=1 → sorok mentén (oszlopokat “összevonjuk”)

Példa:

```
print(matrix.sum(axis=0))
print(matrix.sum(axis=1))
```

Feladat: Adott egy jegytábla:

```
jegyek = np.array([
    [4, 5, 3],
    [2, 4, 5],
    [5, 5, 4]
])
```

- Számold ki tanulónként az átlagot. - Számold ki tantárgyanként az átlagot.

```
import numpy as np

jegyek = np.array([
    [4, 5, 3],
    [2, 4, 5],
    [5, 5, 4]
])

print("Tanulónkénti átlag:", jegyek.mean(axis=1))
print("Tantárgyankénti átlag:", jegyek.mean(axis=0))
```

Mátrixműveletek

Elemenkénti műveletek

- `A = np.array([[1, 2], [3, 4]])`
- `B = np.array([[5, 6], [7, 8]])`
    - `print(A + B)`
    - `print(A * B)`
- Mátrixszorzás
    - `print(A @ B)`

Feladat: Hozz létre két 2×2-es mátrixot. Számold ki az összegüket. Számold ki a mátrixszorzatot

```
import numpy as np

A = np.array([[1, 2], [3, 4]])
B = np.array([[2, 0], [1, 2]])

print("Összeg:", A + B)
print("Mátrixszorzat:", A @ B)
```

Valósabb adat példa (fájlból 2D)

Tegyük fel, hogy az adat2d.txt fájl tartalma:

```
1 2 3
4 5 6
7 8 9
```

Beolvasás:

```
import numpy as np

adat = np.loadtxt("adat2d.txt")

print("Shape:", adat.shape)
print("Oszlopátlag:", adat.mean(axis=0))
```

Összefoglaló feladatok

1.Feladat: Egy cég havi bevételeit tároljuk táblázatban (3 év × 12 hónap).

- Unordered List ItemGenerálj véletlen adatokat (pl. 100–500 közötti számokkal)
- Unordered List ItemSzámold ki:
  - Évenkénti összbevétel
  - Havi átlagbevétel (3 év átlaga)

```
import numpy as np

np.random.seed(0)
bevetel = np.random.randint(100, 501, size=(3, 12))

print("Évenkénti összes bevétel:", bevetel.sum(axis=1))
print("Havi átlagbevétel:", bevetel.mean(axis=0))
```

---

2. Feladat: Vizsgaeredmények elemzése (axis mély megértése)

Adott egy 8×5-ös mátrix, ahol:

- sorok = diákok
- oszlopok = tantárgyak pontszámai

Feladat:

1. Generálj 8×5-ös véletlen pontszám mátrixot (0–100).
2. Számold ki:
   1. diákonkénti átlagot (soronként)
   2. tantárgyankénti átlagot (oszloponként)
3. Határozd meg:
   1. melyik diák teljesített a legjobban
   2. melyik tantárgy átlaga a legrosszabb
4. Szűrd ki azokat a diákokat, akiknek az átlaga 60 alatt van.
5. Normalizáld a pontokat oszloponként (tantárgyanként).

Kulcs: axis=0, axis=1, argmax, argmin, broadcasting.

```
import numpy as np

np.random.seed(0)
```

```
# 1. Generálás
pontok = np.random.randint(0, 101, (8, 5))
print("Pontok:\n", pontok)
#2. Átlagok
# Diákonként (sorok mentén → axis=1)
diak_atlag = np.mean(pontok, axis=1)

# Tantárgyanként (oszlopok mentén → axis=0)
tantargy_atlag = np.mean(pontok, axis=0)

print("Diák átlag:", diak_atlag)
print("Tantárgy átlag:", tantargy_atlag)

#Miért axis=1 a sor? Mert az aggregálás a második dimenzió mentén történik → soron belül számolunk.

#3. Legjobb diák, legrosszabb tantárgy
legjobb_diak = np.argmax(diak_atlag)
legrosszabb_tantargy = np.argmin(tantargy_atlag)

print("Legjobb diák index:", legjobb_diak)
print("Legrosszabb tantárgy index:", legrosszabb_tantargy)

#4. 60 alatti átlagú diákok
gyenge_diakok = np.where(diak_atlag < 60)[0]
print("60 alatti diákok:", gyenge_diakok)

#5. Oszloponkénti normalizálás
min_vals = np.min(pontok, axis=0)
max_vals = np.max(pontok, axis=0)

normalizalt = (pontok - min_vals) / (max_vals - min_vals)
print("Normalizált:\n", normalizalt)

#Broadcasting automatikusan működik oszloponként.
```

---

3. feladat: Éves hőmérsékleti adatelemzés (idősor mátrix)

Egy 12×30-as mátrix:

- sorok = hónapok
- oszlopok = napok

Feladat:

1. Generálj normál eloszlású hőmérséklet adatokat.
2. Számold ki:
    1. havi átlaghőmérsékletet
    2. napi éves átlaghőmérsékletet
3. Határozd meg:

    1. melyik hónap volt a legmelegebb átlagosan
    2. melyik nap volt az év leghidegebb napja
4. Számold ki, hány nap volt 30°C felett havonta.
5. Standardizáld soronként (z-score normalizálás).

