Machine Learning Course

By Kaustubh Olpadkar

ML Libraries - NumPy

What is NumPy?

The NumPy is the fundamental library for *scientific computing* in Python.

It provides a high-performance multidimensional array object, and tools for working with these arrays.

Developed by: Travis Oliphant & community

An open-source project - written in : Python, C

Key Features

A powerful N-dimensional array object

Sophisticated Broadcasting functions

Tools for integrating C/C++ and Fortran code

Useful Linear algebra, Fourier transform, and random number capabilities

An efficient multi-dimensional container of generic data

Seamless and Speedy integration with a wide variety of databases due to support for arbitrarily defined data-types.

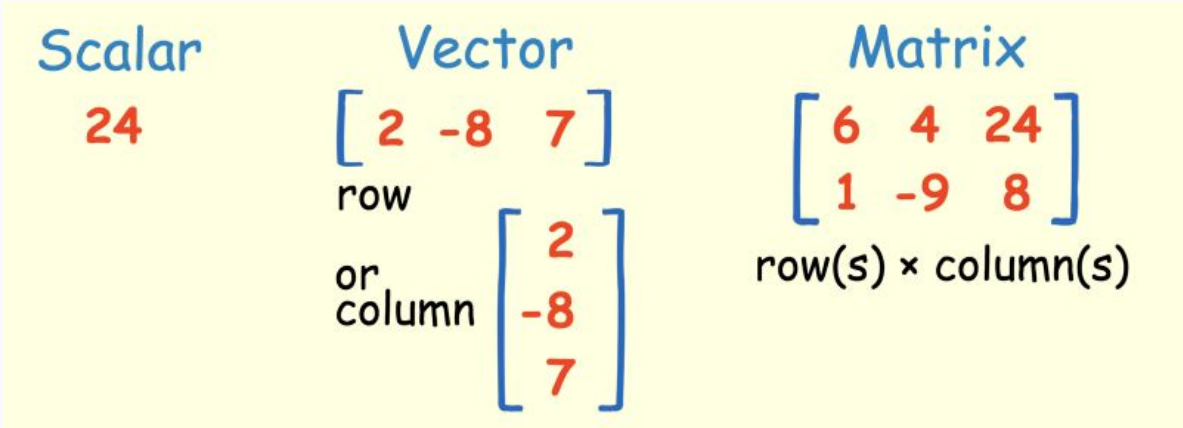
NumPy Arrays

NumPy's main object is the homogeneous multidimensional array.

Array of NumPy is called **ndarray**. It is also known by the alias **array**.

To use NumPy in your program, import the numpy package.

```
import numpy as np
```



`np.array([1,2,3])`	One dimensional array
`np.array([(1,2,3),(4,5,6)])`	Two dimensional array
`np.zeros(3)`	1D array of length 3 all values 0
`np.ones((3,4))`	3 x 4 array with all values 1
`np.eye(5)`	5 x 5 Identity matrix
`np.linspace(0,100,6)`	Array of 6 evenly divided values from 0 to 100
`np.arange(0,10,3)`	Array of values from 0 to less than 10 with step 3
`np.full((2,3),8)`	2 x 3 array with all values 8
`np.random.randn(2)`	Generate 2 random numbers between 0 – 1
`np.random.rand(4,5)`	4 x 5 array of random floats between 0–1
`np.random.randint(5,size=(2,3))`	2 x 3 array with random ints between 0–4

Creating Arrays

`a.shape`	Array Dimensions
`len(a)`	Length of Array
`b.ndim`	Number of array dimensions
`e.size`	Number of array elements
`b.dtype`	Data type of array elements
`b.dtype.name`	Name of data type
`b.astype(int)`	Convert an array to a different type

Array Properties

`a + b`	Addition
`np.add(a, b)`	
`a - b`	Subtraction
`np.subtract(a, b)`	
`a * b`	Multiplication
`np.multiply(a, b)`	
`a / b`	Division
`np.divide(a, b)`	
`np.random.randn(2)`	Generate 2 random numbers between 0 – 1
`np.random.rand(4,5)`	4 x 5 array of random floats between 0–1
`np.random.randint(5,size=(2,3))`	2 x 3 array with random ints between 0–4

Arithmetic Operations

`a.sum()`	Sum of Array elements
`a.min()`	Minimum of Array elements
`a.max()`	Maximum of Array elements
`a.mean()`	Mean / Average
`np.median(a)`	Median
`np.corrcoef(a)`	Correlation Coefficient
`np.std(a)`	Standard Deviation

