numpy.where()

17 August 2025 03:34 PM

We will explore the basics of numpy.where(), how it works, and practical use cases to illustrate its importance in data manipulation and analysis.

Syntax of numpy.where()

Syntax *:numpy.where(condition[, x, y])*

Parameters

- ***condition: A condition that tests elements of the array.***
- *x (optional): Values from this array will be returned where the condition is True.*
- *y (optional): Values from this array will be returned where the condition is False.*

Returns*: If only the condition is provided, numpy.where() will return the indices of elements where the condition is true.*

Table of Content

Uses of Numpy Where

The numpy.where() function is a powerful tool in the NumPy library used for conditional selection and manipulation of arrays. This function enables you to search, filter, and apply conditions to elements of an array, returning specific elements based on the condition provided.

Basic Usage Without x and y

If only the condition is provided, numpy.where() returns the indices of elements that meet the condition.

In this example, numpy.where() checks where the condition arr > 20 is true, and returns the indices [3, 4], meaning the elements at index 3 and 4 (25 and 30) are greater than 20.

```
1  import numpy as np
2  # Create an array
3  arr = np.array([10, 15, 20, 25, 30])
4  # Use np.where to find indices of element greater than 20
5  result=np.where(arr > 20)
6  print(result)
✓ 0.1s
(array([3, 4]),)
```

```
import numpy as np
# Create an array
arr = np.array([10, 15, 20, 25, 30])
# Use np.where() to find indices of elements greater than 20
result = np.where(arr > 20)
print(result)
```

Output:

```
(array([3, 4]),)
```

Using numpy.where() with x and y

By providing x and y as arguments, you can use numpy.where() to return different values depending on whether the condition is true or false.

Here, the numpy.where() function checks the condition arr > 20. For elements that meet the condition, it returns 1, and for those that don't, it returns 0. This is a common technique to apply binary masks in arrays.

Using numpy.where() with x and y

```
1  import numpy as np
2  # Create an array
3  arr = np.array([10, 15, 20, 25, 30])
4  # use np.where to replace values based on condition
5  # if the value is greater than 20, return 1 other wise 0
6  result=np.where(arr > 20, 1, 0)
7  print(result)
✓ 0.0s
[0 0 0 1 1]
```

```
import numpy as np
# Create an array
arr = np.array([10, 15, 20, 25, 30])
# Use np.where() to replace values based on condition
# If the value is greater than 20, return 1, otherwise return 0
result = np.where(arr > 20, 1, 0)
print(result)
```

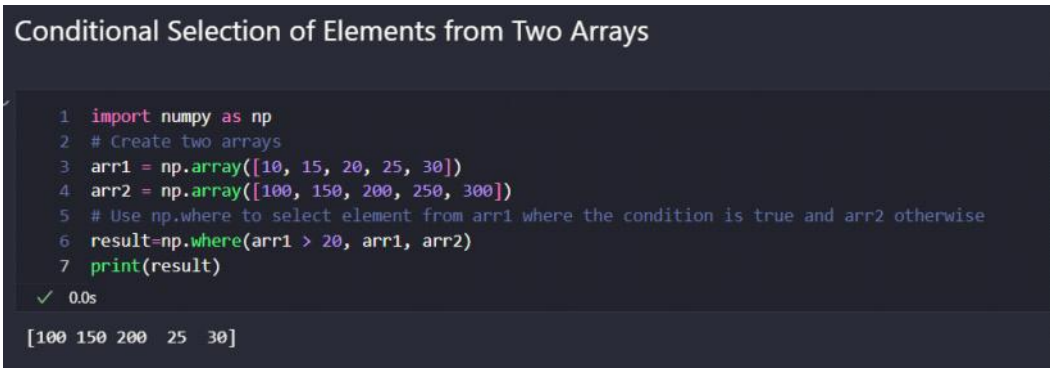
Output:

```
[0 0 0 1 1]
```

Conditional Selection of Elements from Two Arrays

You can also use numpy.where() to choose elements from two different arrays depending on a condition.

In this example, for elements where the condition arr1 > 20 is true, the corresponding element from arr1 is chosen. Otherwise, the element from arr2 is selected.




```
import numpy as np
# Create two arrays
arr1 = np.array([10, 15, 20, 25, 30])
arr2 = np.array([100, 150, 200, 250, 300])
# Use np.where() to select elements from arr1 where the condition is true, and arr2 otherwise
result = np.where(arr1 > 20, arr1, arr2)
print(result)
```

Output:

```
[100 150 200  25  30]
```

From <https://www.geeksforgeeks.org/numpy/numpy-where-in-python/>