Comprehensive List of

NumPy

Library

Rajendra Prasad JM

https://chlorinexxe.github.io/portfolio

1. Array Creation

- np.array(object) : Creates an array from an object
- np.zeros(shape) : Returns a new array of given shape and type, filled with zeros
- np.ones(shape) : Returns a new array of given shape and type, filled with ones
- np.empty(shape) : Returns a new array of given shape and type, without initializing entries
- np.full(shape,fill_value) : Returns a new array of given shape and type, filled with fill_value
- np.arrange(start,stop,step) : Returns evenly spaced values within a given interval
- np.linspace(start,stop,num) : Returns evenly spaced numbers over a specified interval
- np.logspace(start,stop,num) : Returns numbers spaced evenly on a log scale

2. Array Manipulation

- np.reshape( ) : Reshapes an array without changing its data
- np.transpose( ) : Permutes the dimensions of an array
- np.concatenate( ) : Joins a sequence of arrays along an existing axis
- np.stack( ) : Join a Sequence of array along axis
- np.split( ) : Split array into multiple sub-arrays
- np.flip( ) : Reverse the order of elements in an array
- np.roll( ) : Roll array elements along a specified axis

3. Mathematical Functions

- np.sum( ) : Computes the sum of array elements over a specified axis
- np.sqrt( ) : Returns the non-negative square-root of an array, element-wise
- np.sin( ) : Trigonometric sine, element-wise {cos( ) , tan( ) can be used}
- np.abs( ) : Calculate the absolute value element wise
- np.exp( ) : Calculate the exponential of all elements in the input array
- np.log( ) : Calculate the natural logarithm
- np.arcsin( ) : Calculate the Sin inverse , element wise {arccos( ) , arctan( ) can be used}
- np.power( ) : First array elements raised to powers from second array

4. Statistical Functions

- np.median( ) : Computes the median along the specified axis
- np.std( ) : Computes the standard deviation along the specified axis
- np.histogram( ) : Computes the histogram of a set of data
- np.percentile( ) : Computes the q-th percentile of the data along the specified axis
- np.mean( ) : Compute the arithmetic mean along the specified axis
- np.var( ) : Compute the variance along the specified axis
- np.std( ) : Compute the standard deviation along the specified axis
- np.corrcoef( ) : Return Pearson product-moment correlation coefficients

5. Linear Algebra

- np.dot( ) : Dot product of two arrays
- np.linalg.inv( ) : Computes the multiplicative inverse of a matrix
- np.linalg.eig( ) : Computes the eigenvalues and right eigenvectors of a square array
- np.linalg.svd( ) : Singular Value Decomposition
- np.linalg.det( ) : Compute the Determinant of a square matrix
- np.linalg.solve( ) : Solve a linear matrix equation or system of linear scalar equations
- np.linalg.norm( ) : Compute the matrix or Vector norm
- np.matmuls( ) : Compute the dot product of two arrays
- np.cross( ) : Returns the Cross product of 2 arrays
- np.inner( ) : Inner product of 2 arrays

6. Random Sampling

- np.random.rand( ) : Random values in a given shape
- np.random.randn( ) : Return a sample from the "standard normal" distribution
- np.random.randint( ) : Returns random integers from low to high
- np.random.choice( ) : Generates a random sample from a given 1-D array
- np.random.shuffle( ) : Modifies a sequence in-place by shuffling its contents
- np.random.permutation( ) : Randomly permute a sequence , or return a permuted range
- np.random.seed( ) : Seed the generator
- np.random.normal( ) : Draw random sample from a normal (Gaussian) Distribution.
- np.random.binomial( ) : Draw samples from a binomial distribution
- np.random.exponential( ) : Draw samples from a exponential distribution
- np.radom.poisson( ) : Draw samples from a Poisson distribution

7. Advanced Features

- np.broadcast_to( ) : Broadcasts an array to a new shape
- np.delete( ) : Deletes elements along an axis of an array
- np.insert( ) : Inserts values into an array at specified indices along an axis
- np.append( ) : Appends values to the end of an array
- np.isclose( ) : Returns a boolean array where two arrays are element-wise equal within a tolerance
- np.gradient( ) : Computes the gradient of a multidimensional array
- np.eye( ) : Creates a identity matrix

8. File Input and Output

- np.save( ) : Saves an array to a binary file in NumPy .npy format
- np.load( ) : Loads arrays or pickled objects from .npy, .npz, or pickled files
- np.loadtxt( ) : Load data from a text file
- np.savetxt( ) : Save array to a text file
- np.fromfile( ) : Read data from a binary file
- np.memmap( ) : Memory-mapped files for accessing large arrays stored on disk
- np.frombuffer( ) : to create arrays from raw bytes
- np.savez_compressed( ) : For saving multiple arrays into a single compressed .npz file

9. Special Functions

- np.special.( ) : A module containing various special mathematical functions such as Bessel functions, gamma functions, etc

  These are now in scipy.special

Scipy is a Python library used for scientific and technical computing. It builds on top of NumPy, another Python library, and provides a wide range of modules for tasks such as numerical integration, optimization, signal processing, statistics, and more.

“ I hope you found this information helpful!
Feel free to save this post for future reference.
Let's continue to learn and grow together! 🚀 ”

Thank You for Your Support

Rajendra Prasad JM

https://chlorinexxe.github.io/portfolio