```
In [4]: # Define the Log Likelihood Function needed
def PROB(params,k,t):
    n,alpha = params
    if k==0:
        return (alpha/(alpha+t))**n
    else:
        return (((n+k-1)*t)/(k*(alpha+t)))*PROB(params,k-1,t)

def LLnbd(params,t,p,K):
    ll = []
    for i in range(len(p)):
        ll = np.append(ll,(p[i]*np.log(PROB(params,K[i],t))))
    return ll

# Negative of Log Likelihood for minimization function
def NLLnbd(params,t,p,K):
    return (-1)*(np.sum(LLnbd(params,t,p,K)))

# Defining p = 'No of people' and K = 'No of Packs'
p = np.array(df['People'])
K = np.array(df['Packs'])
t = 1

# Maximising the Log Likelihood
soln_nbd = minimize(NLLnbd,
                    args = (t,p,K),
                    x0 = np.array((1,1)),
                    bounds = [(0.000001,None),(0.000001,None)],
                    tol=1e-10,
                    options={'ftol' : 1e-8},)

# Get the optimal solution
n = soln_nbd.x[0]
alpha = soln_nbd.x[1]
ll_nbd = (-1)*(soln_nbd.fun)
print(f"Optimal Solution for NBD Model:")
print(f"Shape Parameter, n = {round(n,6)}")
print(f"Scale Paramenter, alpha = {round(alpha,6)}")
print(f"Log likelihood = {round(ll_nbd,2)}")
```

```
Optimal Solution for NBD Model:
Shape Parameter, n = 0.997636
Scale Paramenter, alpha = 0.24996
Log likelihood = -1140.02
```

(c) The Zero Inflated NBD model:

The zero inflated NBD model considers the zeros in the observed data. Since this data has highest number of people who tried 0 packs, there is a scope of improvement if we build a model considering zeros coming from two sources: π- from a fraction π who will never purchase the candy. 1-π:from a fraction who are likely to buy but have not done so yet. The optimal values pie = 0.11309, n = 1.503988, alpha = 0.334199, Log likelihood = -1136.17.

```
In [5]: # Define the Log Likelihood Function needed
def PROB(alpha,n,k,t):
    if k==0:
        return (alpha/(alpha+t))**n
    else:
        return (((n+k-1)*t)/(k*(alpha+t)))*PROB(alpha,n,k-1,t)

def LLzinbd(params,p,K):
    pie,alpha,n = params
    ll = p[0]*np.log(pie + ((1-pie)*PROB(alpha,n,K[0],1)))
    for i in range(1,len(p)):
        ll += p[i]*np.log((1-pie)*PROB(alpha,n,K[i],1))
    return ll

# Negative of log likelihood for minimization function
def NLLzinbd(params,p,K):
    return (-1)*((LLzinbd(params,p,K)))

# Defining p = 'No of people' and K = 'No of Packs'
p = np.array(df['People'])
K = np.array(df['Packs'])

# Maximising the log likelihood
soln_zinbd = minimize(NLLzinbd,
                      args = (p,K),
                      x0 = np.array((1,1,1)),
                      bounds = [(0.000001,0.999999),(0.000001,None),(0.000001,None)],
                      tol=1e-10,
                      options={'ftol' : 1e-8},)

# Get the optimal solution
pie = soln_zinbd.x[0]
n = soln_zinbd.x[2]
alpha = soln_zinbd.x[1]
ll_zinbd = (-1)*(soln_zinbd.fun)
print(f"Optimal Solution for Zero-Inflated NBD Model :")
print(f"Fraction, pie = {round(pie,6)}")
print(f"Shape Parameter, n = {round(n,6)}")
print(f"Scale Paramenter, alpha = {round(alpha,6)}")
print(f"Log likelihood = {round(ll_zinbd,2)}")
```

```
Optimal Solution for Zero-Inflated NBD Model :
Fraction, pie = 0.11309
Shape Parameter, n = 1.503988
Scale Paramenter, alpha = 0.334199
Log likelihood = -1136.17
```

(d) Finite Mixture models for 2, 3, and 4 segments:

The Finite Mixture models relaxes the assumption that the rate parameter Lambda follows gamma distribution and segments the individuals into S discrete segments with πs: probability that an individual is from segment S.

