Broadcasting

NumPy'da **Broadcasting** (Türkçe'ye bazen "yayınlama" veya "genişletme" olarak çevrilebilir), farklı şekillere (shape) sahip NumPy dizileri arasında aritmetik işlemler (+, -, *, / vb.) yapılmasına olanak tanıyan çok güçlü bir mekanizmadır.

Normalde, iki NumPy dizisi arasında eleman bazında işlem yapmak için dizilerin şekillerinin **tam olarak aynı** olması beklenir. Ancak bu her zaman pratik değildir (örneğin, bir vektörün tüm satırlarına bir matris eklemek veya bir dizinin tüm elemanlarını bir skalerle çarpmak gibi).

Broadcasting, NumPy'nin bu tür durumlarla **otomatik olarak başa çıkmasını** sağlar. NumPy, daha küçük olan diziyi, bellekte gerçekten kopyalamadan, daha büyük diziyle uyumlu hale gelecek şekilde sanal olarak "genişletir" veya "yayınlar". Bu, hem bellek kullanımını verimli hale getirir hem de kodun daha kısa ve okunaklı olmasını sağlar.

Broadcasting Nasıl Çalışır? (Kurallar)

NumPy, iki dizinin şekillerini karşılaştırarak broadcasting yapılıp yapılamayacağına ve nasıl yapılacağına karar verir. Karşılaştırma, dizilerin şekillerinin **sağdan sola (sondan başlayarak)** elemanları üzerinden yapılır:

1. **Boyut Sayısı Eşitleme:** Eğer iki dizinin boyut sayısı (`ndim`) farklıysa, daha az boyuta sahip olan dizinin şeklinin **başına**, boyut sayıları eşitlenene kadar `1` eklenir.
    - Örneğin, `(2, 3)` şeklindeki bir dizi ile `(3,)` şeklindeki bir dizi karşılaştırılırken, `(3,)` şekli `(1, 3)` olarak ele alınır.
2. **Boyut Uyumluluğu Kontrolü:** Şekiller sağdan sola karşılaştırılırken, her bir boyut çifti için aşağıdaki koşullardan biri geçerli olmalıdır:
    - a) Boyutlar **eşittir**.
    - b) Boyutlardan **biri 1**'dir.
    - Eğer bu koşullar karşılanmazsa, diziler uyumsuz demektir ve NumPy bir `ValueError` hatası verir (örn: `operands could not be broadcast together with shapes ...`).
3. **Genişletme (Yayınlama):** Eğer boyutlar uyumluysa (yani eşitse veya biri 1 ise), karşılaştırma yapılan boyutta 1 olan dizi, diğer dizinin boyutuna uyacak şekilde **sanal olarak genişletilir (kopyalanmadan)**. Yani, boyutu 1 olan

eksendeki değerler, diğer dizinin o eksendeki boyutu kadar tekrarlanıyormuş gibi davranılır.

4. **Sonuç:** İşlem sonucunda ortaya çıkan dizinin şekli, karşılaştırılan iki dizinin her bir eksendeki maksimum boyutlarından oluşur.

Örnekler:

1. **Skaler ve Dizi:**Python

```
import numpy as np
a = np.array([1, 2, 3])  # Şekil: (3,)
b = 5                    # Skaler (Şekil: () gibi düşünülebilir)
c = a + b                # Broadcasting uygulanır
print(f"a: {a.shape} {a}")
print(f"b: (skaler)")
print(f"c = a + b: {c.shape} {c}") # Çıktı: c = a + b: (3,) [6 7 8]
```

   - `b` skaler olduğu için `a`'nın her elemanına eklenir. `b` sanal olarak `[5, 5, 5]` gibi davranır.
   - Şekiller: `(3,)` vs `()` → `(3,)` vs `(1,)` (Kural 1) → Boyutlar uyumlu (3 vs 1 → 1 genişler) → Sonuç `(3,)`.

2. **1D ve 2D Dizi:**Python

```
matris = np.array([[1, 2, 3],
                   [4, 5, 6]]) # Şekil: (2, 3)
vektor = np.array([10, 20, 30]) # Şekil: (3,)
sonuc = matris + vektor
print(f"\nMatris:\n{matris}")
print(f"Vektör: {vektor}")
print(f"Matris + Vektör Sonucu:\n{sonuc}")
# Çıktı:
# [[11 22 33]
#  [14 26 36]]
```

   - Şekiller: `(2, 3)` vs `(3,)`
   - Kural 1: `(3,)` → `(1, 3)` yapılır. Şekiller `(2, 3)` vs `(1, 3)` olur.
   - Sağdan ilk boyut: 3 == 3 (Uyumlu).
   - İkinci boyut: 2 vs 1 (Uyumlu, 1 olan genişletilecek).
   - `vektor` (şimdi `(1, 3)` şeklinde), matrisin her satırına (`(2, 3)` şekline uyacak şekilde) eklenir. `[[10, 20, 30]]` dizisi 2 satıra genişletilmiş gibi davranır. Sonuç şekli `(2, 3)` olur.

3. **Farklı Yönlerde Broadcasting:**Python

```
a = np.array([[0], [10], [20]]) # Şekil: (3, 1)
b = np.array([1, 2, 3])      # Şekil: (3,)
c = a + b
```

```
print(f"\na:\n{a}")
print(f"b: {b}")
print(f"a + b:\n{c}")
# Çıktı:
# [[ 1  2  3]
#  [11 12 13]
#  [21 22 23]]
```

- Şekiller: `(3, 1)` vs `(3,)`
- Kural 1: `(3,)` → `(1, 3)`. Şekiller `(3, 1)` vs `(1, 3)` olur.
- Sağdan ilk boyut: 1 vs 3 (Uyumlu, 1 olan genişletilecek). `a`'nın sütunu 3 kez tekrarlanır.
- İkinci boyut: 3 vs 1 (Uyumlu, 1 olan genişletilecek). `b`'nin satırı 3 kez tekrarlanır.
- Sonuç şekli `(3, 3)` olur.

4. **Uyumsuz Şekiller (Hata Örneği):**Python

```
a = np.array([[1, 2], [3, 4]]) # Şekil: (2, 2)
b = np.array([1, 2, 3])       # Şekil: (3,)
try:
    c = a + b
except ValueError as e:
    print(f"\nHata: {e}")
# Çıktı: Hata: operands could not be broadcast together with shapes (2,2) (3,)
```

- Şekiller: `(2, 2)` vs `(3,)`
- Kural 1: `(3,)` → `(1, 3)`. Şekiller `(2, 2)` vs `(1, 3)`.
- Sağdan ilk boyut: 2 vs 3 (Uyumsuz! Ne eşitler ne de biri 1). Hata alınır.

Broadcasting'in Faydaları:

- **Bellek Verimliliği:** Dizileri eşleştirmek için büyük kopyalar oluşturmaktan kaçınır.
- **Hız:** Operasyonlar hala optimize edilmiş C kodunda çalışır.
- **Kod Kısalığı:** Manuel döngüler veya `tile` gibi fonksiyonlarla diziyi genişletme ihtiyacını ortadan kaldırır, kodu daha okunaklı yapar.

Broadcasting, NumPy'nin temel taşlarından biridir ve verimli ve etkili NumPy kodu yazmak için anlaşılması önemlidir.