Fresher Academy

NumPy

What Will You Learn

- Introduction
- NumPy Arrays
- Shape Manipulation
- NumPy Array Copy
- Subsetting
- Random Numbers with
- Basic Statistics with Nur

NumPy

FPT Software | FRESHER ACADEMY

Introduction

NumPy Introduction

Python Lists

- Different types
- Random access
- Change, add, remove

How to perform mathematical operations on lists?

NumPy Introduction

Exercises

- Given rectangles having lengths: [1, 4, 6, 9, 11, 10] and widths: [2, 3, 7, 4, 8, 12]. Calculate rectangle areas and store in another list
- Given two sequences: A = [34, 56, 32, 87, 65, 29] and B = [26, 78, 45, 38, 85, 92]. Compare A and B in elements having the same index to store in another bool list C. For instance 34 > 26 so C[0] = True
- Given circle radius as [12, 24.5, 23.5, 26.7, 30, 19.4, 25.6]. Calculate circle areas that the radius is longer than 25 and store in another list

NumPy Introduction

Alternative to Python List

- NumPy Array
- Numeric Python
- Elementwise
- Easy and Fast
- Installation: In the terminal: pip3 install numpy

NumPy

FPT Software | FRESHER ACADEMY

NumPy Arrays

NumPy Arrays

Array Creation

- Create arrays from a list or tuple using the array() function

```
>>> score_list = [2.3, 3.5, 1.9, 2.6, 3.4, 3.1]
>>> scores = np.array(score_list)
>>> scores
array([2.3, 3.5, 1.9, 2.6, 3.4, 3.1])
```

NumPy Arrays

Array Creation

- NumPy could recognize data types for arrays

```
>>> scores.dtype
dtype('float64')
```

- Data types can be explicitly specified on the creation statement

```
>>> ages = np.array([23, 26, 32, 39, 18], dtype=np.int16)
>>> ages.dtype
dtype('int16')
```

NumPy Arrays

Array Creation

- The function zeros() creates an array of zeros

```
>>> np.zeros((2,3))
array([[0., 0., 0.],
       [0., 0., 0.]])
```

By default, the dtype of the created array is float64.

NumPy Arrays

Array Creation

- The function ones() creates an array of ones

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

By default, the dtype of the created array is float64.

NumPy Arrays

Array Creation

- The function empty() creates an random-value array and depends on the state of the memory.

```
>>> np.empty((2,3))
array([[0., 0., 0.],
       [0., 0., 0.]])
```

By default, the dtype of the created array is float64.

NumPy Arrays

Create arrays from number ranges: np.arange(start, stop, step, dtype)

- start : number, optional
- stop : number
- step : number, optional
- dtype : dtype

```
>>> np.arange(3)
array([0, 1, 2])
>>> np.arange(3,7)
array([3, 4, 5, 6])
>>> np.arange(3,7,2)
array([3, 5])
```

NumPy Arrays

NumPy's array attributes

- ndarray.ndim: the number of axes (dimensions) of the array.
- ndarray.shape: the dimensions of the array.
- ndarray.size: the total number of elements of the array.
- ndarray.dtype: an object describing the type of the elements in the array. (numpy.int32, numpy.int16, numpy.float64)
- ndarray.itemsize: the size in bytes of each element of the array.
- ndarray.data: the buffer containing the actual elements of the array.

NumPy Arrays

NumPy's array attributes

```
>>> numbers
array([[0, 1, 2],
       [3, 4, 5]])
>>> numbers.ndim
2
>>> numbers.shape
(2, 3)
>>> numbers.size
6
>>> numbers.dtype
dtype('int64')
>>> numbers.itemsize
8
```

NumPy Arrays

Basic Operations

- Arithmetic operators on arrays apply elementwise. (+,-,*,/,**,%)

```
>>> numbers = np.arange(6).reshape(2,3)
>>> numbers * 2
array([[ 0,  2,  4],
       [ 6,  8, 10]])
>>> numbers + 4
array([[4, 5, 6],
       [7, 8, 9]])
```

NumPy Arrays

Matrix Operations (*, @, .dot())

```
>>> A = np.array([[2,3], [0,1]])
>>> B = np.array([[1,0], [5,6]])
>>> A * B     # Elementwise product
array([[2, 0],
       [0, 6]])
>>> A @ B     # Matrix product
array([[17, 18],
       [ 5,  6]])
>>> A.dot(B) # Matrix product
array([[17, 18],
       [ 5,  6]])
```

NumPy

Shape Manipulation

FPT Software | FRESHER ACADEMY

Shape Manipulation

The shape attribute

```
>>> numbers = np.arange(6).reshape(2,3)
>>> numbers
array([[0, 1, 2],
       [3, 4, 5]])
>>> numbers.shape
(2, 3)
```

Shape Manipulation

The ravel() method

```
>>> numbers
array([[0, 1, 2],
       [3, 4, 5]])
>>> numbers.ravel()    # Return a flattened array
array([0, 1, 2, 3, 4, 5])
```

Shape Manipulation

The reshape() method

```
>>> numbers
array([[0, 1, 2],
       [3, 4, 5]])
>>> numbers.reshape(3,2)
array([[0, 1],
       [2, 3],
       [4, 5]])
```

