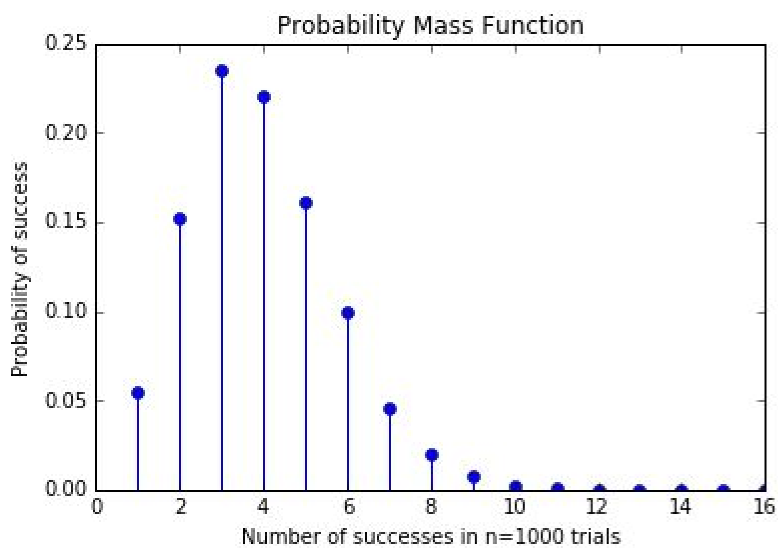
Roberto Sanchez (014587792)
October 17, 2017
EE 381 – Probability and Statistics with Applications to Computing
Lab 3 - Project on Binomial and Poisson Distributions

1) Experimental Bernoulli Trials
   a) **INTRODUCTION:**
      i) The experiment follows tossing 3 dices 1000 times and recording whenever 3 sixes in a roll. Then create a probability mass function plot.
   b) **METHODOLOGY:**
      i) Create several for loops to simulate a toss then doing it multiple times. If we toss 3 sixes we record that success by first checking if the dice roll equals to 6. Once we got the successes we store that on a list and send it to the plotting function.
   c) **RESULTS AND CONCLUSIONS:**
      i) From running the program we get the following Distribution chart:

```
In [28]: runfile('/home/beryl/Downloads/untitled0.py', w
```



2) Calculation using the Binomial Distribution

  a) **INTRODUCTION:**

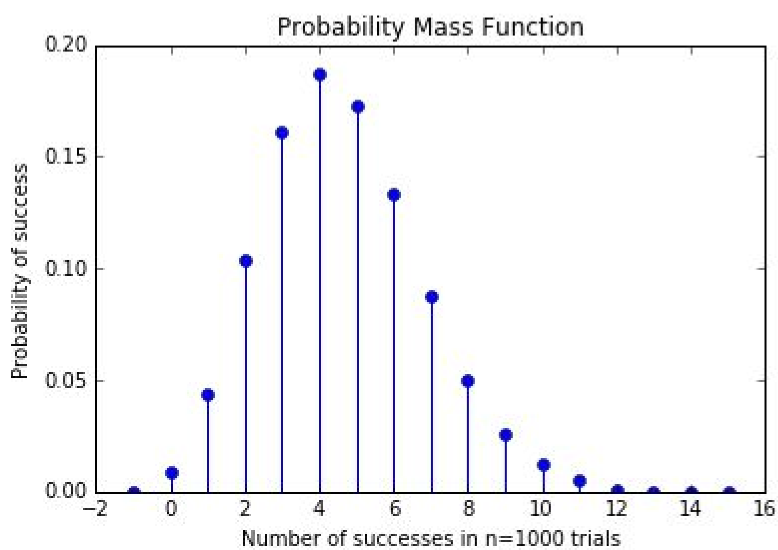
    i) The problem is the same as the previous one however we now use the Binomial distribution to calculate the probability of success of getting 3 sixes in one roll.

  b) **METHODOLOGY:**

    i) We start by having the probability of getting 3 dices to each be 6 in a roll is 1/216. With that we can calculate the Binomial Distribution by getting the values of p,q, and c. Once we have that we can get the calculation and print to a graph.

