Probability (distribution) from the numpy.random.choice() incorrect #12509

Closed **Cuban-Pete** opened this issue on Dec 8, 2018 · 6 comments

Cuban-Pete commented on Dec 8, 2018 • edited

```
# Probability (distribution) from the random.choice function in Numpy does not work correctly
# Added the Python (random.choice) "way" as example which does work correctly
# tested on version '1.15.4' numpy and python 3.7

import numpy as np
import random

values = [1, 2, 3, 4] # pick from 4 values
p_values = [0, .5, 0, .5] # use this probability distribution. NOTE: for 1 and 3 the probability :

results_np = np.random.choice(values, 10, p_values)

results_py = []
for i in range(0,10):
    results_py.append(random.choices(values, p_values))

print(results_np)
print (results_py)

# Gives for example:
#[3 4 2 4 3 4 1 2 3 2] # <--- this is wrong! 3 and 1 should not be there. probability is zero!
#[[4], [2], [2], [4], [2], [4], [4], [4], [2], [4]] #<-- correct!
```

seberg commented on Dec 8, 2018

OK, that is unlucky that cython (and most other things) are not more picky about casting to booleans. You are passing the `replace=True` kwarg here, since the p-value list evaluates to True -- that is instead of the actual `p` kwarg.

I think we should make the replace error out more aggressively probably (i.e. see if it can be safely cast to an int, that should cover most cases, or even stricter check like in indexing, but it is probably unnecessary here).

1

Cuban-Pete commented on Dec 8, 2018 • edited

Ah, I see what I'm doing wrong. However changing it from:

```
results_np = np.random.choice(values, 10,p_values)
```

to

```
results_np = np.random.choice(values, 10, replace=False, p_values)
```

will give this error:

> SyntaxError: positional argument follows keyword argument

Edit:
With this it works:

```
results_np = np.random.choice(values, 10, 1, p_values)
```

So it is not a bug after all. Sorry for wasting time.

rkern commented on Dec 9, 2018

`np.random.choice(values, 10, p=p_values)` is the idiomatic way to write that.

2

rkern closed this on Dec 9, 2018

christabelle commented on May 13

```
values = [0, 1]
p_values = [.8, .2]
print(np.random.choice(values, 20, 1, p=p_values))
```

I'm trying this, to try generating ones 20% of the time. And I get this:

`[0 0 0 1 0 1 1 1 0 0 0 0 1 0 0 0 1 1 0 0] [0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0]`

Is there a soln to this? Thanks!

seberg commented on May 13

There seems nothing wrong with your output. If you want exactly 4 values to be True, maybe try shuffling the array instead, since that is a random order, not a random drawing.

matthew-brett commented on May 13

@christabelle - for your code - I get this:

```
[0 1 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0]
```

That's what I was expecting - 20 values where each one has a 20% chance of being 1. Obviously that's hard to see with only 20 values, but:

```
>>> choices = np.random.choice(values, 20000, 1, p=p_values)
>>> np.count_nonzero(choices) / len(choices)
0.20225
```

But - were you expecting something else?

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked pull requests

Successfully merging a pull request may close this issue.

None yet

5 participants