Numpy, Pandas

講者：Isaac

Outline

- Numpy
- Pandas

Numpy

Numpy

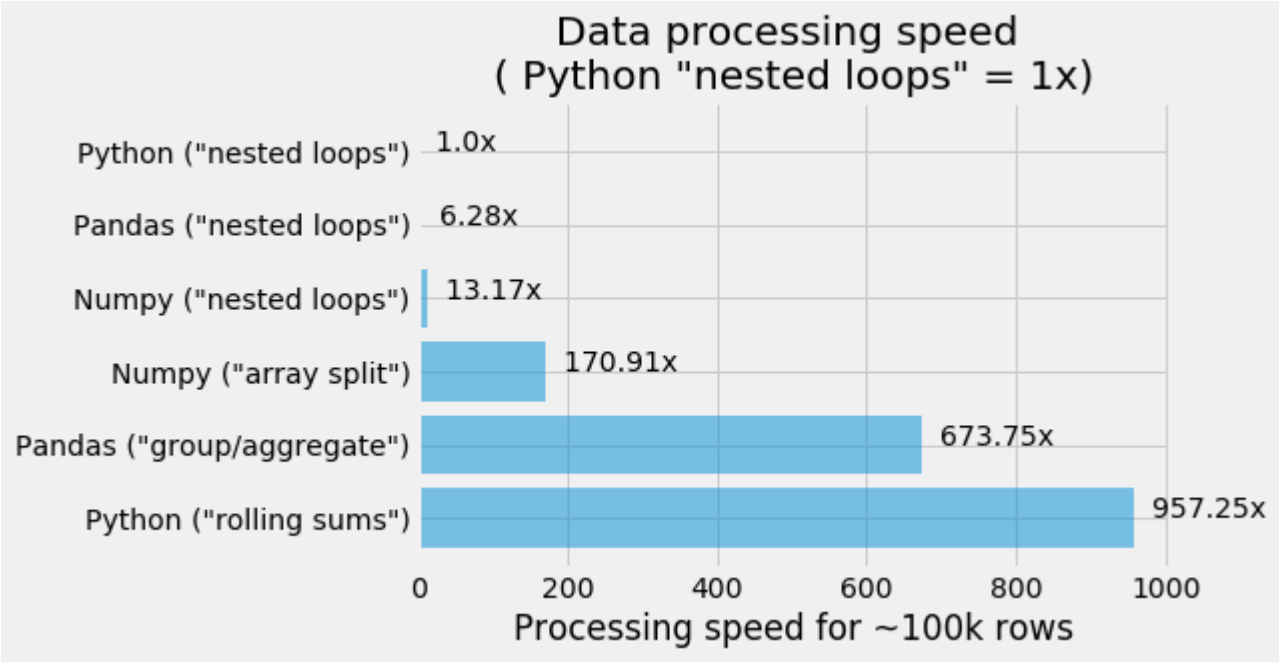
- a library for the Python that support for large, multi-dimensional arrays/matrices
  - http://www.numpy.org/
- core functionality of NumPy is its "ndarray“ data structure
  - for *n*-dimensional array
- all elements of a single array must be of the same type

N-dimensional array

Dimensions	Example	Terminology
1	0 1 2	Vector
2	0 1 2 3 4 5 6 7 8	Matrix
3	0 1 2 3 4 5 6 7 8	3D Array (3rd order Tensor)
N	... ...	ND Array

Numpy

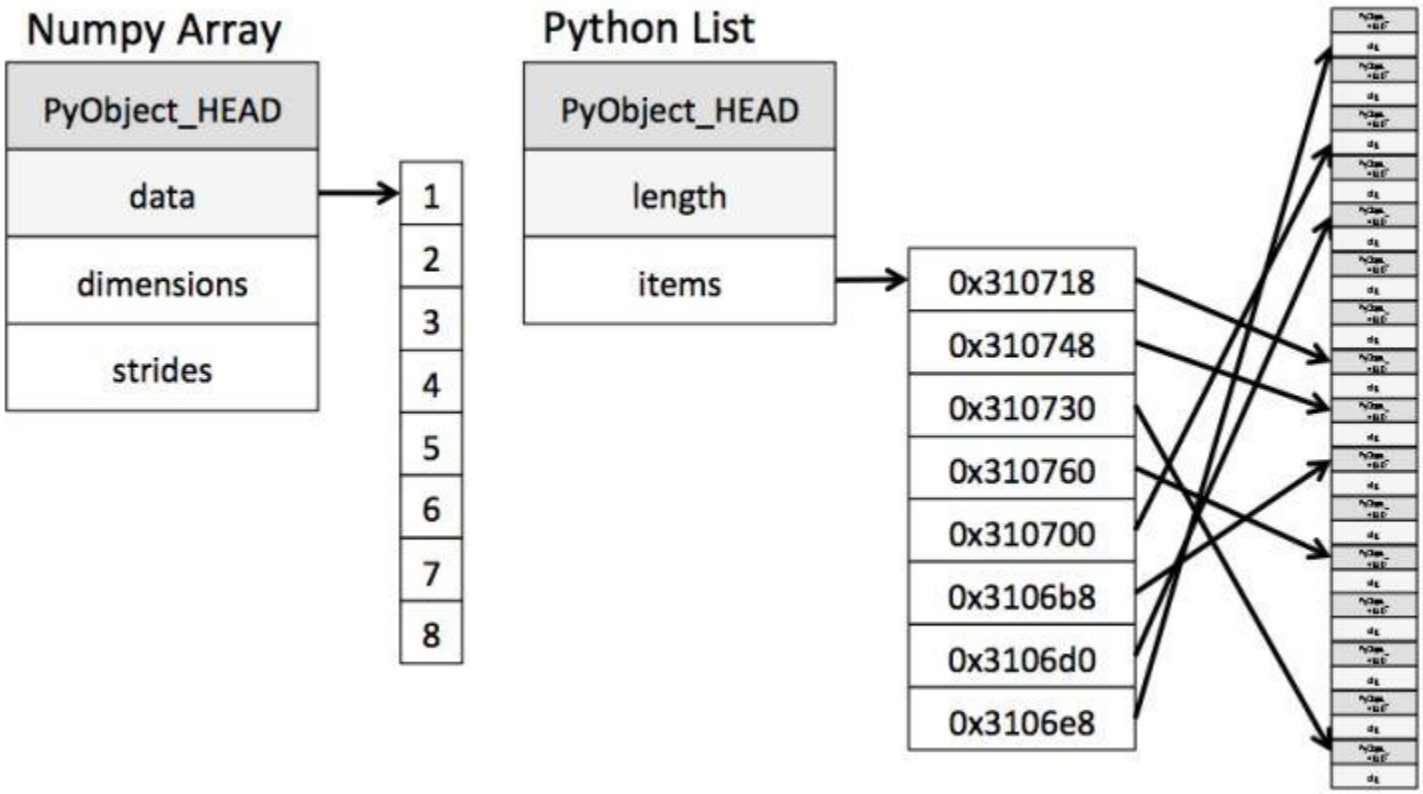
- Why we use numpy?



http://machinelearningexp.com/data-science-performance-of-python-vs-pandas-vs-numpy/

Numpy

- Why numpy?



