NumPy

Introdução

Matemática com arrays NumPy

NumPy

Matemática com arrays NumPy

Vetorização é como chamamos o recurso de expressar operações em lote nos dados de um array sem precisar escrever qualquer laço for.

As operações matemáticas entre arrays de mesmo tamanho são aplicadas em todos os elementos do array.

NumPy

Matemática com arrays NumPy

Os arrays permitem realizar operações matemáticas em blocos inteiros de dados usando uma sintaxe semelhante às operações equivalentes entre elementos escalares.

```
In [21]: myList = [[1,2,3],[4,5,6],[7,8,9]]
         myArray = np.array(myList)
         print(f"myArray: {myArray}")

         multiplica = myArray * 10
         print(f"multiplica: {multiplica}")

         soma = myArray + 1
         print(f"soma: {soma}")
```

```
myArray: [[1 2 3]
 [4 5 6]
 [7 8 9]]
multiplica: [[10 20 30]
 [40 50 60]
 [70 80 90]]
soma: [[ 2  3  4]
 [ 5  6  7]
 [ 8  9 10]]
```

NumPy

Matemática com arrays NumPy

Quando efetuamos operações com um valor, este valor é aplicado a todos elementos do array. Quando efetuamos operações com outro array, os elementos do array são utilizados em cada posição.

```
Cmder
In [1]: import numpy as np

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

In [3]: a * 2
Out[3]:
array([[ 2.,  4.,  6.],
       [ 8., 10., 12.]])

In [4]: a - 1
Out[4]:
array([[0., 1., 2.],
       [3., 4., 5.]])

In [5]: a / 2
Out[5]:
array([[0.5, 1. , 1.5],
       [2. , 2.5, 3. ]])
```

```
In [17]: a + 3
Out[17]:
array([[4., 5., 6.],
       [7., 8., 9.]])
```

```
Cmder
In [22]: a = np.array([[1.,2.,3.],[4.,5.,6.]])

In [23]: b = np.array([[3.,3.,3.],[3.,3.,3.]])

In [24]: a * b
Out[24]:
array([[ 3.,  6.,  9.],
       [12., 15., 18.]])

In [25]: a - b
Out[25]:
array([[-2., -1.,  0.],
       [ 1.,  2.,  3.]])

In [26]: a / b
Out[26]:
array([[0.33333333, 0.66666667, 1.        ],
       [1.33333333, 1.66666667, 2.        ]])
```

```
In [27]: a + b
Out[27]:
array([[4., 5., 6.],
       [7., 8., 9.]])
```

```
In [28]: a ** 2
Out[28]:
array([[ 1.,  4.,  9.],
       [16., 25., 36.]])

In [29]: a ** b
Out[29]:
array([[  1.,   8.,  27.],
       [ 64., 125., 216.]])
```

NumPy

Matemática com arrays NumPy

As comparações entre arrays de mesmo tamanho resultam em arrays booleanos.

```
In [30]: a = np.array([[1.,2.,3.],[4.,5.,6.]])

In [31]: b = np.array([[1.,6.,3.],[2.,5.,9.]])

In [32]: a > b
Out[32]:
array([[False, False, False],
       [ True, False, False]])

In [33]: a < b
Out[33]:
array([[False,  True, False],
       [False, False,  True]])
```

FIM