Programming with Python
Part 3: Numpy

Introduction

- NumPy stands for ‘Numerical Python’.
- It is a package for data analysis and scientific computing with Python.
- NumPy uses a multidimensional array object, and has functions and tools for working with these arrays.
- The powerful n-dimensional array in NumPy speeds-up data processing.

Installing Numpy

- NumPy can be installed by typing following command:

```
pip install numpy
```

- However, most installations of Python such as Anaconda comes with preinstalled numpy
- To check is it is already installed, type

```
import numpy
```

in the console

Arrays

- An array is a data type used to store multiple values using a single identifier (variable name).
- An array contains an ordered collection of data elements where each element is of the same type and can be referenced by its index (position).
- The important characteristics of an array are:
  - Each element of the array is of same data type, though the values stored in them may be different.
  - The entire array is stored contiguously in memory. This makes operations on array fast.
  - Each element of the array is identified or referred using the name of the Array along with the index of that element, which is unique for each element. The index of an element is an integral value associated with the element, based on the element's position in the array. For example consider an array with 5 numbers: [10, 9, 99, 71, 90]

Numpy Arrays

- numpy arrays are used to store lists of numerical data, vectors and matrices.
- The numpy package has a large set of built-in functions for creating, manipulating, and transforming arrays.
- Python language also has an array data structure, but it is not as versatile, efficient and useful as the numpy array.

Creation of a Numpy Array from List

- There are several ways to create arrays. To create an array and to use its methods, first we need to import the NumPy library.

```
#NumPy is loaded as np (we can assign any #name), numpy must be written in lowercase
>>> import numpy as np
```

The NumPy's array() function converts a given list into an array. For example,

```
#Create an array called array1 from the #given list.
>>> array1 = np.array([10,20,30])
#Display the contents of the array
>>> array1
array([10, 20, 30])
```

Creating 1-D array

An array with only single row of elements is called 1-D array. Let us try to create a 1-D array from a list which contains numbers as well as strings.

```
array2 = np.array([5,-7.4,'a',7.2])
```

Observe that since there is a string value in the list, all integer and float values have been promoted to string, while converting the list to array.

Creating 2-D array

We can create a two dimensional (2-D) arrays by passing nested lists to the array() function.

```
array3 = np.array([[2.4,3], [4.91,7],[0,-1]])
```

#To display array3

```
 array3
array([[2.4,3.],
[ 4.9 1, 7. ],
[ 0. , -1. ]])
```

Observe that the integers 3, 7, 0 and -1 have been promoted to floats.

Slicing Arrays

Sometimes we need to extract part of an array. This is done through slicing. We can define which part of the array to be sliced by specifying the start and end index values using [start : end] along with the array name.

```
>>> array2
array([-2, 2, 6, 10, 14, 18, 22])
# excludes the value at the end index
>>> array2[3:5]
array([10, 14])
# reverse the array
>>> array2[ : : -1]
array([22, 18, 14, 10, 6, 2, -2])
```

Slicing 2-D arrays

Let us create a 2-D array called my2Darray having 3 rows and 4 columns.

```
>>> my2Darray = np.array([[ -7, 0, 10, 20],
    [ -5, 1, 40, 200],
    [ -1, 1, 4, 30]])
```

```
# access all the elements in the 3rd column
```

```
>>> my2Darray[0:3,2]
array([10, 40, 4])
```

```
# access elements of 2nd and 3rd row from 1st # and 2nd column
```

```
>>> my2Darray[1:3,0:2]
array([[-5, 1],
[-1, 1]])
```

If row indices are not specified, it means all the rows are to be considered. Likewise, if column indices are not specified, all the columns are to be considered.

```
>>>my2Darray[:,2]
array([10, 40, 4])
```

Numpy Append

Numpy append is used to add values to an existing array.

