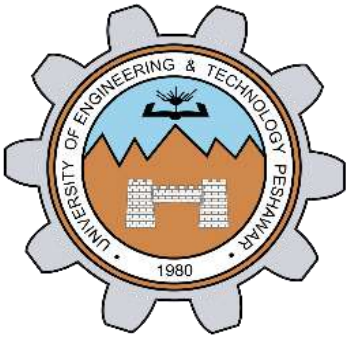
Probability Methods in Engineering

Assignment 5 & 6



Submitted by:

MOEEN KHAN (21PWCSE2069)

AWAIS SADDIQUI (21PWCSE1993)

Class Section: A

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Student Signature: _____

Submitted to:

Dr AMAD KHALIL

June 14th ,2023

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Task 01:

Conditional Probability:

An urn contains two black balls and three white balls. Two balls are selected at random from the urn without replacement and the sequence of colors is noted. Find the probability that both balls are black. **P[A|b]=P[A inter B]/P[B]**

Code:

```
sample_space=["Black","Black","White","White","White"]
W=len([x for x in sample_space if x == 'White'])
B=len([x for x in sample_space if x == 'Black'])
size_of_S=len(sample_space)

#P[B2 intersection B1]= P[B2|B1]/P[B1];

probability_B1=B/size_of_S
probability_W1=W/size_of_S

def Prob_B2_B1(sample_space, W, B):
    print("After one draw of black")
    newArray=sample_space.copy()
    newArray.remove("Black")
    newSize=len(newArray)
    B=B-1
    Prob_B2_B1=B/newSize
    Prob_W2_B1=W/newSize
    print(f"Probabilty of B2 given B1= {B}/{newSize} = ",Prob_B2_B1)
    print(f"Probabilty of W2 given B1= {W}/{newSize} = ",Prob_W2_B1)
    print(newArray)
    #print(newSize)
    return Prob_B2_B1

probabilty_B2_B1=Prob_B2_B1(sample_space, W, B)
P_B1_inter_B2=probabilty_B2_B1 * probability_B1
print("P[B1 |-| B2] = ",P_B1_inter_B2)
```

Output:

```
[Running] python -u "e:\Computer_System-Engineering\Fourth Semester\PME\Python
py"
After one draw of black
Probabilty of B2 given B1= 1/4 =  0.25
Probabilty of W2 given B1= 3/4 =  0.75
['Black', 'White', 'White', 'White']
P[B1 ⊢⊣ B2] =  0.1
```

Task 02:

A coin is thrown 3 times .what is the probability that atleast one head is obtained?

Code:

```
sample_space=["HHH","HHT","HTH","THH","TTH","THT","HTT","TTT"]
size_of_sample_space=len(sample_space)


probability_no_head = 1 / size_of_sample_space

# Calculate the probability of at least one head
probability_of_Atleast_one_head = 1 - probability_no_head
print("S= ",sample_space)
print("Size of sample space is : ",size_of_sample_space)
print("Probability of at least one head:",
probability_of_Atleast_one_head)
```

Output:

```
[Running] python -u "e:\Computer_System-Engineering\Fourth Semester\PME\Python
S=  ['HHH', 'HHT', 'HTH', 'THH', 'TTH', 'THT', 'HTT', 'TTT']
Size of sample space is :  8
Probability of at least one head: 0.875
```

Task 03:

Total Probability:

An urn contains two black balls and three white balls. Two balls are selected at random from the urn without replacement and the sequence of colors is noted. Find the probability that the second ball is white (irrespective of first outcome).

Code:

```
container=["white","white","white","black","black"]
sizeOfContainer=len(container)

whiteBall = len([x for x in container if x == 'white'])
blackBall = len([x for x in container if x == 'black'])

def find_prob_black(array, size , W, B):
    prob_B1=B/size
    array.remove("black")
    size-=1
    B-=1
    prob_W2_B1= W/size
    prob_B2_B1=B/size
    print("Probabilty_B1= ",prob_B1)
    print("After one withDraw : P[B2|B1]= ",prob_B2_B1)
    print("After one withDraw : P[W2|B1]= ",prob_W2_B1)
    return prob_W2_B1, prob_B1

def find_prob_white(array, size , W, B):
    prob_W1=W/size
    array.remove("white")
    size-=1
    W-=1
    prob_B2_W1= B/size
    prob_W2_W1=B/size
    print("Probabilty_W1= ",prob_W1)
    print("After one withDraw : P[W2|W1]= ",prob_W2_W1)
    print("After one withDraw : P[B2|W1]= ",prob_B2_W1)
    return prob_W2_W1,prob_W1

P_W2_B1,P_B1=find_prob_black(container, sizeOfContainer, whiteBall,
blackBall)
P_W2_W1,P_W1=find_prob_white(container, sizeOfContainer, whiteBall,
blackBall)
def find_Prob_W2(P_W2B1,P_B1,P_W2W1,P_W1):
    probability_W2=(P_W2B1*P_B1)+(P_W2W1*P_W1)
    print("probability that the second ball is white P[W2]=
",probability_W2)

find_Prob_W2(P_W2_B1,P_B1,P_W2_W1,P_W1)
```

Output:

```
[Running] python -u "e:\Computer_System-Engineering\Fourth Semester\PME\Python
Probabilty_B1=  0.4
After one withDraw : P[B2|B1]=  0.25
After one withDraw : P[W2|B1]=  0.75
Probabilty_W1=  0.6
After one withDraw : P[W2|W1]=  0.5
After one withDraw : P[B2|W1]=  0.5
probability that the second ball is white P[W2]=  0.6000000000000001
```

Task 04:

Bayes Theorem:

suppose the probability of the weather being cloudy is 40%. The probability of rain on given day is 20%. Also, the prob of clouds on a rainy day is 85%.If it's cloudy outside on a given day, what is the probability that it will rain that day?

