NumPy Roadmap: From Basic to Advanced

1. Creating a NumPy Array

You can create arrays from Python lists or tuples using numpy.array(). NumPy arrays are faster and more memory-efficient than Python lists.

```
import numpy as np

# Create a 1D array
a = np.array([1, 2, 3, 4, 5])
print(a)

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

2. Array Indexing and Slicing

NumPy allows for efficient indexing and slicing of arrays. You can access specific elements, rows, or columns of the array.

```
import numpy as np

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

# Access elements
print(arr[0])  # First element
print(arr[-1]) # Last element

# Slicing
print(arr[1:4])  # Elements from index 1 to 3
```

3. Broadcasting

Broadcasting allows NumPy to perform operations on arrays of different shapes efficiently. Smaller arrays are 'broadcast' to match the shape of larger arrays.

```
import numpy as np

arr1 = np.array([1, 2, 3])
arr2 = np.array([[1], [2], [3]])

# Add arrays with broadcasting
result = arr1 + arr2
print(result)  # Outputs a 3x3 array
```

4. Vectorized Operations

Vectorized operations in NumPy eliminate the need for explicit loops, providing better performance. Use element-wise operations instead of Python loops.

```
import numpy as np

# Create arrays
a = np.array([1, 2, 3, 4])
b = np.array([10, 20, 30, 40])

# Element-wise addition
result = a + b
print(result)

# Element-wise square
print(a ** 2)
```