(i) 2 Segment Finite Mixture Model

In [6]:

```
# Define the Log Likelihood Function needed
def LL2seg(params,p,k):
    pie,lambda1,lambda2 = params
    ll = []
    for i in range(len(p)):
        ll = np.append(ll,p[i]*(np.log((pie*(poisson.pmf(k[i],lambda1)))+((1-pie)*(poisson.pmf(k[i],lambda2))))))
    return ll

# Negative of log likelihood for minimization function
def NLL2seg(params, p, k):
    return (-1)*(np.sum(LL2seg(params, p, k)))

# Defining p = 'No of people' and k = 'No of Packs'
p = np.array(df['People'])
k = np.array(df['Packs'])

# Maximising the log likelihood
soln_2seg = minimize(NLL2seg,
                     args = (p,k),
                     x0 = np.array((1,1,1)),
                     bounds = [(0.000001,0.999999),(0.000001,None),(0.000001,None)],
                     tol=1e-10,
                     options={'ftol' : 1e-8},)

# Get the optimal solution
pie_2seg = soln_2seg.x[0]
lambda1_2seg = soln_2seg.x[1]
lambda2_2seg = soln_2seg.x[2]
ll_2seg = (-1)*(soln_2seg.fun)
print(f"Optimal Solution for 2 Segment Finite Mixture Model:")
print(f"pie1 = {round(pie_2seg,4)}")
print(f"pie2 = {round(1-pie_2seg,4)}")
print(f"Lambda1 = {round(lambda1_2seg,4)}")
print(f"Lambda2 = {round(lambda2_2seg,4)}")
print(f"Log likelihood = {round(ll_2seg,2)}")
```

```
Optimal Solution for 2 Segment Finite Mixture Model:
pie1 = 0.2991
pie2 = 0.7009
Lambda1 = 9.1207
Lambda2 = 1.8022
Log likelihood = -1188.83
```

(ii) 3 Segment Finite Mixture Model

In [7]:
```
# Define the Log Likelihood Function needed
def LL3seg(params,theta3,p,k):
    theta1,theta2,lambda1,lambda2,lambda3 = params
    ll = []
    for i in range(len(p)):
        ll = np.append(ll,p[i]*(np.log((((e**(theta1))/((e**(theta1))+(e**(theta2))+(e**(theta3))))*(poisson.pmf(k[i],lambda1)))-
    return ll

# Negative of log likelihood for minimization function
def NLL3seg(params,theta3,p,k):
    return (-1)*(np.sum(LL3seg(params,theta3,p,k)))

# Defining p = 'No of people' and k = 'No of Packs'
p = np.array(df['People'])
k = np.array(df['Packs'])
theta3 = 0

# Maximising the log likelihood
soln_3seg = minimize(NLL3seg,
                     args = (theta3,p,k),
                     x0 = np.array((1,2,1,1,1)),
                     bounds = [(None,None),(None,None),(0.000001,None),(0.000001,None),(0.000001,None)],
                     tol=1e-10,
                     options={'ftol' : 1e-8},)

# Get the optimal solution
pie1_3seg = e**(soln_3seg.x[0])/(e**(soln_3seg.x[0])+e**(soln_3seg.x[1])+1)
pie2_3seg = e**(soln_3seg.x[1])/(e**(soln_3seg.x[0])+e**(soln_3seg.x[1])+1)
lambda1_3seg = soln_3seg.x[2]
lambda2_3seg = soln_3seg.x[3]
lambda3_3seg = soln_3seg.x[4]
ll_3seg = (-1)*(soln_3seg.fun)
print(f"Optimal Solution for 3 Segment Finite Mixture Model:")
print(f"pie1 = {round(pie1_3seg,4)}")
print(f"pie2 = {round(pie2_3seg,4)}")
print(f"pie3 = {round(1-pie1_3seg-pie2_3seg,4)}")
print(f"Lambda1 = {round(lambda1_3seg,4)}")
print(f"Lambda2 = {round(lambda2_3seg,4)}")
print(f"Lambda3 = {round(lambda3_3seg,4)}")
print(f"Log likelihood = {round(ll_3seg,2)}")
```

```
Optimal Solution for 3 Segment Finite Mixture Model:
pie1 = 0.5433
pie2 = 0.18
pie3 = 0.2768
Lambda1 = 3.4833
Lambda2 = 11.2158
Lambda3 = 0.2906
Log likelihood = -1132.04
```