Code:

```
def bayesTheorem(pR, pC, pCR):
    return pR * pCR / pC

#define probabilities
pRain = 0.2
pCloudy = 0.4
pCloudyRain = 0.85


prob_rain_cloudy=bayesTheorem(pRain, pCloudy, pCloudyRain)
print("P[Rainy| cloudy]= ",prob_rain_cloudy)
```

Output:

```
[Running] python -u "e:\Computer_System-Engineering\Fourth Semester\PME\Python
py"
P[Rainy| cloudy]=  0.425
```

Task 05:

Binomial Probability Law:

What is the probability of getting heads more than 4 times if you flip a coin 6 times having 0.6 as the probability of heads?

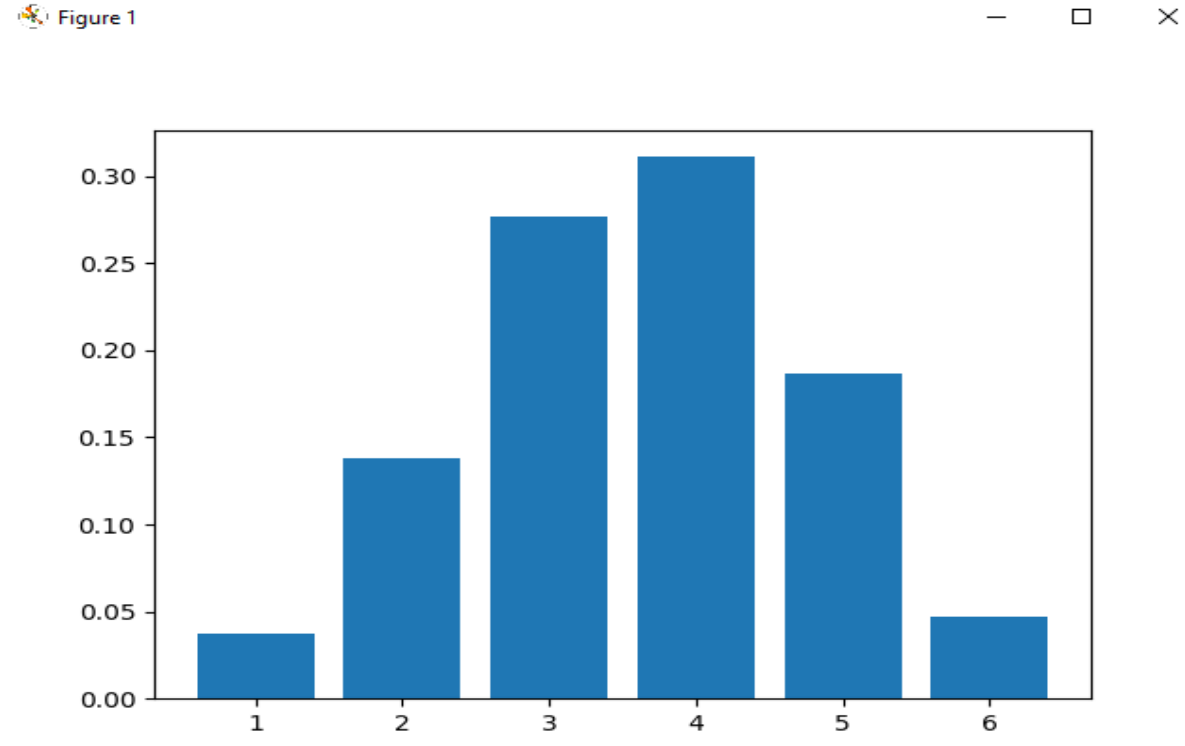
Code:

```
from scipy import stats
import matplotlib.pyplot as plot
#The function stats.binom.pmf takes three arguments:  k,n and p
k=[1,2,3,4,5,6]
prob_success=0.6
num_trail=6

binomial_prob=stats.binom.pmf(k,num_trail,prob_success)
print("Prob_heads more than 4 times =",binomial_prob)
plot.bar(k, binomial_prob)
plot.show()
```

Output:

```
[Running] python -u "e:\Computer_System-Engineering\Fourth Semester\PME\Python
Assignment\Binomial_probability_Law.py"
Prob_heads more than 4 times = [0.036864 0.13824  0.27648  0.31104  0.186624 0.046656]
```



Task 06:

Geometric Probability Law:

What is the probability that the coin has to be flipped i) 4 times ii) more than 4 times, for getting heads for the first time? The probability of heads is 0.6 and the probability of tails is 0.4.

