Week-1

Day-5 (Assignment)

Topic: Advance Numpy

Task: Build a NumPy-Based Mini ML Engine for Linear Regression with Synthetic Data.

Objective:

Create a small Python app using NumPy that generates synthetic data, performs linear regression using matrix operations, and supports saving/loading the data and model weights.

Key Requirements (Mapped to Topics)

1) Linear Algebra Operations (ML-Relevant)

- Use element-wise multiplication and dot product to compute predictions.
- Apply transpose and inverse for solving the Normal Equation:

```
# 2x + y = 3
# x + 3y = 4
```

- Use np.sum(np.diag(...)) to compute sum of diagonal elements in matrix covariance.
- Use np.linalg.solve as an alternative method for solving linear systems.

2) Random Number Generation

- Use np.random.seed() for reproducibility.
- Use np.random.random() or np.random.randint() to generate synthetic feature data.
- Use np.random.choice() to randomly select samples for testing/training splits.
-

3) NumPy I/O Operations

- Save and load model weights using **`np.save()`** and **`np.load().`**
- Save the dataset to `.txt` format using **`np.savetxt()`** and load it back using **`np.loadtxt().`**
- Use **`np.savez()`** to store multiple arrays in a single file, e.g., features, targets, and weights.

4) Arithmetic Operations and Array Slicing

Apply vectorized arithmetic operations to perform statistical analysis.
Use slicing to separate train/test sets or to manipulate arrays during operations.

Bonus (Optional)

- Allow user input to choose between .npy, .txt, or .npz formats for saving data.