Tensor

- What is Tensor?
  - Tensor = multidimensional array

Formally, tensors are multilinear maps from vector spaces to the real numbers (V vector space, and V^* dual space)

$$f : \underbrace{V^* \times \cdots V^*}_{p \text{ copies}} \times \underbrace{V \times \cdots V}_{q \text{ copies}} \to \mathbb{R}$$

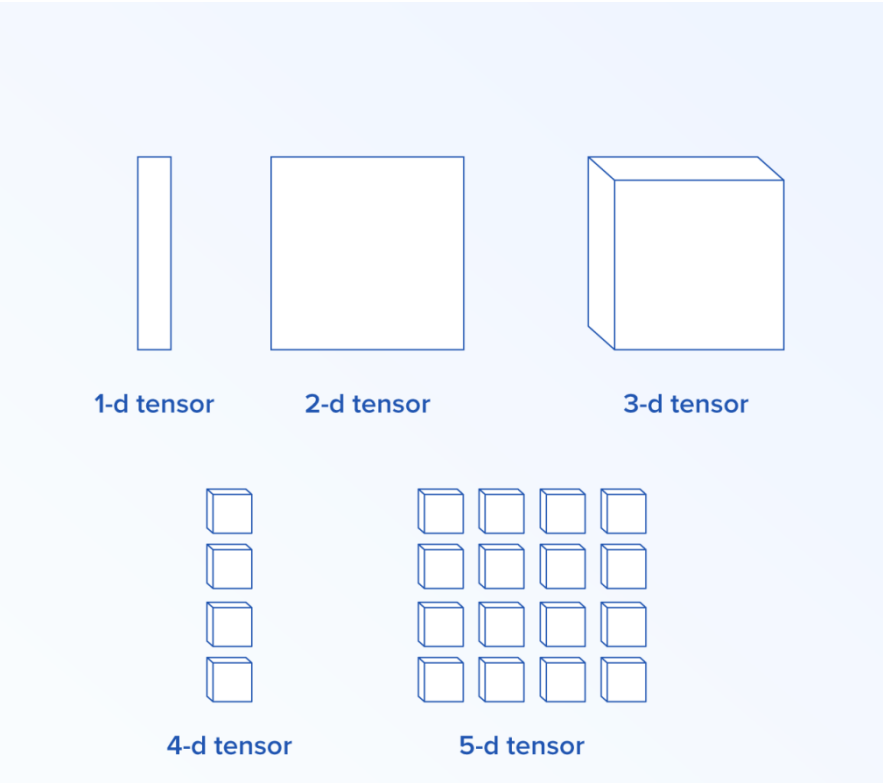
A scalar is a tensor ($f : \mathbb{R} \to \mathbb{R},\ f(e_1) = c$)

A vector is a tensor ($f : \mathbb{R}^n \to \mathbb{R},\ f(e_i) = v_i$)

A matrix is a tensor ($f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R},\ f(e_i, e_j) = A_{ij}$)

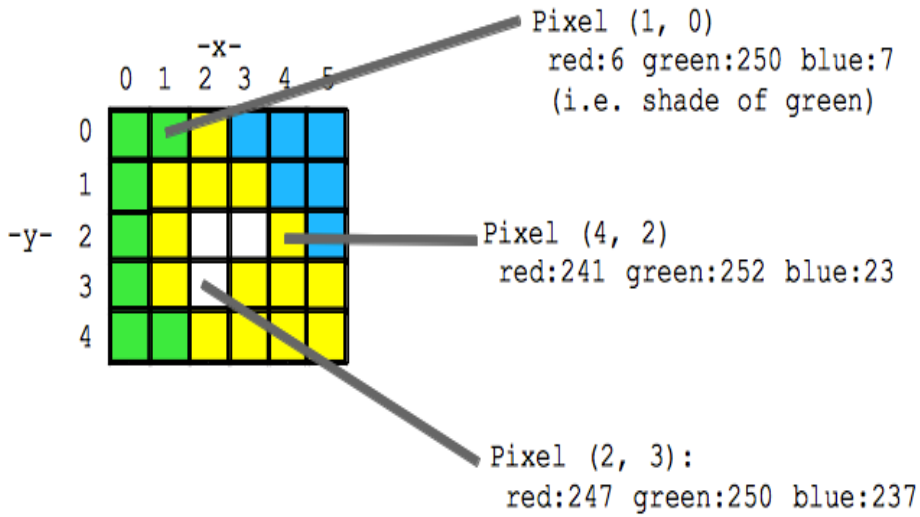
Common to have fixed basis, **so a tensor can be represented as a multidimensional array of numbers.**

Tensor

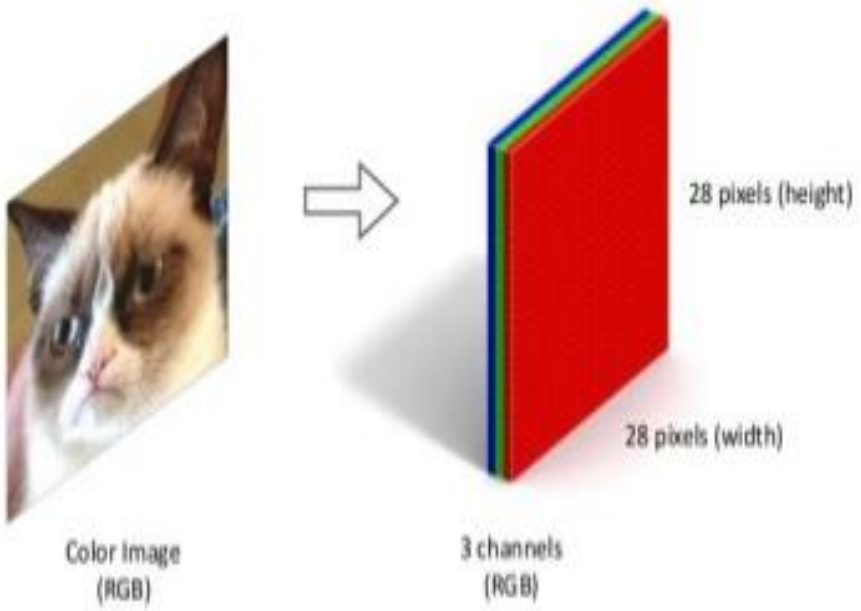


3D Tensor Example

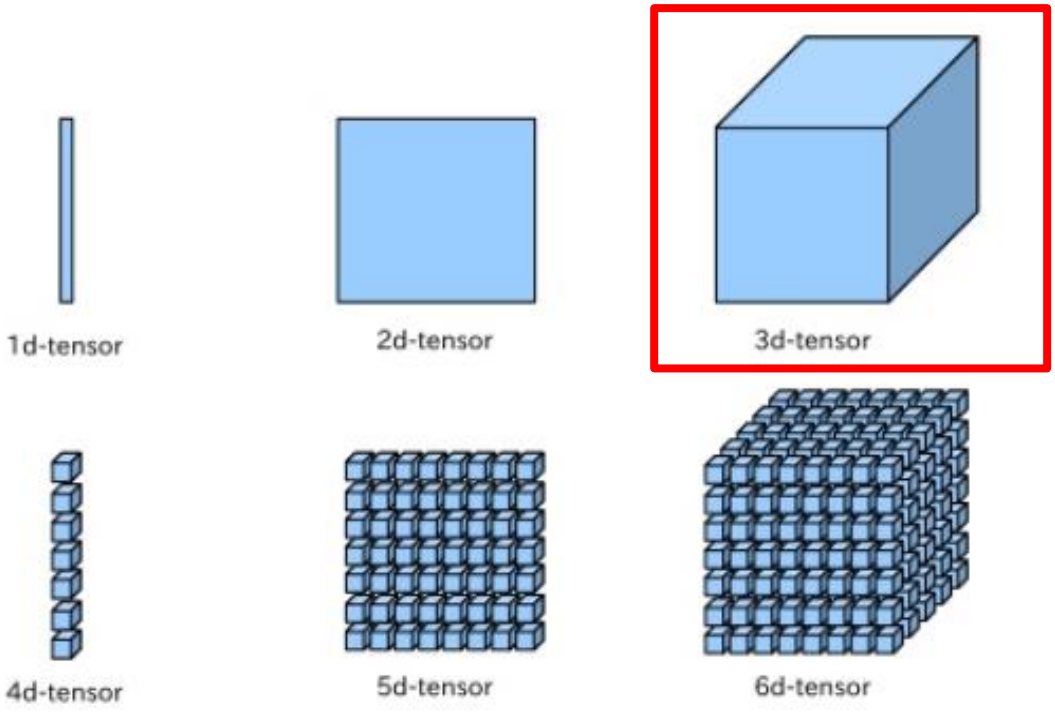
- Each image contain many pixels
  - Each pixels compose red, green, blue(RGB)
- Each channel have brightness levels between 0~255



