**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)

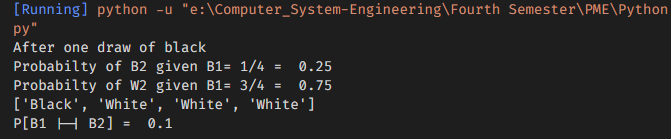
    return Prob\_B2\_B1

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:**



**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

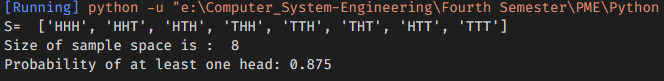
probability\_of\_Atleast\_one\_head = 1 - probability\_no\_head

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:**



**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)

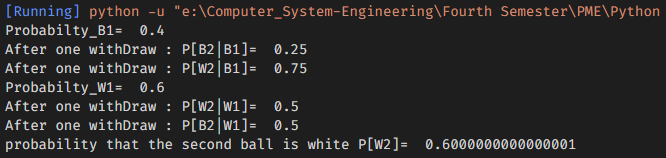
def find\_Prob\_W2(P\_W2B1,P\_B1,P\_W2W1,P\_W1):

    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:**



**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

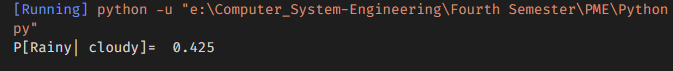
pCloudy = 0.4

pCloudyRain = 0.85

prob\_rain\_cloudy=bayesTheorem(pRain, pCloudy, pCloudyRain)

print("P[Rainy| cloudy]= ",prob\_rain\_cloudy)

**Output:**



**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

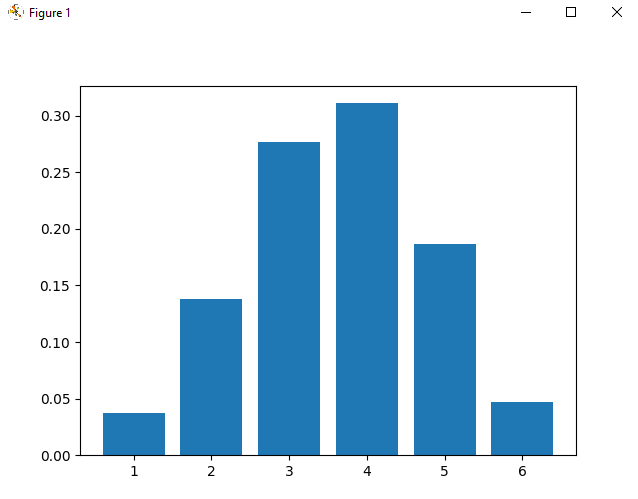
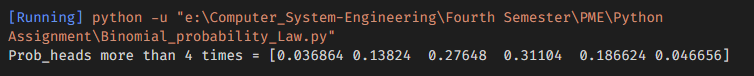
binomial\_prob=stats.binom.pmf(k,num\_trail,prob\_success)

print("Prob\_heads more than 4 times =",binomial\_prob)

plot.bar(k, binomial\_prob)

plot.show()

**Output:**



**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

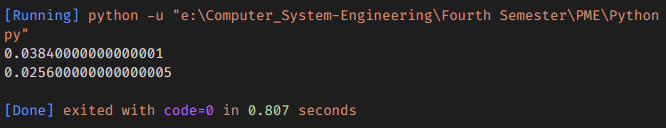
result=geom.pmf(k1,prob\_head\_success,loc=0)

coin\_flip\_more\_4=pow(prob\_tail\_failer,k1)

print(result)

print(coin\_flip\_more\_4)

**Output:**



**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)

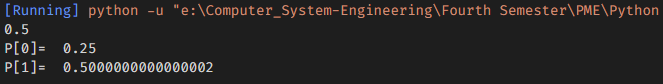
for i in range(len(result)):

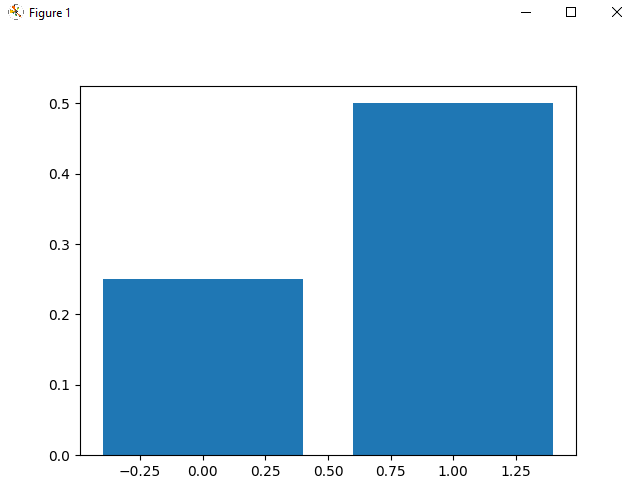
    print(f"P[{i}]= ",result[i])

plot.bar(outcome\_Zeta, result)

plot.show()

**Output:**





**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:**

