

◆ NumPy কী?

****NumPy (Numerical Python)**** হলো Python-এর সবচেয়ে fast numerical library

👉 array, matrix, vector নিয়ে কাজ করে

👉 ML, DL, CV, Pandas—সবকিছুর base

◆ Python List vs NumPy Array

Feature	List	NumPy Array
Speed	Slow	🔥 Very Fast
Type	Mixed	Same dtype
Math	Manual loop	Built-in
ML	✗	✓

```
import numpy as np
a = [1,2,3]
b = np.array([1,2,3])
```

◆ Array Creation (সব ধরনের)

```
np.array([1,2,3])
np.array([[1,2],[3,4]])

np.zeros((3,3))
np.ones((2,4))
np.full((2,2), 7)

np.arange(0, 10, 2)
np.linspace(0, 1, 5)

np.random.rand(3,3)
np.random.randn(3,3)
np.random.randint(1, 10, (3,3))
```

◆ Array Properties (Must Know)

```
a = np.array([[1,2,3],[4,5,6]])
```

```
a.shape    # (2,3)
a.ndim     # 2
a.size     # 6
a.dtype    # int64
```

◆ Indexing & Slicing (🔥 Very Important)

1D

```
a = np.array([10,20,30,40,50])

a[0]
a[-1]
a[1:4]
```

2D

```
b = np.array([[1,2,3],[4,5,6]])

b[0,1]
b[:,1]
b[1,:]
```

◆ Reshape / Flatten

```
a = np.arange(12)

a.reshape(3,4)
a.reshape(-1,1)
a.flatten()
```

◆ Mathematical Operations

```
a = np.array([1,2,3])
b = np.array([4,5,6])

a + b
a - b
a * b
a / b
a ** 2

np.sqrt(a)
np.log(a)
np.exp(a)
```

Aggregation Functions

```
a = np.array([10,20,30])

np.sum(a)
np.mean(a)
np.max(a)
np.min(a)
np.std(a)
np.var(a)
```

Axis Explained (Confusing but Must)

```
a = np.array([[1,2,3],[4,5,6]])

np.sum(a, axis=0) # column
np.sum(a, axis=1) # row
```

 Rule:

- axis=0 → vertically
- axis=1 → horizontally

Broadcasting (🔥 Interview Favorite)

```
a = np.array([[1,2,3],[4,5,6]])  
b = np.array([1,2,3])  
  
a + b
```

👉 ছোট array বড় array-এর সাথে auto adjust হয়

◆ Boolean Indexing (Very Important)

```
a = np.array([10,20,30,40,50])  
  
a[a > 30]  
a[(a > 20) & (a < 50)]
```

✗ Wrong:

```
a > 20 & a < 50
```

◆ Copy vs View

```
a = np.array([1,2,3])  
b = a.copy()  
  
b[0] = 100
```

👉 copy করলে original change হয় না

◆ Stack / Split

```
a = np.array([1,2])  
b = np.array([3,4])  
  
np.hstack((a,b))  
np.vstack((a,b))  
  
np.split(np.array([1,2,3,4]), 2)
```

Sorting & Searching

```
a = np.array([40, 10, 30, 20])

np.sort(a)
np.argsort(a)
np.where(a > 25)
```

Linear Algebra (ML Base)

```
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])

np.dot(A, B)
A.T
np.linalg.det(A)
np.linalg.inv(A)
```

Random & Seed

```
np.random.seed(42)
np.random.rand(3)
```

NumPy in Machine Learning

```
X = np.array([[1, 2], [3, 4], [5, 6]])
y = np.array([0, 1, 1])

X.shape
y.reshape(-1, 1)
```

Mini Project (Exam + Viva)

```
marks = np.array([45, 67, 89, 32, 55])
```

```
avg = marks.mean()  
topper = marks.max()  
passed = marks[marks >= 40]  
failed = marks[marks < 40]
```

MUST-REMEMBER (Exam)