We use the keyword append()

Example

#To add a new row to the existing array. Ensure that the number of columns matches that of the existing array

```
newRowArray= np.append(my2Darray, [[2, 7, 8, 2]], axis=0)

print(newRowArray)
```

#To add a new column to the existing array. Ensure that the number of rows matches that of the existing array

```
newColumnArray= np.append(my2Darray, [[2], [7], [8]], axis=1)

print(newColumnArray)
```

Numpy Sort

We can sort an numpy aray in both ascending and descending order using the function sort()

Example

```
myarray= np.array([22,33,6, -2,44, 2, 10 ])

newNumpyArraySort = np.sort(myarray)
```

In order to sort in descending order, we use the reverse slicing [::-1]

```
ReversedArraySort = np.sort(myarray)[::-1]
```

Reshaping Arrays

We can modify the shape of an array using the reshape() function. Reshaping an array cannot be used to change the total number of elements in the array.

```
array1 = np.arange(10,22)

print(array1 )

array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21])


#reshape

array2 = array1.reshape(3,4)

print(array2)

array([[10, 11, 12, 13],

       [14, 15, 16, 17],

       [18, 19, 20, 21]])
```

Statistical Operations on Arrays

- Let us consider two arrays:

```
>>> arrayA = np.array([1,0,2,-3,6,8,4,7])

>>> arrayB = np.array([[3,6],[4,2]])
```

The `max()` function finds the maximum element from an array.

```
>>> arrayA.max()
```

The `min()` function finds the minimum element from an array.

```
>>> arrayA.min()

-3

>>> arrayB.min(axis=0)

array([3, 2])
```

- The `sum()` function finds the sum of all elements of an array.

```
>>> arrayA.sum()

25

#axis is used to specify the dimension #on which sum is to be made. Here axis = 1 #means the sum of elements on the first row

>>> arrayB.sum(axis=1)

array([9, 6])
```

The `mean()` function finds the average of elements of the array.

```
>>> arrayA.mean()

3.125
```

Questions

- What is NumPy ? How to install it?
- What is an array and how is it different from a list? What is the name of the built-in array class in NumPy ?
- What do you understand by rank of an ndarray?
- Create the following NumPy arrays:
  - a) A 1-D array called `zeros` having 10 elements and all the elements are set to zero.
  - b) A 1-D array called `vowels` having the elements 'a', 'e', 'i', 'o' and 'u'.
  - c) A 2-D array called ones having 2 rows and 5 columns and all the elements are set to 1 and dtype as int.
  - d) Use nested Python lists to create a 2-D array called `myarray1` having 3 rows and 3 columns and store the following data:
    - 2.7, -2, -19
    - 0, 3.4, 99.9
    - 10.6, 0, 13
  - e) A 2-D array called `myarray2` using `arange()` having 3 rows and 5 columns with start value = 4, step size 4 and dtype as float.

Using the arrays created in Question 4 above, write NumPy commands for the following:

- a) Find the dimensions, shape, size, data type of the items and itemsize of arrays zeros, vowels, ones, myarray1 and myarray2.
- b) Reshape the array ones to have all the 10 elements in a single row.
- c) Display the 2nd and 3rd element of the array vowels.
- d) Display all elements in the 2nd and 3rd row of the array myarray1.
- e) Display the elements in the 1st and 2nd column of the array myarray1.
- f) Display the elements in the 1st column of the 2nd and 3rd row of the array myarray1.
- g) Reverse the array of vowels.

Using the arrays created in Question 4 above, write NumPy commands for the following:

a) Divide all elements of array ones by 3.

b) Add the arrays myarray1 and myarray2.

c) Subtract myarray1 from myarray2 and store the result in a new array.

d) Multiply myarray1 and myarray2 elementwise.

Using the arrays created in Question 4 above, write NumPy commands for the following:

- a) Sort the array vowels in reverse.
- b) Sort the array myarray1 such that it brings the lowest value of the column in the first row and so on.

Create a 2-D array called myarray4 using arange() having 14 rows and 3 columns with start value = -1, step size 0.25 having.

Using the myarray4 created in the above questions, write commands for the following:

- a) Find the sum of all elements.
- b) Find the sum of all elements row wise.
- c) Find the sum of all elements column wise.
- d) Find the max of all elements.
- e) Find the min of all elements in each row.
- f) Find the mean of all elements in each row.
- g) Find the standard deviation column wise.