3D Tensor Example

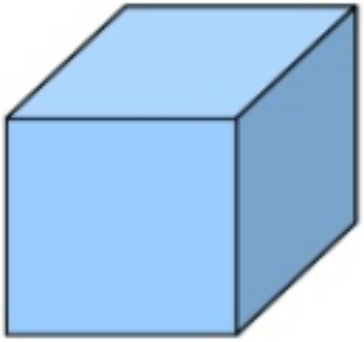


3D Tensor Example



An RGB image = 3D tensor

3D Tensor Example



= **[image width, image height, image channel]**

A image is a 3D-tensor

4D Tensor Example

= **[batch size, image width, image height, image channel]**

A batch of images is a 4D-tensor

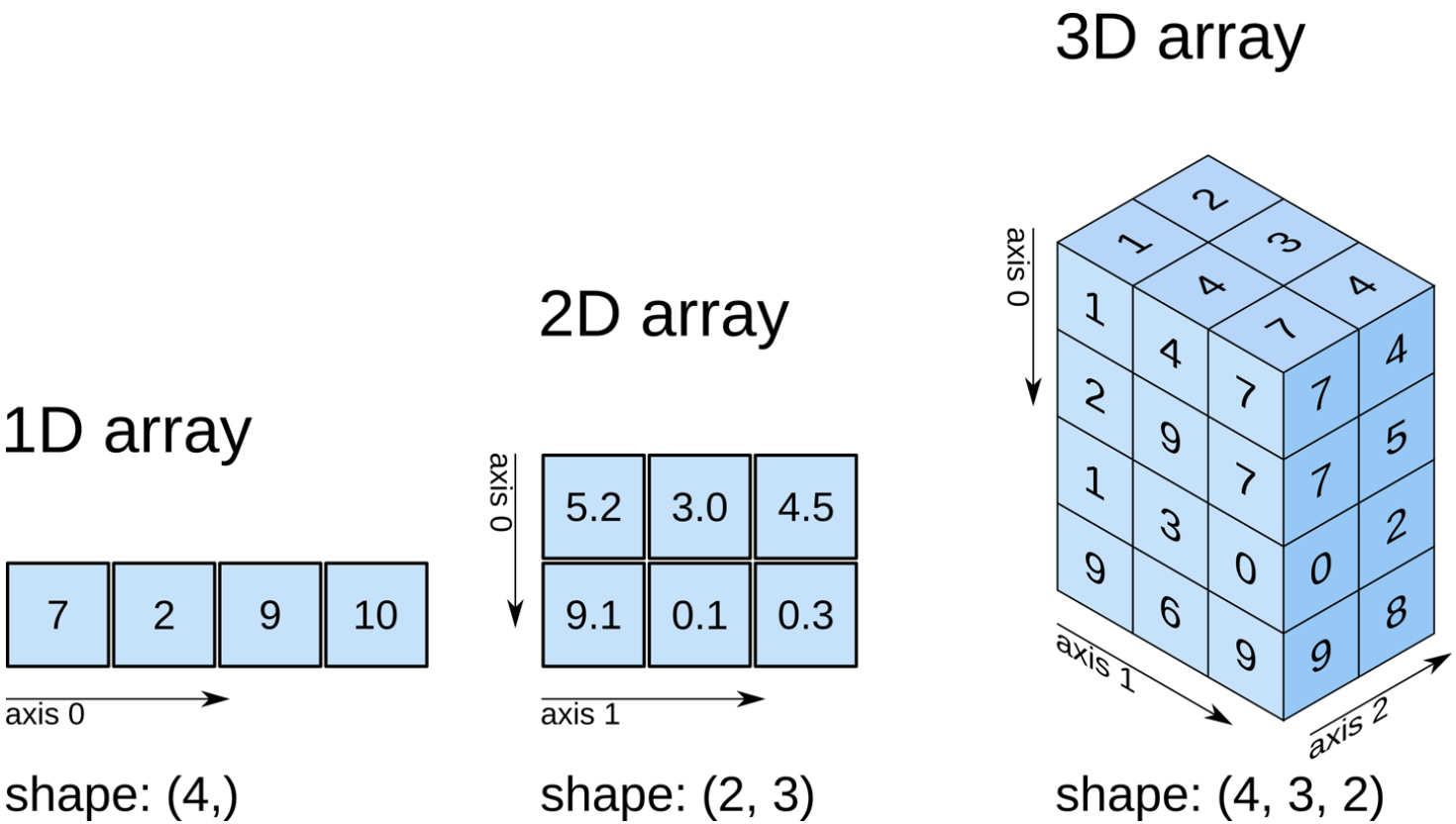
Data type in Numpy

bool	Boolean (True or False) stored as a byte
int	Platform integer (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
complex128	Complex number, represented by two 64-bit floats

Numpy

- What's axis?



Numpy array creation

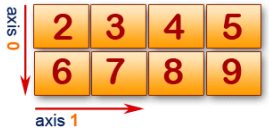
1D array

```
>>> import numpy as np
>>> x =  np.arange(2, 5).reshape(3)
>>> x
array( [ 2, 3, 4 ] )
>>>
```

Shape : (4)

2D array

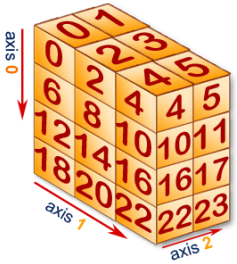
```
>>> import numpy as np
>>> x =  np.arange(2, 10).reshape(2, 4)
>>> x
array( [ [ 2, 3, 4, 5 ] )
       [2, 3, 4, 5 ] ] )
>>>
```



Shape : (2, 4)

3D array

```
>>> import numpy as np
>>> x =  np.arange(24).reshape(4, 3, 2)
>>> x
array( [ [ [ 0, 1 ] ),  [ [ 6, 7] ),   [ [ 12, 13] ), [ [ 18, 19] ),
         [ 2, 3] ),     [ 8, 9] ),     [ 14, 15] ),   [ 20, 21] ),
         [ 4, 5] ),     [ 10, 11] ),   [ 16, 17] ),   [ 22, 23] ] ] )
>>>
```



Shape : (4, 3, 2)

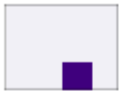
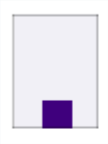
Numpy array creation

Code	Result		Code	Result
`Z = zeros(9)`			`Z = zeros((5,9))`	
`Z = ones(9)`			`Z = ones((5,9))`	

`Z = array(`
`[0,0,0,0,0,0,0,0,0] )`
			`Z = array(`
			`[[0,0,0,0,0,0,0,0,0],`
			`[0,0,0,0,0,0,0,0,0],`
			`[0,0,0,0,0,0,0,0,0],`
			`[0,0,0,0,0,0,0,0,0],`
			`[0,0,0,0,0,0,0,0,0]])`
				

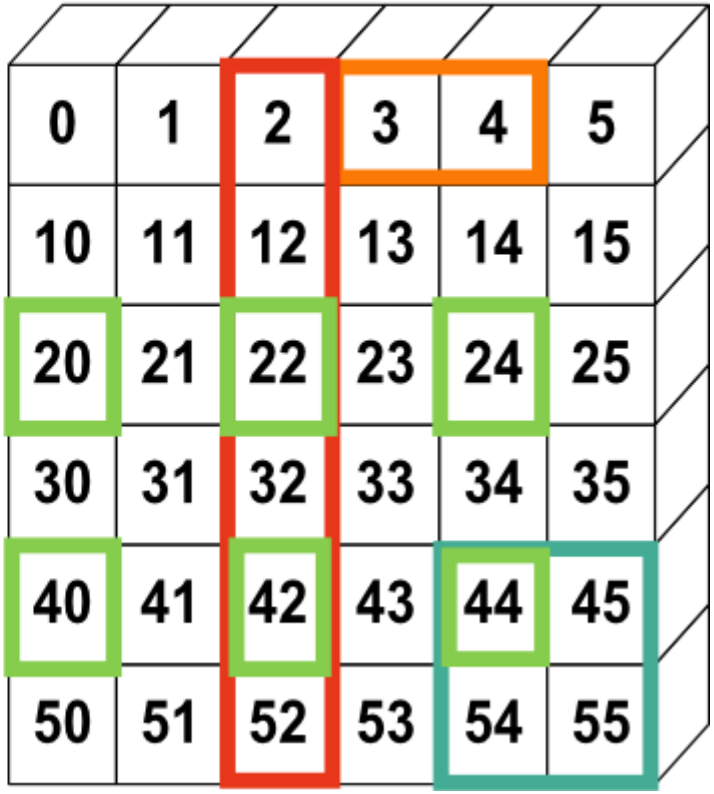
`Z = arange(9)`			`Z = arange(5*9).reshape(5,9)`	
`Z = random.uniform(0,1,9)`			`Z = random.uniform(0,1,(5,9))`	

