DAY-27

OCTOBER-15

Arrays in numpy:

- Array is homogenous in nature.
  Examples:
  1.arr3 = np.array([1,2,3.7,8,4.6])
  o/p: array([1. , 2. , 3.7, 8. , 4.6])
  The whole array is converted into float because in float we can represent both integer values and float values.
  2. arr4 = np.array([1,'a',4.6,9])
  o/p: array(['1', 'a', '4.6', '9'], dtype='<U32')
- To find the type of the variable: type(arr4) o/p:numpy.ndarray
- To create 2 dimensional array:
  arr_2d = np.array([[9,5,1],[8,2,0],[4,3,7]])
  o/p:
  [[9 5 1]
  [8 2 0]
  [4 3 7]]

- To find the dimensions of the array we use array_name.ndim
- To check the number of elements in the array we use array_name.size
- To check the dimensions of matrix we use array_name.shape
- To check the type of elements of array we use array_name.dtype
- To create 3 dimensional array:
  arr_3d = np.array([[[1,2,3],[4,5,6]]])
  o/p:
  array([[[1, 2, 3],
  [4, 5, 6]]])
- Create column matrix using built-in functions
  col_mat = np.array([[1],[2],[3],[6],[6]])
  o/p:
  array([[1],
  [2],
  [3],
  [6],

[6]])

- To convert row matrix to column matrix
  row_mat = np.array([[1,2,3,4,5,6]])
  row_mat.shape
  cm = row_mat.reshape(6,1)
  o/p:
  array([[1],
  [2],
  [3],
  [4],
  [5],
  [6]])
- Different types of matrices
  1. Zeros matrix: np.zeros((3,2),dtype=int)
  2. Ones matrix: np.ones((3,3),dtype=int)
  3. Matrix with desired element: np.full((2,2),7)
  4. Matrix with range: np.arange(1,7,1)
  5. Matrix with range in column format: np.arange(1,10).reshape(3,3)
- Different functions related to matrices:
  Linspace: np.linspace(1,2,5,dtype=int)
  Eye: np.eye(5,dtype=int)
  random.rand: np.random.rand(3,3)
  random.randint: np.random.randint(10,50,size=(3,3))
  empty: np.empty((3,3),dtype=int)
  identity: np.identity(5,dtype=int)

- Linspace- creates matrix with same difference between one element to the other.
- Eye – creates identity matrix of desired size
- random.rand – creates matrix with random values of desired size
- random.randint – creates matrix with random integer values of desired size
- empty – creates matrix of desired shape with garbage values
- identity - creates identity matrix of desired size

1. create 2 arrays perform all basic math operations
   a1 = np.array([[2,3,4]])
   a2 = np.array([[3,4,5]])

   print(a1+a2)
   print(a2-a1)
   print(a1*a2)
   print(a1/a2)
   print(a1//a2)
   o/p:
   [[5 7 9]]
   [[1 1 1]]
   [[6 12 20]]
   [[0.66666667 0.75   0.8   ]]
   [[0 0 0]]

2. Take one array and perform all these functions
   Universal Functions:
   l = np.array([[20,30,40]])
   Sqrt() - sr = np.sqrt(l)
   print(sr)
   [[4.47213595 5.47722558 6.32455532]]
   exp() - exp = np.exp(l)
   [[4.85165195e+08 1.06864746e+13 2.35385267e+17]]
   log()- log = np.log(l)
   [[2.99573227 3.40119738 3.68887945]]
   sin()- sin = np.sin(l)
   [[0.91294525 -0.98803162 0.74511316]]
   median()- med = np.median(l)
   30.0
   mean()-mean = np.mean(l)
   30.0
   std()-std = np.std(l)
   8.16496580927726