BATCH : B150 Data Science

LESSON : **NUMPY**

DATE : 30.03.2023

SUBJECT : **Session 1- Introduction**

ZOOM GİRİŞLERİNİZİ LÜTFEN **LMS** SİSTEMİ ÜZERİNDEN YAPINIZ

DATA SCIENCE LIFECYCLE

sudeep.co

01
BUSINESS UNDERSTANDING
Ask relevant questions and define objectives for the problem that needs to be tackled.

02
DATA MINING
Gather and scrape the data necessary for the project.

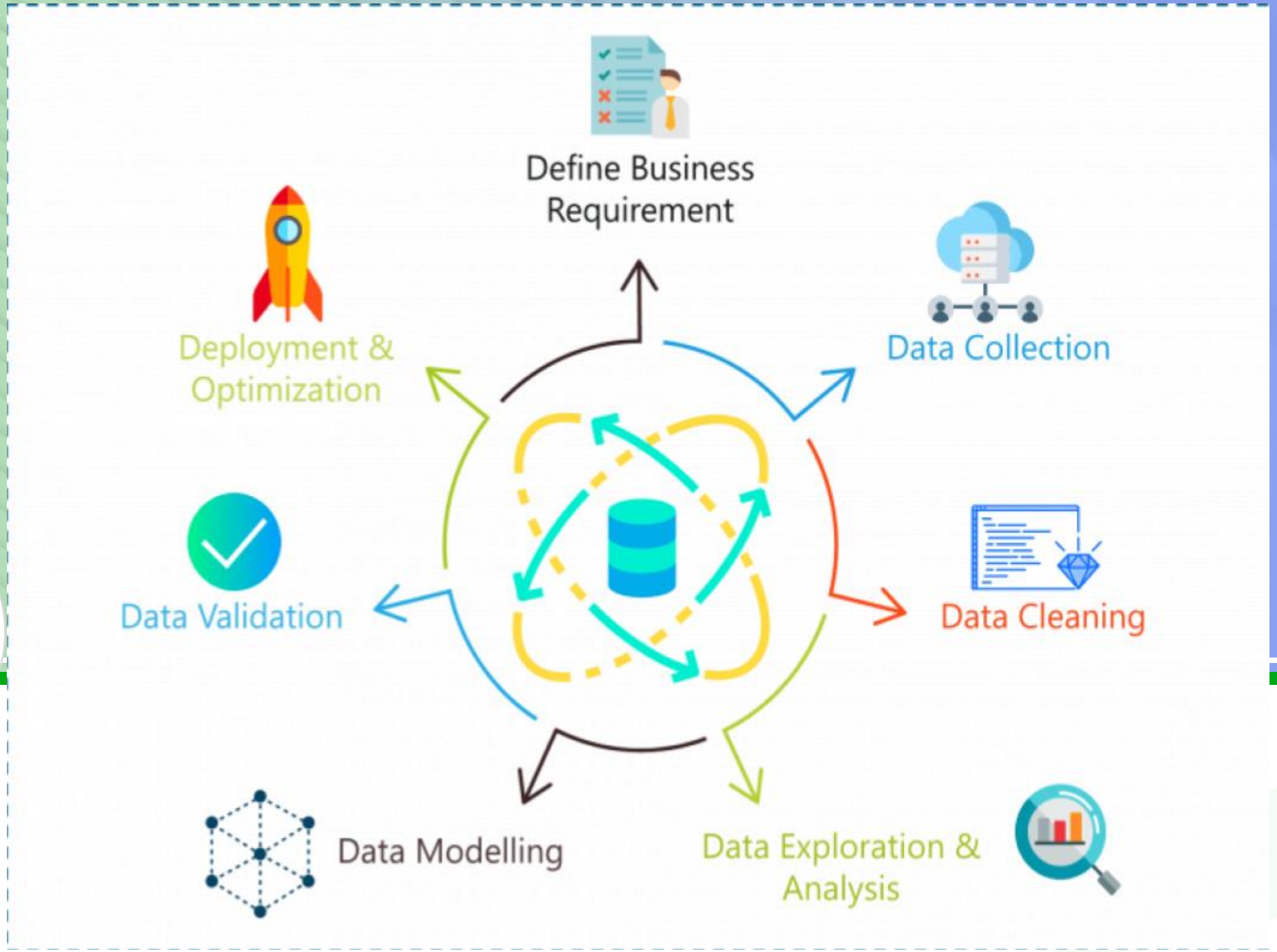
03
DATA CLEANING
Fix the inconsistencies within the data and handle the missing values.

04
DATA EXPLORATION
Form hypotheses about your defined problem by visually analyzing the data.

05
FEATURE ENGINEERING
Select important features and construct more meaningful ones using the raw data that you have.

06
PREDICTIVE MODELING
Train machine learning models, evaluate their performance, and use them to make predictions.

07
DATA VISUALIZATION
Communicate the findings with key stakeholders using plots and interactive visualizations.


Define Business Requirement
Data Collection
Data Cleaning
Data Exploration & Analysis
Data Modelling
Data Validation
Deployment & Optimization

Data Analytics

- Excel/Google Spreadsheets
- SQL
- BI Tools (Tableau, Power BI)
- Python ...

Artificial Intelligence

- Modelling
  - Prediction/Forecasting
    - Regression
    - Classification
    - Clustering...

NumPy Ecosystem

OpenCV | PySAL | numexpr | astropy

PyTables | statsmodels | Biopython

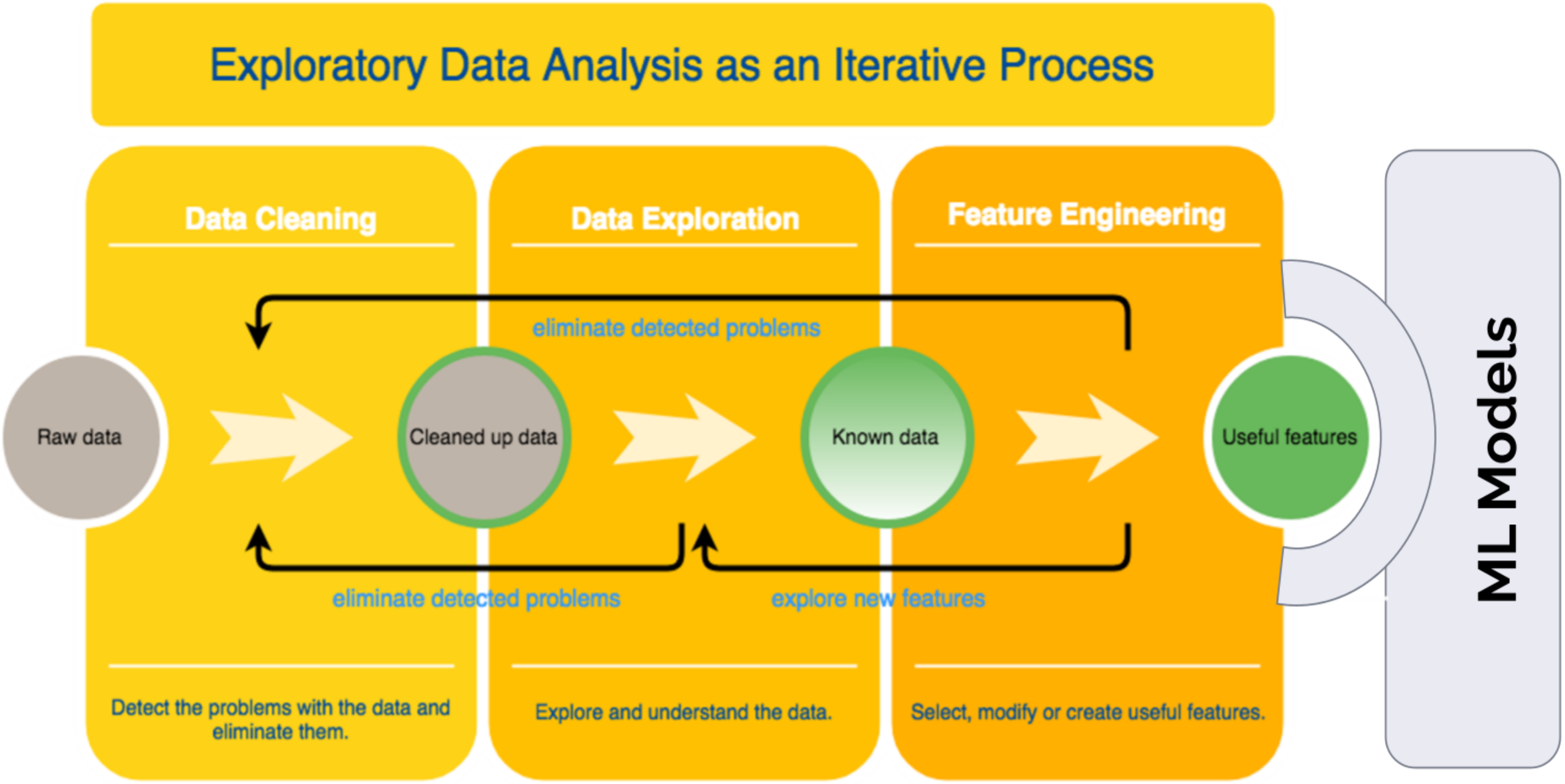
scikit-image | scikit-learn | Numba

Scipy | Pandas | Matplotlib

NumPy

Python Packages for Data Analysis

- **Numpy and Scipy** – fundamental scientific computing.
- **Pandas** – data manipulation and analysis.
- **Matplotlib** – plotting and visualization.
- **Scikit-learn**– machine learning and data mining.
- **StatsModels** – statistical modeling, testing, and analysis.