Code:

```
from scipy.stats import geom

k1=4
prob_head_success=0.6
prob_tail_failer=0.4

result=geom.pmf(k1,prob_head_success,loc=0)
coin_flip_more_4=pow(prob_tail_failer,k1)
print(result)
print(coin_flip_more_4)
```

Output:

```
[Running] python -u "e:\Computer_System-Engineering\Fourth Semester\PME\Python
py"
0.03840000000000001
0.025600000000000005

[Done] exited with code=0 in 0.807 seconds
```

Task 07:

Random Variable:

Let X be the number of heads in three independent tosses of a coin. Find the pmf of X. (Binomial RV)

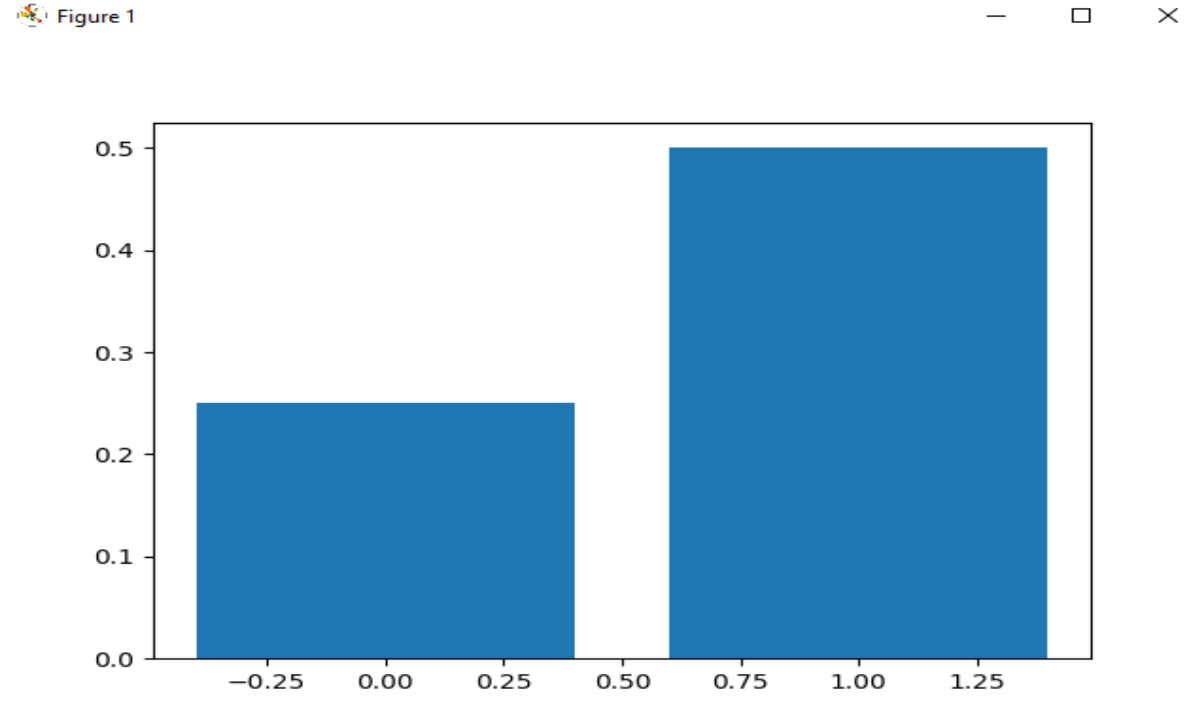
Code:

```
from scipy import stats
import matplotlib.pyplot as plot
outcome_Zeta=[0,1]
size=len(outcome_Zeta)
prob=1/size
print(prob)
result=stats.binom.pmf(outcome_Zeta,2,prob)
for i in range(len(result)):
    print(f"P[{i}]= ",result[i])

plot.bar(outcome_Zeta, result)
plot.show()
```

Output:

```
[Running] python -u "e:\Computer_System-Engineering\Fourth Semester\PME\Python
0.5
P[0]=  0.25
P[1]=  0.5000000000000002
```



Task 08:

Expected Values:

Let X be the number of heads in three tosses of a fair coin. Find E[X].

Code:

```
import numpy as np
def expected_value(values, prob):
    values = np.asarray(values)
    prob = np.asarray(prob)
    return (values * prob).sum() / prob.sum()


zeta=["HHH", "HHT", "HTH", "HTT", "THH", "TTH", "THT", "TTT"]
values = [0, 1, 2, 3]
size=len(values)

probs  = [1/size, 3/size, 3/size, 1/size]


result=expected_value(values, probs)
print("S_x= ",zeta)
print("E[X]= ",result)
```

Output:

```
[Running] python -u "e:\Computer_System-Engineering\Fourth Semester\PME\Python Assignment\Expected_value.py"
S_x=  ['HHH', 'HHT', 'HTH', 'HTT', 'THH', 'TTH', 'THT', 'TTT']
E[X]=  1.5

[Done] exited with code=0 in 0.331 seconds
```