- list vs array
- reshape vs flatten
- axis meaning
- broadcasting
- boolean indexing
- dot vs *


Next Step (Strong Path for You)


NumPy → Pandas → Matplotlib → ML → CV

(তুমি already Pandas/ML করছো, তাই NumPy mastery খুব দরকার)

চাও তো আমি next করে দিতে পারি:

- 1 50 NumPy Practice Problems + Solution
- 2 Exam MCQ + Short Questions
- 3 NumPy দিয়ে ML from scratch (KNN / Linear Regression)

বললেই শুরু করবো 

ঠিক আছে 

এগুলোই NumPy-র সবচেয়ে important exam + interview questions।

আমি simple English + Bengali explanation + example দিচ্ছি।

◆ 1 List vs NumPy Array

 Difference Table

Feature	Python List	NumPy Array
Speed	Slow	🔥 Very Fast
Data Type	Mixed allowed	Same dtype
Memory	More	Less
Math Ops	Loop লাগে	Direct
ML Use	❌	✅

✅ Example

```
# List
a = [1, 2, 3]
print(a * 2)    # [1,2,3,1,2,3]

# NumPy Array
import numpy as np
b = np.array([1, 2, 3])
print(b * 2)    # [2 4 6]
```

📝 Exam Line

NumPy array is faster and supports vectorized operations, unlike Python list.

◆ 2 Reshape vs Flatten

◆ reshape()

- Shape change করে
- Data same থাকে
- Size same হতে হবে

```
a = np.arange(6)
a.reshape(2,3)
```

Output:

```
[[0 1 2]
 [3 4 5]]
```

◆ flatten()

- 2D → 1D বানায়
- New copy তৈরি করে

```
b = np.array([[1,2],[3,4]])
b.flatten()
```

Output:

```
[1 2 3 4]
```

 Difference

reshape	flatten
Shape change	1D করে
Same data	New copy
ML input prep	Data cleaning

 Exam Line

reshape changes array shape, flatten converts multi-dimensional array into 1D.

◆ **3** Axis Meaning (🔥 Most Confusing)

Rule:

- **axis = 0** → column-wise
- **axis = 1** → row-wise

Example

```
a = np.array([[1,2,3],
               [4,5,6]])
```

axis = 0

```
np.sum(a, axis=0)
```

➡ (1+4, 2+5, 3+6) = [5 7 9]

axis = 1


```
np.sum(a, axis=1)
```

→ $(1+2+3, 4+5+6) = [6 \ 15]$

🧠 Trick:

| axis যেকোনো কাজ করে, সেই direction বাদ যায়

◆ 4 Broadcasting

📌 What is Broadcasting?

Small array automatically expands to match larger array shape.

Example

```
a = np.array([[1,2,3],
               [4,5,6]])
b = np.array([1,2,3])

a + b
```

Output:

```
[[2 4 6]
 [5 7 9]]
```

📝 Exam Line

| Broadcasting allows arithmetic operations on arrays of different shapes.

◆ 5 Boolean Indexing

📌 Condition দিয়ে data filter করা

```
a = np.array([10,20,30,40,50])

a[a > 30]
```

Output:

```
[40 50]
```

Multiple Condition

```
a[(a > 20) & (a < 50)]
```

✗ Wrong

```
a > 20 & a < 50
```

 **Exam Line**

Boolean indexing is used to select elements based on condition.

◆ Dot vs * (🔥 ML Question)

◆ * (Element-wise Multiplication)

```
a = np.array([1,2,3])  
b = np.array([4,5,6])  
  
a * b
```

Output:

```
[4 10 18]
```

◆ dot() (Matrix / Vector Multiplication)

```
np.dot(a, b)
```

Output:

```
32
```

2D Matrix

```
A = np.array([[1,2],
              [3,4]])
B = np.array([[5,6],
              [7,8]])

np.dot(A, B)
```

Difference Summary

*	dot
Element-wise	Matrix multiply
Same shape	Compatible shape
Simple math	ML / DL use




Exam Line


* performs element-wise multiplication, while `dot()` performs matrix multiplication.