Exploratory Data Analysis as an Iterative Process
Data Cleaning
Data Exploration
Feature Engineering
eliminate detected problems
Raw data
Cleaned up data
Known data
Useful features
ML Models
eliminate detected problems
explore new features
Detect the problems with the data and eliminate them.
Explore and understand the data.
Select, modify or create useful features.

Numpy

Numerical Python

Çok boyutlu dizilerle ve matrislerle çalışmamızı sağlar

Matematiksel işlemler yapabiliriz

NumPy arrays

NumPy

Numpy

Neden NumPy Kullanılır?

- Daha hızlı
- Daha az döngü
- Daha açık kod
- Daha kaliteli kod

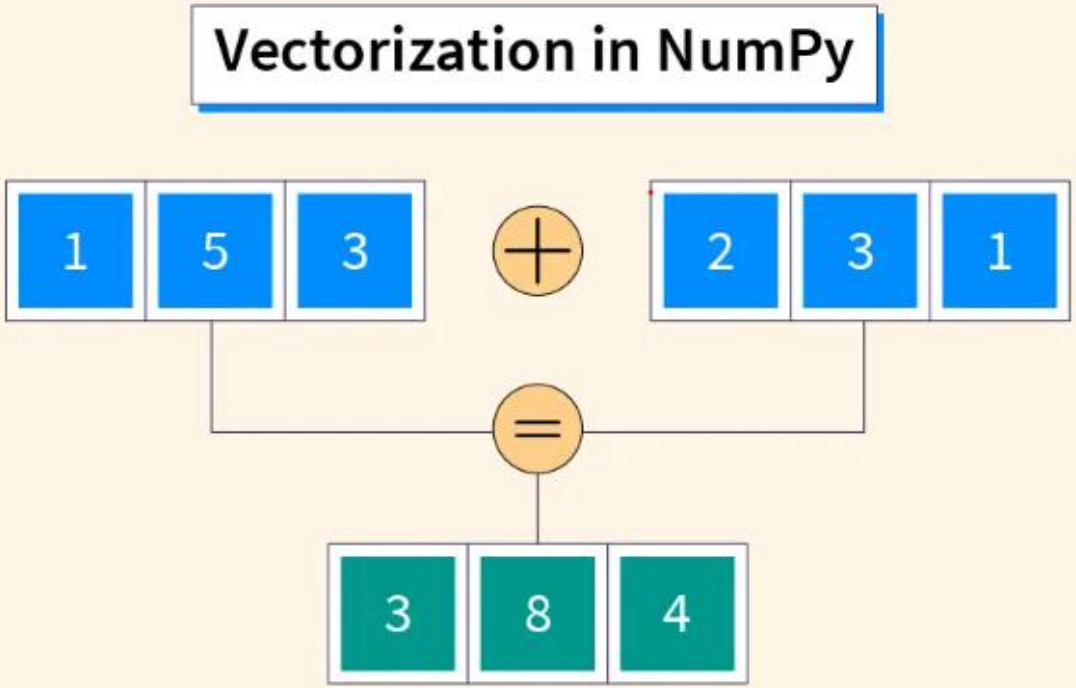
Numpy

NumPy Neden Listelerden Daha Hızlı?

- Bellek yönetimi
- Vektörel işlemler
- Dahili fonksiyonlar
- C veya C ++ dilinde yazılmış alt yapı