(iii) 4 Segment Finite Mixture Model

In [8]:
```
# Define the Log Likelihood Function needed
def LL4seg(params,theta4,p,k):
    theta1,theta2,theta3,lambda1,lambda2,lambda3,lambda4 = params
    ll = []
    for i in range(len(p)):
        ll = np.append(ll,p[i]*(np.log((((e**(theta1))/((e**(theta1))+(e**(theta2))+(e**(theta3))+(e**(theta4))))*(poisson.pmf(k
    return ll

# Negative of Log Likelihood for minimization function
def NLL4seg(params,theta4,p,k):
    return (-1)*(np.sum(LL4seg(params,theta4,p,k)))

# Defining p = 'No of people' and k = 'No of Packs'
p = np.array(df['People'])
k = np.array(df['Packs'])
theta4 = 0

# Maximising the Log Likelihood
soln_4seg = minimize(NLL4seg,
                     args = (theta4,p,k),
                     x0 = np.array((1,2,3,1,1,1,1)),
                     bounds = [(None,None),(None,None),(None,None),(0.000001,None),(0.000001,None),(0.000001,None),(0.000001,None
                     tol=1e-10,
                     options={'ftol' : 1e-8},)

# Get the optimal solution
pie1_4seg = e**(soln_4seg.x[0])/(e**(soln_4seg.x[0])+e**(soln_4seg.x[1])+e**(soln_4seg.x[2])+1)
pie2_4seg = e**(soln_4seg.x[1])/(e**(soln_4seg.x[0])+e**(soln_4seg.x[1])+e**(soln_4seg.x[2])+1)
pie3_4seg = e**(soln_4seg.x[2])/(e**(soln_4seg.x[0])+e**(soln_4seg.x[1])+e**(soln_4seg.x[2])+1)
lambda1_4seg = soln_4seg.x[3]
lambda2_4seg = soln_4seg.x[4]
lambda3_4seg = soln_4seg.x[5]
lambda4_4seg = soln_4seg.x[6]
ll_4seg = (-1)*(soln_4seg.fun)
print(f"Optimal Solution for 4 Segment Finite Mixture Model:")
print(f"pie1 = {round(pie1_4seg,4)}")
print(f"pie2 = {round(pie2_4seg,4)}")
print(f"pie3 = {round(pie3_4seg,4)}")
print(f"pie4 = {round(1-pie1_4seg-pie2_4seg-pie3_4seg,4)}")
print(f"Lambda1 = {round(lambda1_4seg,4)}")
print(f"Lambda2 = {round(lambda2_4seg,4)}")
print(f"Lambda3 = {round(lambda3_4seg,4)}")
print(f"Lambda4 = {round(lambda4_4seg,4)}")
print(f"Log likelihood = {round(ll_4seg,2)}")
```

```
Optimal Solution for 4 Segment Finite Mixture Model:
pie1 = 0.1514
pie2 = 0.2442
pie3 = 0.1017
pie4 = 0.5027
Lambda1 = 7.4187
Lambda2 = 0.2048
Lambda3 = 12.8731
Lambda4 = 3.0021
Log likelihood = -1130.07
```

Evaluate the models developed; explain which of them is best, and why. Are there any significant differences among the results from these models? If so, what exactly are these differences? Discuss what you believe could be causing the differences

To Evaluate the models developed so far, we used the AIC and BIC to measure the models. The best model is selected based on the lowest AIC and BIC values.

AIC:

AIC: 2k-2*LL where K: number of parameters (Varied by model) and LL: Log likelihood

In [10]:
```
# AIC: 2k-2*LL
AIC_p = (2*1)-(2*ll_p)
AIC_nbd = (2*2)-(2*ll_nbd)
AIC_zinbd = (2*3)-(2*ll_zinbd)
AIC_2seg = (2*3)-(2*ll_2seg)
AIC_3seg = (2*5)-(2*ll_3seg)
AIC_4seg = (2*7)-(2*ll_4seg)

# Printing the AIC values for all the models
print(f"AIC for Poisson model is {round(AIC_p,2)}")
print(f"AIC for NBD model is {round(AIC_nbd,2)}")
print(f"AIC for Zero-Inflated NBD model is {round(AIC_zinbd,2)}")
print(f"AIC for 2 Segment Finite Mixture model is {round(AIC_2seg,2)}")
print(f"AIC for 3 Segment Finite Mixture model is {round(AIC_3seg,2)}")
print(f"AIC for 4 Segment Finite Mixture model is {round(AIC_4seg,2)}")
```

```
AIC for Poisson model is 3091.99
AIC for NBD model is 2284.05
AIC for Zero-Inflated NBD model is 2278.33
AIC for 2 Segment Finite Mixture model is 2383.67
AIC for 3 Segment Finite Mixture model is 2274.09
AIC for 4 Segment Finite Mixture model is 2274.14
```