FINAL EXAM SUMMARY (5 lines)

- List is slower; NumPy array is faster and optimized
- reshape changes shape; flatten converts to 1D
- axis=0 → column, axis=1 → row
- Broadcasting handles different shapes
- Boolean indexing filters data
- * is element-wise, `dot()` is matrix multiplication

চাও তো আমি এটাকে:

-  PDF notes
-  MCQ + Viva Q&A
-  ML-based numerical examples

যেকোনোটা বানিয়ে দিতে পারি। বলো 

চল 🦊

নিচে **50 NumPy Practice Problems + Full Solutions** দেওয়া হলো —

👉 **Beginner** → **Intermediate** → **Advanced (ML-ready)**

👉 Exam + Interview + Project সব কভার করবে

Beginner Level (1–15)

1 Create 1D array from 1 to 10

```
np.arange(1, 11)
```

2 Create 3×3 zero matrix

```
np.zeros((3,3))
```

3 Create 2×4 ones matrix

```
np.ones((2,4))
```

4 Create array of 5 random numbers

```
np.random.rand(5)
```

5 Find shape of array

```
a.shape
```

6 Find number of dimensions

```
a.ndim
```

7 Convert list to NumPy array

```
np.array([5, 10, 15])
```

8 Create array of even numbers from 0–20

```
np.arange(0, 21, 2)
```

9 Get first 3 elements

```
a[:3]
```

10 Reverse array

```
a[::-1]
```

1 1 Change datatype to float

```
a.astype(float)
```

1 2 Find max value

```
np.max(a)
```

1 3 Find mean

```
np.mean(a)
```

1 4 Create identity matrix

```
np.eye(3)
```

1 5 Count elements

```
a.size
```

Intermediate Level (16–35)

1 6 Reshape 1D → 2D

```
np.arange(12).reshape(3,4)
```

1 7 Flatten array

```
a.flatten()
```

1 8 Element-wise addition

```
a + b
```

1 9 Element-wise multiplication

```
a * b
```

2 0 Matrix multiplication

```
np.dot(a, b)
```

2 1 Boolean indexing (>50)

```
a[a > 50]
```

2 2 Replace values < 40 with 0

```
a[a < 40] = 0
```

2 3 Sum column-wise

```
np.sum(a, axis=0)
```

2 4 Sum row-wise

```
np.sum(a, axis=1)
```

2 5 Sort array

```
np.sort(a)
```

2 6 Find index of max value

```
np.argmax(a)
```

2 7 Stack vertically

```
np.vstack((a,b))
```

2 8 Stack horizontally

```
np.hstack((a,b))
```

2 9 Split array into 2 parts

```
np.split(a, 2)
```

3 0 Unique elements

```
np.unique(a)
```

3 1 Standard deviation

```
np.std(a)
```

3 2 Variance

```
np.var(a)
```

3 3 Transpose matrix

```
a.T
```

3 4 Generate random integers


```
np.random.randint(1,100,(3,3))
```

3 5 Set random seed

```
np.random.seed(42)
```

Advanced / ML Level (36–50)

3 6 Normalize array (0–1)

```
(a - a.min()) / (a.max() - a.min())
```

3 7 Compute Euclidean distance

```
np.sqrt(np.sum((a-b)**2))
```

3 8 Convert 2D → column vector

```
a.reshape(-1,1)
```

3 9 One-hot encoding

```
np.eye(3)[[0,1,2]]
```

4 0 Replace NaN with 0

```
np.nan_to_num(a)
```

4 1 Check NaN

```
np.isnan(a)
```

4 2 Clip values (0–100)

```
np.clip(a, 0, 100)
```

4 3 Find correlation coefficient

```
np.corrcoef(a, b)
```

4 4 Calculate cosine similarity

```
np.dot(a,b)/(np.linalg.norm(a)*np.linalg.norm(b))
```