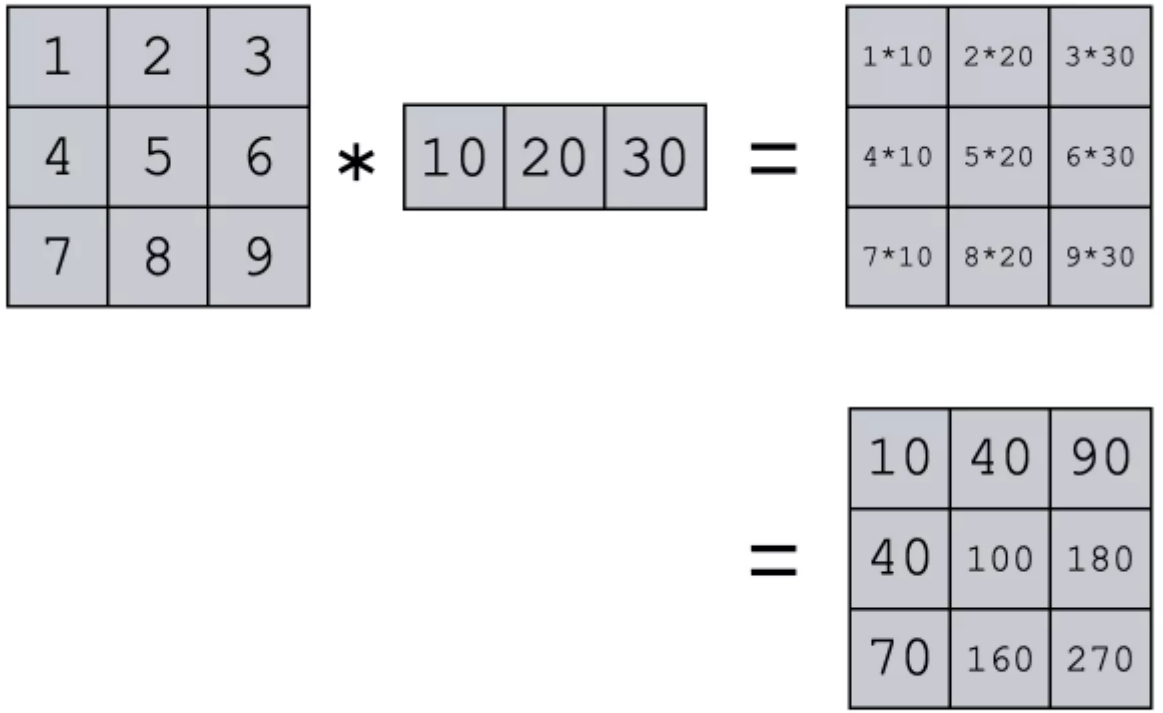
Numpy

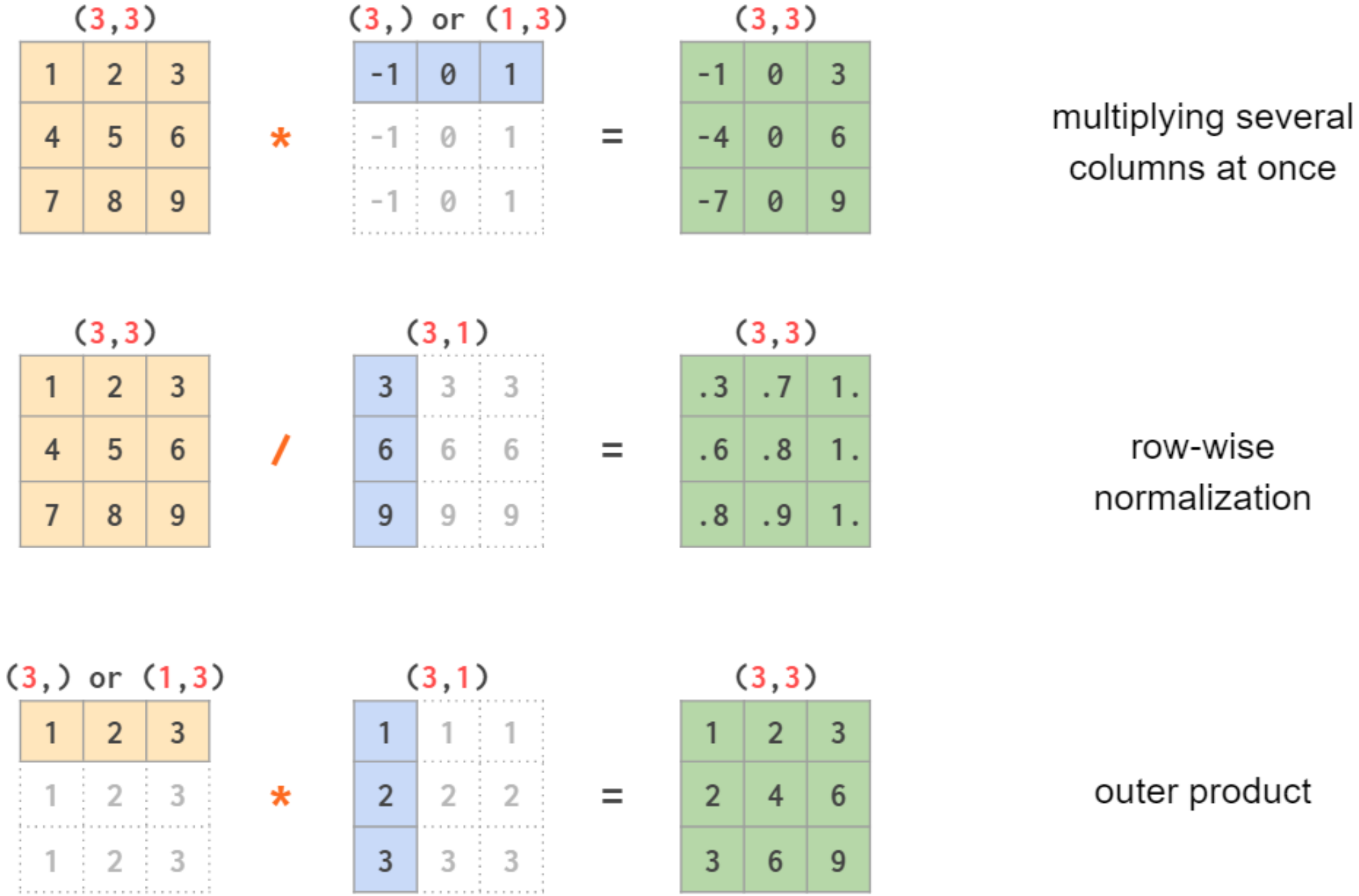
- **Vectorization**



Numpy

- **Broadcasting**

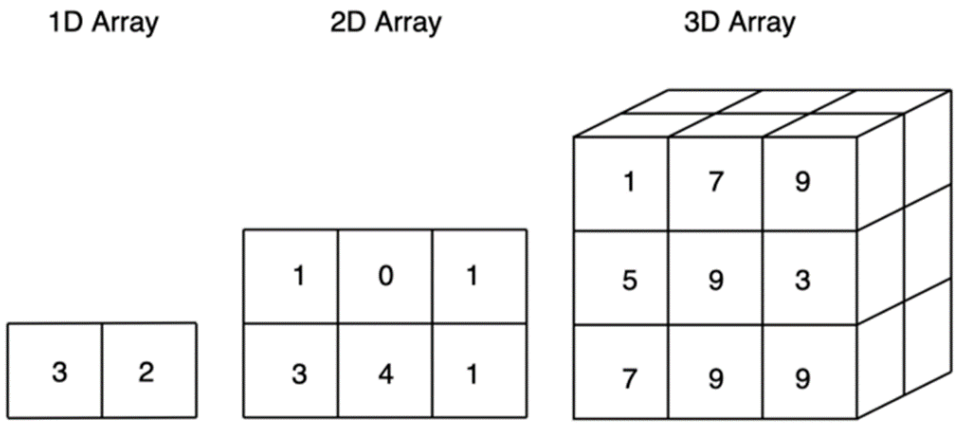



(3,3)
1 2 3
4 5 6
7 8 9
*
(3,) or (1,3)
-1 0 1
-1 0 1
-1 0 1
=
(3,3)
-1 0 3
-4 0 6
-7 0 9
multiplying several columns at once
(3,3)
1 2 3
4 5 6
7 8 9
/
(3,1)
3 3 3
6 6 6
9 9 9
=
(3,3)
.3 .7 1.
.6 .8 1.
.8 .9 1.
row-wise normalization
(3,) or (1,3)
1 2 3
1 2 3
1 2 3
*
(3,1)
1 1 1
2 2 2
3 3 3
=
(3,3)
1 2 3
2 4 6
3 6 9
outer product

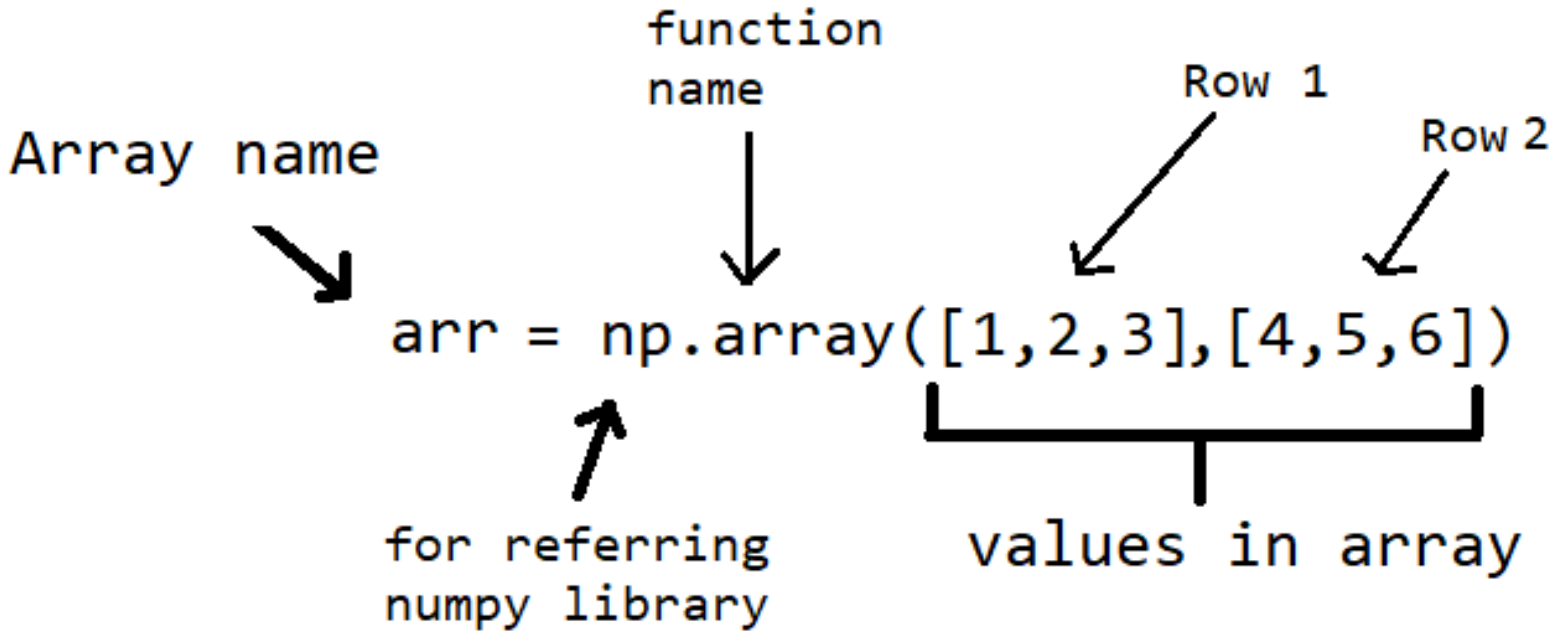
Numpy

- **pip install numpy**
- **import numpy as np**

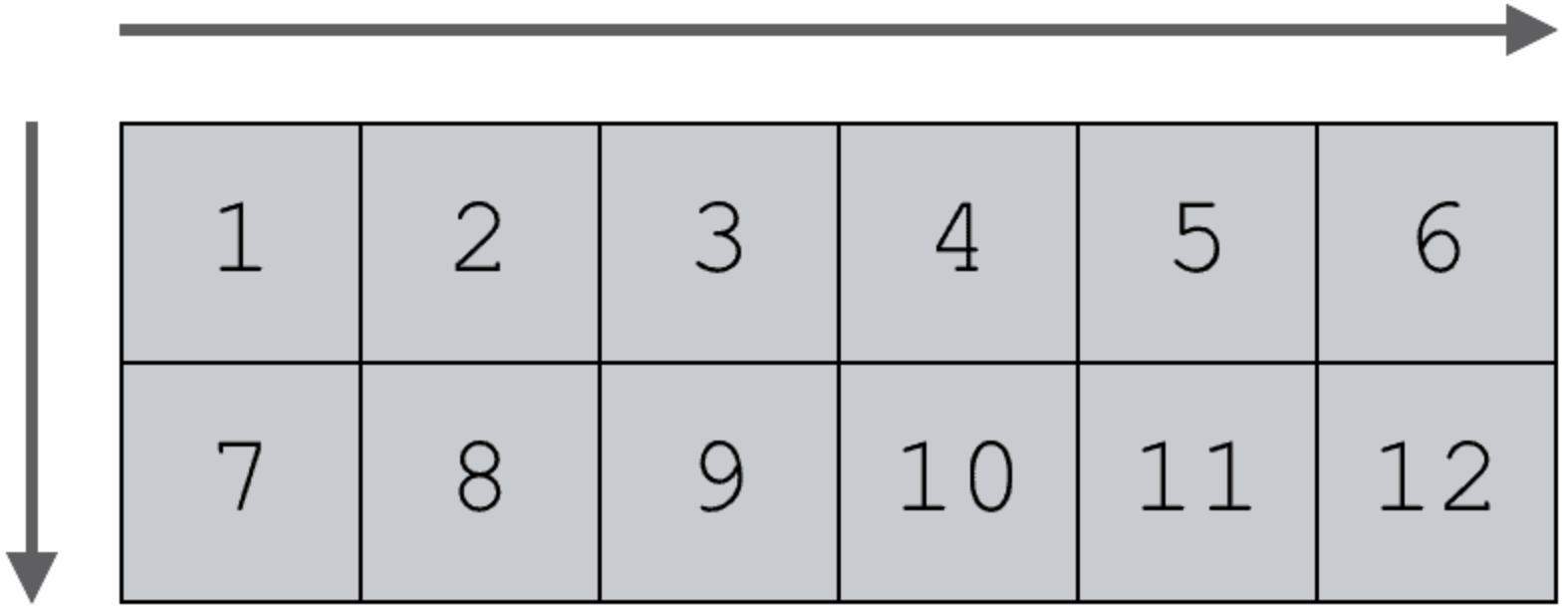
numpy.array()