BIC:

BIC: k*ln(n)-2*LL where K: number of parameters (Varied by model)

n: number of records

LL: Log likelihood

```
In [9]: # BIC: k*ln(n)-2*LL
BIC_p = (1*np.log(np.sum(df['People'])))-(2*ll_p)
BIC_nbd = (2*np.log(np.sum(df['People'])))-(2*ll_nbd)
BIC_zinbd = (3*np.log(np.sum(df['People'])))-(2*ll_zinbd)
BIC_2seg = (3*np.log(np.sum(df['People'])))-(2*ll_2seg)
BIC_3seg = (5*np.log(np.sum(df['People'])))-(2*ll_3seg)
BIC_4seg = (7*np.log(np.sum(df['People'])))-(2*ll_4seg)


# Printing the BIC values for all the models
print(f"BIC for Poisson model is {round(BIC_p,2)}")
print(f"BIC for NBD model is {round(BIC_nbd,2)}")
print(f"BIC for Zero-Inflated NBD model is {round(BIC_zinbd,2)}")
print(f"BIC for 2 Segment Finite Mixture model is {round(BIC_2seg,2)}")
print(f"BIC for 3 Segment Finite Mixture model is {round(BIC_3seg,2)}")
print(f"BIC for 4 Segment Finite Mixture model is {round(BIC_4seg,2)}")

BIC for Poisson model is 3096.12
BIC for NBD model is 2292.29
BIC for Zero-Inflated NBD model is 2290.7
BIC for 2 Segment Finite Mixture model is 2396.03
BIC for 3 Segment Finite Mixture model is 2294.7
BIC for 4 Segment Finite Mixture model is 2303.0
```

Model	AIC	BIC
Poisson	3091.99	3096.12
NBD	2284.05	2292.29
Zero-Inflated NBD	2278.33	**2290.7**
2-Segment	2383.67	2396.03
3-segment	**2274.09**	2294.7
4-segment	2274.14	2303.0

Based on the above calculations of AIC and BIC, the best model according to the lowest **AIC** is **3-segment finite mixture model** and with respect to lowest **BIC** it is the **Zero-inflated NBD Model**. The BIC imposes a heavier penalty for increasing the number of parameters used for model estimation and it prefers the model with a lesser number of parameters. The Zero-inflated NBD model accounts for the inflation caused by a greater number of zeroes present in the data and is hence the better model in this context. The 3-segment mixture model also fits the data well but is using more number of parameters in doing so. Hence, the Zero-inflated NBD model is the preferred one in this case.

3. Based on the 2, 3, and 4-segment finite mixture models, how many packs are the following customers likely to purchase over the next 8 weeks?

(a) a customer who purchased 5 packs in the past week:

- ***2-Segment Finite mixture model:***

(i) 2 Segment Finite Mixture model

```
In [11]: P2_seg1 = ((pie_2seg)*(poisson.pmf(5,lambda1_2seg)))/(((pie_2seg)*(poisson.pmf(5,lambda1_2seg)))+((1-pie_2seg)*(poisson.pmf(5,lar
         P2_seg2 = ((1-pie_2seg)*(poisson.pmf(5,lambda2_2seg)))/(((pie_2seg)*(poisson.pmf(5,lambda1_2seg)))+((1-pie_2seg)*(poisson.pmf(5,l

         Exp_2seg = 8*((lambda1_2seg*P2_seg1)+(lambda2_2seg*P2_seg2))

         print(f"For the 2 Segment Finite Mixture model:")
         print(f"Probability of customer being in segment 1 is {round(P2_seg1,4)}")
         print(f"Probability of customer being in segment 2 is {round(P2_seg2,4)}")
         print(f"Expected number of purchases in 8 weeks is {round(Exp_2seg,2)}")

         For the 2 Segment Finite Mixture model:
         Probability of customer being in segment 1 is 0.4845
         Probability of customer being in segment 2 is 0.5155
         Expected number of purchases in 8 weeks is 42.78
```

```
In [12]: # If classifying customers into segment 2
         Exp_2seg_ = 8*(lambda2_2seg)

         print(f"If classifying customers into segment 2:")
         print(f"Expected number of purchases in 8 weeks is {round(Exp_2seg_,2)}")

         If classifying customers into segment 2:
         Expected number of purchases in 8 weeks is 14.42
```