Kulcs: mean(axis=1), mean(axis=0), min, sum(axis=1).

```
#Generálás
np.random.seed(1)
homerseklet = np.random.normal(20, 10, (12, 30))

#Havi és napi átlag
havi_atlag = np.mean(homerseklet, axis=1)
napi_atlag = np.mean(homerseklet, axis=0)

#Legmelegebb hónap
legmelegebb_honap = np.argmax(havi_atlag)

#Leghidegebb nap az évben
leghidegebb_nap = np.unravel_index(np.argmin(homerseklet),
homerseklet.shape)

#30°C feletti napok havonta
forro_napok = np.sum(homerseklet > 30, axis=1)

#Z-score standardizálás soronként
mean = np.mean(homerseklet, axis=1, keepdims=True)
std = np.std(homerseklet, axis=1, keepdims=True)

standardizalt = (homerseklet - mean) / std
```

---

4. feladat: Bevételi mátrix elemzése (összetett aggregáció)

Egy 6×4-es mátrix:

- sorok = hónapok
- oszlopok = üzletek

Feladat:

1. Generálj bevételi adatokat.
2. Számold ki:
    1. üzletenkénti éves bevételt
    2. havi összbevételt
3. Határozd meg:
    1. melyik üzlet teljesített legjobban összesítve
    2. melyik hónap volt a legerősebb

4. Számold ki az egyes üzletek piaci részesedését (%) az éves összbevételből.
5. Skálázd a mátrixot 0–1 közé globálisan.

Kulcs: sum(axis=0), sum(axis=1), argmax, broadcasting.

```
#Generálás
np.random.seed(2)
bevetel = np.random.randint(1000, 5000, (6, 4))

#Éves bevétel üzletenként
uzlet_eves = np.sum(bevetel, axis=0)

#Havi összbevétel
havi_osszes = np.sum(bevetel, axis=1)

#Legjobb üzlet
legjobb_uzlet = np.argmax(uzlet_eves)

#Legjobb hónap
legjobb_honap = np.argmax(havi_osszes)

#Piaci részesedés
osszes_bevetel = np.sum(bevetel)
reszesedes = uzlet_eves / osszes_bevetel * 100

#Globális normalizálás
min_v = np.min(bevetel)
max_v = np.max(bevetel)

skala = (bevetel - min_v) / (max_v - min_v)
```

---

5. feladat: Képmanipulációs alapfeladat (mátrix mint kép)

Egy 100×100-as mátrix egy szürkeárnyalatos képet reprezentál (0–255).

Feladat:

1. Generálj egy ilyen mátrixot.
2. Számold ki:
    1. teljes kép átlagfényességét
    2. soronkénti átlagfényességet
3. Küszöbölés: minden 128 alatti érték legyen 0, felette 255.
4. Normalizáld oszloponként.
5. Számold ki, hány pixel világosabb az átlagfényességnél.

Kulcs: boolean indexelés + mean(axis=...).

```
#Generálás
np.random.seed(3)
kep = np.random.randint(0, 256, (100, 100))

#Átlagfényesség
global_atlag = np.mean(kep)
sor_atlag = np.mean(kep, axis=1)

#Küszöbölés
binary = np.where(kep < 128, 0, 255)

#Oszloponkénti normalizálás
min_col = np.min(kep, axis=0)
max_col = np.max(kep, axis=0)

norm_kep = (kep - min_col) / (max_col - min_col)

#Átlag feletti pixelek száma
vilagos_db = np.sum(kep > global_atlag)
```

---

6. feladat: Többdimenziós adat – Diák × Tantárgy × Félév

Generálj egy 3D tömböt: 2×10×4

- dimenzió = félév
- dimenzió = diák
- dimenzió = tantárgy

1. Számold ki:
    1. diákonkénti átlagot félévenként
    2. tantárgyankénti átlagot teljes évre
2. Határozd meg:
    1. melyik diák javított legtöbbet két félév között
3. Szűrd ki azokat a diákokat, akiknek bármely félévben 50 alatti az átlaguk.
4. Aggregáld az egész rendszert egyetlen globális átlagra.
5. Normalizáld a 3D tömböt félévenként külön-külön.

Kulcs: többtengelyes aggregáció (axis=(...)), mean, sum, any.

```
#Geneálás
np.random.seed(4)
adat = np.random.randint(0, 101, (2, 10, 4))

#Diák átlag félévenként
diak_atlag = np.mean(adat, axis=2) #Eredmény shape: (2, 10)

#Tantárgy átlag teljes évre
```

```
tantargy_atlag = np.mean(adat, axis=(0,1))

#Legnagyobb javulás
javulas = diak_atlag[1] - diak_atlag[0]
legnagyobb_javulas = np.argmax(javulas)

#50 alatti átlag bármely félévben
gyenge = np.any(diak_atlag < 50, axis=0)
gyenge_indexek = np.where(gyenge)[0]

#Globális átlag
global_atlag = np.mean(adat)

#Félévenkénti normalizálás
min_vals = np.min(adat, axis=(1,2), keepdims=True)
max_vals = np.max(adat, axis=(1,2), keepdims=True)

normalizalt = (adat - min_vals) / (max_vals - min_vals)
```

From:
https://edu.iit.uni-miskolc.hu/ - **Institute of Information Science - University of Miskolc**

Permanent link:
https://edu.iit.uni-miskolc.hu/tanszek:oktatas:muszaki_informatika:numpy_ii._statisztika_matrixmuveletek

Last update: **2026/02/27 07:50**