Numpy


function
name
Row 1
Row 2
Array name
arr = np.array([1,2,3],[4,5,6])
for referring
numpy library
values in array

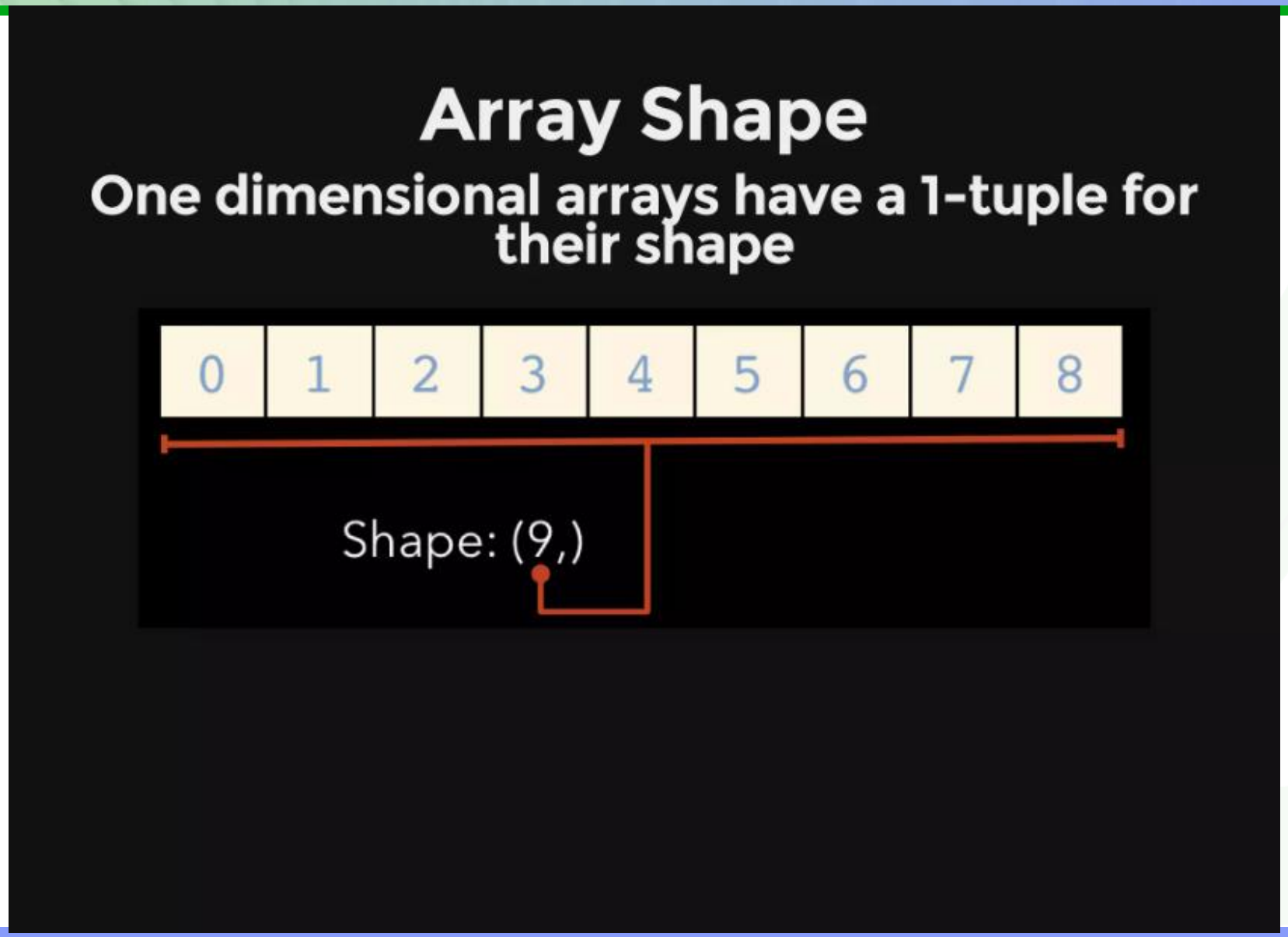
There are 6 columns



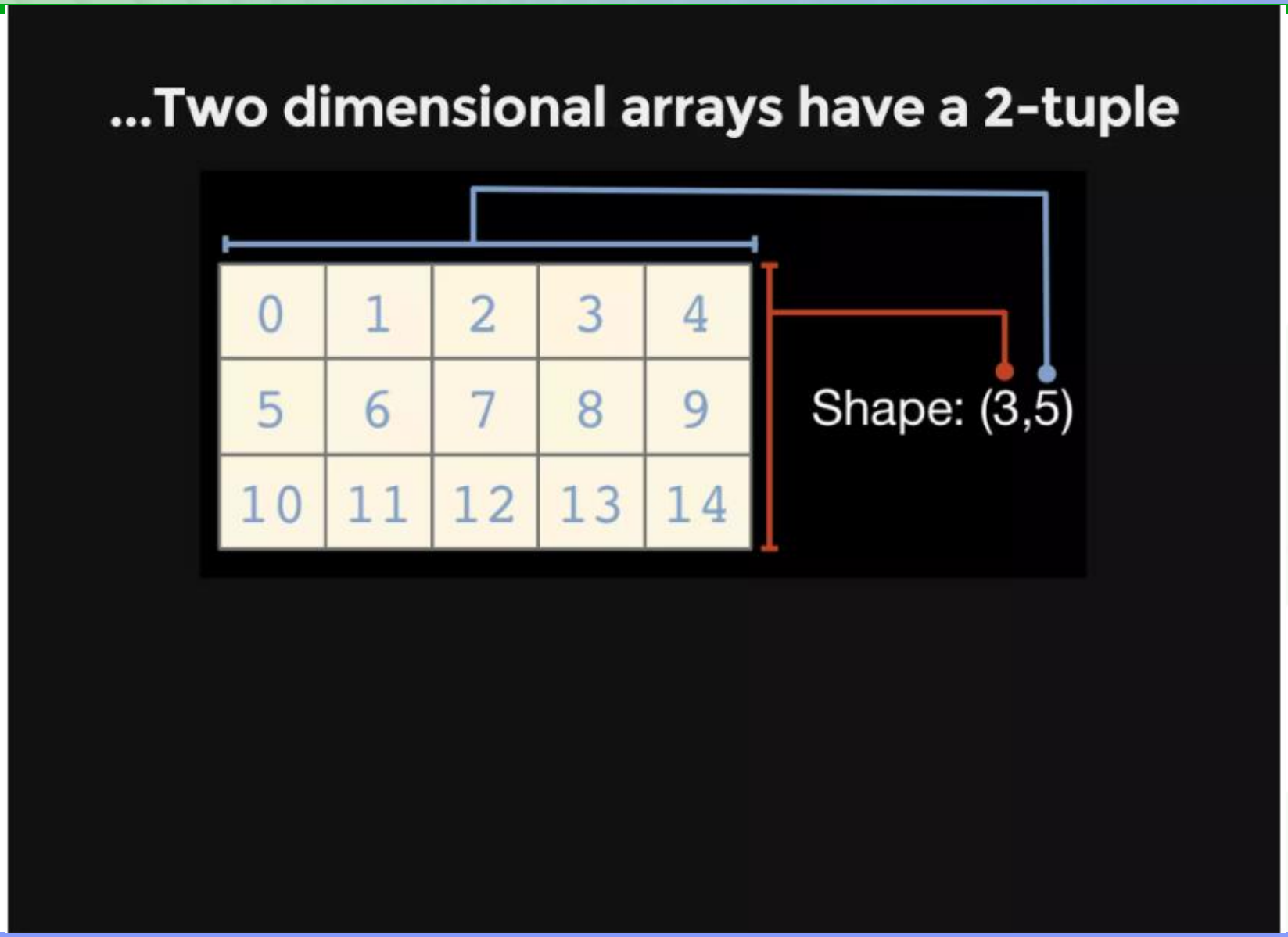
There are
2 rows

1	2	3	4	5	6
7	8	9	10	11	12

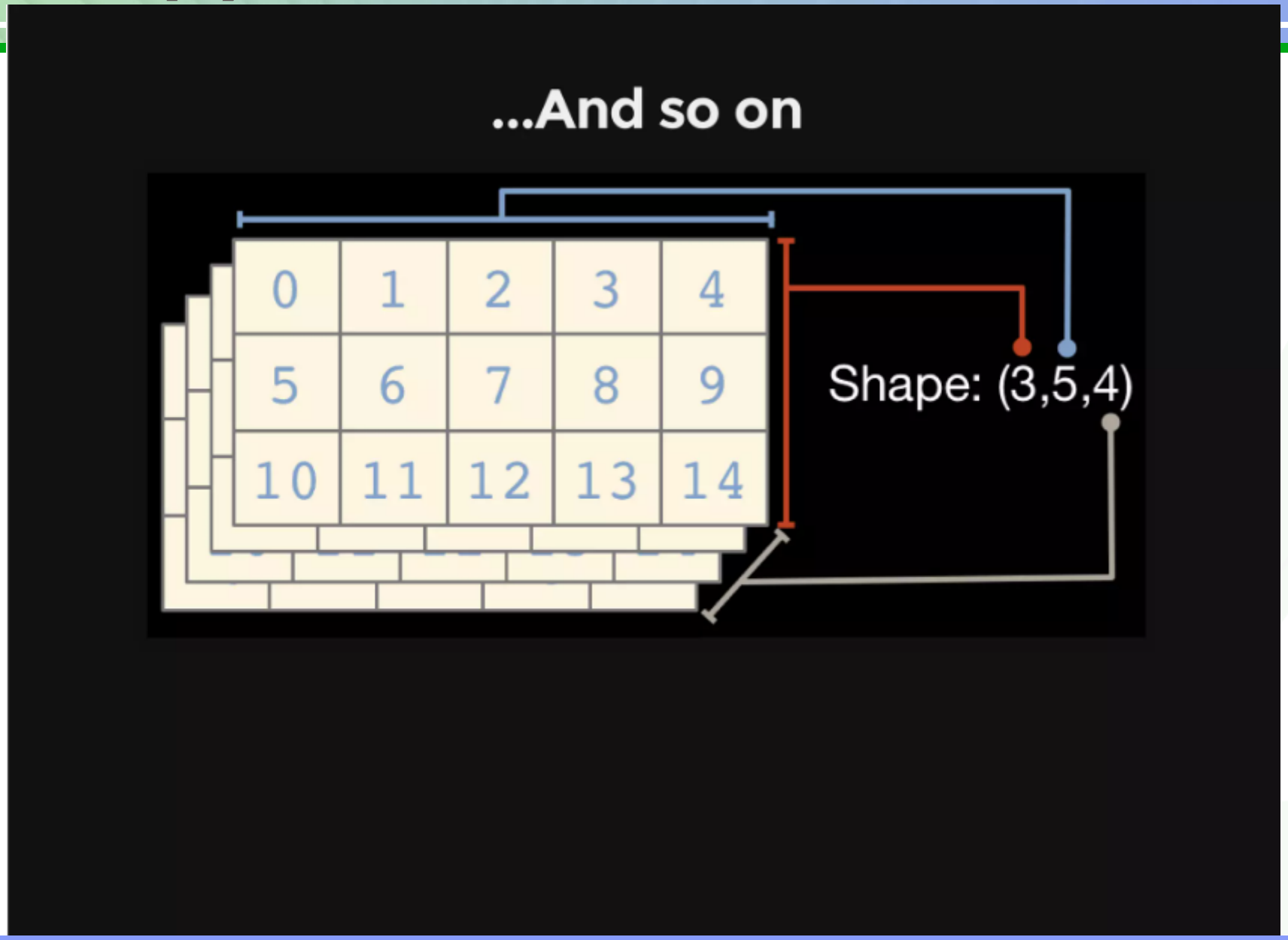
Numpy


Array Shape
One dimensional arrays have a 1-tuple for their shape
0
1
2
3
4
5
6
7
8
Shape: (9,)

Numpy

