Commonly Used NumPy functions List:

Array Creation:

- np.array(): Create an array from a list or tuple.
- np.zeros(shape): Create an array filled with zeros.
- np.ones(shape): Create an array filled with ones.
- np.arange(start, stop, step): Create an array with a range of values.
- np.linspace(start, stop, num): Create an array with evenly spaced values.
- np.random.rand(shape): Create an array with random values from a uniform distribution.
- np.random.randn(shape): Create an array with random values from a standard normal distribution.
- np.random.randint(low, high, size): Create an array with random integers within a specified range.

Array Manipulation:

- np.reshape(array, new_shape): Reshape an array.
- np.concatenate((array1, array2), axis): Concatenate arrays along a specified axis.
- np.split(array, indices_or_sections, axis): Split an array into multiple sub-arrays.
- np.hstack((array1, array2)): Stack arrays horizontally (column-wise).
- np.vstack((array1, array2)): Stack arrays vertically (row-wise).
- np.transpose(array): Transpose of an array.
- array.T: Shortcut for array transpose.

Mathematical Functions:

- np.sum(array, axis): Sum of array elements along a specified axis.
- np.mean(array, axis): Mean of array elements along a specified axis.
- np.std(array, axis): Standard deviation of array elements along a specified axis.
- np.min(array), np.max(array): Minimum and maximum values in an array.
- np.argmin(array), np.argmax(array): Indices of minimum and maximum values.
- np.dot(array1, array2): Dot product of two arrays.
- np.linalg.inv(array): Inverse of a square matrix.
- np.linalg.det(array): Determinant of a square matrix.

Universal Functions (ufuncs):

- Arithmetic functions: np.add(), np.subtract(), np.multiply(), np.divide(), np.power(), etc.
- Trigonometric functions: np.sin(), np.cos(), np.tan(), etc.
- Exponential and logarithmic functions: np.exp(), np.log(), np.log10(), etc.

Random Module Functions:

- np.random.seed(seed): Seed the random number generator for reproducibility.
- np.random.choice(array, size): Randomly sample from an array.
- np.random.shuffle(array): Shuffle the elements of an array in place.
- np.random.normal(mean, std_dev, size): Draw random samples from a normal distribution.

Linear Algebra:

- np.linalg.eig(array): Eigenvalues and eigenvectors of a square matrix.
- np.linalg.svd(array): Singular Value Decomposition of a matrix.
- np.linalg.solve(a, b): Solve a linear matrix equation, or system of linear scalar equations.
- np.linalg.qr(array): QR decomposition of a matrix.

Statistical Functions:

- np.percentile(array, q): Compute the qth percentile of the data along the specified axis.
- np.median(array, axis): Compute the median of the data along the specified axis.
- np.histogram(array, bins): Compute the histogram of a set of data.
- np.var(array, axis): Compute the variance of the data along the specified axis.
- np.corrcoef(array1, array2): Compute the correlation coefficient between two arrays.

Array Comparison:

- np.equal(array1, array2): Element-wise comparison if two arrays are equal.
- np.allclose(array1, array2): Compare if two arrays are element-wise equal within a tolerance.
- np.array_equal(array1, array2): Check if two arrays have the same shape and elements.

Set Operations:

- np.unique(array): Find the unique elements of an array.
- np.intersect1d(array1, array2): Find the intersection of two arrays.
- np.union1d(array1, array2): Find the union of two arrays.
- np.setdiff1d(array1, array2): Find the set difference of two arrays.

File I/O:

- np.loadtxt(filename, delimiter): Load data from a text file.
- np.genfromtxt(filename, delimiter): Load data with missing values handled as specified.
- np.savetxt(filename, array, delimiter): Save an array to a text file.

- np.save(filename, array): Save an array to a binary file in .npy format.
- np.load(filename): Load an array from a .npy binary file.

Polynomials:

- np.polyval(coefficients, x): Evaluate a polynomial at specific values.
- np.polyfit(x, y, degree): Fit a polynomial of a specified degree to the data.
- np.roots(coefficients): Find the roots of a polynomial equation.

Other Functions:

- np.vectorize(func): Convert a Python function into a vectorized function.
- np.apply_along_axis(func, axis, array): Apply a function along a specified axis of an array.
- np.meshgrid(x, y): Create coordinate matrices from coordinate vectors.
- np.gradient(array): Compute the gradient of an N-dimensional array.