PROJECT 2

BUAN 6383.002 MODELING FOR BUSINESS ANALYTICS

Group 03

Amna Malik
Ayasha Anupam
Bindu Musham
Yen-Ting Liu
Victor Nava Angel

Part I: Replicating Models from Class

1. Consider the hard candy example from class. The associated data is in the file candy.csv. Develop the following models discussed in class using maximum likelihood estimation (MLE). Report your code and all relevant details, including the estimated values of the parameters for each model and the corresponding log-likelihood values. Please add comments to your code to make it easy to understand.

(a) the Poisson model,

```
#POISSON MODEL

def LL(params, retained,k):
    prob = []
    ll = []
    lambda1 = params
    for i in range(len(k)):
        prob.append((pow(lambda1,k[i])*math.exp(-lambda1))/math.factorial(k[i]))
        ll.append(retained[i]*np.log(float(prob[i])))
    return ll
```

```
#Converting

def NLL(params,retained, k):
    return(-np.sum(LL(params,retained, k)))
```

```
#Declaring the parameters

k = candy['Packs']
retained = candy['People']
params = np.array((1))
```

```
#Declaring the minimizing function

soln = minimize(NLL,
                args = (retained, k),
                x0 = np.array((1)),
                bounds = [(0.000001, None)],
                tol = 1e-10,
                options = {'ftol':1e-8}
                )
```

```
#The value of the poisson parameters for the optimal solution

lambda1 = soln.x[0]

print(soln)
print('\n')
print('The optimal value of lambda :',lambda1)
print('The maximum value of Log-Likelihood:',-soln.fun)
AIC = (2*(len(candy.axes[1])-1)) - (2*(-soln.fun))
BIC = ((len(candy.axes[1])-1)*(np.log(len(candy)))) - (2*(-soln.fun))
print('AIC:', AIC)
print('BIC:', BIC)
```

```
      fun: 1544.996390448969
 hess_inv: <1x1 LbfgsInvHessProduct with dtype=float64>
      jac: array([4.54747354e-05])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 18
      nit: 8
     njev: 9
   status: 0
  success: True
        x: array([3.99122787])


The optimal value of lambda : 3.9912278692038283
The maximum value of Log-Likelihood: -1544.996390448969
AIC: 3091.992780897938
BIC: 3093.0373033356614
```

(b) the NBD model,

```
#NBD MODEL

def LL_nbd(params_nbd,retained_nbd,k):
    prob_nbd = []
    ll_nbd = []
    alpha, n = params_nbd
    for i in range(len(k)):
        if i==0:
            prob_nbd.append(pow((alpha/(alpha+1)),n))
        else:
            prob_nbd.append(prob_nbd[i-1]*(n+k[i]-1)/(k[i]*(alpha+1)))
        ll_nbd.append(retained_nbd[i]*np.log(float(prob_nbd[i])))
    return ll_nbd
```

```
#Converting

def NLL_nbd(params_nbd,retained_nbd,k):
    return(-np.sum(LL_nbd(params_nbd,retained_nbd,k)))
```

```
k = candy['Packs']
retained_nbd = candy['People']
params_nbd = np.array((1,1))
```

```
soln_nbd = minimize(NLL_nbd,
                args = (retained_nbd,k),
                x0 = np.array((1,1)),
                bounds = [(0.000001, None),(0.000001, None)],
                tol = 1e-10,
                options = {'ftol':1e-8}
                )
```

```
print(soln_nbd)
print('\n')
print('The optimal Log-Likelihood Value:', -soln_nbd.fun)
print('The optimal value of alpha:',soln_nbd.x[0])
print('The optimal value of n:',soln_nbd.x[1])
AIC = (2*(len(candy.axes[1])-1)) - (2*(-soln_nbd.fun))
BIC = ((len(candy.axes[1])-1)*(np.log(len(candy)))) - (2*(-soln_nbd.fun))
print('AIC:', AIC)
print('BIC:', BIC)
```

```
      fun: 1140.0237462468867
 hess_inv: <2x2 LbfgsInvHessProduct with dtype=float64>
      jac: array([ 0.0188038 , -0.00741238])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 51
      nit: 11
     njev: 17
   status: 0
  success: True
        x: array([0.24996035, 0.99763594])


The optimal Log-Likelihood Value: -1140.0237462468867
The optimal value of alpha: 0.24996034884773632
The optimal value of n: 0.9976359355786394
AIC: 2282.0474924937735
BIC: 2283.092014931497
```