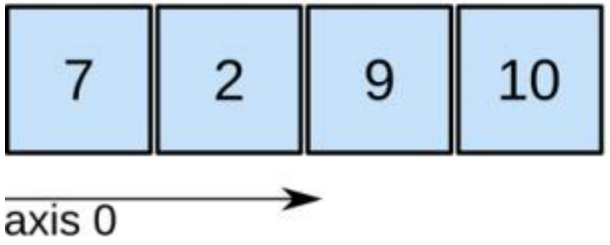

...Two dimensional arrays have a 2-tuple
0 1 2 3 4
5 6 7 8 9
10 11 12 13 14
Shape: (3,5)

Numpy



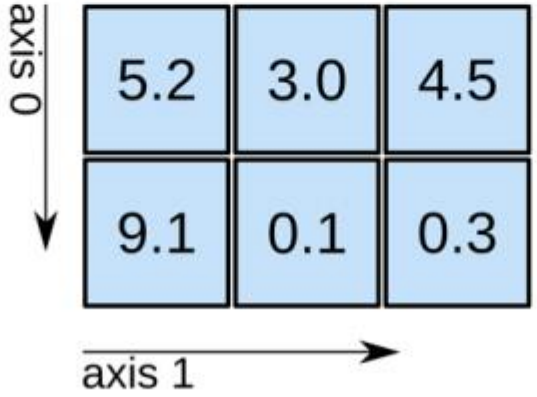
Numpy

1D array



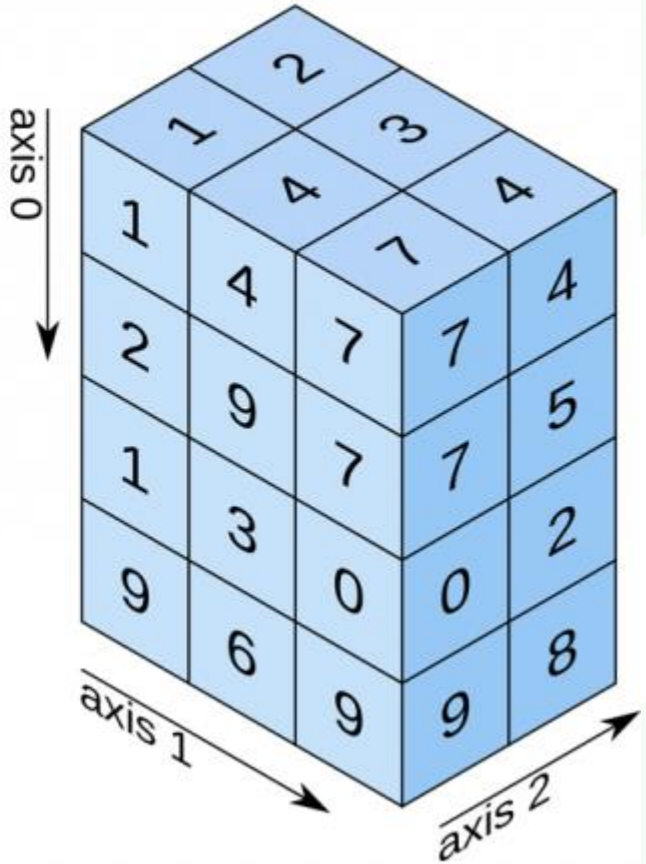
shape: (4,)

2D array



shape: (2, 3)

3D array

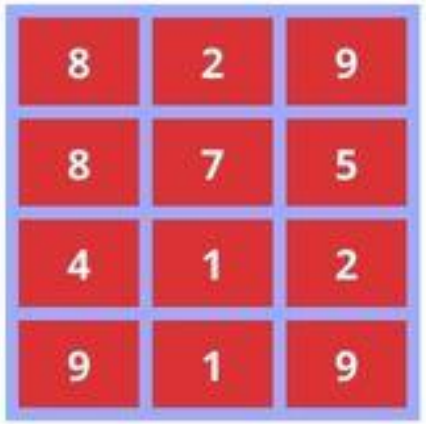


shape: (4, 3, 2)

Numpy

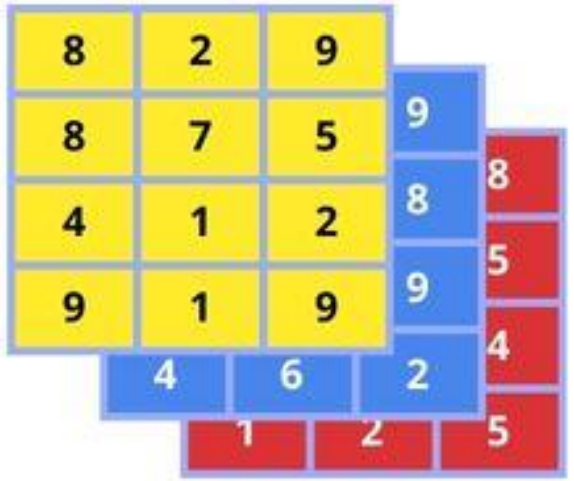
1

my_np.array([4,5,9])



