Task 2: NumPy

1. What is vectorization in NumPy?

Vectorization in NumPy refers to applying operations on entire arrays without using explicit loops. Vectorization is important because it:

- Improves Performance: Eliminates Python-level loops and leverages fast low-level implementations.
- Produces Cleaner Code: Fewer lines, easier to maintain.
- Scales Better: Can efficiently handle large scientific data and machine learning workloads.

Example 1: Add a number to each element

Performs element-wise addition across the entire array without using loops, making the operation fast and efficient.

```
import numpy as np

a1 = np.array([2, 4, 6, 8, 10])
num = 2
res = a1 + num
print(res)
```

Output

```
[ 4  6  8 10 12]
```

2. What is the difference between reshape() and resize()?

Both the numpy.reshape() and numpy.resize() methods are used to change the size of a NumPy array. The difference between them is that the reshape() does not changes the original array but only returns the changed array, whereas the resize() method returns nothing and directly changes the original array.

3. What are NumPy dimensions and axes?

In NumPy, dimensions and axes describe the shape and direction of data stored in an array.

Dimensions (ndim)

- A dimension refers to the number of levels (or nested layers) of data in an array.

- It tells how many directions are needed to locate an element.
- Also called the rank of an array.
- Given by the attribute ndim.
  Examples in words:
- 1D array → single line of elements
- 2D array → rows and columns (like a table)
- 3D array → multiple 2D tables stacked together

Axes

- An axis represents a specific direction along which data is arranged or operated on.
- Axes are numbered starting from 0.
- Each axis corresponds to one dimension.
  For a 2D array:
- Axis 0 → vertical direction (rows)
- Axis 1 → horizontal direction (columns)
  For higher dimensions:
- Axis numbers increase as dimensions increase.
- Operations like sum, mean, or max can be applied along a specific axis.

4. What is slicing in NumPy arrays?

Slicing in NumPy is a way to extract a portion (subset) of an array by specifying a range of indices.It allows you to access:
Specific rows or columns
Subarrays from multidimensional arrays
Continuous blocks of data efficiently
Important Characteristics

- **Zero-based indexing**
  Indexing starts from 0.
- **End index is excluded**
  The stop index is not included in the result.
- **Works across dimensions**
  Each dimension can be sliced independently.
- **Returns a view, not a copy**
  Changes to the sliced array affect the original array (important difference from Python lists).

5. How does NumPy help in mathematical computations for AI?

NumPy is a core library for AI and machine learning because it provides fast, efficient, and expressive numerical computation.

1. Efficient Handling of Large Data

   AI models work with large datasets (vectors, matrices, tensors). NumPy stores data in contiguous memory blocks, making access and computation much faster than native Python structures.

2. Fast Mathematical Operations

   NumPy performs computations using optimized C and Fortran libraries under the hood.This makes operations like addition, multiplication, dot products, and matrix inversion significantly faster, which is critical for training AI models.

3. Linear Algebra Support

   AI heavily relies on linear algebra. NumPy provides built-in support for:

   - Matrix multiplication
   - Transpose
   - Determinants
   - Eigenvalues and eigenvectors
   - Solving linear equations

   These are essential for algorithms like regression, neural networks, and dimensionality reduction.

4. Foundation for AI Libraries

   Most AI and ML libraries are built on top of NumPy, including:

   - Data preprocessing tools
   - Scientific computing libraries
   - Deep learning frameworks (internally)

   Understanding NumPy makes it much easier to learn and use these advanced AI tools.