- ***3-Segment Finite mixture model:***

(ii) 3 Segment Finite Mixture model

```
In [13]: P3_seg1 = ((pie1_3seg)*(poisson.pmf(5,lambda1_3seg)))/(((pie1_3seg)*(poisson.pmf(5,lambda1_3seg)))+((pie2_3seg)*(poisson.pmf(5,la
         P3_seg2 = ((pie2_3seg)*(poisson.pmf(5,lambda2_3seg)))/(((pie1_3seg)*(poisson.pmf(5,lambda1_3seg)))+((pie2_3seg)*(poisson.pmf(5,la
         P3_seg3 = ((1-pie1_3seg-pie2_3seg)*(poisson.pmf(5,lambda3_3seg)))/(((pie1_3seg)*(poisson.pmf(5,lambda1_3seg)))+((pie2_3seg)*(pois

         Exp_3seg = 8*((lambda1_3seg*P3_seg1)+(lambda2_3seg*P3_seg2)+(lambda3_3seg*P3_seg3))

         print(f"For the 3 Segment Finite Mixture model:")
         print(f"Probability of customer being in segment 1 is {round(P3_seg1,4)}")
         print(f"Probability of customer being in segment 2 is {round(P3_seg2,4)}")
         print(f"Probability of customer being in segment 3 is {round(P3_seg3,4)}")
         print(f"Expected number of purchases in 8 weeks is {round(Exp_3seg,2)}")

         For the 3 Segment Finite Mixture model:
         Probability of customer being in segment 1 is 0.9521
         Probability of customer being in segment 2 is 0.0478
         Probability of customer being in segment 3 is 0.0
         Expected number of purchases in 8 weeks is 30.83
```

```
In [14]: # If classifying customers into segment 1
         Exp_3seg_ = 8*(lambda1_3seg)

         print(f"If classifying customers into segment 1:")
         print(f"Expected number of purchases in 8 weeks is {round(Exp_3seg_,2)}")

         If classifying customers into segment 1:
         Expected number of purchases in 8 weeks is 27.87
```

- ***4- Segment Finite mixture model:***

(iii) 4 Segment Finite Mixture model

```
In [15]: P4_seg1 = ((pie1_4seg)*(poisson.pmf(5,lambda1_4seg)))/(((pie1_4seg)*(poisson.pmf(5,lambda1_4seg)))+((pie2_4seg)*(poisson.pmf(5,la
         P4_seg2 = ((pie2_4seg)*(poisson.pmf(5,lambda2_4seg)))/(((pie1_4seg)*(poisson.pmf(5,lambda1_4seg)))+((pie2_4seg)*(poisson.pmf(5,la
         P4_seg3 = ((pie3_4seg)*(poisson.pmf(5,lambda3_4seg)))/(((pie1_4seg)*(poisson.pmf(5,lambda1_4seg)))+((pie2_4seg)*(poisson.pmf(5,la
         P4_seg4 = ((1-pie1_4seg-pie2_4seg-pie3_4seg)*(poisson.pmf(5,lambda4_4seg)))/(((pie1_4seg)*(poisson.pmf(5,lambda1_4seg)))+((pie2_4

         Exp_4seg = 8*((lambda1_4seg*P4_seg1)+(lambda2_4seg*P4_seg2)+(lambda3_4seg*P4_seg3)+(lambda4_4seg*P4_seg4))

         print(f"For the 4 Segment Finite Mixture model:")
         print(f"Probability of customer being in segment 1 is {round(P4_seg1,4)}")
         print(f"Probability of customer being in segment 2 is {round(P4_seg2,4)}")
         print(f"Probability of customer being in segment 3 is {round(P4_seg3,4)}")
         print(f"Probability of customer being in segment 4 is {round(P4_seg4,4)}")
         print(f"Expected number of purchases in 8 weeks is {round(Exp_4seg,2)}")
```

```
For the 4 Segment Finite Mixture model:
Probability of customer being in segment 1 is 0.2482
Probability of customer being in segment 2 is 0.0
Probability of customer being in segment 3 is 0.0112
Probability of customer being in segment 4 is 0.7406
Expected number of purchases in 8 weeks is 33.67
```

```
In [16]: # If classifying customers into segment 4
         Exp_4seg_ = 8*(lambda4_4seg)

         print(f"If classifying customers into segment 4:")
         print(f"Expected number of purchases in 8 weeks is {round(Exp_4seg_,2)}")
```

```
If classifying customers into segment 4:
Expected number of purchases in 8 weeks is 24.02
```

