Description of NumPy Arrays in Data Science

Farivar Zarvandeh

July 28, 2023

1 Description of Topic

NumPy is a powerful library in Python used extensively in data science for numerical computations. One of the fundamental data structures in NumPy is the NumPy array, also known as ndarray. NumPy arrays are N-dimensional homogeneous containers for data, allowing efficient operations on large datasets.

1.1 Usages and Functionality of NumPy Arrays in Data Science

- NumPy arrays provide a fast and memory-efficient way to perform mathematical operations on large datasets, making them ideal for scientific computing and data analysis tasks.
- NumPy arrays can represent vectors, matrices, and higher-dimensional arrays, making them versatile for various applications.
- Data manipulation, element-wise operations, broadcasting, and aggregation are efficiently performed on NumPy arrays.
- NumPy arrays are the building blocks for other data structures like pandas DataFrames, which are widely used in data science.

2 List of Methods to Generate NumPy Arrays

1. `numpy.array()`: Creates an array from a sequence-like object.
2. `numpy.zeros()`: Generates an array of all zeros.
3. `numpy.ones()`: Generates an array of all ones.
4. `numpy.full()`: Creates an array of a specified shape filled with a specified value.
5. `numpy.empty()`: Generates an array without initializing its elements to any particular value.

6. `numpy.arange()`: Creates an array with evenly spaced values within a given interval.

7. `numpy.linspace()`: Generates an array with evenly spaced values over a specified range.

8. `numpy.random.rand()`: Generates an array with random values from a uniform distribution over the interval [0, 1).

9. `numpy.random.randn()`: Generates an array with random values from a standard normal distribution.

```
Using numpy.array():
-----------------------------------
import numpy as np

# Create a NumPy array from a list
arr1 = np.array([1, 2, 3, 4, 5])
print(arr1)  # Output: [1 2 3 4 5]

# Create a NumPy array from a tuple
arr2 = np.array((6, 7, 8, 9, 10))
print(arr2)  # Output: [ 6  7  8  9 10]
```

```
#Using numpy.zeros() and numpy.ones():
-----------------------------------------

import numpy as np

# Create an array of zeros with shape (2, 3)
zeros_arr = np.zeros((2, 3))
print(zeros_arr)
# Output:
# [[0. 0. 0.]
#  [0. 0. 0.]]

# Create an array of ones with shape (3, 2)
ones_arr = np.ones((3, 2))
print(ones_arr)
# Output:
# [[1. 1.]
#  [1. 1.]
#  [1. 1.]]
```

```
Using numpy.full():
------------------------------
```

```python
import numpy as np

# Create a 3x3 array filled with the value 7
filled_arr = np.full((3, 3), 7)
print(filled_arr)
# Output:
# [[7 7 7]
#  [7 7 7]
#  [7 7 7]]
```

```
Using numpy.empty():
-----------------------------------------------
```

```python
import numpy as np

# Create an empty array of shape (2, 4)
empty_arr = np.empty((2, 4))
print(empty_arr)
# Output:
# [[6.23042070e-307 4.67296746e-307 1.69121096e-306 8.01097889e-307]
#  [1.29062229e-306 1.33512173e-306 1.33511562e-306 2.22518251e-306]]
```

```
Using numpy.arange():
-------------------------------------------------
```

```python
import numpy as np

# Create an array with values from 0 to 9 (exclusive)
range_arr = np.arange(10)
print(range_arr)  # Output: [0 1 2 3 4 5 6 7 8 9]

# Create an array with values from 5 to 15 (exclusive) with a step of 2
range_arr2 = np.arange(5, 15, 2)
print(range_arr2)  # Output: [ 5  7  9 11 13]
```

```
Using numpy.linspace():

-------------------------------------
```

```python
import numpy as np

# Create an array with 5 equally spaced values between 0 and 1 (inclusive)
linspace_arr = np.linspace(0, 1, 5)
print(linspace_arr)  # Output: [0.   0.25 0.5  0.75 1.  ]
```

```
Using numpy.random.rand() and numpy.random.randn():
-----------------------------------------------------------
```

```python
import numpy as np

# Create a 2x3 array with random values from a uniform distribution [0, 1)
rand_arr = np.random.rand(2, 3)
print(rand_arr)
# Output:
# [[0.71533423 0.76827301 0.73318471]
#  [0.00424766 0.60982305 0.51866837]]

# Create a 3x3 array with random values from a standard normal distribution
randn_arr = np.random.randn(3, 3)
print(randn_arr)
# Output:
# [[ 0.31584827 -0.20669599 -0.38921971]
#  [-0.26437596 -0.12347281  0.52574252]
#  [-0.89023117 -0.61714791  1.02422545]]
```