Python Numpy

What is NumPy?

NumPy is a powerful library for numerical computing in Python. It provides support for arrays, matrices, and many mathematical functions to operate on these data structures efficiently.

Installation

You can install NumPy using pip:

bash
```
pip install numpy
```

Importing NumPy

To use NumPy, you need to import it into your script:

```
import numpy as np
```

Creating Arrays

1. Creating a 1D Array

```
import numpy as np

# Creating a 1D array
arr = np.array([1, 2, 3, 4, 5])
print(arr)
```

2. Creating a 2D Array

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

3. Using Built-in Functions

```
# Array of zeros
zeros = np.zeros((3, 3))
print(zeros)

# Array of ones
ones = np.ones((2, 4))
print(ones)

# Array of a specific shape filled with a specific value
full = np.full((3, 3), 7)
print(full)

# Array with a range of values
range_arr = np.arange(0, 10, 2)
print(range_arr)

# Array with evenly spaced numbers over a specified
interval
linspace_arr = np.linspace(0, 1, 5)
print(linspace_arr)
```

Array Operations

1. Element-wise Operations

```
# Element-wise addition
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
c = a + b
print(c)

# Element-wise multiplication
d = a * b
```

```
print(d)
```

2. Mathematical Functions

```
# Applying mathematical functions
arr = np.array([0, np.pi/2, np.pi])

sin_values = np.sin(arr)
print(sin_values)

exp_values = np.exp(arr)
print(exp_values)
```

Array Indexing and Slicing

1. Indexing

```
# Accessing elements in a 1D array
arr = np.array([1, 2, 3, 4, 5])
print(arr[0]) # Output: 1
print(arr[-1]) # Output: 5

# Accessing elements in a 2D array
arr_2d = np.array([[1, 2, 3], [4, 5, 6]])
print(arr_2d[0, 0]) # Output: 1
print(arr_2d[1, 2]) # Output: 6
```

2. Slicing

```
# Slicing a 1D array
arr = np.array([1, 2, 3, 4, 5])
print(arr[1:4]) # Output: [2, 3, 4]

# Slicing a 2D array
```

```
arr_2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print(arr_2d[0:2, 1:3]) # Output: [[2, 3], [5, 6]]
```

Array Manipulation

1. Reshaping Arrays

```
# Reshaping an array
arr = np.array([1, 2, 3, 4, 5, 6])
reshaped_arr = np.reshape(arr, (2, 3))
print(reshaped_arr)
```

2. Concatenation

```
# Concatenating arrays
a = np.array([1, 2])
b = np.array([3, 4])
concat_arr = np.concatenate((a, b))
print(concat_arr)

# Concatenating 2D arrays along rows (axis=0)
a_2d = np.array([[1, 2], [3, 4]])
b_2d = np.array([[5, 6]])
concat_2d = np.concatenate((a_2d, b_2d), axis=0)
print(concat_2d)

# Concatenating 2D arrays along columns (axis=1)
concat_2d_cols = np.concatenate((a_2d, b_2d.T), axis=1)
print(concat_2d_cols)
```

Statistical Operations

1. Basic Statistics

```
# Basic statistical operations
arr = np.array([1, 2, 3, 4, 5])

mean = np.mean(arr)
print("Mean:", mean)

std_dev = np.std(arr)
print("Standard Deviation:", std_dev)

sum_arr = np.sum(arr)
print("Sum:", sum_arr)
```

2. Aggregation Functions

```
# Aggregation functions
arr_2d = np.array([[1, 2, 3], [4, 5, 6]])

sum_axis0 = np.sum(arr_2d, axis=0)
print("Sum along axis 0:", sum_axis0)

sum_axis1 = np.sum(arr_2d, axis=1)
print("Sum along axis 1:", sum_axis1)
```

Random Numbers

1. Generating Random Numbers

```
# Generating random numbers
random_arr = np.random.rand(3, 3)
print(random_arr)

# Generating random integers
random_ints = np.random.randint(0, 10, (3, 3))
print(random_ints)
```

```
# Setting a seed for reproducibility
np.random.seed(42)
random_arr_seeded = np.random.rand(3, 3)
print(random_arr_seeded)
```

Features of Numpy

NumPy contains many features, some of which are listed below:

- NumPy is a combination of Python and the C language as it is partially written in Python, and most of its parts are written in C or C++.
- The object-oriented approach is also fully supported by Numpy.
- NumPy functions can be used to work with code that is written in other programming languages and provides tools for integrating with languages such as C, Fortran, etc.
- NumPy is an open-source core Python package for scientific computing.
- NumPy uses less space and stores data in contiguous memory.
- It offers a multidimensional array object with excellent performance as well as methods for working with these arrays.
- Arrays can be reshaped into different dimensions using the reshape function provided by NumPy.
- We can work with different data types using NumPy and can determine the type of data using the dtype function.
- NumPy comes with a plethora of built-in functions such as sum, sort, max, and so on, allowing users to write fewer lines of code while improving the quality of their work.
- NumPy provides a broadcasting technique by which we can perform arithmetic operations on arrays of different shapes.
- NumPy with SciPy is also used as an alternative to MATLAB.

Applications of Numpy

- Numpy arrays are used as an alternative for Python lists.
- Numpy in Python is used for performing mathematical operations on Multidimensional arrays.
- Numpy is used with different libraries like Scipy, Pandas, Tkinter, etc.
- Numpy is also used for reshaping the arrays called Broadcasting for performing operations on different sized arrays.
- Scipy with Numpy in Python can be used in place of MATLAB.