Guia do Iniciante para NumPy

Tempo Estimado : 10 Minutos

Objetivo:

Nesta leitura, você aprenderá:

- Fundamentos do NumPy
- Como criar arrays NumPy
- Atributos e indexação de arrays
- Operações básicas como adição e multiplicação

O que é NumPy?

NumPy, abreviação de **Num**erical **Py**thon, é uma biblioteca fundamental para computação numérica e científica em Python. Ela oferece suporte para grandes arrays e matrizes multidimensionais, juntamente com uma coleção de funções matemáticas de alto nível para operar nesses arrays. NumPy serve como a base para muitas bibliotecas de ciência de dados e aprendizado de máquina, tornando-se uma ferramenta essencial para análise de dados e pesquisa científica em Python.

Aspectos-chave do NumPy em Python:

- **Estruturas de dados eficientes**: O NumPy introduz estruturas de array eficientes, que são mais rápidas e mais econômicas em termos de memória do que listas do Python. Isso é crucial para lidar com grandes conjuntos de dados.
- **Arrays multi-dimensionais**: O NumPy permite trabalhar com arrays multi-dimensionais, possibilitando a representação de matrizes e tensores. Isso é particularmente útil em computação científica.
- **Operações elemento a elemento**: O NumPy simplifica operações matemáticas elemento a elemento em arrays, facilitando a realização de cálculos em conjuntos de dados inteiros de uma só vez.
- **Geração de números aleatórios**: Ele fornece uma ampla gama de funções para gerar números aleatórios e dados aleatórios, o que é útil para simulações e análises estatísticas.
- **Integração com outras bibliotecas**: O NumPy se integra perfeitamente com outras bibliotecas de ciência de dados como SciPy, Pandas e Matplotlib, aumentando sua utilidade em várias áreas.
- **Otimização de desempenho**: As funções do NumPy são implementadas em linguagens de baixo nível como C e Fortran, o que aumenta significativamente seu desempenho. É uma escolha ideal quando a velocidade é essencial.

Instalação

Se você ainda não instalou o NumPy, pode fazê-lo usando `pip`:

```
pip install numpy
```

Criando arrays NumPy

Você pode criar arrays NumPy a partir de listas Python. Esses arrays podem ser unidimensionais ou multidimensionais.

Criando um array 1D

```
import numpy as np
```

import numpy as np: Nesta linha, a biblioteca NumPy é importada e atribuída um alias `np` para facilitar a referência no código.

```
# Creating a 1D array
arr_1d = np.array([1, 2, 3, 4, 5]) # **np.array()** is used to create NumPy arrays.
```

arr_1d = np.array([1, 2, 3, 4, 5]): Nesta linha, um array unidimensional do NumPy chamado `arr_1d` é criado. Ele utiliza a função `np.array()` para converter uma lista Python [1, 2, 3, 4, 5] em um array do NumPy. Este array contém cinco elementos, que são 1, 2, 3, 4 e 5. `arr_1d` é um array 1D porque possui uma única linha de elementos.

Criando um array 2D

```
import numpy as np
```

import numpy as np: Nesta linha, a biblioteca NumPy é importada e atribuída um alias `np` para facilitar a referência no código.

```
# Creating a 2D array
arr_2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
```

arr_2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]): Nesta linha, um array NumPy bidimensional chamado `arr_2d` é criado. Ele utiliza a função `np.array()` para converter uma lista de listas em um array NumPy 2D. A lista externa contém três listas internas, cada uma representando uma linha de elementos. Assim, `arr_2d` é um array 2D com três linhas e três colunas. Os elementos deste array formam uma matriz com valores de 1 a 9, organizados em uma grade 3x3.