Numpy array reshape

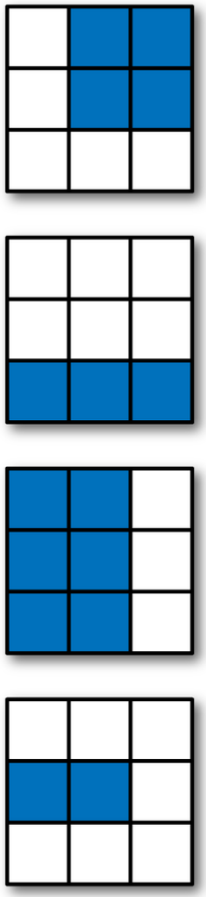
Code	Result	Code	Result
`Z[2,2] = 1`		`Z = Z.reshape(1,12)`	
`Z = Z.reshape(4,3)`		`Z = Z.reshape(12,1)`	
`Z = Z.reshape(6,2)`			
`Z = Z.reshape(2,6)`			

Numpy array indexing/slicing

```
>>> a[0,3:5]
array([3,4])

>>> a[4:,4:]
array([[44, 45],
       [54, 55]])

>>> a[:,2]
array([2,12,22,32,42,52])

>>> a[2::2,::2]
array([[20,22,24]
       [40,42,44]])
```



Numpy array indexing/slicing

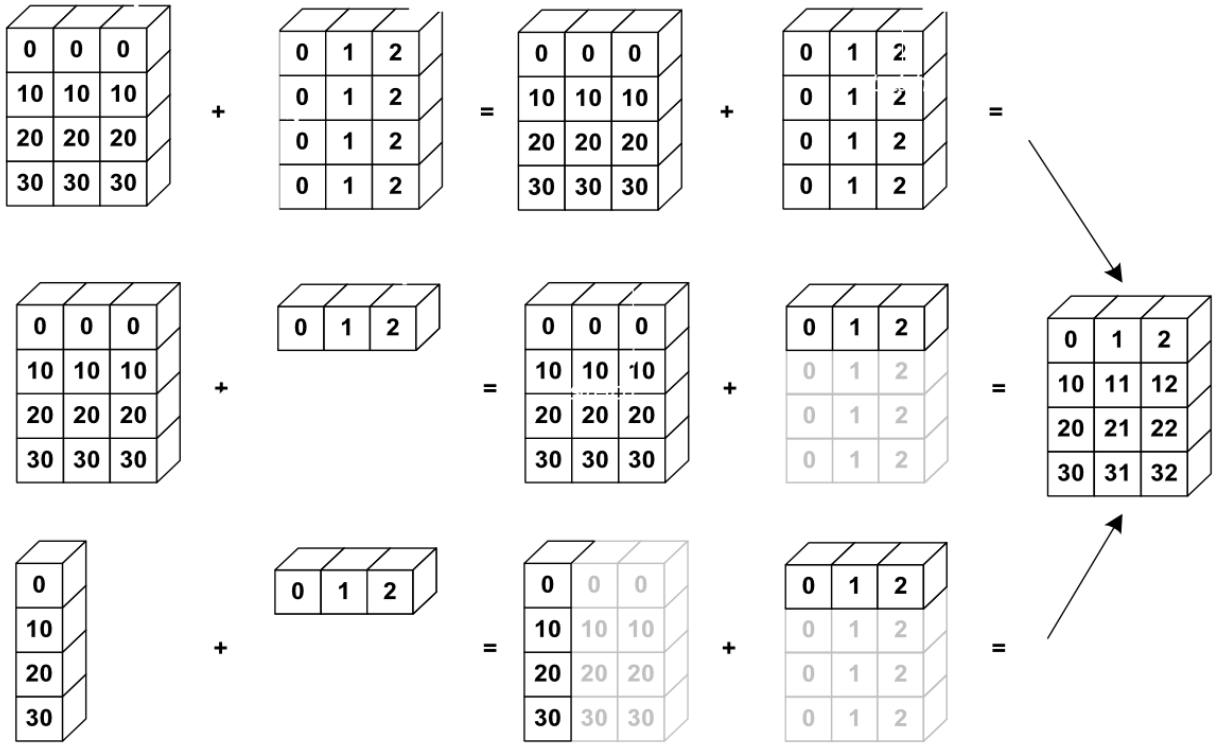


Expression	Shape
`arr[:2, 1:]`	`(2, 2)`
`arr[2]`	`(3,)`
`arr[2, :]`	`(3,)`
`arr[2:, :]`	`(1, 3)`
`arr[:, :2]`	`(3, 2)`
`arr[1, :2]`	`(2,)`
`arr[1:2, :2]`	`(1, 2)`

