NUMERI RANDOM

- np random
- np default - nrf

np random

- Genera numeri casuali

Numeri random tra 0 e 7

np.random.~~randint~~RANDINT(7) = np.random.randint(0, 7)

INCLUSO ESCLUSO

> 7

> 5

> 4

np.random.

- Numeri casuali tra 0 e 1

np.random.rand()

- Numeri casuali tra 0.0 e 1.0

np.random.random()

GENERAZIONE ARRAY CASUALI

- ARRAY 1D

~~np.random.array~~

np.random.randint(6, 42, 8)

- 6: inf incluso
- 42: sup escluso
- 8: quanti elementi compongono l'array

ES

[6, 6, 13, 5, 8, 40, 7, 22]

8 elementi.

- Numeri ~~distribuiti~~ casuali tra 0 e 1 formanti un array

np.random.rand(6)

6: lunghezza dell'array

[0,233, 0,8982, ..., 0,556] ← 6 elementi

B

- ARRAY BIDIMENSIONALE

np.random.randint $(6, 17, (4, 2))$

4 triple e 2 colonne

6 incluso, 17 escluso

ESEMPIO OUTPUT

```
[
  [10, 12],
  [15, 3],
  [1, 16],
  [5, 5]
]
```

- SELEZIONE ELEMENTO CASUALE DA ARRAY

```
listaA = [6, 10, 15, 8]
arrayA = np.array(listaA)
np.random.choice(arrayA)
```

OUTPUT ESEMPIO

10

Pg

NUMPY. RANDOM

~~Con~~ random.random()

Si genera un float compreso tra 0 e 1.

Ovviamente si può usare il seed per ottenere sempre lo stesso numero:

```
random.seed(10)
random.random()
```

> 0,5714

random.random((2,4))

Genera un array costituito ~~tra~~ con 2 righe e 4 colonne con valori compresi tra 0.0 e 1.0 (Float).

default_rng

import numpy as np

rng = np.random.default_rng(seed 221)

arr_random = rng.random(5) → Nr elementi dell'array

> array([0.45, 0.26, 0.99, 0.67, 0.32])

Nr casuali tra 0.0 e 1.0

random
~~default_rng~~ è una funzione presente in np.random.default_rng(seed)

Possiamo anche generare numeri casuali in base ad una distribuzione Normale. Invece di np.random.default_rng(seed).random(),

Se usa np.random.default_rng(seed).normal

E

~~rng = default_rng(221)~~ rng = np.random.default_rng(221)

N = 5

G = 1

normal_random = ~~rng~~ rng.normal(N, G, 10)

nr elementi

> array([numeri casuali])

Generiamo numeri casuali interi tra

un minimo e un max

np.random.default_rng(221).~~random~~ integers(3, 7, 5)

min max nr elementi

> array([4, 4, 4, 4, 4])