**Array**

import numpy as np

array1 = np.array ([10,12,14,16,20])

print (array1)

**2d Array**

import numpy as np

list1 = [[10,20,30],[50,60,70]]

print (list1)

a= np.array ([[10,20,30],[50,60,70]])

print (a)

**Array with different number of elements**

ar1 = np.array([[10,20,30,40],[50,60,70, np.nan]])

ar1

import numpy as np

list1 = [[10,20,30],[50,60,70,80]]

print (list1)

a= np.array ([[10,20,30],[50,60,70,80]])

print (a)

print (type(a))

list1 = [10,20,30,40,50]

import numpy as np

array1 = np.array (list1)

print (type(array1))

print (type(list1))

print (array1)

list1 = [10,20,30,40,50,'X']

import numpy as np

array1 = np.array (list1)

print (type(array1))

print (type(list1))

print (list1)

print (array1)

**Because string all elements converted to string**

list1.append ('x')

import numpy as np

array1 = np.array (list1)

print (type(array1))

print (type(list1))

print (array1)

**With Tuple**

Tup1 = (10,20,30,40,50,'x','y')

import numpy as np

array1 = np.array (Tup1)

print (type(array1))

print (type(Tup1))

print (Tup1)

print (array1)

**Changes in the sub list won’t impact main list**

list1 = [10,20,30,40,50,60,70,80]

list2 = list1 [:3]

print (list2)

list2[1]=500

print (list2)

print (list1)

**If we update sliced array , parent array also gets impacted**

array1 = np.array (list1)

print (array1)

array2 = array1 [:3]

print (array2)

array2 [0] =500

print (array1)

print (array2)

**But in Copy parent array won’t impacted**

print (array1)

array2 = (array1 [0:3]).copy()

print (array2)

array2[1]=200

print (array1)

print (array2)

**Matrix**

import numpy as np

a = np.arange(9)

a = a.reshape((3,3))

print (a)

print (a[1][1])

b = np.zeros((5,5))

b = np.ones((3,5))

b

**F for Columns , C for rows**

mat1 = np.arange (0,30)

mat2 =np.reshape (mat1,(6,5),'F')

print (mat1)

print (mat2)

**Create Matrix from Three List and Transpose**

list1 = [10,20,30,40]

list2 = [100,110,120,130]

list3 = [200,210,220,230]

Mat1 = np.array ([list1, list2,list3])

print (Mat1)

print (Mat1.T)

**MAT Division**

list1 = [500,510,520,530]

list2 = [100,110,120,130]

list3 = [200,210,220,230]

Mat1 = np.array ([list1, list2,list3])

list6 = [10,11,12,13]

list7 = [20,21,22,23]

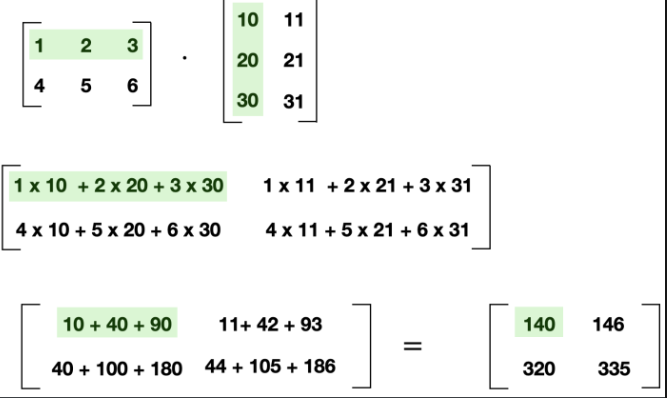
list8 = [30,31,32,33]

Mat2 = np.array ([list6, list7,list8])

Mat3 = Mat1/Mat2

print (Mat3)

**MAT Multiplication**



**Flatten the Array**

print (mat2.ravel())

a = np.arange(30)

b = a.reshape(2,5,3)

print (b)

new1 = b.reshape(30,)

print (new1)

**Linear Spaced Vector**

vector = np.linspace(0,30,11)

print (vector)

**Access Sales with dictionary**

EU\_Sales = [1002, 4523,4123,3584]

APAC\_Sales = [7854,9654,4582,8543]

US\_Sales = [7845,4512,3652,7452]

Sales = np.array ([EU\_Sales, APAC\_Sales,US\_Sales])

print (Sales)

region\_dict= {'EU':0,'APAC':1,'US':2}

Q\_dict ={'Q1':0,'Q2':1,'Q3':2,'Q4':3}

Sales [region\_dict['APAC']][Q\_dict['Q2']]

Slicing

import numpy as np

arr1= np.arange (100)

print (arr1[10:],'\n')

print (arr1[:20],'\n')

print (arr1[10:20],'\n')

print (arr1[10:50:5])

print (arr1[::10])

import numpy as np

arr1 = np.arange(20)

print (arr1)

arr\_slice = slice (1,10,2)

print (arr\_slice)

print (arr1[arr\_slice])

**Slicing**

import numpy as np

arr1= np.arange (100)

print (arr1[10:],'\n')

print (arr1[:20],'\n')

print (arr1[10:20],'\n')

print (arr1[10:50:5])

print (arr1[::10])

import numpy as np

arr1 = np.arange(20)

print (arr1)

arr\_slice = slice (1,10,2)

print (arr\_slice)

print (arr1[arr\_slice])

**Extract 2 X2 array from 3 X 3**

arr1 = np.arange(9)

print (arr1)

arr1=arr1.reshape(3,3)

print ('\n')

print (arr1)

print ('\n')

arr2 = arr1[0:2,0:2]

print (arr2)

arr1 = np.arange(9)

arr1=arr1.reshape(3,3)

print (arr1)

print ('\n')

arr2 = arr1[1:3,1:3]

print (arr2)