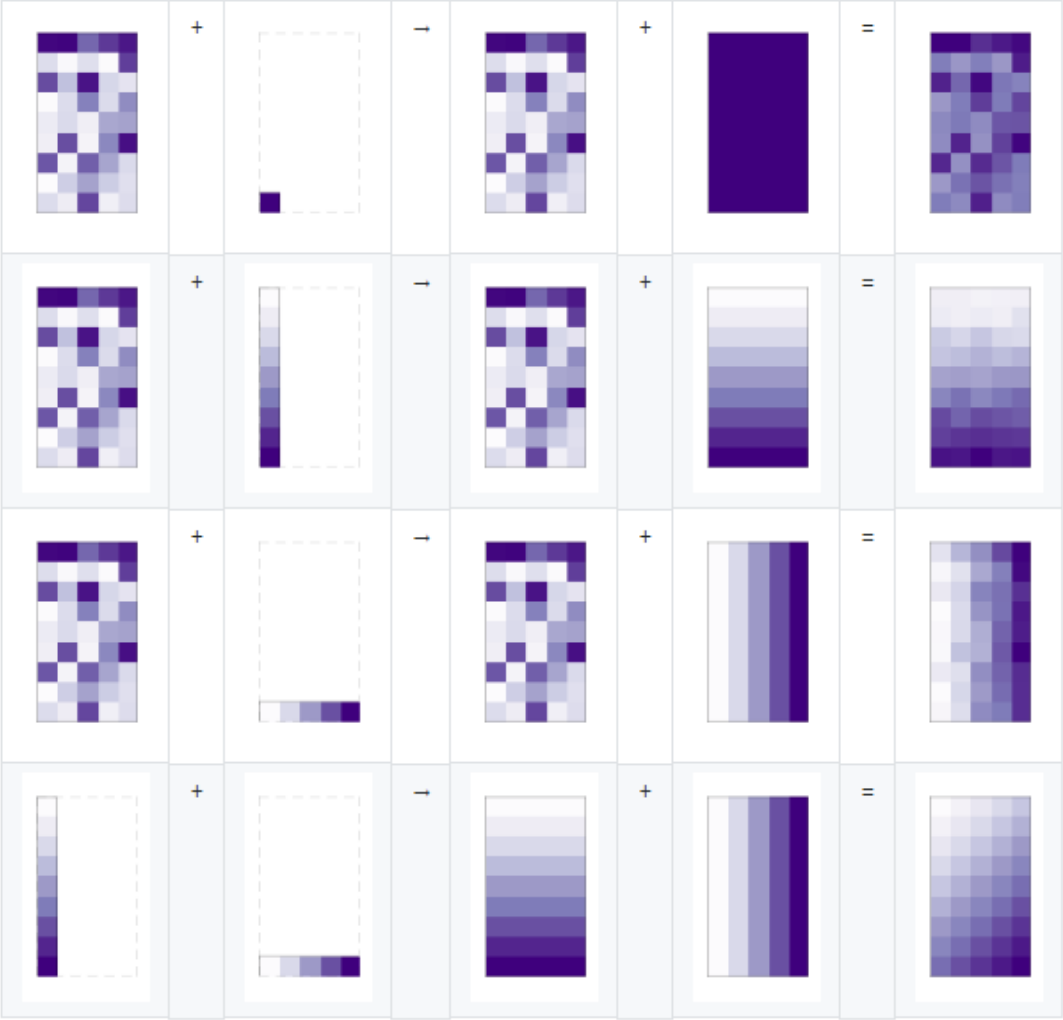
Numpy array indexing/slicing

Code	Result		Code	Result
`Z`			`Z[...] = 1`	
`Z[1,1] = 1`			`Z[:,0] = 1`	
`Z[0,:] = 1`			`Z[2:,2:] = 1`	
`Z[:,::2] = 1`			`Z[::2,:] = 1`	
`Z[:-2,:-2] = 1`			`Z[2:4,2:4] = 1`	
`Z[::2,::2] = 1`			`Z[3::2,3::2] = 1`	

Numpy array broadcasting



Numpy array broadcasting



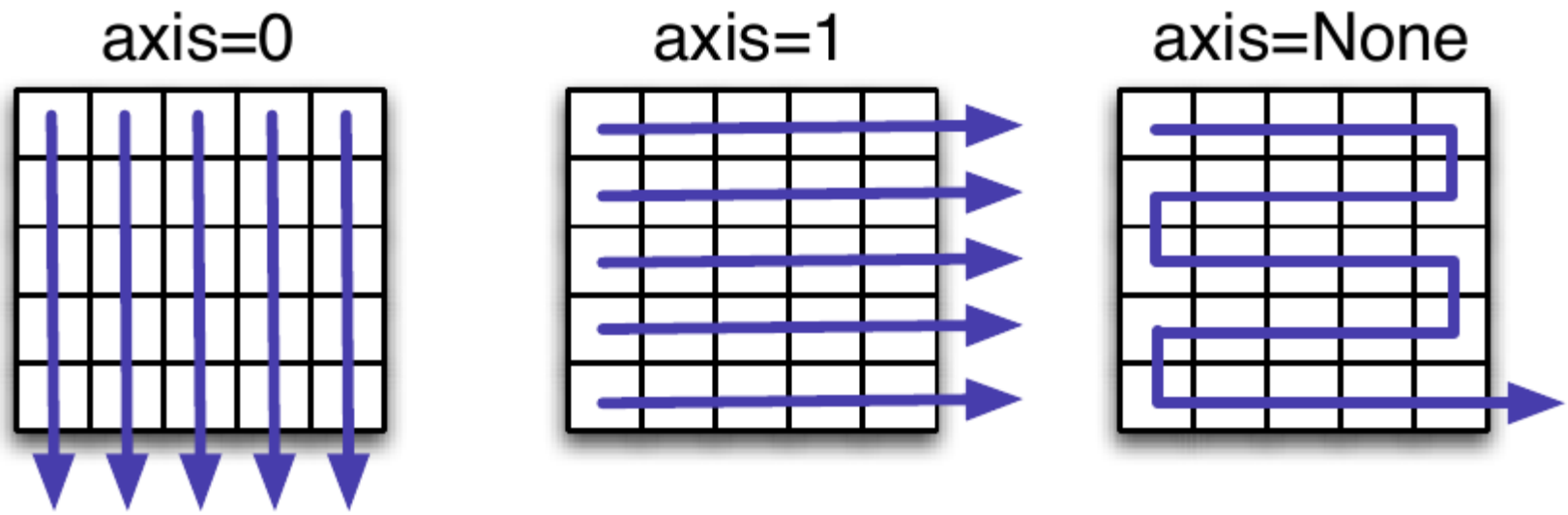
Rules of broadcasting

- Rule 1
  - If the two arrays differ in their number of dimensions, the shape of the one with fewer dimensions is *padded* with ones on its leading (left) side.
- Rule 2
  - If the shape of the two arrays does not match in any dimension, the array with shape equal to 1 in that dimension is stretched to match the other shape.
- Rule 3
  - If in any dimension the sizes disagree and neither is equal to 1, an error is raised.

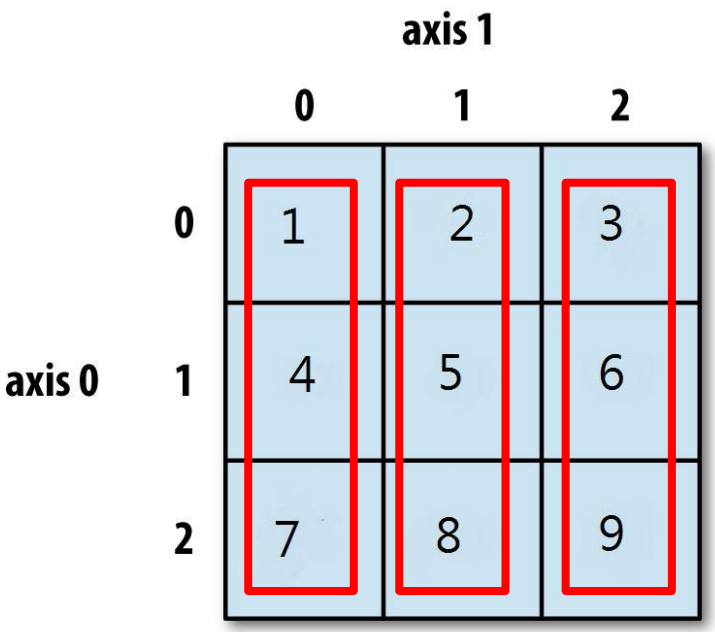
Numpy array operations

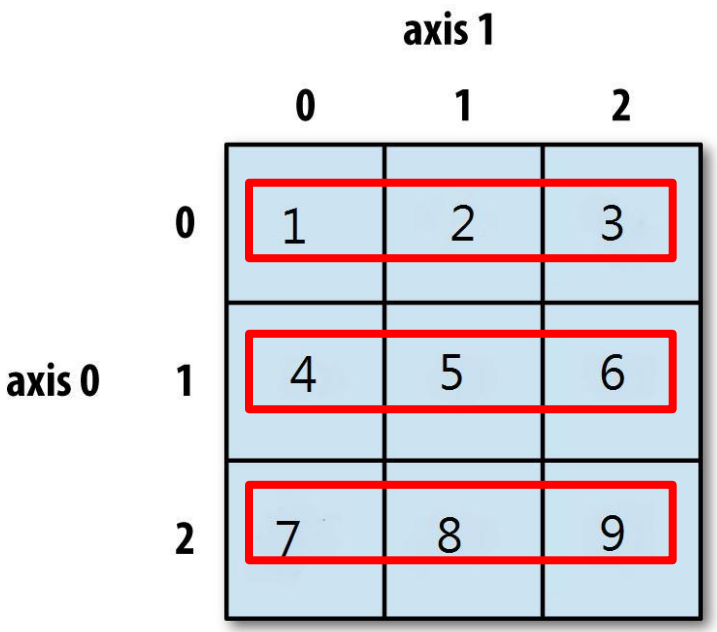
Code	Before	After
`Z = np.where(Z > 0.5, 0, 1)`		
`Z = np.maximum(Z, 0.5)`		
`Z = np.minimum(Z, 0.5)`		
`Z = np.sum(Z, axis=0)`		

