

Assignment-1

Based on NumPy

Q1: Questions on Basic NumPy Array

- (a) Reverse the NumPy array: `arr = np.array([1, 2, 3, 6, 4, 5])`
- (b) Flatten the NumPy arr: `array1 = np.array([[1, 2, 3], [2, 4, 5], [1, 2, 3]])` using any two NumPy in-built methods
- (c) Compare the following numpy arrays:
`arr1 = np.array([[1, 2], [3, 4]])`
`arr2 = np.array([[1, 2], [3, 4]])`
- (d) Find the most frequent value and their indice(s) in the following arrays:
 - i. `x = np.array([1,2,3,4,5,1,2,1,1,1])`
 - ii. `y = np.array([1, 1, 1, 2, 3, 4, 2, 4, 3, 3,])`
- (e) For the array `gfg = np.matrix('4, 1, 9; 12, 3, 1; 4, 5, 6')`, find
 - i. Sum of all elements
 - ii. Sum of all elements row-wise
 - iii. Sum of all elements column-wise
- (f) For the matrix: `n_array = np.array([[55, 25, 15],[30, 44, 2],[11, 45, 77]])`, find
 - i. Sum of diagonal elements
 - ii. Eigen values of matrix
 - iii. Eigen vectors of matrix
 - iv. Inverse of matrix
 - v. Determinant of matrix
- (g) Multiply the following matrices and also find covariance between matrices using NumPy:
 - i. `p = [[1, 2], [2, 3]]`
`q = [[4, 5], [6, 7]]`
 - ii. `p = [[1, 2], [2, 3], [4, 5]]`
`q = [[4, 5, 1], [6, 7, 2]]`
- (h) For the matrices: `x = np.array([[2, 3, 4], [3, 2, 9]])`; `y = np.array([[1, 5, 0], [5, 10, 3]])`, find inner, outer and cartesian product?

Q2: Based on NumPy Mathematics and Statistics

- (a) For the array: `array = np.array([[1, -2, 3],[-4, 5, -6]])`
 - i. Find element-wise absolute value
 - ii. Find the 25th, 50th, and 75th percentile of flattened array, for each column, for each row.
 - iii. Mean, Median and Standard Deviation of flattened array, of each column, and each row
- (b) For the array: `a = np.array([-1.8, -1.6, -0.5, 0.5, 1.6, 1.8, 3.0])`. Find floor, ceiling and truncated value, rounded values

Q3: Based on Searching and Sorting

- (a) For the array: `array = np.array([10, 52, 62, 16, 16, 54, 453])`, find
- Sorted array
 - Indices of sorted array
 - 4 smallest elements
 - 5 largest elements
- (b) For the array: `array = np.array([1.0, 1.2, 2.2, 2.0, 3.0, 2.0])`, find
- Integer elements only
 - Float elements only

Q4:

- (a) Write a function named `img_to_array(path)` that reads an image from a specified *path* and save it as text file on local machine? (Note: use separate cases for RGB and Grey Scale images)
- (b) Load the saved file into jupyter notebook?