Consulta de vectores y matrices en NumPy

Alejandra Díaz Navarro - 2240063

Marzo 3, 2025

1 ¿Qué es NumPy?

NumPy (Num erical Python) es una biblioteca de código abierto de Python que se utiliza ampliamente en ciencia e ingeniería. La biblioteca NumPy contiene estructuras de datos de matriz multidimensionales, como la homogénea, la N-dimensional ndarrayy una gran biblioteca de funciones que operan de manera eficiente en estas estructuras de datos.

1.1 Ventajas de NumPy sobre las listas de Python

- **Mayor velocidad:** NumPy está optimizado y escrito en C, lo que lo hace más rápido que las listas estándar de Python.
- **Menor consumo de memoria:** Usa estructuras eficientes para almacenar datos numéricos.
- **Operaciones vectorizadas:** Se pueden realizar cálculos en múltiples elementos sin necesidad de bucles.
- **Funciones matemáticas avanzadas:** Soporta álgebra lineal, estadísticas y transformadas de Fourier.

2 Creación de Arrays

NumPy introduce el tipo de datos `ndarray`, que es un array N-dimensional. Se pueden crear de varias formas:

2.1 Crear un array a partir de una lista

```
import numpy as np

# Crear un array a partir de una lista
a = np.array([1, 2, 3, 4, 5])
print(a)  # Salida: [1 2 3 4 5]
```

2.2 Arrays de ceros, unos y aleatorios

```
# Crear un array de ceros
ceros = np.zeros((3,3))

# Crear un array de unos
unos = np.ones((2,4))

# Crear una matriz con valores aleatorios entre 0 y 10
aleatorio = np.random.randint(0, 10, (3,3))
```

3 Operaciones Matemáticas en Arrays

NumPy permite realizar operaciones matemáticas de manera eficiente sin necesidad de usar bucles:

```
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

# Operaciones b sicas
suma = a + b  # [5 7 9]
resta = a - b  # [-3 -3 -3]
producto = a * b  # [4 10 18]
division = a / b  # [0.25 0.4 0.5]
```

4 Funciones Estadísticas con NumPy

NumPy ofrece funciones útiles para obtener información estadística de arrays:

```
datos = np.array([10, 20, 30, 40, 50])

media = np.mean(datos)   # Promedio
mediana = np.median(datos)  # Mediana
desviacion = np.std(datos)  # Desviaci n est ndar
suma_total = np.sum(datos)  # Suma de todos los elementos
maximo = np.max(datos)  # Valor m ximo
minimo = np.min(datos)  # Valor m nimo
```

5 Funciones para Manipulación de Arrays

```
arr = np.array([[1, 2, 3], [4, 5, 6]])

# Cambiar forma de un array
arr_reshaped = arr.reshape((3,2))

# Concatenaci n de arrays
concat_horizontal = np.hstack((arr, arr))
concat_vertical = np.vstack((arr, arr))
```

```
# Divisi n de arrays
split = np.split(arr, 2, axis=1)

# Ordenar elementos
ordenado = np.sort(arr, axis=0)
```

6 Creación de Matrices

NumPy permite crear matrices de diferentes maneras:

6.1 Matriz a partir de una lista anidada

```
import numpy as np

# Crear una matriz 3x3 con listas anidadas
matriz = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print(matriz)
```

6.2 Matrices de ceros y unos

```
# Matriz de ceros de 3x3
ceros = np.zeros((3,3))

# Matriz de unos de 2x4
unos = np.ones((2,4))
```

6.3 Matriz identidad

```
# Matriz identidad de tama o 4x4
identidad = np.eye(4)
```

6.4 Matriz con valores aleatorios

```
# Matriz de 3x3 con valores aleatorios entre 0 y 10
aleatoria = np.random.randint(0, 10, (3,3))
```

7 Operaciones con Matrices

Las matrices en NumPy permiten realizar operaciones matemáticas fácilmente.

7.1 Suma y Resta

```
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])

# Suma de matrices
suma = A + B

# Resta de matrices
resta = A - B
```

7.2 Multiplicación de matrices

NumPy permite dos tipos de multiplicación:

- Elemento a elemento (`*`)
- Producto matricial (`@` o `np.dot()`)

```
# Multiplicaci n elemento a elemento
producto_elemento = A * B

# Producto matricial
producto_matriz = A @ B  # Tambi n se puede usar np.dot(A, B)
```

7.3 Transposición de una matriz

```
# Transponer una matriz
A_transpuesta = A.T
```

7.4 Inversa de una matriz

```
# Inversa de una matriz
A_inversa = np.linalg.inv(A)
```

7.5 Determinante de una matriz

```
# Determinante de una matriz
determinante = np.linalg.det(A)
```