Numpy array operations



Numpy array operations

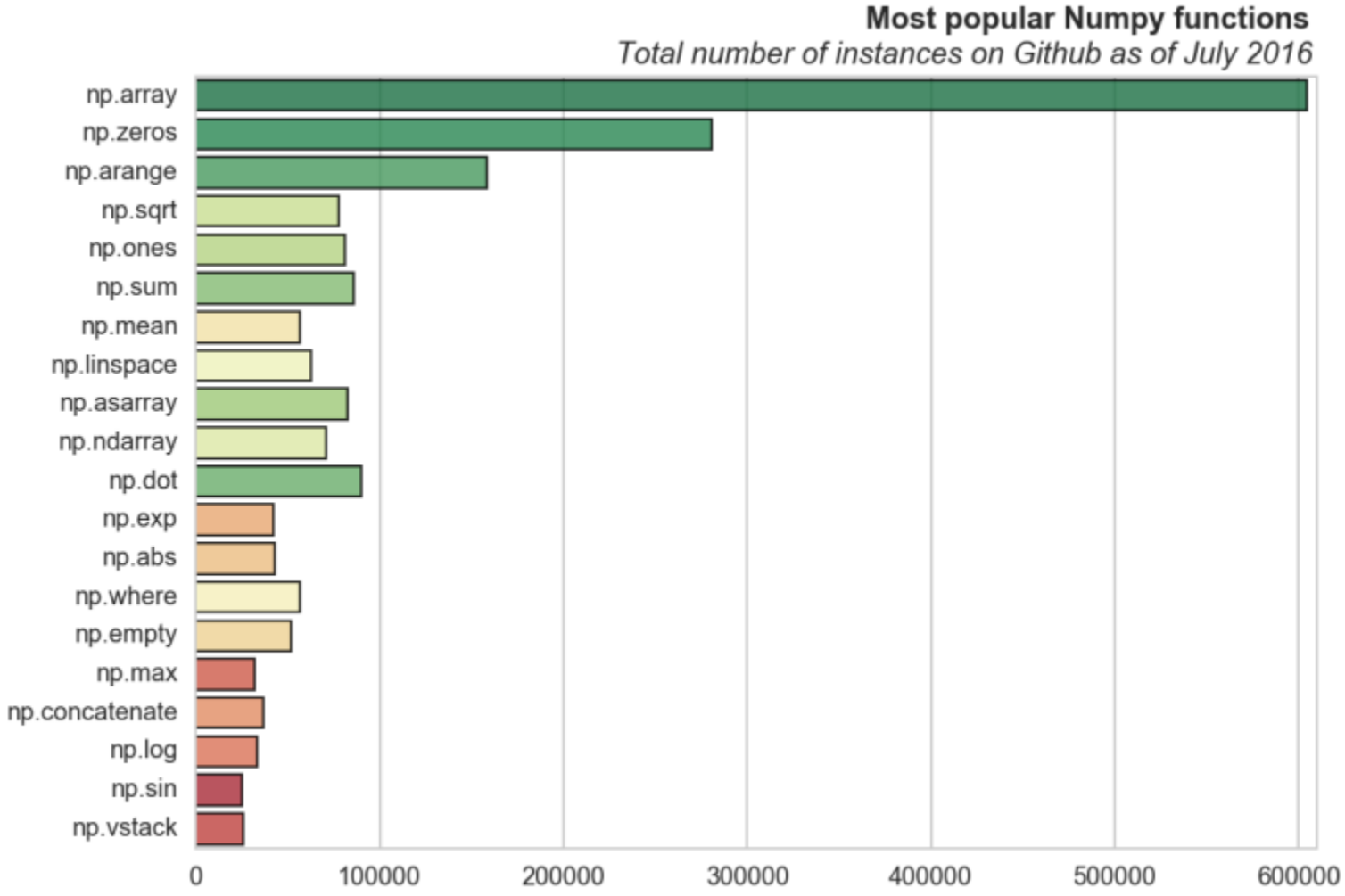




ndarray.sum(axis = 0) -> array([12, 15, 18])

ndarray.sum(axis = 1) -> array([6, 15, 24])

Most popular Numpy functions

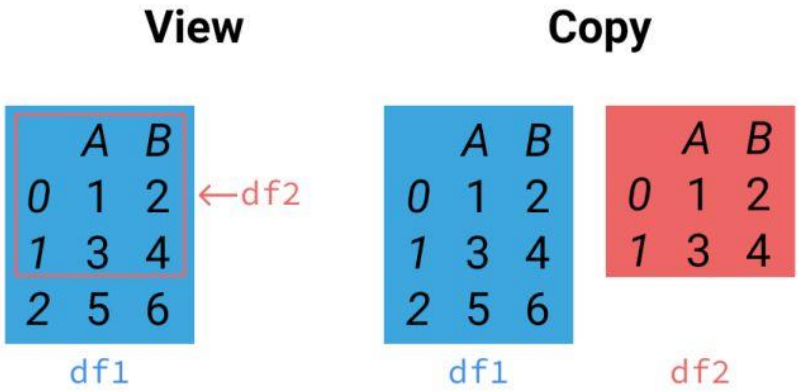


Numpy V.S. TensorFlow

Numpy	TensorFlow
`a = np.zeros((2,2)); b = np.ones((2,2))`	`a = tf.zeros((2,2)), b = tf.ones((2,2))`
`np.sum(b, axis=1)`	`tf.reduce_sum(a,reduction_indices=[1])`
`a.shape`	`a.get_shape()`
`np.reshape(a, (1,4))`	`tf.reshape(a, (1,4))`
`b * 5 + 1`	`b * 5 + 1`
`np.dot(a,b)`	`tf.matmul(a, b)`
`a[0,0], a[:,0], a[0,:]`	`a[0,0], a[:,0], a[0,:]`

Copy and view

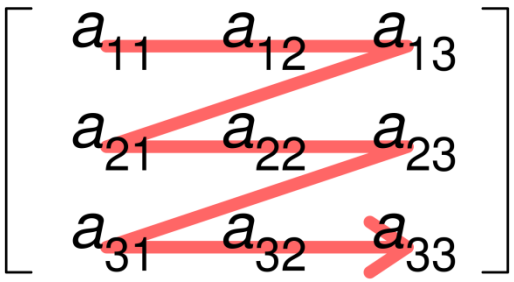
- while executing the functions
  - some of them return a copy of the input array, while some return the view
- return contents are physically stored in another location
  - it is called copy
- return contents are in the same memory as origin contents
  - it is called it as view
- view is faster than copy



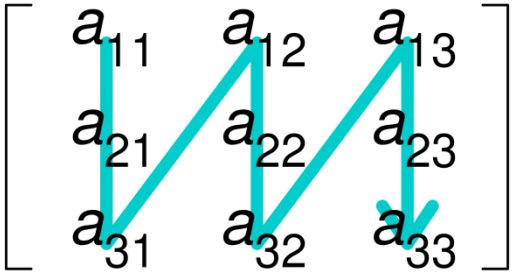
High performance numpy

- reference
  - https://zhuanlan.zhihu.com/p/28626431
  - https://morvanzhou.github.io/tutorials/data-manipulation/np-pd/4-1-speed-up-numpy/
  - https://www.slideshare.net/skarl86/numpy-tutorialfinal-20160303