Atributos de array

Os arrays do NumPy possuem vários atributos úteis:

```
# Array attributes
print(arr_2d.ndim)  # ndim : Represents the number of dimensions or "rank" of the array.
# output : 2
print(arr_2d.shape)  # shape : Returns a tuple indicating the number of rows and columns in the array.
# Output : (3, 3)
print(arr_2d.size) # size: Provides the total number of elements in the array.
# Output : 9
```

Indexação e fatiamento

Você pode acessar elementos de um array NumPy usando indexação e fatiamento:

Nesta linha, o terceiro elemento (índice 2) do array 1D `arr_1d` é acessado.

```
# Indexing and slicing
print(arr_1d[2])          # Accessing an element (3rd element)
```

Nesta linha, o elemento na 2ª linha (índice 1) e 3ª coluna (índice 2) do array 2D `arr_2d` é acessado.

```
print(arr_2d[1, 2])       # Accessing an element (2nd row, 3rd column)
```

Nesta linha, a 2ª linha (índice 1) do array 2D `arr_2d` é acessada.

```
print(arr_2d[1])          # Acessando uma linha (2ª linha)
```

```
In this line, the 2nd column (index 1) of the 2D array `arr_2d` is accessed.
```python
print(arr_2d[:, 1])       # Accessing a column (2nd column)
```

Operações básicas

NumPy simplifica operações básicas em arrays:

Operações aritméticas elemento a elemento:

Adição, subtração, multiplicação e divisão de arrays com escalares ou outros arrays.

Adição de Array

```
# Array addition
array1 = np.array([1, 2, 3])
array2 = np.array([4, 5, 6])
result = array1 + array2
print(result)  # [5 7 9]
```

Multiplicação escalar

```
# Scalar multiplication
array = np.array([1, 2, 3])
result = array * 2 # each element of an array is multiplied by 2
print(result)  # [2 4 6]
```

Multiplicação elemento a elemento (Produto de Hadamard)

```
# Element-wise multiplication (Hadamard product)
array1 = np.array([1, 2, 3])
array2 = np.array([4, 5, 6])
result = array1 * array2
print(result)  # [4 10 18]
```

Multiplicação de matrizes

```
# Matrix multiplication
matrix1 = np.array([[1, 2], [3, 4]])
matrix2 = np.array([[5, 6], [7, 8]])
result = np.dot(matrix1, matrix2)
print(result)
# [[19 22]
#  [43 50]]
```
```

NumPy simplifica essas operações, tornando-as mais fáceis e eficientes do que as listas tradicionais do Python.

Operações com NumPy

Aqui está a lista de operações que podem ser realizadas usando Numpy

Operação	Descrição	Exemplo
Criação de Array	Criando um array NumPy.	`arr = np.array([1, 2, 3, 4, 5])`
Aritmética Elementar	Adição, subtração, etc., elemento a elemento.	`result = arr1 + arr2`
Aritmética Escalar	Adição, subtração, etc., escalar.	`result = arr * 2`
Funções Elementares	Aplicando funções a cada elemento.	`result = np.sqrt(arr)`
Soma e Média	Calculando a soma e a média de um array.	`total = np.sum(arr) average = np.mean(arr)`
Valores Máximos e Mínimos	Encontrando os valores máximos e mínimos.	`max_val = np.max(arr) min_val = np.min(arr)`
Redimensionamento	Mudando a forma de um array.	`reshaped_arr = arr.reshape(2, 3)`
Transposição	Transpondo um array multidimensional.	`transposed_arr = arr.T`
Multiplicação de Matrizes	Realizando a multiplicação de matrizes.	`result = np.dot(matrix1, matrix2)`

Conclusão

NumPy é uma biblioteca fundamental para ciência de dados e cálculos numéricos. Este guia cobre o básico do NumPy, e há muito mais para explorar. Visite numpy.org para mais informações e exemplos.

Autor

Akansha Yadav

Registro de Mudanças

Data	Versão	Alterado por	Descrição da Mudança
2023-10-12	1.0	Akansha Yadav	Leitura criada inicialmente
2023-02-16	2.0	Gagandeep Singh	Revisão de ID
2024-02-21	2.1	Mercedes Schneider	Revisão de QA com edições