Python Libraries for Data Science

By Vaishali KUNJIR

Python Libraries for Data Science

- NumPy(Numerical Python)
- Pandas
- Matplotlib
- SciPy (Scientific Python)
- Scikit-learn

Data Science Python Libraries Applications

There are multiple stages in the data science process, and different libraries are used to help with each stage. Following are the statges of data science process:

- Extract, transform, load (ETL)
- Data exploration
- Data evaluation
- Data modeling
- Data presentation (or visualization)

NumPy
(https://numpy.org/devdocs/user/index.html#user)

What is Numpy?

1 Efficient scientific computing

Numpy is an open-source numerical computing library for Python. It enables fast and efficient computation with arrays and matrices.

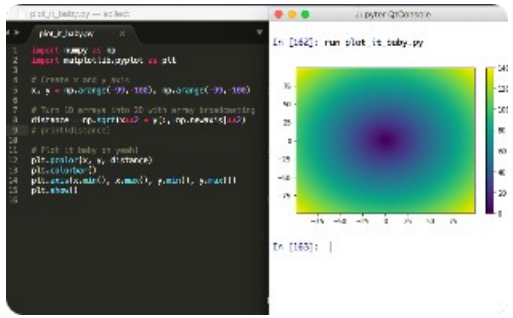
2 Central to data analysis

Numpy provides the backbone for many data analysis libraries, making it the foundation for several machine learning models and algorithms.

3 Trusted by the scientific community

Numpy has been actively developed and refined for over 15 years, making it a powerful tool for scientific computing and research.

Key Features of Numpy



Arrays and Matrices

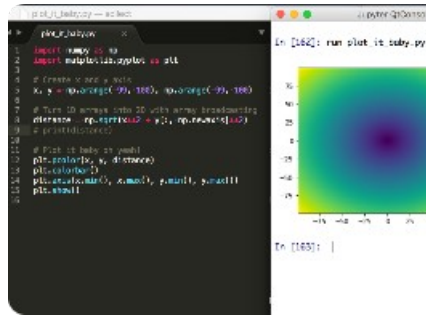
Numpy provides powerful data structures for performing mathematical operations on arrays

Mathematical Operations

Numpy offers a comprehensive set of mathematical functions for performing complex operations on arrays and matrices.

Random Number Generation

Numpy provides several functions for generating arrays with random numbers, which is useful in many applications such as



Broadcasting

Numpy's broadcasti feature improves co readability and enab us to perform matrix operations without actually creating a second matrix.

Common Numpy Functions and Examples

Creating Arrays and Matrices

Numpy makes creating and initializing arrays and matrices a breeze. For example, you can create an array filled with zeros with np.zeros((3,3)) or an identity matrix with np.eye(3).

Indexing and Slicing Arrays

Numpy offers advanced indexing and slicing features, enabling us to extract subsets of data based on specific criteria.

Statistical Operations with Numpy

Numpy offers several statistical functions, including mean, median, variance, and standard

Practical Applications of Numpy

1

Using Numpy for Data Analysis

Numpy is an essential tool for data analysis in fields ranging from finance to scientific research. You can use Numpy to perform computations on large datasets and perform advanced statistical analysis quickly and efficiently.

2

Implementing Machine Learning Algorithms with Numpy

Numpy's powerful array and matrix operations make it a robust tool for implementing machine learning models, such as linear regression, logistic regression, and neural networks.

3

Optimizing Numpy Performance

With careful implementation, numpy can offer blazing-fast performance, particularly

NumPy (Ndarray object)

- Ndarray is a collection of same data type elements
- Every element in a ndarray takes the same size of block in the memory
- Elements of array accessed using zero-based index
- Syntax: numpy.array(object*, dtype = None, copy = True, order = None, subok = False, ndmin = 0)

Sr.No.	Parameter & Description
1	**object** Any object exposing the array interface method returns an array, or any (nested) sequence.
2	**dtype** Desired data type of array, optional
3	**copy** Optional. By default (true), the object is copied
4	**order** C (row major) or F (column major) or A (any) (default)
5	**subok** By default, returned array forced to be a base class array. If true, sub-classes passed through
6	**ndmin** Specifies minimum dimensions of resultant array

More on Numpy Array operations

- numpy.sort() ==> Sorts elements
- numpy.concatenate() ==> Joins 2 arrays
- numpy.reshape(number of rows, number of columns) ==> Reshape the array
- ndarray.ndim ==> will tell you, number of axes/dimension of the array
- ndarray.size==> will tell you the total number of elements of the array
- ndarray.shape ==> will display a tuple of integers that indicate the number of elements stored along each dimension of the array. If, for example, you have a 2-D array with 2 rows and 3 columns, the shape of your array is (2, 3)

Creation of numpy array using existing data

Syntax: ***numpy.asarray(data, dtype=None, order=None)***

data = Input data in the form of list, list of tuples, tuples,
Dtype = Default data type of input data is applied
Order = C(row major and its default) or F(column major)

Example 1:

```
# convert list to ndarray
import numpy as np
x = [1,2,3]
a = np.asarray(x)
print a  #output [1 2 3]
```

Example 2:

```
# ndarray from tuple
import numpy as np
x = (1,2,3)
a = np.asarray(x)
print a # output [1 2 3]
```

What is difference between python list and numpy array?