Row-major order



Column-major order



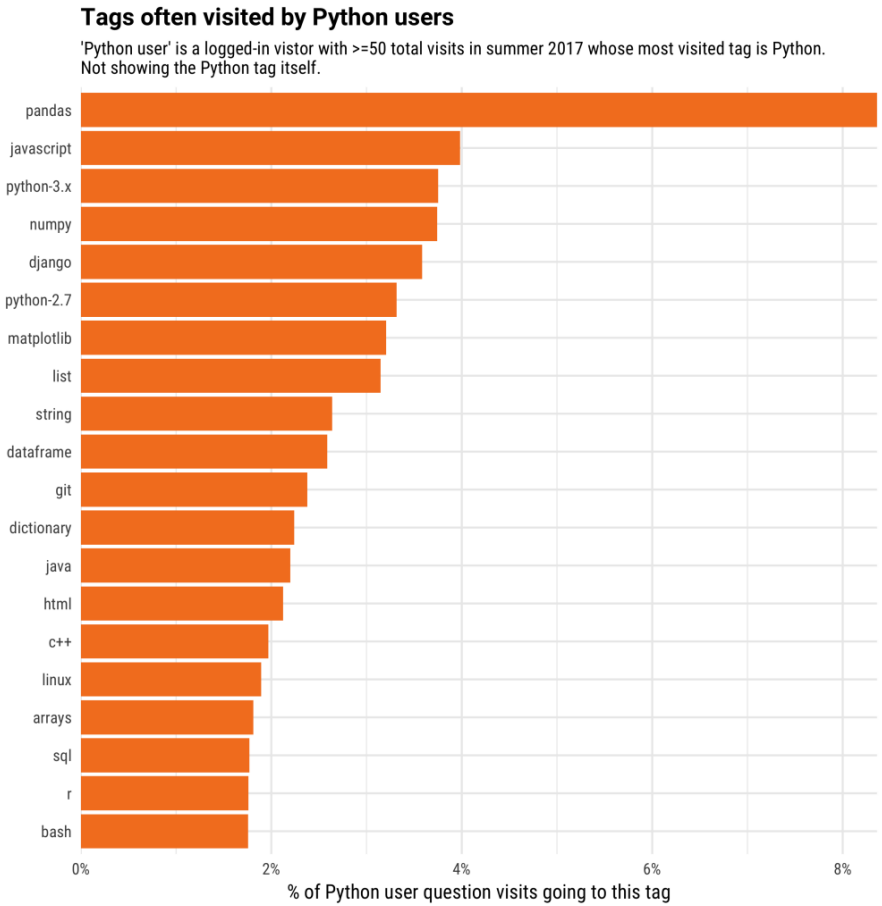
Pandas

Pandas

- Pandas is python library that is very useful to manipulate data, especially structure data
- Provide data structures and operations for manipulating numerical tables and time series

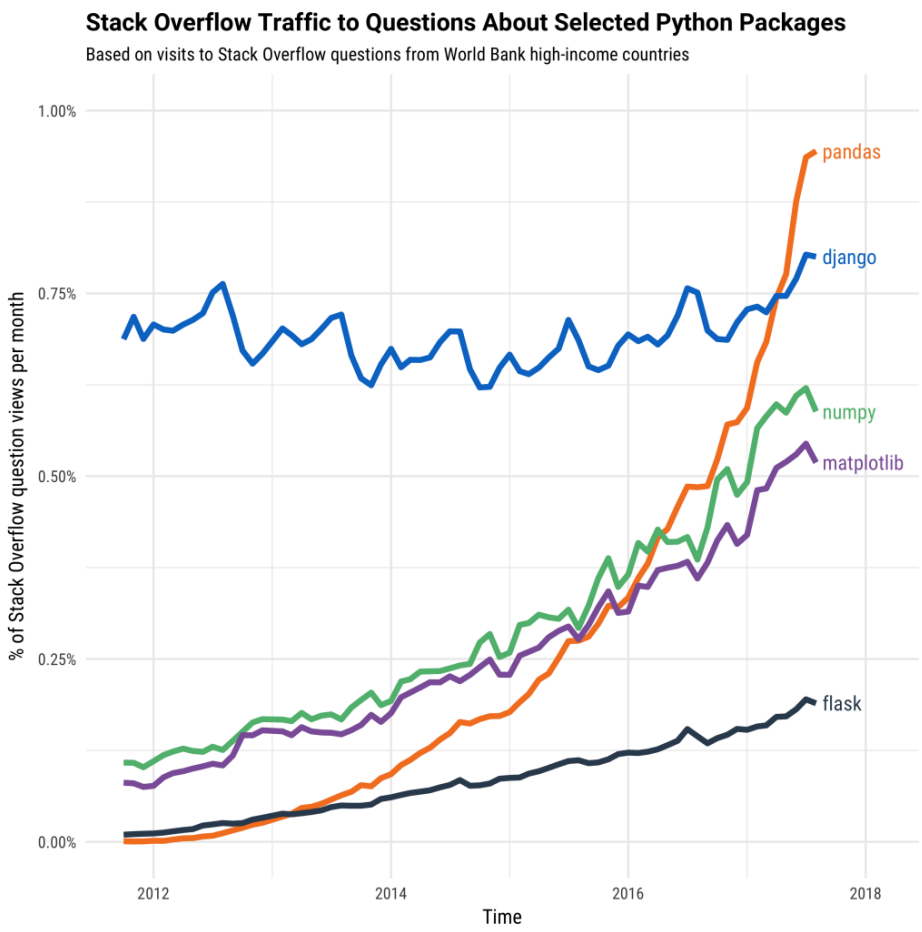
Pandas

Popularity in Pandas



https://stackoverflow.blog/2017/09/14/python-growing-quickly/

Popularity in Pandas



Pandas Data structure

- Pandas series
  - 1-D data
- Pandas dataframe
  - 2-D data
- Pandas panel
  - 3-D data

Series

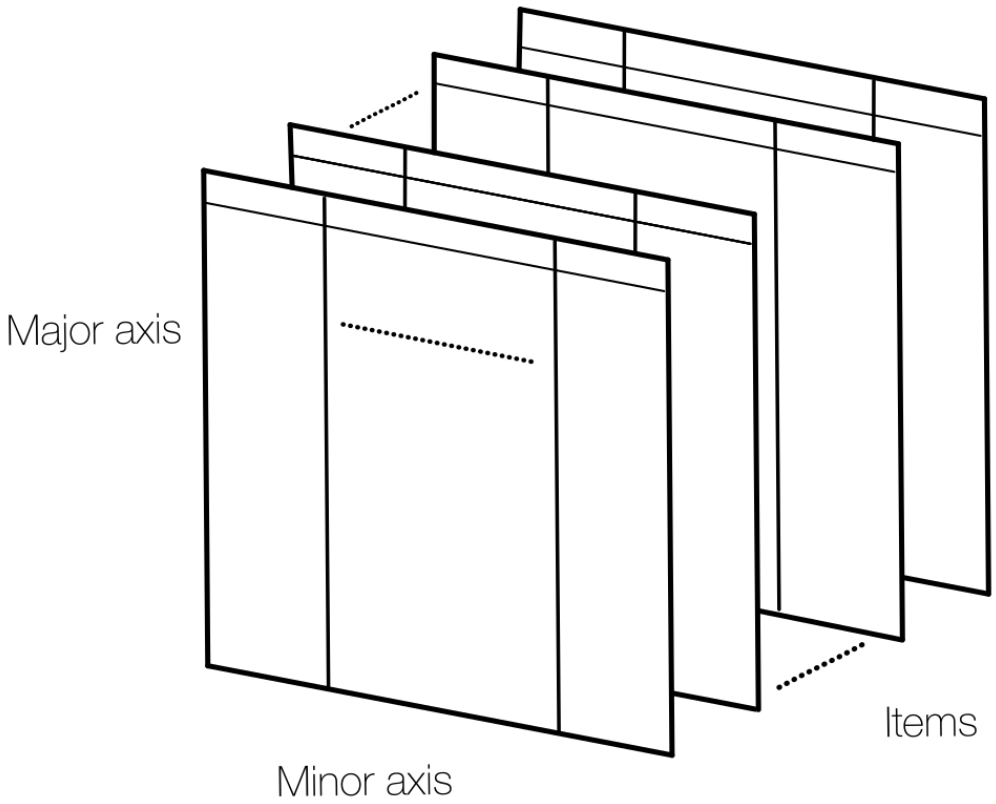
Index	Data
1	‘A’
2	‘B’
3	‘C’
4	‘D’
5	‘E’

Dataframe

	GEOID	State	2005	2006	2007	2008	2009	2010	2011	2012	2013
0	04000US01	Alabama	37150	37952	42212	44476	39980	40933	42590	43464	41381
1	04000US02	Alaska	55891	56418	62993	63989	61604	57848	57431	63648	61137
2	04000US04	Arizona	45245	46657	47215	46914	45739	46896	48621	47044	50602
3	04000US05	Arkansas	36658	37057	40795	39586	36538	38587	41302	39018	39919
4	04000US06	California	51755	55319	55734	57014	56134	54283	53367	57020	57528