2

my_np.array([[8,2,9],
[8,7,5],
[4,1,2],
[9,1,9]])



3

my_np.array([[[8,2,9],
[8,7,5],[4,1,2],[9,1,9]],
[[4,5,9],[4,2,8],
[2,1,9],[4,6,2]],
[[6,8,8],[2,4,5],
[7,8,4],[1,2,5]]])

Numpy

Array Creation Methods

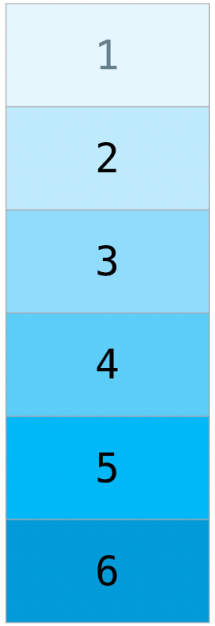
- arange
- zeros
- ones
- full
- empty
- linspace
- logspace
- eye
- random.rand
- random.randn
- random.randint

Numpy

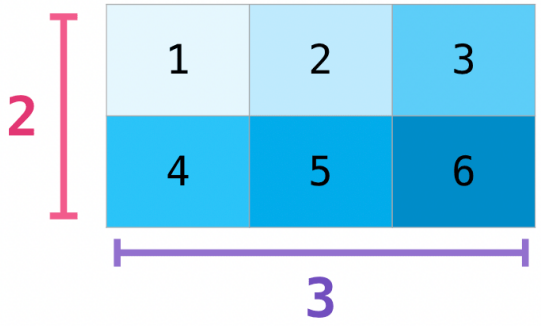
Array Methods

- shape
- reshape
- size
- resize
- ndim
- dtype-astype
- itemsize
- copy
- ravel
- transpose
- max-min

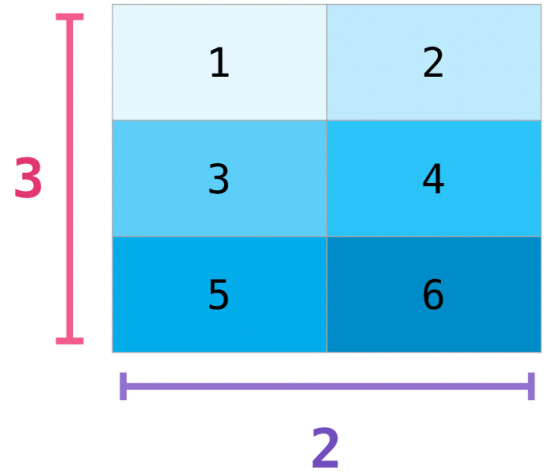
data

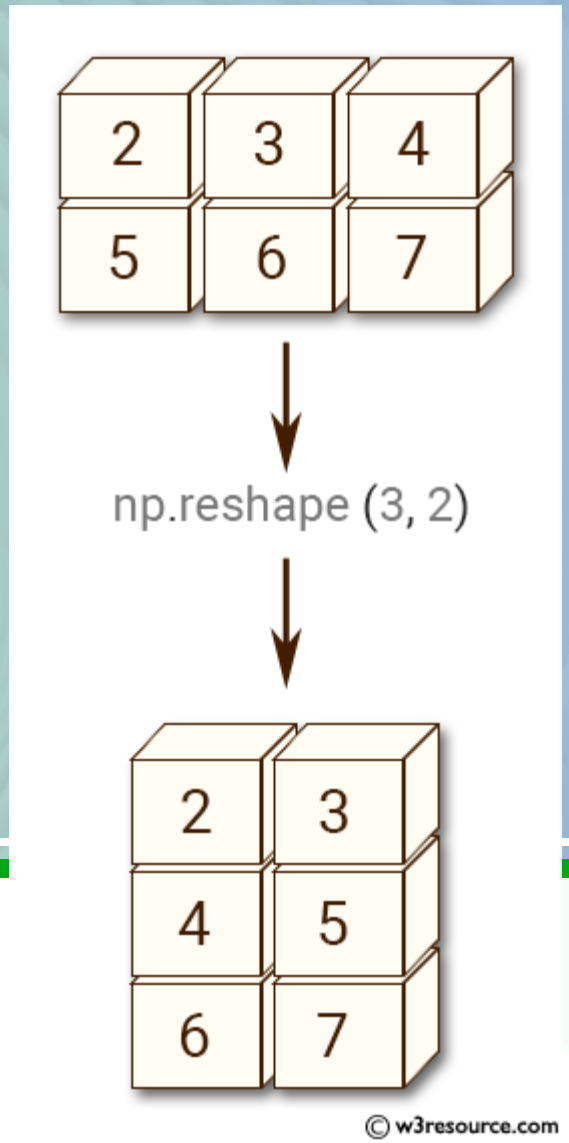


data.reshape(2,3)

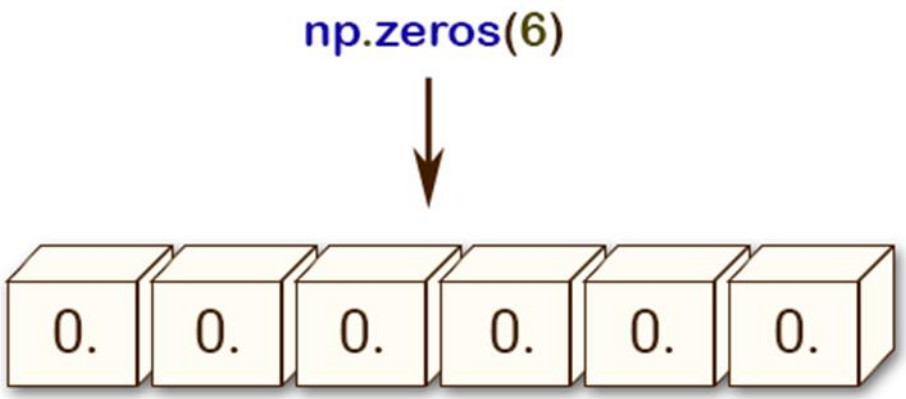


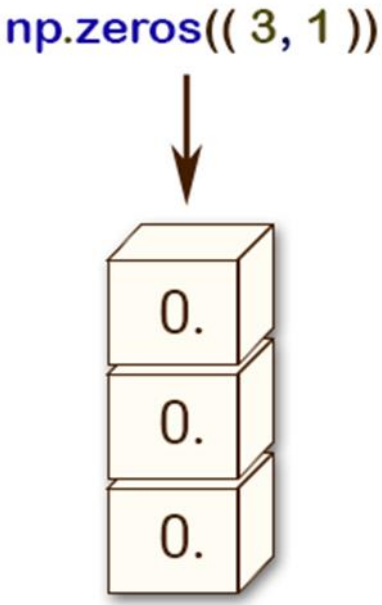
data.reshape(3,2)