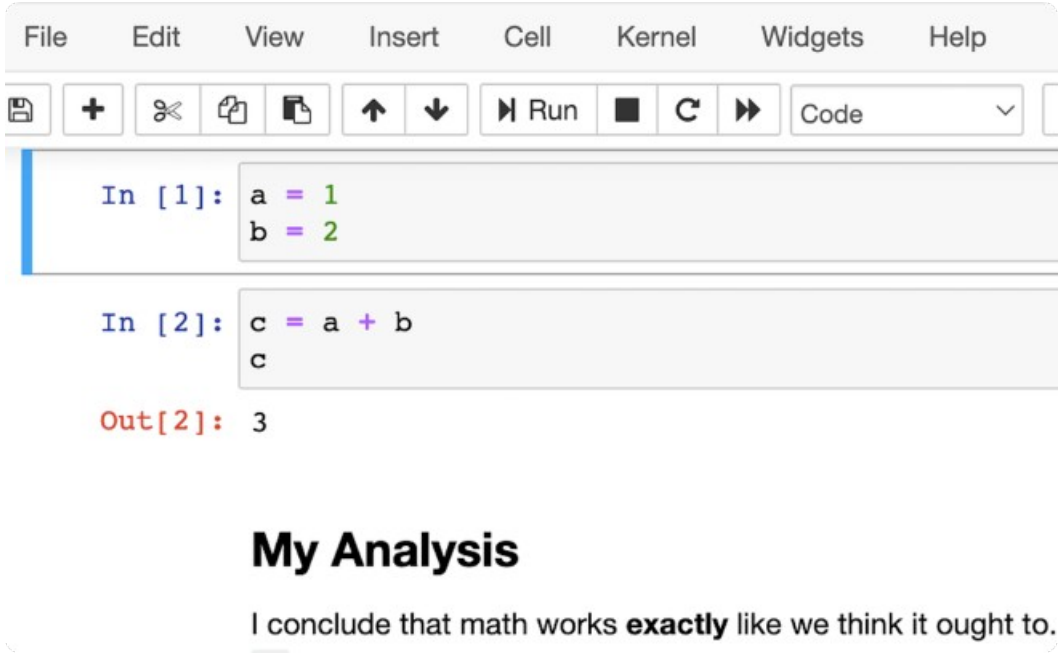
Numpy Array (https://numpy.org/doc/stable/reference/generated/numpy.array.html)

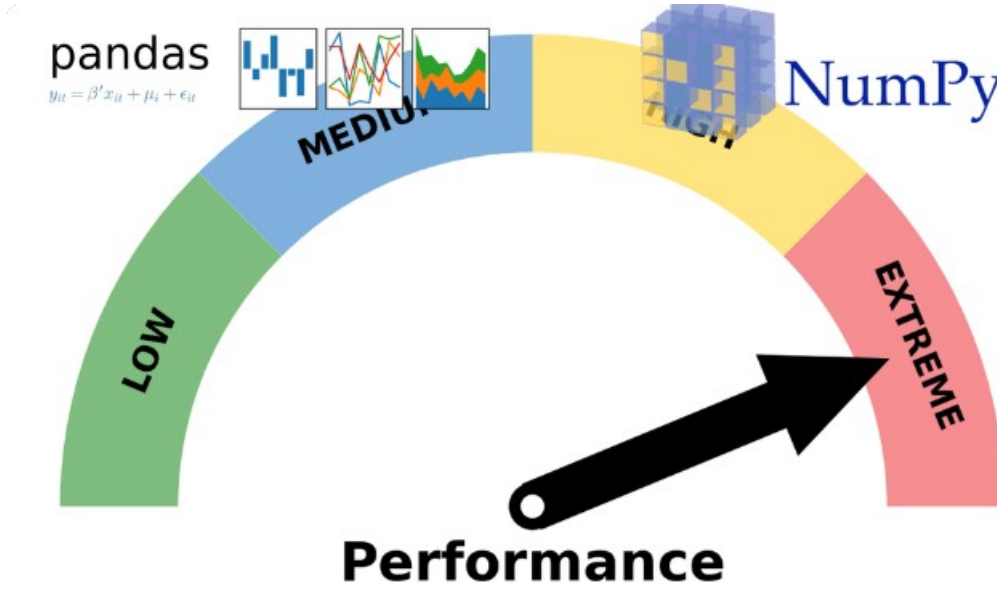
- NumPy array gives you an enormous range of fast and efficient ways of creating arrays and manipulating numerical data inside them.
- All the elements of numpy array are **homogeneous**.
  Example: [1,2,3,4,5]
- NumPy arrays are faster and more compact than Python lists
- An array consumes less memory and is convenient to use. NumPy uses much less memory to store data and it provides a mechanism of specifying the data types. This allows the code to be optimized even further.

Python List

- While a Python list can contain different data types within a single list.
- All the elements of python list are **heterogeneous**.
  Example: [1, 2.8, 3, 'hello']

Tips and Tricks for Using Numpy Effectively





Avoiding Common Pitfalls and Errors

One of the most significant pitfalls is improper array

Optimizing Numpy Performance

Numpy provides many built-in functions and

Useful links

- https://numpy.org/devdocs/user/absolute_beginners.html#adding-removing-and-sorting-elements
- https://numpy.org/devdocs/user/absolute_beginners.html#how-do-you-know-the-shape-and-size-of-an-array
- https://numpy.org/devdocs/user/absolute_beginners.html#indexing-and-slicing
- https://numpy.org/devdocs/user/absolute_beginners.html#how-to-create-an-array-from-existing-data
- https://numpy.org/devdocs/user/absolute_beginners.html#broadcasting
- https://numpy.org/devdocs/user/absolute_beginners.html#basic-array-operations
- https://numpy.org/devdocs/user/absolute_beginners.html#more-useful-array-operations
- https://numpy.org/devdocs/user/absolute_beginners.html#creating-matrices
- https://numpy.org/devdocs/user/absolute_beginners.html#transposing-and-reshaping-a-matrix
- https://numpy.org/devdocs/user/absolute_beginners.html#how-to-reverse-an-array
- https://numpy.org/devdocs/user/absolute_beginners.html#working-with-mathematical-formulas