NumPy for Machine Learning & Interviews

Chapter 1: NumPy Basics

1.1 Why NumPy?

- Python lists are slow for large datasets.
- NumPy arrays are optimized, use **vectorized operations**.
- Essential for ML data preprocessing and computation.

1.2 Creating Arrays

```
import numpy as np

# 1D array
arr = np.array([1, 2, 3, 4])
print(arr)  # [1 2 3 4]
print(type(arr))  # <class 'numpy.ndarray'>

# 2D array
matrix = np.array([[1,2,3],[4,5,6]])
print(matrix)
```

Bengali: এখানে `np.array` দিয়ে Python list কে NumPy array তে convert করা হলো।

1.3 Array Properties

```
print(arr.shape)  # আকার (4,)
print(arr.ndim)   # dimension, এখানে 1
print(arr.dtype)  # data type, যেমন int64
```

1.4 Special Arrays

```
np.zeros((3,3))        # 3x3 zero matrix
np.ones((2,4))         # 2x4 ones matrix
np.arange(0,10,2)      # 0 থেকে 10, step 2
np.linspace(0,1,5)     # 0 থেকে 1, 5 equal parts
np.random.rand(3,3)    # random 0-1
np.random.randint(0,10,(2,3)) # random integers
```

Chapter 2: Array Operations

2.1 Basic Element-wise Operations

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

print(a + b)  # [5 7 9]
print(a * b)  # [4 10 18]
print(a / 2)  # [0.5 1.0 1.5]
```

2.2 Broadcasting

```
a = np.array([[1,2,3],[4,5,6]])
b = np.array([10,20,30])
print(a + b)
```

Bengali: ছোট array বড় array এর সাথে automatically match করে element-wise operation করে।

2.3 Reshape & Flatten

```
a = np.arange(6)
print(a.reshape(2,3))  # 2x3 matrix
print(a.flatten())      # 1D array
```

2.4 Indexing & Slicing

```
a = np.array([10,20,30,40,50])
print(a[0])       # 10
print(a[-1])      # 50
print(a[1:4])     # [20 30 40]
print(a[a>25])    # [30 40 50]
```

Chapter 3: Advanced Array Operations

3.1 Matrix Operations

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

```
print(A @ B)             # matrix multiplication
print(np.dot(A,B))       # same
print(np.transpose(A))   # transpose
print(np.linalg.inv(A))  # inverse
```

3.2 Aggregation / Statistics

```
a = np.array([1,2,3,4,5])
print(np.sum(a))
print(np.mean(a))
print(np.median(a))
print(np.std(a))
print(np.var(a))
print(np.max(a))
print(np.min(a))
```

3.3 Logical Operations

```
a = np.array([1,2,3,4,5])
print(np.where(a>3, 'Yes', 'No'))  # ['No' 'No' 'No' 'Yes' 'Yes']
```

3.4 Random Sampling for ML

```
np.random.seed(42)       # reproducible
X = np.random.rand(5,3)  # 5 rows, 3 features
print(X)
```

Chapter 4: Common NumPy Interview Questions

4.1 Array Creation & Shape

1. Create a 3x3 identity matrix.

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

2. Difference between `arange` and `linspace`.
3. `arange(start, stop, step)` → uses step size
4. `linspace(start, stop, num)` → divides interval into `num` equal parts

4.2 Array Operations

1. Sum all elements along axis 0 and 1 of a 2D array.

```
arr = np.array([[1,2,3],[4,5,6]])
print(np.sum(arr, axis=0))  # column sum [5 7 9]
print(np.sum(arr, axis=1))  # row sum [6 15]
```

2. Multiply two matrices without a loop.

```
A @ B
```

4.3 Indexing / Slicing

1. Extract all even numbers.

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

2. Replace negative numbers with 0.

```
arr = np.array([-1,2,-3,4])
arr[arr<0] = 0  # [0 2 0 4]
```

4.4 Broadcasting

1. Add a 1D array to each row of 2D array.

```
arr2d = np.array([[1,2,3],[4,5,6]])
arr1d = np.array([10,20,30])
print(arr2d + arr1d)
```

4.5 Statistics

1. Normalize a dataset (Z-score).

```
arr = np.array([1,2,3,4,5])
normalized = (arr - np.mean(arr)) / np.std(arr)
print(normalized)
```

4.6 Random & Sampling

1. Create random 3x4 array and shuffle.

```
arr = np.random.randint(0,10,(3,4))
np.random.shuffle(arr)
print(arr)
```

4.7 Linear Algebra

1. Dot product.

```
np.dot(np.array([1,2]), np.array([3,4]))  # 11
```

2. Eigenvalues & Eigenvectors.

```
A = np.array([[1,2],[2,3]])
vals, vecs = np.linalg.eig(A)
print(vals)
print(vecs)
```

4.8 ML-focused

1. Standardize features.

```
X = np.random.rand(5,3)
X_std = (X - X.mean(axis=0)) / X.std(axis=0)
print(X_std)
```

2. Linear regression formula.

```
# y = Xb
X = np.random.rand(5,3)
y = np.random.rand(5,1)
b = np.linalg.inv(X.T @ X) @ X.T @ y
print(b)
```

3. Vectorized sigmoid.

```
def sigmoid(z):
    return 1 / (1 + np.exp(-z))

z = np.array([-1,0,1])
print(sigmoid(z))
```

Chapter 5: Practice Exercises

1. Create a 1D array 1–20, reshape to 4x5.
2. Find multiples of 3 in a random 1D array (size 15).
3. Sum of each column in 2D array.
4. Normalize a 1D array.
5. Create 3x3 matrix and compute inverse.
6. Generate 10x3 dataset, compute mean & std of each column.
7. Dot product between two matrices.
8. Implement Z-score normalization for 2D array.
9. Generate synthetic dataset and compute covariance matrix.
10. Vectorize sigmoid for logistic regression.

Tips & Tricks for Interviews

- Always **use vectorized operations**, avoid loops.
- Understand **axis=0 vs axis=1**.
- Practice **boolean indexing** for filtering.
- Learn **matrix operations** for ML algorithms.
- Be ready to **create synthetic data** for testing.

This PDF can now be converted and used as a complete NumPy guide for ML practice and interviews.