learn-co-curriculum / dsc-bernoulli-and-binomial-distribution-lab Public

View license

0 stars 166 forks

Star Watch

Code Issues 1 Pull requests Actions Projects Security Insights

solution

This branch is 9 commits ahead, 9 commits behind master.

lmcm18 fixed random seed used in solutions for consistency; added li… … on Oct 28, 2019 13

View code

README.md

Bernoulli and Binomial Distribution - Lab

Introduction

In this lab, you'll practice your newly gained knowledge on the Bernoulli and Binomial Distribution.

Objectives

You will be able to:

- Apply the formulas for the Binomial and Bernoulli distribution to calculate the probability of a specific event
- Use `numpy` to randomly generate Binomial and Bernoulli trials
- Use `matplotlib` to show the output of generated Binomial and Bernoulli trials

Apply the formulas for the Binomial and Bernoulli distributions

When playing a game of bowling, what is the probability of throwing exactly 3 strikes in a game with 10 rounds? Assume that the probability of throwing a strike is 25% for each round. Use the formula for the Binomial distribution to get to the answer. You've created this before, so we provide you with the function for factorials again:

```
def factorial(n):
    prod = 1
    while n >= 1:
        prod = prod * n
        n = n - 1
    return prod
```

```
p_3_strikes = (factorial(10)/(factorial(7)*factorial(3)))*(0.25)**3*(0.75)**7
p_3_strikes
```

```
0.25028228759765625
```

Now, create a function for the Binomial distribution with three arguments n, p and k just like in the formula:

$$P(Y=k)=\binom{n}{k}p^k(1-p)^{(n-k)}$$

```
def binom_distr(n,p,k):
    p_k = (factorial(n)/(factorial(k)*factorial(n-k)))*(p**k*(1-p)**(n-k))
    return p_k
```

Validate your previous result by applying your new function.

```
binom_distr(10,0.25,3)
```

```
0.25028228759765625
```

Now write a `for` loop along with your function to compute the probability that you have five strikes or more in one game. You'll want to use `numpy` here!

```
import numpy as np
prob = 0
for i in np.arange(5,11):
     prob += binom_distr(10,0.25,i)
prob
```

```
0.07812690734863281
```

Use a simulation to get the probabilities for all the potential outcomes

Repeat the experiment 5000 times.

```
np.random.seed(123)
n = 5000
iteration = []
for loop in range(n):
    iteration.append(np.random.binomial(10, 0.25))
    np_it = np.array(iteration)
```

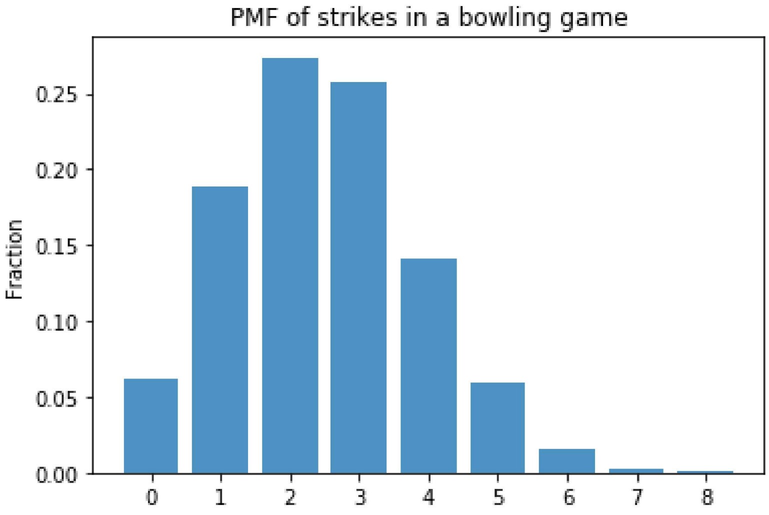
```
values, counts = np.unique(np_it, return_counts=True)
print(values)
print(counts)
```

```
[0 1 2 3 4 5 6 7 8]
[ 310  941 1368 1286  707  297   78   11    2]
```

Visualize these results

Create the PMF using these empirical results (that is, the proportions based on the values we obtained running the experiment 5000 times).

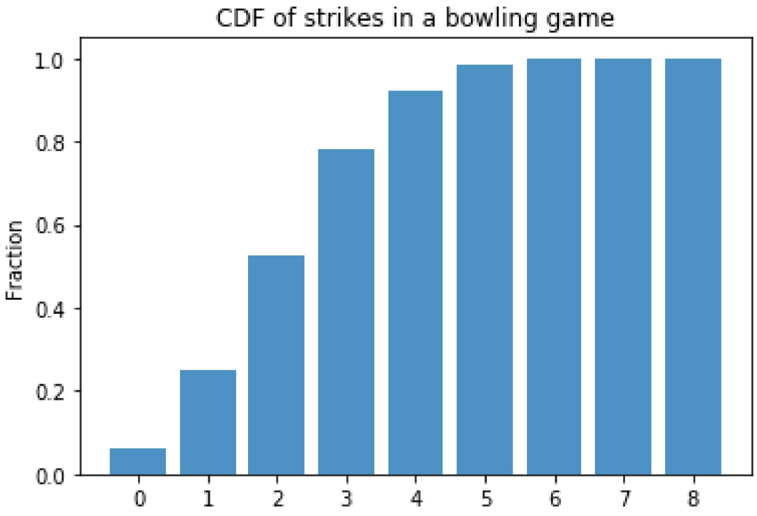
```
import matplotlib.pyplot as plt
%matplotlib inline
plt.bar(values, counts/5000, align='center', alpha=0.8)
plt.xticks(values)
plt.ylabel('Fraction')
plt.title('PMF of strikes in a bowling game');
```



You should see that, with a 25% strike hit rate, even when simulating 5000 times, an almost perfect and/or perfect game of 9 and 10 strikes didn't even occur once! If you change the random seed, however, you'll see that perfect games will show up occasionally.

Next, let's create the CDF based on these results. You can use `np.cumsum` to obtain cumulative probabilities.

```
import matplotlib.pyplot as plt
plt.bar(values, np.cumsum(counts/5000), align='center', alpha=0.8)
plt.xticks(values)
plt.ylabel('Fraction')
plt.title('CDF of strikes in a bowling game');
```



Summary

Congratulations! In this lab, you practiced your newly gained knowledge of the Bernoulli and Binomial Distribution.

Releases

No releases published

Packages

No packages published

Contributors 4

- **LoreDirick** Lore Dirick
- **PeterBell** Peter Bell
- **lmcm18**
- **mas16** matt

Languages

● **Jupyter Notebook** 89.9% ● **Python** 10.1%