Dataframe is consists of rows or columns

Panel



Create dataframe

Dictionary

Row Oriented

```
sales = [{'account': 'Jones LLC', 'Jan': 150, 'Feb': 200, 'Mar': 140},
         {'account': 'Alpha Co', 'Jan': 200, 'Feb': 210, 'Mar': 215},
         {'account': 'Blue Inc', 'Jan': 50, 'Feb': 90, 'Mar': 95 }]
df = pd.DataFrame(sales)
```

default

List

```
sales = [('Jones LLC', 150, 200, 50),
         ('Alpha Co', 200, 210, 90),
         ('Blue Inc', 140, 215, 95)]
labels = ['account', 'Jan', 'Feb', 'Mar']
df = pd.DataFrame.from_records(sales, columns=labels)
```

from_records

	account	Jan	Feb	Mar
0	Jones LLC	150	200	140
1	Alpha Co	200	210	215
2	Blue Inc	50	90	95

Column Oriented

```
sales = {'account': ['Jones LLC', 'Alpha Co', 'Blue Inc'],
         'Jan': [150, 200, 50],
         'Feb': [200, 210, 90],
         'Mar': [140, 215, 95]}
df = pd.DataFrame.from_dict(sales)
```

from_dict

```
sales = [('account', ['Jones LLC', 'Alpha Co', 'Blue Inc']),
         ('Jan', [150, 200, 50]),
         ('Feb', [200, 210, 90]),
         ('Mar', [140, 215, 95]) ]
df = pd.DataFrame.from_items(sales)
```

from_items

When using a dictionary, column order is not preserved.
Explicitly order them:
`df = df[['account', 'Jan', 'Feb', 'Mar']]`

DataFrame Basic Functionality

Attribute/Method	Description
T	Transposes rows and columns.
axes	Returns a list with the row axis labels and column axis labels as the only members.
dtypes	Returns the dtypes in this object.
empty	True if NDFrame is entirely empty [no items]; if any of the axes are of length 0.
ndim	Number of axes / array dimensions.
shape	Returns a tuple representing the dimensionality of the DataFrame.
size	Number of elements in the NDFrame.
values	Numpy representation of NDFrame.
head()	Returns the first n rows.
tail()	Returns last n rows.

DataFrame Basic Functionality

Function	Description
count()	Number of non-null observations
sum()	Sum of values
mean()	Mean of Values
median()	Median of Values
mode()	Mode of values
std()	Standard Deviation of the Values
min()	Minimum Value
max()	Maximum Value
abs()	Absolute Value
prod()	Product of Values
cumsum()	Cumulative Sum
cumprod()	Cumulative Product

Pandas merge

LEFT	key	A	B
0	K0	A0	B0
1	K1	A1	B1
2	K2	A2	B2
3	K3	A3	B3

RIGHT	key	C	D
0	K0	C0	D0
1	K1	C1	D1
2	K1	C2	D2
3	K4	C3	D3

Left Merge

RESULT	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K1	A1	B1	C1	D1
2	K1	A1	B1	C2	D2
3	K2	A2	B2	NaN	NaN
4	K3	A3	B3	NaN	NaN

Right Merge

RESULT	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K1	A1	B1	C1	D1
2	K1	A1	B1	C2	D2
3	K4	NaN	NaN	C3	D3

Inner Merge

RESULT	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K1	A1	B1	C1	D1
2	K1	A1	B1	C2	D2

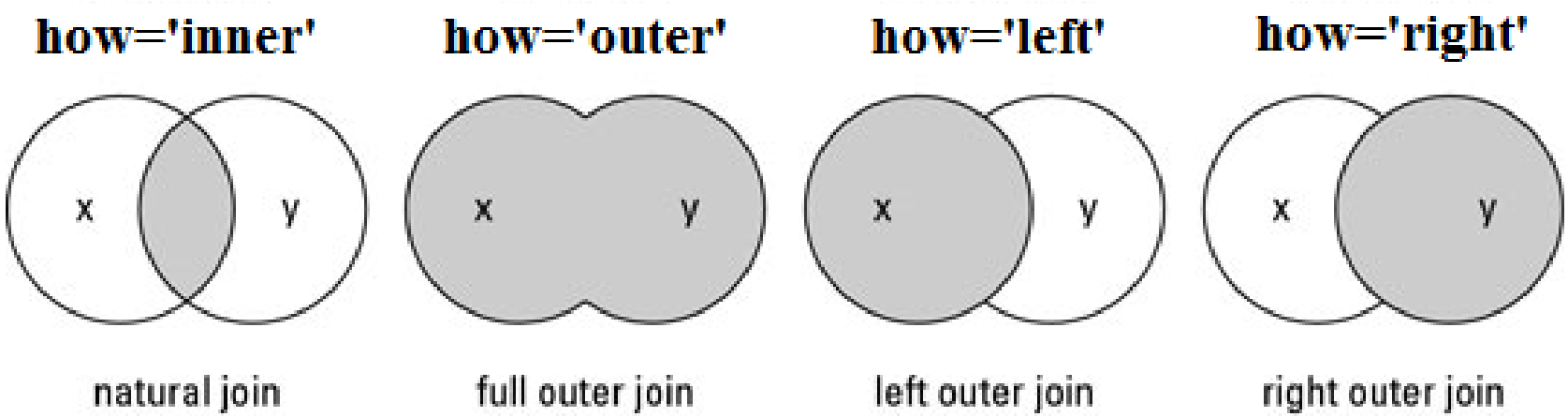
Outer Merge

RESULT	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K1	A1	B1	C1	D1
2	K1	A1	B1	C2	D2
3	K2	A2	B2	NaN	NaN
4	K3	A3	B3	NaN	NaN
5	K4	NaN	NaN	C3	D3

Pandas merge

Merge method	SQL Join Name	Description
left	LEFT OUTER JOIN	Use keys from left frame only
right	RIGHT OUTER JOIN	Use keys from right frame only
outer	FULL OUTER JOIN	Use union of keys from both frames
inner	INNER JOIN	Use intersection of keys from both frames

Pandas merge



Pandas append

df1

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3

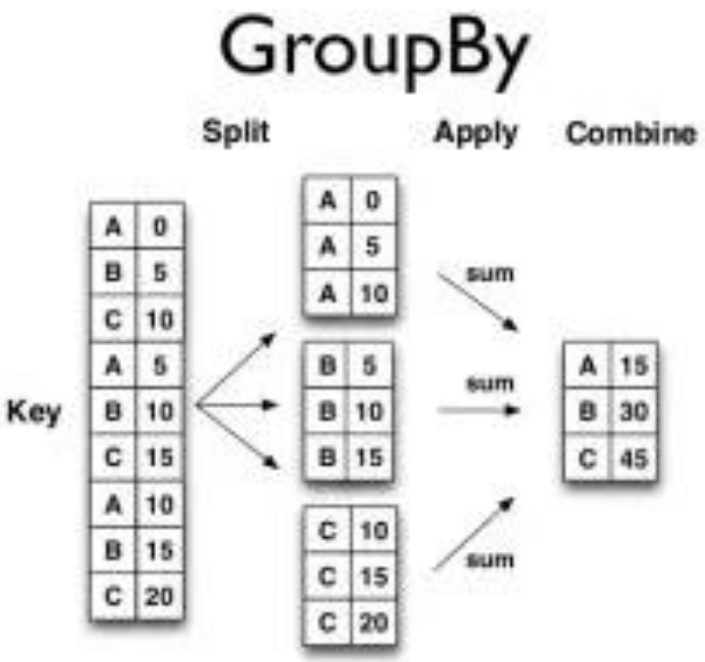
df2

	A	B	C	D
4	A4	B4	C4	D4
5	A5	B5	C5	D5
6	A6	B6	C6	D6
7	A7	B7	C7	D7

Result

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3
4	A4	B4	C4	D4
5	A5	B5	C5	D5
6	A6	B6	C6	D6
7	A7	B7	C7	D7

Pandas groupby



What's time series

- a series of data points indexed in time order
  - Stock price, ECG, ……

Pandas in time series