Optimal values: n=0.998, alpha = 0.250, LL: -1140.024

(c) the Zero Inflated NBD model, and

(c) the Zero Inflated NBD model,

```
#Zero Inflated NBD Model

def LL_znbd(params_znbd,retained_znbd,k):
    nbd = []
    prob_znbd = []
    ll_znbd = []
    alpha, n, pi = params_znbd
    for i in range(len(k)):
        if i==0:
            nbd.append(pow((alpha/(alpha+1)),n))
            prob_znbd.append(pi+(1-pi)*nbd[i])
        else:
            nbd.append((n+k[i]-1)/(k[i]*(alpha+1))*nbd[i-1])
            prob_znbd.append((1-pi)*nbd[i])
        ll_znbd.append(retained_znbd[i]*np.log(float(prob_znbd[i]), where=0<prob_znbd[i]))
    return ll_znbd
```

```
#Converting

def NLL_znbd(params_znbd,retained_znbd,k):
    return(-np.sum(LL_znbd(params_znbd,retained_znbd,k)))
```

```
k = candy['Packs']
retained_znbd = candy['People']
params_znbd = np.array((1,1,1))
```

```
soln_znbd = minimize(NLL_znbd,
                args = (retained_znbd,k),
                x0 = np.array((1,1,1)),
                bounds = [(0.000001, None),(0.000001, None),(0.000001,0.999999)],
                tol = 1e-10,
                options = {'ftol':1e-8}
                )
```

```
print(soln_znbd)
print('\n')
print('The optimal Log-Likelihood Value:', -soln_znbd.fun)
print('The optimal value of alpha:',soln_znbd.x[0])
print('The optimal value of n:',soln_znbd.x[1])
print('The optimal value of pi:',soln_znbd.x[2])
```

```
      fun: 1136.1656408772878
 hess_inv: <3x3 LbfgsInvHessProduct with dtype=float64>
      jac: array([ 0.01025455, -0.00334239,  0.01291482])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 132
      nit: 26
     njev: 33
   status: 0
  success: True
        x: array([0.33418757, 1.50391062, 0.11310919])


The optimal Log-Likelihood Value: -1136.1656408772878
The optimal value of alpha: 0.3341875671769899
The optimal value of n: 1.5039106236302846
The optimal value of pi: 0.11310919456086455
```

Optimal values: n:1.50399220, alpha:0.334197, pi:0.0.113099, LL:-1136.1656

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

2 Segments

```
#Finite Mixture Model for 2 segments

def LL_fmm2(params_fmm2,retained_fmm2,k):
    seg1 = []
    seg2 = []
    prob_fmm2 = []
    ll_fmm2 = []
    lambda1, lambda2, pi = params_fmm2
    for i in range(len(k)):
        seg1.append((pow(lambda1,k[i])*math.exp(-lambda1))/math.factorial(k[i]))
        seg2.append((pow(lambda2,k[i])*math.exp(-lambda2))/math.factorial(k[i]))
        prob_fmm2.append(pi*seg1[i]+(1-pi)*seg2[i])
        ll_fmm2.append(retained_znbd[i]*np.log(float(prob_fmm2[i])))
    return ll_fmm2
```

```
#Converting

def NLL_fmm2(params_fmm2,retained_fmm2,k):
    return(-np.sum(LL_fmm2(params_fmm2,retained_fmm2,k)))
```

```
k = candy['Packs']
retained_fmm2 = candy['People']
params_fmm2 = np.array((1,1,1))
```

```
soln_fmm2 = minimize(NLL_fmm2,
                args = (retained_fmm2,k),
                x0 = np.array((1,1,1)),
                bounds = [(0.000001, None),(0.000001, None),(0.000001, 0.99999)],
                tol = 1e-10,
                options = {'ftol':1e-8}
                )
```

```
print(soln_fmm2)
print('\n')
print('The optimal Log-Likelihood Value:', -soln_fmm2.fun)
print('The optimal value of lambda1:',soln_fmm2.x[0])
print('The optimal value of lambda2:',soln_fmm2.x[1])
print('The optimal value of pi:',soln_fmm2.x[2])
```

```
      fun: 1188.83282722193
 hess_inv: <3x3 LbfgsInvHessProduct with dtype=float64>
      jac: array([0.00022737, 0.0013415 , 0.01305125])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 56
      nit: 13
     njev: 14
   status: 0
  success: True
        x: array([9.12067683, 1.80215459, 0.29912153])


The optimal Log-Likelihood Value: -1188.83282722193
The optimal value of lambda1: 9.120676826389044
The optimal value of lambda2: 1.8021545902197404
The optimal value of pi: 0.29912152799598396
```

Optimal values: lambda1 = 9.12067683, lambda2 = 1.80215459, pi = 0.29912153 LL: -1188.83282722193

