NUMPY IN PYTHON

19 March 2025 19:18

NumPy is a fundamental library for scientific computing in Python. It provides support for arrays and matrices, along with a collection of mathematical functions to operate on these data structures. In this lesson, we will cover the basics of NumPy, focusing on arrays and vectorized operations.

TO INSTALL NUMPY

pip install numpy

CODE

```
Import numpy as np

#create array using numpy
#create a 1D array
Arr1=np.array([1,2,3,4,5])
Print(arr1)
Print(type(arr1))
Print(arr1.shape)
```

OUTPUT

```
[1,2,3,4,5]
<class 'numpy.ndarray'>
(5,)
```

No. of element

RESHAPE

```
Arr2=np.array([1,2,3,4,5])
Arr2.reshape(1,5)    #1row and 5 column
```

OUTPUT

```
array( [ [1,2,3,4,5] ] )
```

NESTED LIST

```
Arr2=np.array( [ [1,2,3,4,5 ] ] )
Arr2.shape
```


```
#2D array
Arr2=np.array([ [1,2,3,4,5],[2,3,4,5,6] ])
Print(arr2)
Print(arr2.shape)
```

```
[ [1,2,3,4,5] , [2,3,4,5,6] ]
(2,5)
```

```
#array with inbuilt function
Np.arrange(0,10,2).reshape(5,1)
```

```
Array([[0],
       [2],
       [4],
       [6],
       [8]])
```

```
np.ones((3,4))
```

```
array([[1., 1., 1., 1.),
       (1., 1., 1., 1.),
        (1., 1., 1., 1.]])
```

```
##identity matrix
np.eye(3)
```

```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

CODE

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

print("Array:\n", arr)
print("Shape:",  arr.shape)                          #Output: (2, 3)
print("Number of dimensions:",  arr.ndim)           #Outputs 2
print("Size (number of elements):",  arr.size)      # Output: 6
print("Data type:",  arr.dtype)                     # Output: Int64 (may vary based on platform)
print("Item size (in bytes):",  arr.itemsize)      #Output: 8 (may vary based on platform)
```

OUTPUT

```
Array:
        [[123]
        [456]]
Shape:   (2, 3)
Number of dimensions:  2
Size (number of elements): 6
Data type:     int32
Item size (in bytes): 4
```

CODE

```
###Numpy Vectorized Operation

arr1=np.array([1,2,3,4,5])
arr2=np.array([10,20,30,40,50])

###Element Wise addition
print("Addition: ", arr1+arr2)

##Element Wise Substraction
print("Substraction: ", arr1-arr2)

#Element-wise multiplication
print("Multiplication: ", arr1 *arr2)

#Element-wise division
print("Division: ", arr1  / arr2)
```

OUTPUT

```
Addition:       [11 22 33 44 55]
Substraction: [-9-18-27-36-45]
Multiplication:[10 48 90 160 250]
Division:       [0.1 0.1 0.1 0.1 0.1]
```

CODE

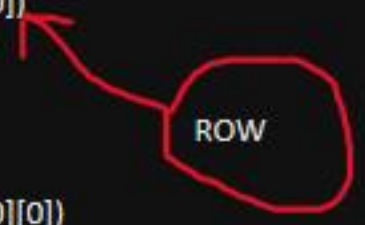
```
## Universal Function
arr=np.array([2,3,4,5,6])

## square root
print(np.sqrt(arr))

##Exponential
print(np.exp(arr))

##sine
print(np.sin(arr))

##natural log
print(np.log(arr))
```

OUTPUT

```
[1.41421356 1.73205081 2.   2.23606798 2.44948974]

[7.3890561 20.08553692 54.59815003 148.4131591
403.42879349]

[0.90929743
0.14112001 -0.7568025 -0.95892427 -0.2794155]

[0.69314718 1.09861229 1.38629436 1.60943791
1.79175947]
```

CODE

```
#array slicing and Indexing
Arr=np.array([[1,2,3,4], [5,6,7,8),
(9,10,11,12]])

print("Array: \n",  arr)


Print(arr[0])


Print(arr[0][0])


Print(arr[1:])


Print(arr[1:,2:])
```



OUTPUT

```
Array:
      [[1   2   3    4]
      [5   6    7    8]
      [ 9   10   11   12]]


Array([1,  2,  3,  4])


1


array([[ 5,   6,    7,   8],
       [9,    10,    11,    12]])


array([[ 7,  8],
       [11, 12]])
```

```
Print(arr[0:2,2:])
```

```
[[3  4]
 [7  8]]
```

```
#modifying array
Arr[0,0]=100
Print(arr)
```

```
Array:
      [[100  2   3    4]
      [5     6    7    8]
      [ 9    10   11   12]]
```

CODE

```
### statistical concepts Normalization
##to have a mean of 8 and standard deviation of 1
data = np.array([1, 2, 3, 4, 5])

#Calculate the mean and standard deviation
mean = np.mean(data)
std_dev = np.std(data)

#Normalize the data
normalized_data = (data mean) / std_dev
print("Normalized data:", normalized_data)


#Output!
#Normalized data: 1.26491106 0.03215553 0. 0.63245553 1.26191186]
```

OUTPUT

```
Normalized data: [-1.41421356 -0.70710678 0.8.70710678 1.41421356]
```

CODE

```
data np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

#mean
mean np.mean(data)
print("Mean:", mean)

# Median
median np.median(data)
print("Median:", median)

#Standard deviation
std_dev = np.std(data)
print("Standard Deviation:", std_dev)

$Variance
variance = np.var(data)
print("Variance:", variance)
```

Output

```
Mean: 5.5
Median: 5.5
Standard Deviation: 2.8722813232690143
Variance: 8.25
```

CODE

```
## Logical operation
data=np.array([1,2,3,4,5,6,7,8,9,10))

data[(data>=5)  &  (data<=8)]
```

OUTPUT

```
Array([5,6,7,8])
```