NumPy sort()

Summary: in this tutorial, you'll learn how to use the numpy `sort()` function to sort elements of an array.

Introduction to the NumPy sort() function

The `sort()` function returns a sorted copy of an array (https://www.pythontutorial.net/python-numpy/create-numpy-array/) . Here's the syntax of the `sort()` function:

```
numpy.sort(a, axis=- 1, kind=None, order=None)
```

In this syntax:

- `a` is a numpy array to be sorted. Also, it can be any object that can be converted to an array.
- `axis` specifies the axis along which the elements will be sorted. If the axis is None, the function flattens the array before sorting. By default, the axis is -1 which sorts elements along the last axis.
- `kind` specifies the sorting algorithm which can be 'quicksort', 'mergesort', 'heapsort', and 'stable'.

- `order` specifies which fields to compare first, second, etc when sorting an array with fields defined. The `order` can be a string that represents the field to sort or a list of strings that represent a list of fields to sort.

If you want to sort the elements of an array in place, you can use the `sort()` method of the `ndarray` object with the following syntax:

```
ndarray.sort(axis=- 1, kind=None, order=None)
```

NumPy sort() function examples

Let's take some examples of using the NumPy `sort()` function.

1) Using the sort() function to sort a 1-D array

The following example uses the `sort()` function to sort numbers in a 1-D array:

```
import numpy as np

a = np.array([2, 3, 1])
b = np.sort(a)
print(b)
```

Output:

```
[1 2 3]
```

In this example, the `sort()` function sorts the elements of the array from low to high.

To sort elements of an array from high to low, you can use the `sort()` function to sort an array from low to high and use a slice to reverse the array. For example:

```
import numpy as np

a = np.array([2, 3, 1])
```

```
b = np.sort(a)[::-1]
print(b)
```

Output:

```
[3 2 1]
```

In this example:

- First, the `sort()` function sorts the elements in the array `a` in ascending order (from low to high)
- Then, the slice [::-1] reverses the sorted array so that the elements of the result array are in descending order.

2) Using the numpy sort() function to sort a 2-D array example

The following example uses the `sort()` funciton to sort a 2-D array:

```
import numpy as np

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

b = np.sort(a)
print(b)
```

Output:

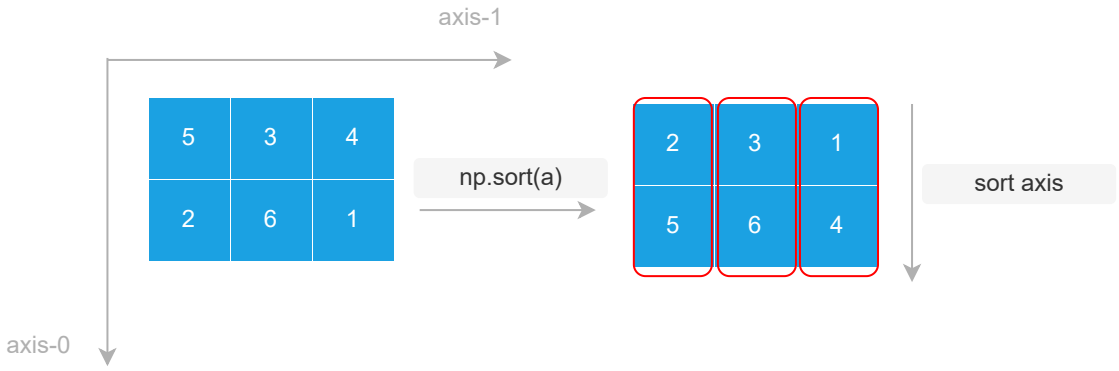
```
[[1 2 3]
 [4 5 6]]
```

The following example uses the `sort()` function to sort elements on axis 0:

```
import numpy as np

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

b = np.sort(a, axis=0)
print(b)
```



Output:

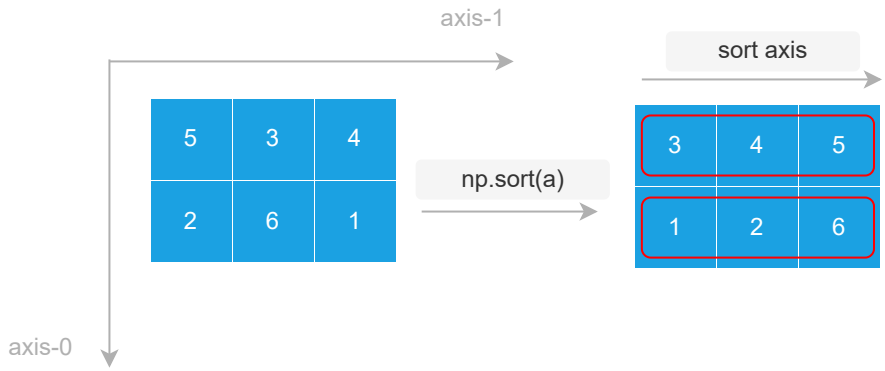
```
[[2 3 1]
 [5 6 4]]
```

Similarly, you can sort elements of the array on axis 1:

```
import numpy as np

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

b = np.sort(a, axis=1)
print(b)
```



Output:

```
[[3 4 5]
 [1 2 6]]
```

3) Using the numpy sort() function to sort a structured array example

The following example sorts the employees by year of services and then salary:

```
import numpy as np

dtype = [('name', 'S10'),
         ('year_of_services', float),
         ('salary', float)]

employees = [
    ('Alice', 1.5, 12500),
    ('Bob', 1, 15500),
    ('Jane', 1, 11000)
]

payroll = np.array(employees, dtype=dtype)

result = np.sort(
    payroll,
    order=['year_of_services', 'salary']
)
```

```
print(result)
```

Output:

```
[(b'Jane', 1. , 11000.) (b'Bob', 1. , 15500.) (b'Alice', 1.5, 12500.)]
```

Summary

- Use the numpy `sort()` function to sort elements of an array.