4 5 Shuffle array

```
np.random.shuffle(a)
```

4 6 Count non-zero elements

```
np.count_nonzero(a)
```

4 7 Find diagonal elements

```
np.diag(a)
```

4 8 Create mask (>mean)

```
a > a.mean()
```

4 9 Convert boolean to int

```
(a > 50).astype(int)
```

5 0 Mini ML Prep (Feature + Label)

```
X = np.array([[1,2],[3,4],[5,6]])  
y = np.array([0,1,1])
```

🔥 How to Practice (Best Way)

- 1 Copy each problem
 - 2 Try without seeing solution
 - 3 Compare
 - 4 Modify values
-

📌 Want Next?

- 📄 PDF Notes
- ? MCQ + Viva Questions
- 🤖 NumPy → ML (KNN / Linear Regression from scratch)

বললেই শুরু করবো ✔

ভালো প্রশ্ন 👍

NumPy শেখার পর “আর কী জানা দরকার”—এইটাই আসল turning point।

তুমি যেহেতু Pandas, ML, CV project করছো, তাই আমি honest + practical roadmap দিচ্ছি।

✅ NumPy শেষ করলে এখন যেগুলো জানা MUST

🔥 1 NumPy Advanced (Must for ML)

যেগুলো না জানলে ML কোডে আটকে যাবে:

- `vectorization` (loop বাদ দিয়ে কাজ)
- `where()` vs boolean indexing
- `einsum()` (advanced but powerful)
- `memory view vs copy`
- `performance comparison` (loop vs numpy)

📌 Example:

```
np.where(a > 50, 1, 0)
```

🔥 2 Pandas (You already started – go deeper)

এখন NumPy + Pandas combine করতে হবে:

Must know:

- `loc` vs `iloc`
- `groupby`
- `merge` / `join`
- `apply` vs `vectorized ops`
- `datetime` handling
- `missing value pipeline`

👉 Real skill = **data cleaning + feature engineering**

🔥 3 Matplotlib + Seaborn (Data Visualization)

ML model বুঝাতে visualization লাগে।

Must know:

- `line`, `bar`, `histogram`
- `boxplot` (outlier)
- `heatmap` (correlation)

👉 Project + Viva = visualization required

4 Statistics (Very Important for ML)

এটা অনেকেই skip করে ❌

You must know:

- mean, median, mode
- variance, std
- probability basics
- correlation vs covariance
- normal distribution

👉 Model evaluation এখানেই depend করে

5 Machine Learning Core (Next Big Step)

এখন NumPy knowledge directly কাজে লাগবে:

Learn order:

1. Linear Regression (from scratch using NumPy)
2. Logistic Regression
3. KNN
4. K-Means
5. Evaluation metrics

📌 Example:

```
y_pred = X.dot(w) + b
```

6 Scikit-learn (Industry Tool)

From scratch বোঝার পর:

- train_test_split
 - StandardScaler
 - model.fit(), predict()
 - confusion matrix
 - cross-validation
-

7 Computer Vision (Your Strong Area)

Since তুমি YOLO / CV নিয়ে কাজ করছো:

- NumPy image as array
 - OpenCV
 - image normalization
 - bounding box operations
 - batch processing
-

Smart Learning Path (For You)

```
NumPy ✓  
↓  
Pandas (Advanced)  
↓  
Statistics  
↓  
Matplotlib  
↓  
ML from Scratch (NumPy)  
↓  
scikit-learn  
↓  
CV / DL
```

Now I Recommend (Next Action)

তুমি এখন ready এইগুলোর জন্য 🙌


Option 1

 NumPy → Linear Regression from Scratch

Option 2

 Advanced Pandas (Real Dataset Cleaning Project)

Option 3

 ML Interview + Viva Preparation

বল কোনটা আগে করবো—আমি step-by-step শুরু করে দেবো 💪