3 Segments

```
#Finite Mixture Model for 3 segments

def LL_fmm3(params_fmm3,retained_fmm3,k):
    seg1 = []
    seg2 = []
    seg3 = []
    prob_fmm3 = []
    ll_fmm3 = []
    lambda1, lambda2, lambda3,theta1, theta2 = params_fmm3
    theta3 = 0
    t1 = math.exp(theta1)/(math.exp(theta1)+math.exp(theta2)+math.exp(theta3))
    t2 = math.exp(theta2)/(math.exp(theta1)+math.exp(theta2)+math.exp(theta3))
    t3 = math.exp(theta3)/(math.exp(theta1)+math.exp(theta2)+math.exp(theta3))
    for i in range(len(k)):
        seg1.append((pow(lambda1,k[i])*math.exp(-lambda1))/math.factorial(k[i]))
        seg2.append((pow(lambda2,k[i])*math.exp(-lambda2))/math.factorial(k[i]))
        seg3.append((pow(lambda3,k[i])*math.exp(-lambda3))/math.factorial(k[i]))
        prob_fmm3.append(np.dot([t1,t2,t3],[seg1[i],seg2[i],seg3[i]]))
        ll_fmm3.append(retained_fmm3[i]*np.log(float(prob_fmm3[i])))
    return ll_fmm3
```

```
#Converting
def NLL_fmm3(params_fmm3,retained_fmm3,k):
    return(-np.sum(LL_fmm3(params_fmm3,retained_fmm3,k)))
```

```
k = candy['Packs']
retained_fmm3 = candy['People']
params_fmm3 = np.array((1,1,1,2,1))
```

```
soln_fmm3 = minimize(NLL_fmm3,
               args = (retained_fmm3,k),
               x0 = np.array((1,1,1,2,1)),
               bounds = [(0.000001, None),(0.000001, None),(0.000001, None),(None,None),(None,None)],
               tol = 1e-10,
               options = {'ftol':1e-8}
               )
```

```
print(soln_fmm3)
print('\n')
print('The optimal Log-Likelihood Value:', -soln_fmm3.fun)
print('The optimal value of lambda1:',soln_fmm3.x[0])
print('The optimal value of lambda2:',soln_fmm3.x[1])
print('The optimal value of lambda3:',soln_fmm3.x[2])
print('The optimal value of theta1:',soln_fmm3.x[3])
print('The optimal value of theta2:',soln_fmm3.x[4])
print('The optimal value of theta3:', 0.000)
```

```
      fun: 1132.0429842770327
 hess_inv: <5x5 LbfgsInvHessProduct with dtype=float64>
      jac: array([9.09494627e-05, 9.09494707e-05, 2.16004992e-03, 8.18545232e-04,
       4.09272614e-04])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 156
      nit: 22
     njev: 26
   status: 0
  success: True
        x: array([11.21582588,  3.48332532,  0.29055433, -0.43040886,  0.6744319 ])


The optimal Log-Likelihood Value: -1132.0429842770327
The optimal value of lambda1: 11.215825878532442
The optimal value of lambda2: 3.4833253155799135
The optimal value of lambda3: 0.29055433476787973
The optimal value of theta1: -0.4304088578864968
The optimal value of theta2: 0.6744319047118253
```

