07-01-22 -NumPy and Pandas

07 January 2022 11:39

Numpy

Numpy – Python

→ Modules in Py like packages in Java
- → Pandas
- → Numpy
- → Sklearn etc

Numpy → array

→ import numpy as np (alias)
or
import numpy

numpy.array

"dynamically typed lang."

"contigous locations – data stored using np array"
50 X faster than list

constructor

a = numpy.array([1, 2, 3]) ✓
(→ tuple / list)

a = np.array((1, 2, 3)) ✓

(1, 2, 3) X

or a = np.array([1, 2, 3])

or — get only array class.

from numpy import array

a = array([1,2,3])

rank → dimensions → 1

shape → tuple of int showing no. of el. in each dimension → (3,)

or

c = array([[10,20,30],[30,40,50]])

rank = 2

shape = (3,3) → (2,3) X error in Jupyter

c.shape or np.shape(c)

z = np.full((3,2), -1) → fill with "-1"

(3,2) → 3 row, 2 columns

b = np.zeros((3,3)) ⇒ fill with "0"

or np.ones

np.eye(n) → identity matrix → n row & columns

type(n) → data type of 'n'

↳ Base Class function [Built-in module]

in python - datatypes are also classes

n.dtype → type of values : int64 or float64 etc.

↳ bits

x.dtype → type of values : int 64 / … / float
(64 → 8 bytes)

force dtype

`x = np.array([10, 20, 30], dtype = 'f')` — 'f' → float

or 'F' → complex

'i' → int32

'i8' → int 64

or `dtype = np.float64`

Elementwise ADD :

x + y → np.array of same shapes (x: np.array)

or

`np.add(x1, x2)`

SUB : x − y **or** `np.subtract(x, y)`

Similarly, * and /

MUL : (element-wise) x * y **or** `np.multiply(x, y)`

DIV : x / y or `np.divide(x, y)`

Transpose : **x.T**

Comments : line : #
block : ''' … '''

`np.sqrt(x)` : element wise $\sqrt[2]{\ }$

Matrix Ops.

Multiply matrix : `np.dot(x,y)`

or

matrix & vector both

or `x.dot(y)`

shape of x & y should be like:

- (2,3) & (3,2)
- or
- (3,3) & (3,3)

`a = np.arange(start, stop, step, dtype)`

- start → default '0'
- stop → mandatory, end limit '10' → 0,1,2,3,4,5,6,7,8,9
- step → Default → 1
- dtype → Default → 1

`a = a.reshape(2, 5)` (2 rows, 5 columns)

or `a = np.arange(...).reshape`

linspace Vs. arange

- linspace → no step
- no. of vals we want in this range

`b = np.linspace(10, 20, 4)` also endpoint parameter?

- 10 → start
- 20 → end
- 4 → no. of values i want
- endpoint → excludes end

param: retstep

step = (20 − 10)/4 or (19 − 10)/4 if endpoint=false

Pandas

dataframe ???

{ "weight": [50, 60, 70], "height": [150, 160, 170] }

data = {"weight": [50, 60, 70], "height": [150, 160, 170]}
↳ dictionary

import pandas as pd
df = pd.DataFrame(data) # dataframe object

print(df.head)

df.loc[0] ⇒ print 0th row

Also specify index name

df = pd.DataFrame(data, index = ["P1", "P2", "P3"])

df.loc["P2"]

Slicing :

a = np.array([[1,2,3], [4,5,6], [7,8,9]])

a[start : end : step] ↳ default: 1 } for 1D

default 0

This index is excluded

for 2D or multi : Slicing for each dimension seperately

a[1:3 , 3:5]

rows ↓ → columns → column 3 & 4

row 1 & 2

All rows or All columns a[:, 3:5]

all rows included

a [:, :]

↪ whole array