PROJECT 3 - BINOMIAL AND POISSON DISTRIBUTIONS

EE 381 – Probability and Stats Computing

Abstract

This project explores random experiments that can be described by probability distributions. These distributions are the Binomial and Poisson distributions.

Brian Nguyen

015188926

1. Experimental Bernoulli Trials

Introduction

This project will simulate the rolling of three multi-sided unfair dice *n* = 1,000 times. The probability vector for the multi-sided die is p = [0.2, 0.1, 0.15, 0.3, 0.2, 0.05]. A roll is considered a "success" if the first die rolls a "one," the second die rolls a "two," and the third die rolls a "three." The experiment is repeated for *N* = 10,000 times in order to generate a PMF plot.

Methodology

To create a roll, I used the function

- `np.random.choice(c, n, p)`

where c = [1, 2, 3, 4, 5, 6], p = [0.2, 0.1, 0.15, 0.3, 0.2, 0.05], and n = 1000. This generates a list of size 1000 that contains the side that is landed on in each roll. I created three variables for each roll.

- `roll_1 = np.random.choice(c, n, p)`
- `roll_2 = np.random.choice(c, n, p)`
- `roll_3 = np.random.choice(c, n, p)`

In order to track the successes of each experiment, I implemented a while loop within the range of *n* that will iterate through each index. If the first roll is a "one," the second roll is a "two" and the third roll is a "three," I incremented the success counter.

Results and Conclusion

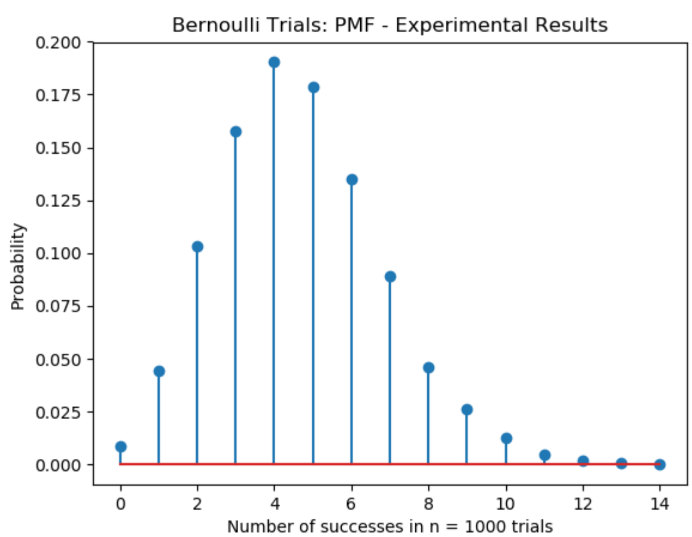

Figure 1

Appendix

```
import numpy as np
import matplotlib.pyplot as plt


def bernoulli_trials(N):
    c = [1, 2, 3, 4, 5, 6]
    p = [0.2, 0.1, 0.15, 0.3, 0.2, 0.05]
    n = 1000
    the_list = []
    success = 0
    for x in range(N):
        roll_1 = np.random.choice(c, n, p)
        roll_2 = np.random.choice(c, n, p)
        roll_3 = np.random.choice(c, n, p)
        for i in range(len(roll_1)):
            if roll_1[i] == 1 and roll_2[i] == 2 and roll_3[i]
== 3:
                success += 1
        the_list.append(success)
        success = 0
    # ----------------------------------------------------------
    b = range(0, 16)
    sb = np.size(b)
    h1, bin_edges = np.histogram(the_list, bins=b)
    h1 = h1 / N
    b1 = bin_edges[0:sb - 1]
    plt.close('all')

    plt.figure(1)
    plt.stem(b1, h1)
    plt.title('Bernoulli Trials: PMF - Experimental Results')
    plt.xlabel('Number of successes in n = 1000 trials')
    plt.ylabel('Probability')
    plt.show()


def main():
    bernoulli_trials(10000)
main()
```

2. Calculations Using Binomial Distribution

Introduction

This experiment uses the theoretical formula for the Binomial Distribution to calculate the probabilities of successes for each roll of the unfair n-sided die in the previous experiment. The PMF plot is generated using the Binomial formula.

$$\binom{n}{k} p^k q^{n-k}$$

Methodology

To use the Binomial formula, we first must know what the probability of success is. Given from our probability vector, we can see that

- P(success) = P(1st die = "1" ∩ 2nd die = "2" ∩ 3rd die = 3)
- = P(1st die = "1") x P(2nd die = "2") x P(3rd die = "3")
- = (0.2) (0.1) (0.15)
- = 0.003