Aggregate/Statistical Functions

Subsetting, Slicing, Indexing

Also see Lists

Subsetting

```
>>> a[2]
  3
```

Select the element at the 2nd index

```
>>> b[1,2]
  6.0
```

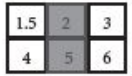
Select the element at row 0 column 2 (equivalent to `b[1][2]`)

Slicing

```
>>> a[0:2]
  array([1, 2])
```

Select items at index 0 and 1

```
>>> b[0:2,1]
  array([ 2.,  5.])
```



Select items at rows 0 and 1 in column 1

```
>>> b[:1]
  array([[1.5, 2., 3.]])
```

Select all items at row 0 (equivalent to `b[0:1, :]`)

```
>>> c[1,...]
  array([[[ 3.,  2.,  1.],
          [ 4.,  5.,  6.]]])
```

Same as `[1,:,:]`

```
>>> a[ : :-1]
  array([3, 2, 1])
```

Reversed array `a`

Boolean Indexing

```
>>> a[a<2]
  array([1])
```

Select elements from `a` less than 2

Fancy Indexing

```
>>> b[[1, 0, 1, 0],[0, 1, 2, 0]]
  array([ 4. , 2. , 6. , 1.5])
```

Select elements (1,0), (0,1), (1,2) and (0,0)

```
>>> b[[1, 0, 1, 0]][:,[0,1,2,0]]
  array([[ 4. ,5. , 6. , 4. ],
         [ 1.5, 2. , 3. , 1.5],
         [ 4. , 5. , 6. , 4. ],
         [ 1.5, 2. , 3. , 1.5]])
```

Select a subset of the matrix's rows and columns

Subsetting, Slicing and Indexing

Function	Description
`abs, fabs`	Compute the absolute value element-wise for integer, floating-point, or complex values
`sqrt`	Compute the square root of each element (equivalent to `arr ** 0.5`)
`square`	Compute the square of each element (equivalent to `arr ** 2`)
`exp`	Compute the exponent e^x of each element
`log, log10, log2, log1p`	Natural logarithm (base *e*), log base 10, log base 2, and log(1 + x), respectively
`sign`	Compute the sign of each element: 1 (positive), 0 (zero), or –1 (negative)
`ceil`	Compute the ceiling of each element (i.e., the smallest integer greater than or equal to that number)
`floor`	Compute the floor of each element (i.e., the largest integer less than or equal to each element)
`rint`	Round elements to the nearest integer, preserving the `dtype`
`modf`	Return fractional and integral parts of array as a separate array
`isnan`	Return boolean array indicating whether each value is NaN (Not a Number)
`isfinite, isinf`	Return boolean array indicating whether each element is finite (non-`inf`, non-NaN) or infinite, respectively
`cos, cosh, sin, sinh, tan, tanh`	Regular and hyperbolic trigonometric functions
`arccos, arccosh, arcsin, arcsinh, arctan, arctanh`	Inverse trigonometric functions
`logical_not`	Compute truth value of `not x` element-wise (equivalent to `~arr`).

Scientific Functions

Function	Description
`diag`	Return the diagonal (or off-diagonal) elements of a square matrix as a 1D array, or convert a 1D array into a square matrix with zeros on the off-diagonal
`dot`	Matrix multiplication
`trace`	Compute the sum of the diagonal elements
`det`	Compute the matrix determinant
`eig`	Compute the eigenvalues and eigenvectors of a square matrix
`inv`	Compute the inverse of a square matrix
`pinv`	Compute the Moore-Penrose pseudo-inverse of a matrix
`qr`	Compute the QR decomposition
`svd`	Compute the singular value decomposition (SVD)

Linear Algebra Functions - numpy.linalg

Function	Description
`seed`	Seed the random number generator
`permutation`	Return a random permutation of a sequence, or return a permuted range
`shuffle`	Randomly permute a sequence in-place
`rand`	Draw samples from a uniform distribution
`randint`	Draw random integers from a given low-to-high range
`randn`	Draw samples from a normal distribution with mean 0 and standard deviation 1 (MATLAB-like interface)
`binomial`	Draw samples from a binomial distribution
`normal`	Draw samples from a normal (Gaussian) distribution
`beta`	Draw samples from a beta distribution
`chisquare`	Draw samples from a chi-square distribution
`gamma`	Draw samples from a gamma distribution
`uniform`	Draw samples from a uniform [0, 1) distribution

Random Numbers & Probability Distributions - numpy.random

NumPy is Base

ML Libraries are built on its Top.

Thanks!

Contact:

Kaustubh Olpadkar
56, Narayan Orbis
Sun Pharma Road
Vadodara, 390012

kaustubh.olpadkar@gmail.com
www.simulations.ml