NumPy: Numerical Python

THE POWERHOUSE FOR NUMERICAL COMPUTING IN PYTHON

What is NumPy?

- NumPy stands for Numerical Python.
- It provides fast and efficient operations on multidimensional arrays.
- It is the foundation for scientific computing in Python.
- Key features: arrays, broadcasting, linear algebra, and more.

What is array?

- An **array** is a **data structure** that stores a collection of items, usually of the same data type, in an organized and structured way. In programming, arrays are commonly used to store sequences of numbers, strings, or other objects.

Arrays in Python (NumPy)

- In Python, the NumPy library provides the ndarray object, which is a powerful array structure for numerical computing.
- **Why NumPy Arrays?**
- **Faster**: NumPy arrays are much faster than Python lists for numerical operations.
- **More Functional**: NumPy arrays support element-wise operations, broadcasting, and advanced features like reshaping and slicing.

Why Are Arrays Useful?

- **Organized Storage**: Arrays help organize data logically.
- **Efficient Computation**: Great for mathematical and data-heavy tasks like machine learning, scientific computing, and image processing.
- **Flexibility**: Arrays support slicing, indexing, reshaping, and many other operations.

Key Features of NumPy

- Multidimensional arrays (ndarray).
- Mathematical operations on arrays.
- Broadcasting for operations on arrays of different shapes.
- Tools for random number generation, linear algebra, and Fourier transforms.
- High performance due to its C-based implementation.

Creating Arrays

- Using np.array():

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

- Predefined arrays:

```
np.zeros((2, 3))  # Array of zeros
np.ones((3, 3))   # Array of ones
np.empty((2, 2))  # Uninitialized array
```

- Generating sequences:

```
np.arange(0, 10, 2)  # [0, 2, 4, 6, 8]
np.linspace(0, 1, 5) # [0., 0.25, 0.5, 0.75, 1.]
```

Indexing and Slicing

- Access elements using indices:

```
arr[0], arr[1]
```

- Slice arrays:

```
arr[1:4], arr[:, 1]
```

- Example:

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

Broadcasting

- Perform operations on arrays of different shapes.
- Example:

```
arr = np.array([[1, 2], [3, 4]])
print(arr + 10)  # Add 10 to every element
```

- Eliminates the need for explicit loops.

Mathematical Operations

- Element-wise operations:

```
arr1 + arr2, arr1 * arr2
```

- Aggregations:

```
arr.sum(), arr.mean(), arr.max(), arr.min()
```

- Example:

```
arr = np.array([1, 2, 3])
print(arr.sum())  # Output: 6
```

Reshaping and Transposing

- Reshape arrays:

```
arr.reshape((2, 3))
```

- Transpose arrays:

```
arr.T
```

- •Example:

```
arr = np.array([[1, 2], [3, 4]])
print(arr.T)  # [[1, 3], [2, 4]]
```

Random Number Generation

- Generate random numbers:

```
np.random.default_rng().random((2, 2))
```

- Example:

```
rng = np.random.default_rng()
print(rng.random((2, 2)))
```

Conclusion

- NumPy is a powerful tool for numerical computations.
- It provides fast, efficient, and flexible array operations.
- Widely used in data analysis, machine learning, and scientific computing