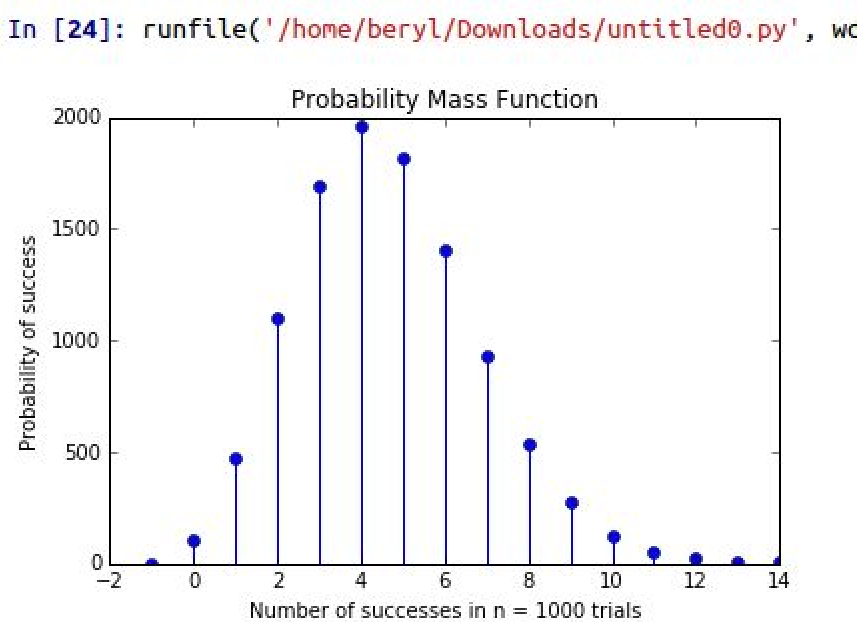
  c) **RESULTS AND CONCLUSIONS:**

```
In [15]: runfile('/home/beryl/Downloads/untitled0.py', w
```



Compared to #1, 4 seem to be higher successes

3) Approximation of Binomial by Poisson Distribution
   a) **INTRODUCTION:**
      i) Same as the previous problem, however we use the Poisson Distribution to approximate the previous answer.
   b) **METHODOLOGY:**
      i) We solve this problem by first declaring all known variables for the Poisson Distribution. We then go and do multiple runs on the equation saving the results to a list which we then print in a distribution.
   c) **RESULTS AND CONCLUSIONS:**



It gave the same result as 3 which means the Poisson gave a pretty good approximation to the binomial

4) Appendix

a) Code for #1

```
# -*- coding: utf-8 -*-

# Name: Roberto Sanchez
# Assignment #3 Part 1
# October 14, 2017

import numpy as np
import random
import matplotlib.pyplot as plt
# Number of rounds ie: number of expiremnets run in this case 10,000
def Experiment(rounds):
    #Tracks all the success per trial
    ListOfSuccesses = []
    # run experiment in number of rounds
    for y in range(rounds):
         #runs one trial, tosses 3 dices and see if its 3 sixes
        TrialSuccessful = 0
        # This is for one trial, repeated 1000 times
        for x in range(1000):
             #tracks if we have a six in a row when tossing 3 dies
             GotaSix = 0
             #Tossing 3 Dies
             for z in range(3):
                 # Is the value 6?
                 if random.randint(1,7) == 6:
                     #track it
                     GotaSix += 1
             #After tossing 3 dies, did we get 3 sixes?
             if GotaSix == 3:
                 #Track it
                 TrialSuccessful += 1;
         #AFter one trial, store value in listS
        ListOfSuccesses.append(TrialSuccessful)
        #print("Success in Trial ",y,": ",TrialSuccessful)
    # This prints out the graph
    b = range(0,17)
    h1, bin_edges = np.histogram(ListOfSuccesses, bins = b)
    b1 = bin_edges[1:17]
    # stem(x,y) cordinates
    plt.stem(b1, h1/rounds)
    plt.title("Probability Mass Function")
    plt.ylabel("Probability of success")
    plt.xlabel("Number of successes in n=1000 trials")

Experiment(10000)
```

b) Code for #2

```

import scipy as sp
import numpy as np
import matplotlib.pyplot as plt
def CalcBinomial(rounds):
    calc = []
    probability = 1/216.
    #does the binomial calculation
    for x in range(0,17):
        c = sp.misc.comb(rounds,x)
        p = np.power(probability, x)
        q = np.power(1-probability, rounds-x)
        for i in range((int)(c * p * q * rounds)):
            calc.append(x)
#This prints out graph
    b = range(-1,17)
    h1, bin_edges = np.histogram(calc, b)
    b1 = bin_edges[0:17]

    plt.stem(b1, h1/rounds)
    plt.title("Probability Mass Function")
    plt.ylabel("Probability of success")
    plt.xlabel("Number of successes in n=1000 trials")

CalcBinomial(1000)
```

c) Code for #3

```
import numpy as np
import matplotlib.pyplot as plt
def CalcPoisson(rounds):
    probability = 1/216
    lamda = probability * rounds
    PoissonNum = []
    e = 2.718281

    for x in range (17):
        lamda2 = np.power(lamda,x)
        e2 = np.power(e,lamda)
        factorial = np.math.factorial(x)
        Poisson = (int)(lamda2*rounds/factorial*e2)
        for i in range(Poisson):
            PoissonNum.append(x)
    #This prints the graph
    b = range(-1,16)
    h1, bin_edges = np.histogram(PoissonNum, bins = b)
    b1 = bin_edges[0:16]

    #plot
    plt.stem(b1, h1/rounds)
    plt.title("Probability Mass Function")
    plt.ylabel("Probability of success")
    plt.xlabel("Number of successes in n = 1000 trials")

    #show graph
    plt.show()
CalcPoisson(1000)
```