(b) a customer who purchased 9 packs in the past week:

- ***2-Segment Finite mixture model:***

(i) 2 Segment Finite Mixture model

```
In [17]: P2_seg1 = ((pie_2seg)*(poisson.pmf(9,lambda1_2seg)))/(((pie_2seg)*(poisson.pmf(9,lambda1_2seg)))+((1-pie_2seg)*(poisson.pmf(9,lam
         P2_seg2 = ((1-pie_2seg)*(poisson.pmf(9,lambda2_2seg)))/(((pie_2seg)*(poisson.pmf(9,lambda1_2seg)))+((1-pie_2seg)*(poisson.pmf(9,l

         Exp_2seg = 8*((lambda1_2seg*P2_seg1)+(lambda2_2seg*P2_seg2))

         print(f"For the 2 Segment Finite Mixture model:")
         print(f"Probability of customer being in segment 1 is {round(P2_seg1,4)}")
         print(f"Probability of customer being in segment 2 is {round(P2_seg2,4)}")
         print(f"Expected number of purchases in 8 weeks is {round(Exp_2seg,2)}")
```

```
For the 2 Segment Finite Mixture model:
Probability of customer being in segment 1 is 0.9984
Probability of customer being in segment 2 is 0.0016
Expected number of purchases in 8 weeks is 72.87
```

```
In [18]: # If classifying customers into segment 1
         Exp_2seg_ = 8*(lambda1_2seg)

         print(f"If classifying customers into segment 1:")
         print(f"Expected number of purchases in 8 weeks is {round(Exp_2seg_,2)}")
```

```
If classifying customers into segment 1:
Expected number of purchases in 8 weeks is 72.97
```

- ***3-Segment Finite mixture model:***

(ii) 3 Segment Finite Mixture model

In [19]:
```
P3_seg1 = ((pie1_3seg)*(poisson.pmf(9,lambda1_3seg)))/(((pie1_3seg)*(poisson.pmf(9,lambda1_3seg)))+((pie2_3seg)*(poisson.pmf(9,la
P3_seg2 = ((pie2_3seg)*(poisson.pmf(9,lambda2_3seg)))/(((pie1_3seg)*(poisson.pmf(9,lambda1_3seg)))+((pie2_3seg)*(poisson.pmf(9,la
P3_seg3 = ((1-pie1_3seg-pie2_3seg)*(poisson.pmf(9,lambda3_3seg)))/(((pie1_3seg)*(poisson.pmf(9,lambda1_3seg)))+((pie2_3seg)*(pois

Exp_3seg = 8*((lambda1_3seg*P3_seg1)+(lambda2_3seg*P3_seg2)+(lambda3_3seg*P3_seg3))

print(f"For the 3 Segment Finite Mixture model:")
print(f"Probability of customer being in segment 1 is {round(P3_seg1,4)}")
print(f"Probability of customer being in segment 2 is {round(P3_seg2,4)}")
print(f"Probability of customer being in segment 3 is {round(P3_seg3,4)}")
print(f"Expected number of purchases in 8 weeks is {round(Exp_3seg,2)}")
```

```
For the 3 Segment Finite Mixture model:
Probability of customer being in segment 1 is 0.1562
Probability of customer being in segment 2 is 0.8438
Probability of customer being in segment 3 is 0.0
Expected number of purchases in 8 weeks is 80.06
```