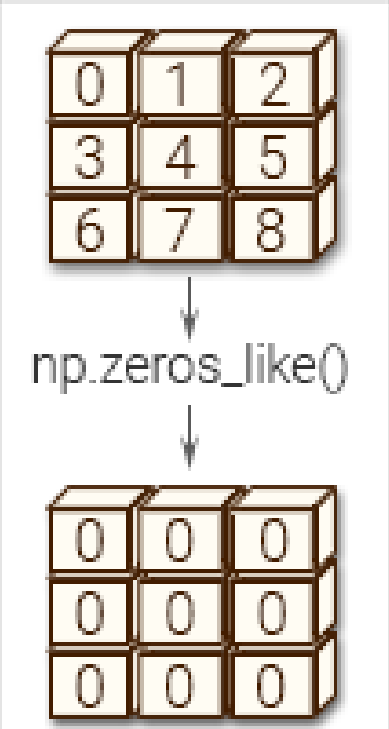

2
3
4
5
6
7
np.reshape (3, 2)
2
3
4
5
6
7
© w3resource.com

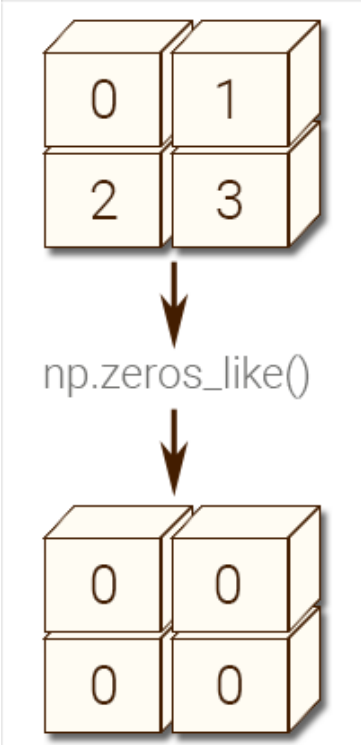
Numpy

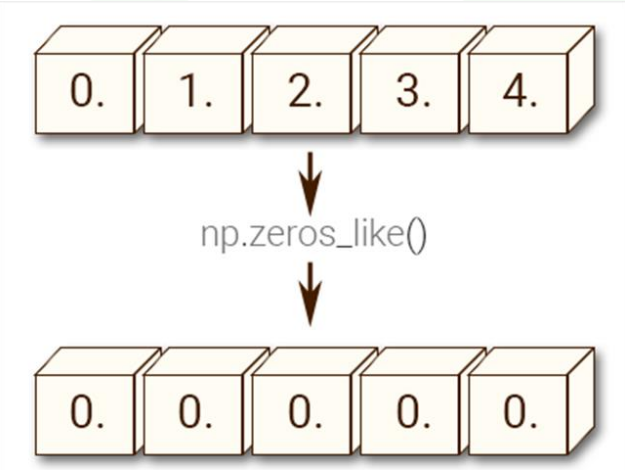




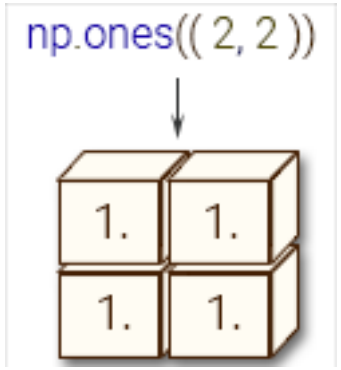
Numpy

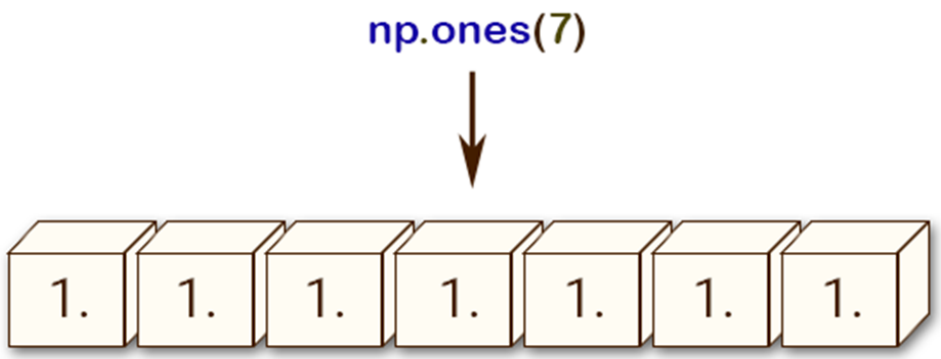

0 1 2
3 4 5
6 7 8
np.zeros_like()
0 0 0
0 0 0
0 0 0

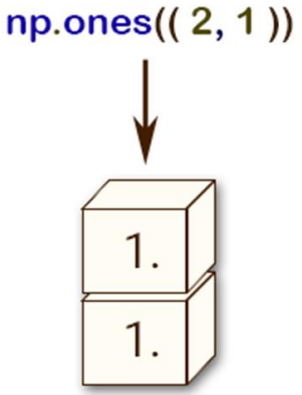

0 1
2 3
np.zeros_like()
0 0
0 0


0. 1. 2. 3. 4.
np.zeros_like()
0. 0. 0. 0. 0.

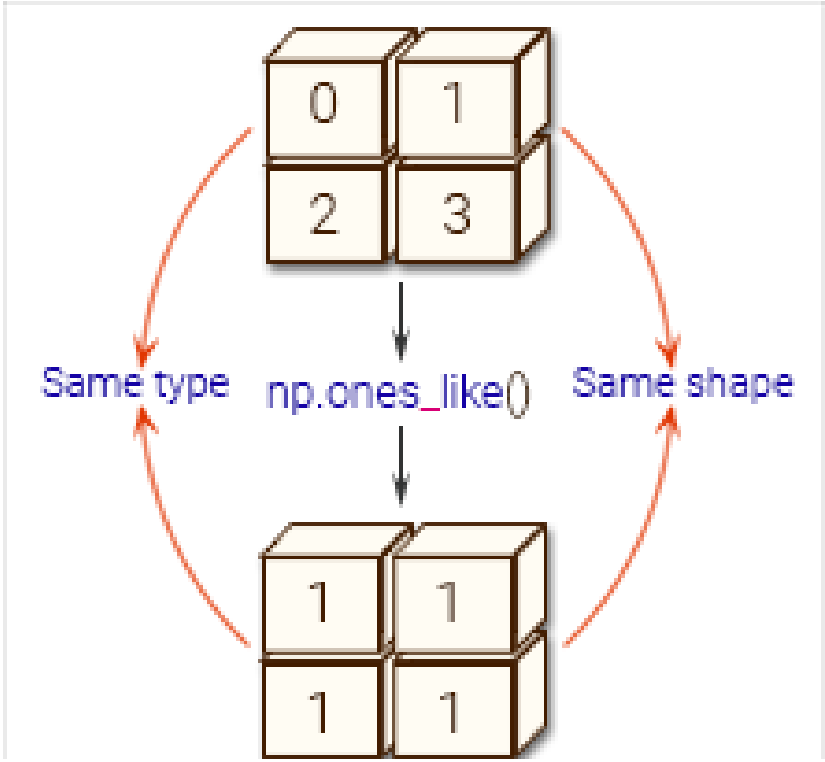
Numpy

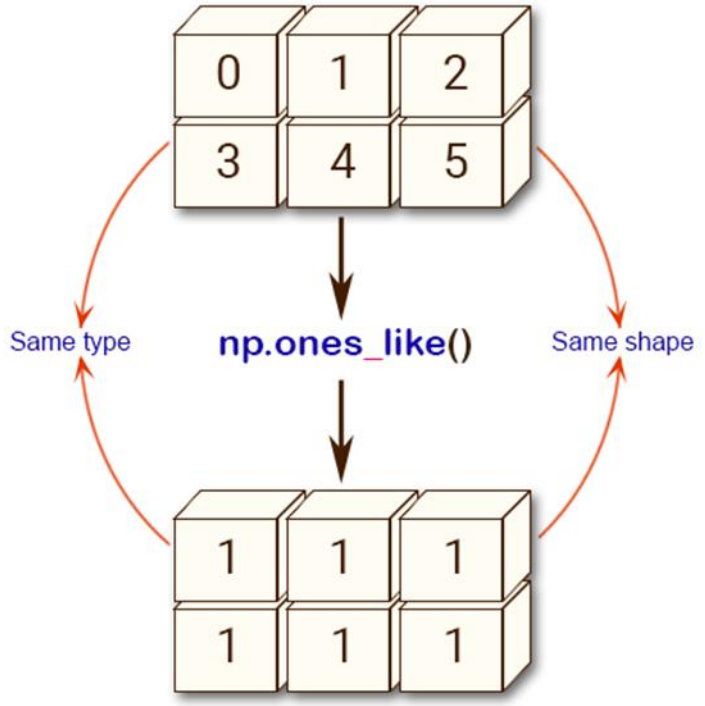






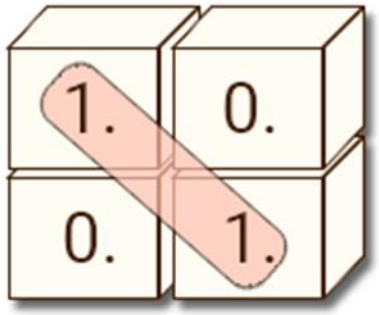
Numpy


0
1
2
3
Same type
np.ones_like()
Same shape
1
1
1
1

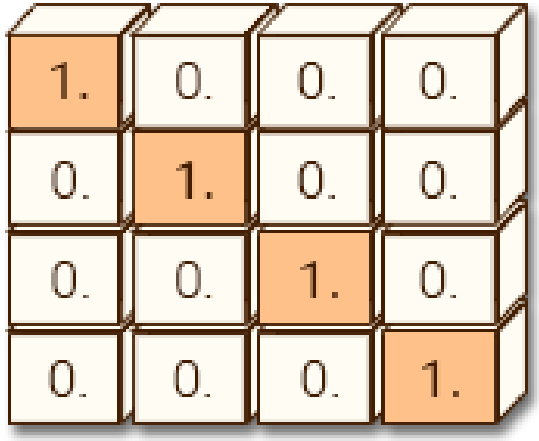

0
1
2
3
4
5
Same type
np.ones_like()
Same shape
1
1
1
1
1
1

Numpy

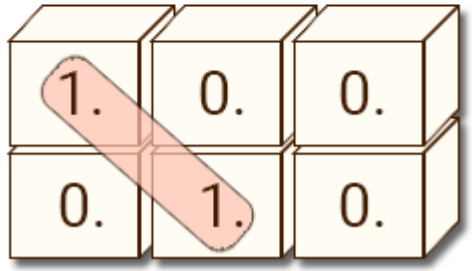
np.eye(2)



