In [ ]:

Python Random Number

Need of Python Random Number

- A Random Number in Python is any number in a range we decide.
- From initializing weights in an ANN to splitting data into random train and test sets, the need for generating random numbers is apparent.
- Another use-case could be the random shuffling of a training dataset in stochastic gradient descent.

So today, we will discuss pseudorandom generators in Python. We will also try doing that with the standard Python library and with NumPy.

How to Generate Python Random Number?

- What we really generate is pseudorandom numbers. These are numbers that appear nearly random but are actually something we generate with a deterministic process.
- Python uses the Mersenne Twister pseudorandom number generator.
- The process of generating random numbers involves deterministically generating sequences and seeding with an initial number.
- The default for the seed is the current system time in seconds/ milliseconds. A different seed will produce a different sequence of random numbers.

1. Import Python Random

Before we can begin, let’s first import the module random from the Python Standard Library. You can directly import it-

import random

To import a piece of functionality from it- say, random, you can:

from random import random

Or for seed, you can:

from random import seed

In [ ]:

In [ ]:

In [ ]:

In [ ]:

2. Random Floating Point Numbers

- Let’s take an example of generating Python Random Number.

```
In [1]: from random import seed
        from random import random
        seed(7)
        random(),random(),random(),random()
```

```
Out[1]: (0.32383276483316237,
         0.15084917392450192,
         0.6509344730398537,
         0.07243628666754276)
```

- Works for us. Now, what if we reseed to the same value and call the random() functions/methods again?

```
In [2]: seed(7)
        random(),random(),random(),random()
```

```
Out[2]: (0.32383276483316237,
         0.15084917392450192,
         0.6509344730398537,
         0.07243628666754276)
```

- You’ll find it gives us the same thing as it did earlier.
- Working with minimum values and multiplying the floats-
- Some days, you may not want to generate Random Number in Python values between 0 and 1.
- In the following piece of code, 2 is the minimum value, and we multiple the random number generated by 10.

```
In [3]: seed(7)
        2+10*random()
```

```
Out[3]: 5.238327648331624
```

3. Python Random Integers

- We use the randint() function to get integers instead, randomly.
- It takes two arguments- the start and the top, and then draws a random value from a uniform distribution. Each value has an equal chance of being picked.

```
In [5]: from random import randint
        seed(7)
        randint(0,9),randint(0,9),randint(0,9)
```

```
Out[5]: (5, 2, 6)
```

- We asked for three random values, this gave us 5, 2, and 6.

```
In [ ]:
```

4. Getting Integers Randomly from a Range

- randrange() randomly selects an element from range(start,stop,step).

```
In [9]: from random import randrange
        randrange(-2,4)

Out[9]: 2
```

```
In [10]: from random import randrange
         randrange(-2,4)

Out[10]: -2
```

5. Random Gaussian Values

- The gauss() function takes in two arguments- the mean and the standard deviation. This gives us a real-valued distribution.

```
In [11]: from random import gauss
         seed(7)
         gauss(0,1),gauss(0,1),gauss(0,1)

Out[11]: (-0.2558802884476004, 0.511431512516514, -0.2260961647831047)
```

6. Choosing Randomly From Lists

- It is possible to randomly pick values from our own custom lists. We have the choice() function/method for this.

```
In [12]: list=[2,4,3,9,6,2,1,0,7,4,3,5,3,6,8]
         from random import choice
         seed(7)
         choice(list),choice(list),choice(list),choice(list),choice(list),choice(list)

Out[12]: (2, 3, 1, 3, 2, 4)
```

- **For a choice of multiple values, you can use choices() instead.**

```
In [13]: from random import choices
         choices(list,k=4)

Out[13]: [3, 4, 7, 6]
```

```
In [ ]:
```

```
In [ ]:
```

7. Randomly Choosing a Subset from a List

- Once sample() puts an item from a list into the sublist, it does not add it back to the original list for picking from.
- This is selection without replacement. Note that this does not modify the original list. This function/ method also takes the size of the subset to create.

```
In [14]: list
```

```
Out[14]: [2, 4, 3, 9, 6, 2, 1, 0, 7, 4, 3, 5, 3, 6, 8]
```

```
In [15]: from random import sample
         sample(list,6)
```

```
Out[15]: [9, 2, 4, 1, 5, 3]
```

8. Shuffling a List Randomly

- We can shuffle a list like a deck of cards with the shuffle() function/ method.
- This shuffles the list in-place. In other words, it does not need to create a new list to put shuffled items into one by one.

```
In [16]: list
```

```
Out[16]: [2, 4, 3, 9, 6, 2, 1, 0, 7, 4, 3, 5, 3, 6, 8]
```

```
In [17]: from random import shuffle
         shuffle(list)
         list
```

```
Out[17]: [0, 3, 3, 3, 6, 5, 2, 8, 6, 4, 2, 1, 7, 4, 9]
```

- Note that this modifies the list.

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

How to Generate Python Random Number with NumPy?

- With the seed() and rand() functions/ methods from NumPy, we can generate random numbers. The functionality is the same as above.

```
In [18]: from numpy.random import seed
         from numpy.random import rand
         seed(7)
         rand(3)

Out[18]: array([0.07630829, 0.77991879, 0.43840923])
```

```
In [19]: seed(7)
         rand(3)

Out[19]: array([0.07630829, 0.77991879, 0.43840923])
```

```
In [ ]:
```

```
In [ ]:
```

Python Interview Questions on Random Numbers

- How to choose random number from list in Python?
- How to generate random number using NumPy?
- Explain the way to get Random Gaussian Numbers.
- How does python generate Random Numbers?
- What is the need to generate random number in Python?