In [20]:
```
# If classifying customers into segment 2
Exp_3seg_ = 8*(lambda2_3seg)

print(f"If classifying customers into segment 2:")
print(f"Expected number of purchases in 8 weeks is {round(Exp_3seg_,2)}")
```

```
If classifying customers into segment 2:
Expected number of purchases in 8 weeks is 89.73
```

- ***4- Segment Finite mixture model:***

(iii) 4 Segment Finite Mixture model

In [21]:
```
P4_seg1 = ((pie1_4seg)*(poisson.pmf(9,lambda1_4seg)))/(((pie1_4seg)*(poisson.pmf(9,lambda1_4seg)))+((pie2_4seg)*(poisson.pmf(9,la
P4_seg2 = ((pie2_4seg)*(poisson.pmf(9,lambda2_4seg)))/(((pie1_4seg)*(poisson.pmf(9,lambda1_4seg)))+((pie2_4seg)*(poisson.pmf(9,la
P4_seg3 = ((pie3_4seg)*(poisson.pmf(9,lambda3_4seg)))/(((pie1_4seg)*(poisson.pmf(9,lambda1_4seg)))+((pie2_4seg)*(poisson.pmf(9,la
P4_seg4 = ((1-pie1_4seg-pie2_4seg-pie3_4seg)*(poisson.pmf(9,lambda4_4seg)))/(((pie1_4seg)*(poisson.pmf(9,lambda1_4seg)))+((pie2_4

Exp_4seg = 8*((lambda1_4seg*P4_seg1)+(lambda2_4seg*P4_seg2)+(lambda3_4seg*P4_seg3)+(lambda4_4seg*P4_seg4))

print(f"For the 4 Segment Finite Mixture model:")
print(f"Probability of customer being in segment 1 is {round(P4_seg1,4)}")
print(f"Probability of customer being in segment 2 is {round(P4_seg2,4)}")
print(f"Probability of customer being in segment 3 is {round(P4_seg3,4)}")
print(f"Probability of customer being in segment 4 is {round(P4_seg4,4)}")
print(f"Expected number of purchases in 8 weeks is {round(Exp_4seg,2)}")
```

```
For the 4 Segment Finite Mixture model:
Probability of customer being in segment 1 is 0.6712
Probability of customer being in segment 2 is 0.0
Probability of customer being in segment 3 is 0.2751
Probability of customer being in segment 4 is 0.0537
Expected number of purchases in 8 weeks is 69.45
```

In [22]:
```
# If classifying customers into segment 1
Exp_4seg_ = 8*(lambda1_4seg)

print(f"If classifying customers into segment 1:")
print(f"Expected number of purchases in 8 weeks is {round(Exp_4seg_,2)}")
```

```
If classifying customers into segment 1:
Expected number of purchases in 8 weeks is 59.35
```

Part II: Analysis of New Data

Articles.csv contains the number of publications by 915 doctoral candidates (articles), along with five predictors:

1. female: 1 if candidate was female, 0 otherwise
2. married: 1 if candidate was married, 0 otherwise
3. kids: number of children aged ≤ 5
4. prestige: prestige of the candidate's department (higher is better)
5. mentorpubs: number of publications by the candidate's mentor over the past 3 years

Your task is to predict the number of articles as a function of the five independent variables

1.) Estimate all relevant parameters for Poisson regression using MLE. Report your code, the estimated parameters and the maximum value of the log-likelihood. What are the managerial takeaways — which customer characteristics seem to be important?

```
os.chdir(r'D:\ACADS\SEMESTER I\BUAN 6383\PROJECTS\PROJECT 2')
df1 = pd.read_csv(r'articles.csv')
df1
```

	articles	female	married	kids	prestige	menpubs
0	0	0	1	0	2.52	7
1	0	1	0	0	2.05	6
2	0	1	0	0	3.75	6
3	0	0	1	1	1.18	3
4	0	1	0	0	3.75	26
...	...	...	...	...	...	...
910	11	0	1	2	2.86	7
911	12	0	1	1	4.29	35
912	12	0	1	1	1.86	5
913	16	0	1	0	1.74	21
914	19	0	1	0	1.86	42

915 rows × 6 columns

```
df1.isnull().sum()
```

```
articles    0
female      0
married     0
kids        0
prestige    0
menpubs     0
dtype: int64
```