CREATIVE INSTITUTE DATA SCIENCE

NUMPY INTORDUCTION

NumPy

NumPy, which stands for Numerical Python

Numpy is the fundamental Python package for scientific computation programming.

Scientific programming

deals with solving scientific or mathematical problems with the help of computers.

Numpy in python used for scientific programming

Tools & techniques to solve mathamatical problems

Basic Scientific programming operations are like

- Array
- Matrices
- Integration
- Statistics

How to Install Numpy

Pip install numpy

Used of Numpy

- Arrays
- Linear algebra
- Random number generation

Similarity between list & Numpy arrays

NUMPY INTORDUCTION

similarity

- Storing data
- Mutable
- Can be indexed
- Slicing operation

Defference

- List: different data types
- Array : similar data types

List is built in library in python while numpy need to install numpy python packages

Array will get output if you devide array by any value..

a=np.array([3,6,9,54])

a/3

list will get error if devide list

l=[2,4,6,8]

l/2   # will get error

Advantages

1. Store less memory
2. Fast
3. More convenient to use when we want to do mathematical operations.
4. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.

NUMPY INTORDUCTION

Arrays

An array is the collection of elements of the same data type

NumPy is used to work with arrays. The array object in NumPy is called ndarray.

array(object, dtype=None, *, copy=True, order='K', subok=False, ndmin=0,   like=None)

- ✓ **one dimestional array**

  a1=np.array([1,2])

- ✓ **Two Dimentional Array**

  a2=np.array([[1,2],[3,4]])

- ✓ **change of data type**

  a4=np.array([1,2],'complex')

NUMPY INTORDUCTION

Array Function in Numpy

Numpy provides us with several built-in functions to create and work with arrays

```
numpy.array(object, dtype=None, copy=True, order='K', subok=False,
ndmin=0)
```

Here, all attributes other than objects are optional.

- **Object:** specify the object for which you want an array
- **Dtype:** specify the desired data type of the array
- **Copy:** specify if you want the array to be copied or not
- **Order:** specify the order of memory creation
- **Subok:** specify if you want a sub-class or a base-class type array
- **Ndmin:** specify the dimensions of an array

Array Creation using np.array() Function

```
import numpy as np
#creating an array a
a = np.array( [[ 1, 2, 3, 4],[ 5, 6, 7,8],[9,10,11,12]])

#printing array a

print ("Array is:",a)

#we can also print the other attributes like dimensions,shape and size of an
array
print ("Dimensions of a are:", a.ndim)
print ("Shape of a is", a.shape)
print ("Size of a is", a.size)
```

NUMPY INTORDUCTION

Creating an Empty Array using empty Function

```
#creating an array an empty square array of dimensions 2X2
empty_array = np.empty([2,2], dtype = int)
#np.empty() creates an array with random values
print ("Array is:", empty_array)
```

Creating an Array with Zeros using zero_like Function

Zero_like function returns an array of zeros with shape and type as input.

np.zeros(shape,dtype)

```
a2=np.zeros([2,2],dtype=int)
print(a2)
```

Array Manipulation

Copying from One Array to Another

copy content from one array to another using the copyto function.

np.copyto(destination, source)

```
zeros_array = np.zeros([2,2], dtype = int)
print ("Array zeros is:", zeros_array)
ones_array = np.ones([2,2], dtype = int)
print ("Array ones is :", ones_array)
#copying content from ones_array to zeros
np.copyto(zeros_array,ones_array)
print ("New zeros array :", zeros_array)
```

NUMPY INTORDUCTION

Changing the Shape of an Array

changes the shape of an array without changing the data in it.

np.reshape(object, shape)

```
#creating an array a 1D array
a = np.array([[1,2],[3,4]])
print ("array a is :", a)
#changing the shape of array from 2D to 1D
print ("reshape array a is:",np.reshape(a,4))
```

Transposing an Array

Transpose_like array functions help in transposing the array.

```
a = np.array([[1,2],[3,4]])
print ("array a is :", a)
#transposing array a using array.T
print ("transposed array a is:", a.T)
```

Joining Two or More Arrays

Concatenate function helps in joining two or more array along the given axis.

np.concatenate((a1, a2, ...), axis=0, out=None)

```
import numpy as np
#creating two arrays a and b
a = np.array([[1, 2], [3, 4]])
b = np.array([[5, 6]])
#joining a and b vertically
print ("concatenated array vertically:", np.concatenate((a, b), axis=0))
#joining a and b horizontally
print ("concatenated array horizontally:", np.concatenate((a, b),
axis=None))
```

Splitting an Array Into Multiple Sub-Arrays

np.split(array, indices)

```
#creating an array using arange function.
a = np.arange(8)
print (a)
#splitting array a into 4 equal parts
print ("sub-parts of array a:", np.split(a, 4))
```

Adding Elements to an Existing Array

np.insert(array, index, value) to insert values along the given axis before the given indices.

```
#creating an array using arange function.
a = np.array([[1,2,3],[1,2,3]])
print ("array a is :", a)
#inserting elements along the y axis at index 1
print ("array a after insertion:", np.insert(a,1,5, axis = 1))
```

Deleting the Elements from an Array

Delete function can be used to delete an axis of the given array and returns a new array with sub-arrays along the deleted axis.

np.delete(array, object, axis)

```
a = np.array([[1,2,3],[1,2,3]])
print ("array a is :", a)
#deleting elements
print ("after deletion:", np.delete(a,[1,2,3],axis = 0))
```