**Array Operators**

print (arr1)

print (arr1.shape)

print (arr1.ndim)

print (arr1.itemsize)

print (arr1.dtype)

print (arr1.size)

print (arr1.min())

print (arr1.max())

print (arr1.sum())

print (np.sqrt(arr1))

print (np.std(arr1))

**Empty Array**

arr5 = np.empty([3,2], dtype = float)

print (arr5)

arr5 = np.empty([3,2], dtype = int)

print (arr5)

arr5 = np.empty([3,2], dtype = bool)

print (arr5)

arr5 = np.empty([3,2], dtype = str)

print (arr5)

**Write Data to File**

arr1 = np.arange (9)

arr2 = arr1.reshape (3,3)

np.savetxt('Sample.txt',arr2)

Load from File

arr3=np.loadtxt('Sample.txt')

print (arr3)

**Save CSV**

np.savetxt ('Sample2.csv',arr2,delimiter = ';' )

**Load Data**

arr5=np.loadtxt('Sample3.csv',delimiter = ';')

print (arr5)

arr5=np.loadtxt('Sample2.csv',delimiter = ',')

print (arr5)

**Save 1D text data with fmt parameter**

ar1 = np.array (['Sun', 'Mon', 'Tue'])

ar1

np.savetxt('Sample5.txt',ar1,fmt=['%s'])

**Print all numpy commands**

import numpy

for name in dir(numpy):

print (name, end = '\t')

**University of South Australia**

**Create Two-Dimensional Array**

import numpy as np

data = np.array ([[5, -10,-2], [4,3,9]])

print (data)

print (type(data))

print (data.ndim)

**Multiply each element in the array**

print (data \*10)

x = data \*10

print (x)

**Addition of two arrays**

data1 = data + data

print (data1)

**Data type of each element**

print (data.dtype)

**Print Array Dimension : Size of Array**

print (data.shape)

**Create Array from a Sequence**

datal = [6, 7.5, 8, 0, 1]

arrl = np. array ( datal )

print (arrl)

**Type of the Array**

print (type (arrl))

**Create a Two dimensional Array with ndim and Shape**

data2 = [[1, 2,3, 4], [5,6,7, 8]]

arr2 = np.array(data2)

print (arr2)

print (arr2.ndim)

print (arr2.shape)

**convert a list of equal-length lists into an ndarray**

**Create One dimensional Array – Vector**

arr1 = np.zeros(10)

print (arr1)

**Create Two dimensional Array**

arr2 = np.zeros((3,6))

print (arr2)

print (arr2.dtype)

**Create Three-dimensional Array**

arr3= np.empty((2, 3, 2))

print (arr3)

print (arr3.dtype)

**Create Array from range() function**

arr4 = np.arange (15)

print (arr4)

**Difference Array and as array**

**Array**

list1= [10,20,30]

arr1= np.array(list1)

arr11= np.array(arr1)

arr11[1] =100

print (list1)

print (arr1)

print (arr11)

**AS Array**

list2 =[50,60,70]

arr2= np.asarray(list2)

arr21= np.asarray(arr2)

arr21[1] =100

print (list2)

print (arr2)

print (arr21)

**Ones Like Example (only integer array cab be referred for ones\_like)**

a= np.arange(30)

a=a.reshape(3,10)

print (a)

x=np.ones\_like(a)

print (x)

**Can not refer string array for ones\_like- It will be error)**

st1 = np.array ([['A','B','C'], ['X','Y','Z']])

print (st1)

st2= np.ones(st1)

print (st2)

**Zeros Like similar to ones Like**

a= np.arange(30)

a=a.reshape(3,10)

print (a)

x=np.zeros\_like(a)

print (x)

**Example Empty**

x = np.empty ([5,2])

print (x)

**Example Empty\_Like**

a = ([1,2,3], [4,5,6])

x= np.empty\_like(a)

print (x)

**Eye - Return a 2-D array with ones on the diagonal and zeros elsewhere.**

x = np.eye(5, dtype=int)

print (x)

x = np.identity(4, dtype=float)

print (x)

**Create Array with dtype**

arr1 = np.array([1, 2, 3], dtype=np.float64)

arr2 = np.array([1, 2, 3], dtype=np.int32)

print (arr1)

print (arrl.dtype)

print ('\n')

print (arr2)

print (arr2.dtype)

A **signed integer** is one with either a plus or minus sign in front. That is it can be either positive or negative. An **unsigned integer** is assumed to be positive. This is important in computing because the numbers are stored (usually) as a fixed number of binary digits

**astype() – Type Casting**

**Convert Int to Float**

arr = np.array([1, 2, 3, 4, 5])

print (arr)

print (arr.dtype)

print ('\n')

float\_arr = arr.astype(np.float64)

print (float\_arr)

print (float\_arr.dtype)

**Convert Float to Int – Truncate Decimal Values**

arr = np.array([3.7, -1.2,-2.6,0.5,12.9,10.1])

print (arr)

print (arr.dtype)

print ('\n')

arr =arr.astype(np.int32)

print (arr)

print (arr.dtype)

print ('\n')

**Convert a list of strings to numbers**

numeric\_strings = np.array(['1.25','-9.6','42'], dtype=np.string\_)

print (numeric\_strings)

print (numeric\_strings.dtype)

print ('\n')

float\_values = numeric\_strings.astype(float)

print (float\_values)

print (float\_values.dtype)

**use another array’s dtype to typecast**

int\_array = np.arange(10)

print (int\_array)

print (int\_array.dtype)

print ('\n')

float\_array = np.array([.22, .270, .357, .380, .44, .50], dtype=np.float64)

# Converting to another array type

int\_array=int\_array.astype (float\_array.dtype)

print (int\_array)

print (int\_array.dtype)

print ('\n')