In order to plot this function, I used the function

- `np.random.binomial(1000, 0.003, size=1000)`

to generate a list of successes in *n* = 1000 trials. The list is then plotted

Results and Conclusion

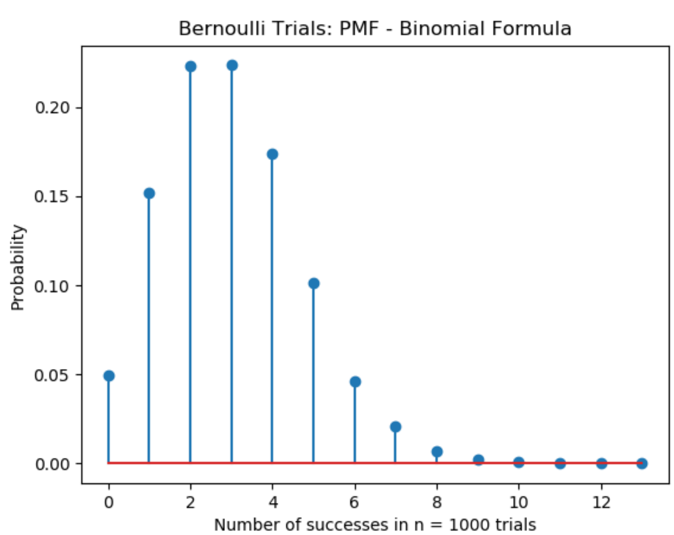

Figure 2

Appendix

```
import numpy as np
import matplotlib.pyplot as plt

# parameters:
#   number of trials, probability of success, repetition of
experiment
s = np.random.binomial(1000, 0.003, size=1000)
print(s)
b = range(0, 15)
sb = np.size(b)
h1, bin_edges = np.histogram(s, bins=b)
h1 = h1 / 10000
b1 = bin_edges[0:sb - 1]
plt.close('all')

plt.figure(2)
plt.stem(b1, h1)
plt.title('Bernoulli Trials: PMF - Binomial Formula')
plt.xlabel('Number of successes in n = 1000 trials')
plt.ylabel('Probability')
plt.show()
```

3. Approximation of Binomial by Poisson Distribution

Introduction

In the event that the probability *p* of success in a Bernoulli trial is small and the number of trials *n* is large, we would use the Poisson distribution formula:

$$f(X = k) \cong \frac{\lambda e^{-\lambda}}{k!}$$

This approximation is only valid for rare events, i.e.:

- $n > 50$ – large number of trials
- $\lambda = np < 5$ – small probability of success

Using λ as a parameter and the formula, the experiment requires to plot the probability distribution function that approximates the distribution of the random variable

$$X = \{number\ of\ successes\ in\ n\ Bernoulli\ trials\}$$

Methodology

The parameter λ is *np*, where *n* is the number of trials and *p* is the probability of success. Since this experiment is 1,000 trials and the probability of success is 0.003, the value of λ is 3. This means out of 1,000 trials, we would expect that we would get an approximation of 3 successes.

In order to generate and plot these values, I used the function

- `np.random.poisson(3, 1000)`

to generate a list of approximated successes in 1000 trials. The values in the list is then plotted

Results and Conclusion

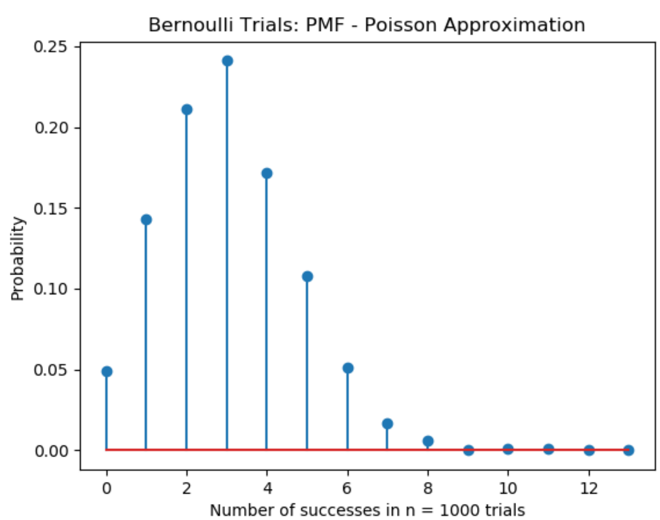

Figure 3

Appendix

```
import numpy as np
import matplotlib.pyplot as plt

# parameters:
#   number of trials, probability of success, repetition of
experiment
s = np.random.binomial(1000, 0.003, size=1000)
print(s)
b = range(0, 15)
sb = np.size(b)
h1, bin_edges = np.histogram(s, bins=b)
h1 = h1 / 10000
b1 = bin_edges[0:sb - 1]
plt.close('all')

plt.figure(2)
plt.stem(b1, h1)
plt.title('Bernoulli Trials: PMF - Binomial Formula')
plt.xlabel('Number of successes in n = 1000 trials')
plt.ylabel('Probability')
plt.show()
```