Shape Manipulation

T: transpose

```
>>> numbers
array([[0, 1, 2],
       [3, 4, 5]])
>>> numbers.T  # Return a transposed array
array([[0, 3],
       [1, 4],
       [2, 5]])
```

Shape Manipulation

The resize() method

```
>>> numbers
array([[0, 1, 2],
       [3, 4, 5]])
>>> numbers.resize(3,2)          # Modify the array itself
>>> numbers
array([[0, 1],
       [2, 3],
       [4, 5]])
```

NumPy

NumPy Array Copy

FPT Software | FRESHER ACADEMY

NumPy Array Copy

Simple Assignment: No Copy at All

```
>>> A = np.array([34, 56, 32, 87, 65, 29])
>>> B = A
>>> B is A
True
```

NumPy Array Copy

View or Shallow copy with the view() method

```
>>> A = np.array([34, 56, 32, 87, 65, 29])
>>> B = A.view()
>>> B
array([34, 56, 32, 87, 65, 29])
>>> B is A
False
>>> B[-1] = 85
>>> A  # Both arrays are modified together
array([34, 56, 32, 87, 65, 85])
```

NumPy Array Copy

Deep copy with the copy() method

```
>>> A = np.array([34, 56, 32, 87, 65, 29])
>>> B = A.copy()
>>> B
array([34, 56, 32, 87, 65, 29])
>>> B is A
False
>>> B[-1] = 85
>>> B
array([34, 56, 32, 87, 65, 85])
>>> A  # Not modify the orignial
array([34, 56, 32, 87, 65, 29])
```

NumPy

Subsetting

FPT Software | FRESHER ACADEMY

Subsetting

Indexing

```
>>> numbers
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11],
       [12, 13, 14, 15]])
>>> numbers[1, 2]
6
>>> numbers[3, 0]
12
>>> numbers[-1, -1]
15
>>> numbers[-2, -3]
9
```

Subsetting

Slicing

```
>>> numbers
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11],
       [12, 13, 14, 15]])
>>> numbers[:,2]
array([ 2,  6, 10, 14])
>>> numbers[1,:]
array([4, 5, 6, 7])
>>> numbers[1:2,2:3]
array([[6]])
>>> numbers[0:2,1:3]
array([[1, 2],
       [5, 6]])
```

Subsetting

Indexing with Boolean Arrays

```
>>> A = np.array([34, 56, 32, 87, 65, 29])
>>> B = np.array([26, 78, 45, 38, 85, 92])
>>> A > B
array([ True, False, False,  True, False, False])
>>> B[A > B]
array([26, 38])
>>> B[A < B]
array([78, 45, 85, 92])
```

NumPy

Random Numbers with NumPy

FPT Software | FRESHER ACADEMY

Random Numbers with NumPy

np.random.random(size=None)

- Return random floats in the half-open interval [0.0, 1.0).
- size : int or tuple of ints, optional

```
>>> np.random.random()
0.28613933495037946
>>> np.random.random(4)
array([0.22685145, 0.55131477, 0.71946897, 0.42310646])
>>> np.random.random((2,3))
array([[0.9807642 , 0.68482974, 0.4809319 ],
       [0.39211752, 0.34317802, 0.72904971]])
```

Random Numbers with NumPy

np.random.randint(low, high=None, size=None, dtype='l')

- Return random integers from low (inclusive) to high (exclusive).

```
>>> np.random.randint(5)
2
>>> np.random.randint(5, 10)
9
>>> np.random.randint(5, 10)
5
>>> np.random.randint(5, 10, 2)
array([5, 6])
```

Random Numbers with NumPy

numpy.random.choice(1-D-array, size=None, replace=True, p=None)

- Generates a random sample from a given 1-D array
- replace : boolean, optional (Whether the sample is with or without replacement)

```
>>> A
array([34, 87, 29, 56, 32, 65])
>>> np.random.choice(A)
56
>>> np.random.choice(A, 3)
array([32, 32, 87])
>>> np.random.choice(A, 3, replace=False)
array([32, 34, 87])
```

Random Numbers with NumPy

np.random.shuffle(array)

- Modify a sequence in-place by shuffling its contents.

```
>>> A
array([34, 87, 29, 56, 32, 65])
>>> np.random.shuffle(A)
>>> A
array([32, 65, 87, 29, 56, 34])
```

Random Numbers with NumPy

np.random.permutation(array)

- Randomly permute a sequence, or return a permuted range.

```
>>> A
array([34, 87, 29, 56, 32, 65])
>>> np.random.permutation(A)
array([65, 32, 34, 56, 87, 29])
```

NumPy

FPT Software | FRESHER ACADEMY

Basic Statistics with NumPy

Basic Statistics with NumPy

```
>>> A = np.array([34, 56, 32, 87, 65, 29])
>>> B = np.array([26, 78, 45, 38, 85, 92])
>>> np.mean(A)
50.5
>>> np.median(A)
45.0
>>> np.std(A)
21.013884299037464
>>> np.corrcoef(A, B)
array([[ 1.        , -0.05991697],
       [-0.05991697,  1.        ]])
>>> np.sum(A)
303
>>> np.sort(A)
array([29, 32, 34, 56, 65, 87])
```

Fresher Academy

Happy Analyzing!