# Numpy

`import numpy as np`

`arr = np.array([3,6,37,7])` 1-D array, accessing 2nd element, `arr[1]`

`arr = np.array([[3,6,37,7]])`, 2-D array, accessing 2nd element, `arr[0,1]`, 0th row 2nd column

`arr.shape` to get the shape of array

`arr.size`, to get total no. of elements in numpy array

`arr.dtype`, data type of elements in the array

`arr.ndim`, number of dimensions of array

`arr[0,1]=45`, modifying an element

`np.zeros((2,5))` an 2,5 matrix with all zero elements

`np.identity(45)`, an identity matrix of shape 45

`np.arange(15)`, creating an array of elements from 0 to 14

`np.linspace(1,5,12)`, dividing 1 to 5 in 12 elements

`np.empty((4,6))`, a (4,6) matrix with random elements

`arr.reshape(3,33)`, reshaping a matrix of size 99 and shape=(99,) to shape of (3,33) , size of both should be equal

`x= [[1,2,3],[4,5,6],[7,1,0]]`

`ar= np.array(x)`, list to np array

`ar.tolist()`, numpy array to list conversion

`ar.sum(axis=0 or axis=1)`, will sum the row/column elements and give a array of sums

`ar.T`, transposing an array

`ar.argmax()`, gives index of max element in whole array

`ar.argmax(axis=0)`, gives index of max element of every column in array form

`ar.argmin()`, gives index of min element

`ar.argsort()`, sorted array indices

`ar.sum()`, sum of all elements

`ar.max()`, maximum among all elements

`ar1+ar2`, summing two numpy arrays, not possible with list

`ar1*ar2`, multiplying two numpy arrays, not possible with list

`np.sqrt(ar)`, array with square root of every element

`np.where(ar>5)`, array of indexex where the condition satisfies, **important can be used as a conditional in a dataframe**

`np.count_nonzero(ar)`, count of non zero elements

`np.nonzero(ar)`, array of index where there's non zero elements