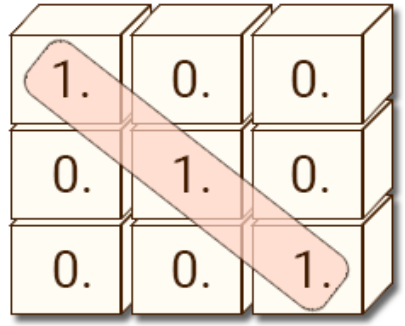
np.eye(4)



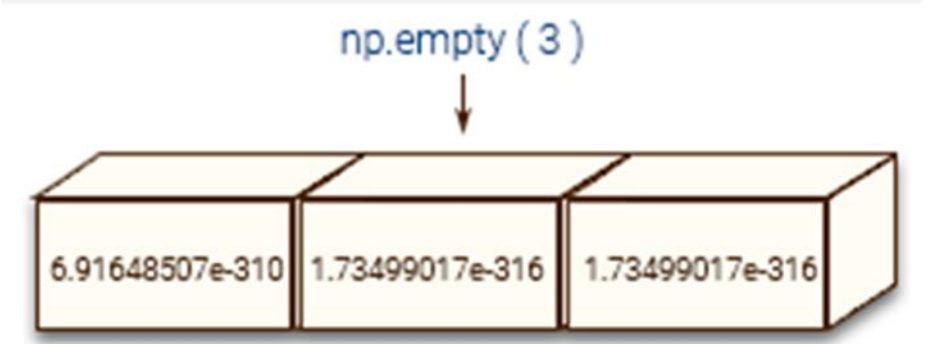
np.eye(2,3)

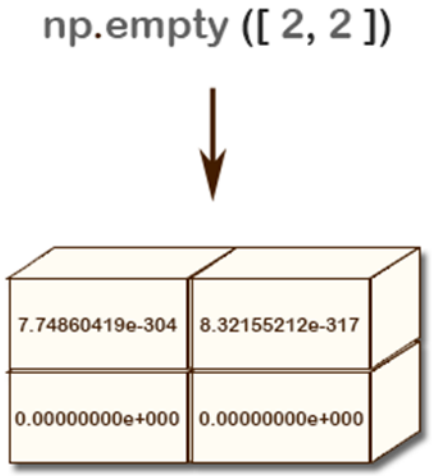


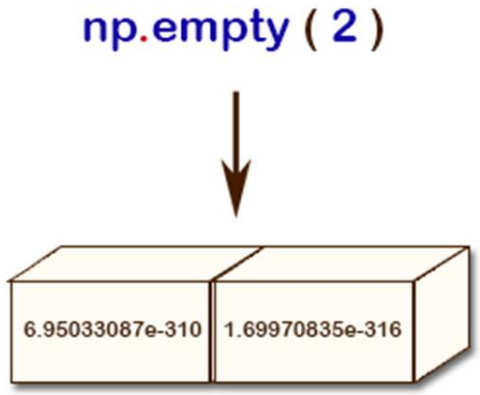
np.eye(3,3)



Numpy

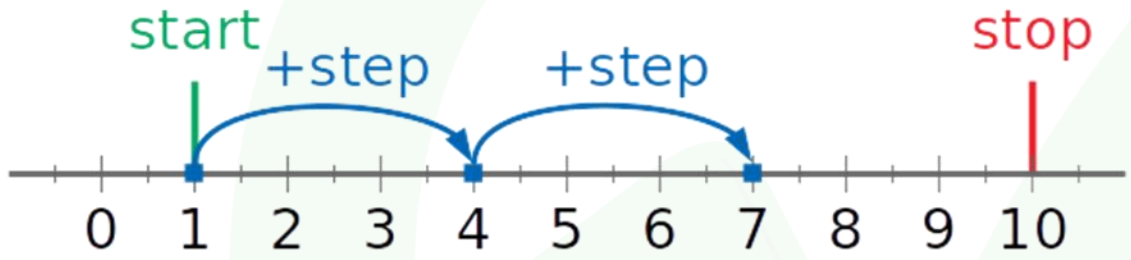

np.empty (3)
6.91648507e-310
1.73499017e-316
1.73499017e-316


np.empty ([2, 2])
7.74860419e-304
8.32155212e-317
0.00000000e+000
0.00000000e+000

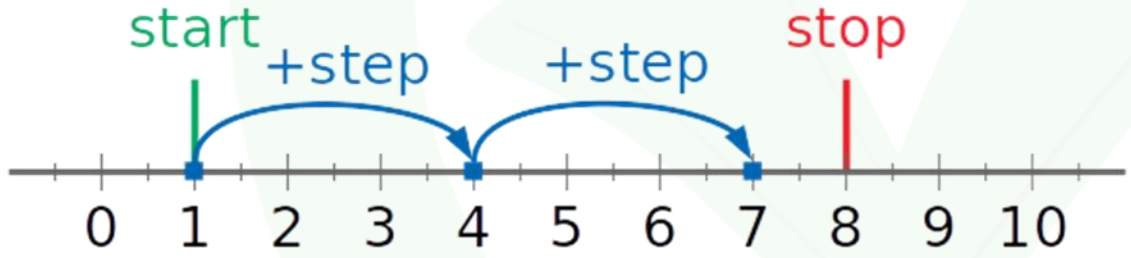

np.empty (2)
6.95033087e-310
1.69970835e-316

Numpy

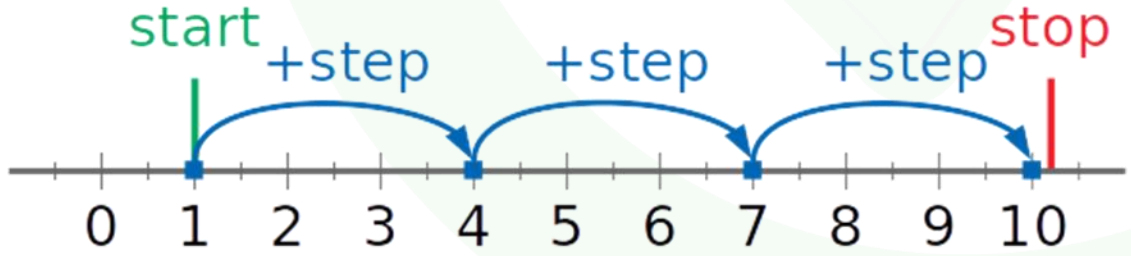
```
>>> np.arange(1, 10, 3)
array([1, 4, 7])
```




```
>>> np.arange(1, 8, 3)
array([1, 4, 7])
```




```
>>> np.arange(1, 10.1, 3)
array([1., 4., 7., 10.])
```



Numpy

numpy.linspace()

	INPUT	OUTPUT
1	x = np.linspace(10,20,5)	array([10. , 12.5, 15. , 17.5, 20.])
2	x = np.linspace(10,20, 5, endpoint = False)	array([10., 12., 14., 16., 18.])
3	x = np.linspace(1, 2, 5, retstep = True)	(array([1. , 1.25, 1.5 , 1.75, 2.]), 0.25)

Numpy

numpy.logspace(start, stop, num, endpoint, base, dtype)

	INPUT	OUTPUT
1	a = np.logspace(1.0, 2.0, num = 10)	array([10., 12.91549665, 16.68100537, 21.5443469 , 27.82559402, 35.93813664, 46.41588834, 59.94842503, 77.42636827, 100.])
2	a = np.logspace(1,10, num = 10, base = 2)	array([2., 4., 8., 16., 32., 64., 128., 256., 512., 1024.])

-> start : [float] start(base ** start) of interval range.
-> stop : [float] end(base ** stop) of interval range
-> endpoint : [boolean, optional]If True, stop is the last sample. By default, True
-> num : [int, optional] No. of samples to generate
-> base : [float, optional] Base of log scale. By default, equals 10.0
-> dtype : type of output array

Numpy

Data Types	Description
bool_	Boolean (True or False) stored as a byte
int_	Default integer type (same as C long; normally either int64 or int32)
intc	Identical to C int (normally int32 or int64)
intp	Integer used for indexing (same as C ssize_t; normally either int32 or int64)
int8	Byte (-128 to 127)
int16	Integer (-32768 to 32767)
int32	Integer (-2147483648 to 2147483647)
int64	Integer (-9223372036854775808 to 9223372036854775807)
uint8	Unsigned integer (0 to 255)
uint16	Unsigned integer (0 to 65535)
uint32	Unsigned integer (0 to 4294967295)
uint64	Unsigned integer (0 to 18446744073709551615)
float_	Shorthand for float64
float16	Half precision float: sign bit, 5 bits exponent, 10 bits mantissa
float32	Single precision float: sign bit, 8 bits exponent, 23 bits mantissa
float64	Double precision float: sign bit, 11 bits exponent, 52 bits mantissa
complex_	Shorthand for complex128
complex64	Complex number, represented by two 32-bit floats (real and imaginary components)
complex128	Complex number, represented by two 64-bit floats (real and imaginary components)

NumPy Veri Tipleri

i	integer
b	boolean
u	unsigned integer – işaretsiz tamsayı
f	float
c	complex float
m	timedelta
M	datetime
O	object
S	string
U	Unicode string
V	diğer türler için sabit bellek yığını (void)

Numpy