Pre-Read: Food Delivery Data Exploration and analysis 3

1. Broadcasting in NumPy

📌 **Description:**
Broadcasting is the method NumPy uses to allow arithmetic operations between arrays of different shapes by automatically “stretching” one to match the shape of the other.

💡 **Importance:**

- Makes operations between differently shaped arrays possible without manual reshaping.
- Reduces memory usage and code complexity.
- Widely used in ML model computations, feature scaling, and image processing.

Example: Adding a 1D array to each row of a 2D array without explicit loops.

---

2. Vectorization in NumPy

📌 **Description:**
Vectorization means applying operations directly on whole arrays instead of looping through elements one by one. This leverages NumPy’s C-level implementation for fast execution.

💡 **Importance:**

- Drastically improves performance and reduces code complexity.
- Minimizes Python-level loops, which are slower.

- Crucial for large datasets and real-time numerical computations.

Example:
Instead of looping to add two lists elementwise, `a + b` directly adds two NumPy arrays `a` and `b`.

---

3. Splitting Arrays in NumPy

📌 **Description:**
Splitting is dividing one array into multiple sub-arrays using functions like `np.split()`, `np.hsplit()`, and `np.vsplit()`.

💡 **Importance:**

- Helpful for partitioning data into training and testing sets.
- Makes it easier to manage and process chunks of large arrays.
- Useful for parallel processing and batch operations.

---

4. Stacking Arrays in NumPy

📌 **Description:**
Stacking is joining multiple arrays along a new or existing axis using `np.stack()`, `np.hstack()`, or `np.vstack()`.

💡 **Importance:**

- Used to combine multiple datasets or features into a single structured array.
- Essential in data preprocessing pipelines (merging features, combining channels).

- Maintains efficient memory layout and fast access for combined data.