4 Segments

```
#Finite Mixture Model for 4 segments

def LL_fmm4(params_fmm4,retained_fmm4,k):
    seg1 = []
    seg2 = []
    seg3 = []
    seg4 = []
    prob_fmm4 = []
    ll_fmm4 = []
    lambda1, lambda2, lambda3,lambda4, theta1, theta2, theta3 = params_fmm4
    theta4 = 0
    t1 = math.exp(theta1)/(math.exp(theta1)+math.exp(theta2)+math.exp(theta3)+math.exp(theta4))
    t2 = math.exp(theta2)/(math.exp(theta1)+math.exp(theta2)+math.exp(theta3)+math.exp(theta4))
    t3 = math.exp(theta3)/(math.exp(theta1)+math.exp(theta2)+math.exp(theta3)+math.exp(theta4))
    t4 = math.exp(theta4)/(math.exp(theta1)+math.exp(theta2)+math.exp(theta3)+math.exp(theta4))
    for i in range(len(k)):
        seg1.append((pow(lambda1,k[i])*math.exp(-lambda1))/math.factorial(k[i]))
        seg2.append((pow(lambda2,k[i])*math.exp(-lambda2))/math.factorial(k[i]))
        seg3.append((pow(lambda3,k[i])*math.exp(-lambda3))/math.factorial(k[i]))
        seg4.append((pow(lambda4,k[i])*math.exp(-lambda4))/math.factorial(k[i]))
        prob_fmm4.append(np.dot([t1,t2,t3,t4],[seg1[i],seg2[i],seg3[i],seg4[i]]))
        ll_fmm4.append(retained_fmm4[i]*np.log(float(prob_fmm4[i])))
    return ll_fmm4
```

```
#Converting

def NLL_fmm4(params_fmm4,retained_fmm4,k):
    return(-np.sum(LL_fmm4(params_fmm4,retained_fmm4,k)))
```

```
k = candy['Packs']
retained_fmm4 = candy['People']
params_fmm4 = np.array((1,1,1,1,2,2,2))
```

```
soln_fmm4 = minimize(NLL_fmm4,
               args = (retained_fmm4,k),
               x0 = np.array((1,1,1,1,1,2,3)),
               bounds = [(0.000001, None),(0.000001, None),(0.000001, None),(0.000001, None),
                         (None, None),(None, None),(None, None)],
               tol = 1e-10,
               options = {'ftol':1e-8}
               )
```

```
print(soln_fmm4)
print('\n')
print('The optimal Log-Likelihood Value:', -soln_fmm4.fun)
print('The optimal value of lambda1:',soln_fmm4.x[0])
print('The optimal value of lambda2:',soln_fmm4.x[1])
print('The optimal value of lambda3:',soln_fmm4.x[2])
print('The optimal value of lambda4:',soln_fmm4.x[3])
print('The optimal value of theta1:',soln_fmm4.x[4])
print('The optimal value of theta2:',soln_fmm4.x[5])
print('The optimal value of theta3:',soln_fmm4.x[6])
print('The optimal value of theta4:', 0.000)
```

```
      fun: 1130.0705911691716
 hess_inv: <7x7 LbfgsInvHessProduct with dtype=float64>
      jac: array([-0.00027285, -0.00020464, -0.00029559,  0.00186446,  0.00075033,
       -0.00022737, -0.00136424])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 616
      nit: 64
     njev: 77
   status: 0
  success: True
        x: array([ 7.41860518,  0.20475954, 12.87272436,  3.00208255, -1.20020759,
       -0.72203176, -1.59821858])


The optimal Log-Likelihood Value: -1130.0705911691716
The optimal value of lambda1: 7.4186051792521885
The optimal value of lambda2: 0.20475954329335422
The optimal value of lambda3: 12.87272436309969
The optimal value of lambda4: 3.0020825472310286
The optimal value of theta1: -1.2002075884580459
The optimal value of theta2: -0.7220317598893029
The optimal value of theta3: -1.5982185751226556
```

2. 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.

```
#Comparing Poisson Model and NBD Model

#Likelihood Ratio Test p-Value: 2(L1-L2)
LLR1 = 2*(-soln_nbd.fun - (-soln.fun))

#p-Value of Log-Likelihood Ratio Test
p_value1 = 1-stats.chi2.cdf(LLR1,1)

#AIC and BIC of Poisson and NBD model
AIC_poisson = (2*(len(candy.axes[1])-1)) - 2*(-soln.fun)
BIC_poisson = (len(candy.axes[1])-1)*np.log(sum(candy['People'])) - 2*(-soln.fun)

AIC_nbd = (2*(len(candy.axes[1]))) - 2*(-soln_nbd.fun)
BIC_nbd = (len(candy.axes[1]))*np.log(sum(candy['People'])) - 2*(-soln_nbd.fun)

print('Comparision Results of Poisson and NBD Models:')
print('1. Log-likelihood ratio: ',LLR1)
print('2. p-Value of the log-likelihood ratio test: ', p_value1)
print('3. AIC:')
print('   i) Poisson Model: ',AIC_poisson)
print('   ii) NBD Model: ',AIC_nbd)
print('4. BIC:')
print('   i) Poisson Model: ',BIC_poisson)
print('   ii) NBD Model: ',BIC_nbd)
```

```
Comparision Results of Poisson and NBD Models:
1. Log-likelihood ratio:  809.9452884054194
2. p-Value of the log-likelihood ratio test:  0.0
3. AIC:
   i) Poisson Model:  3091.992780897938
   ii) NBD Model:  2284.0474924925184
4. BIC:
   i) Poisson Model:  3096.115273707452
   ii) NBD Model:  2292.292478111547
```

1. p-Value < 0.001, so we can reject the null hypothesis that Poisson Model is as good as NBD Model.

2. AIC of NBD model < AIC of Poisson Model, and this shows that the NBD model is better than the Poisson Model as the model with small AIC is preferred.

3. BIC of NBD model < BIC of Poisson Model, and this shows that the NBD model is better than the Poisson Model as the model with small BIC is preferred.

```
##Comparing NBD Model and Zero Inflated NBD Model

#Likelihood Ratio Test p-Value: 2(L1-L2)
LLR2 = 2*(-soln_znbd.fun - (-soln_nbd.fun))

#p-Value of Log-Likelihood Ratio Test
p_value2 = 1-stats.chi2.cdf(LLR2,1)

#AIC and BIC of Poisson and NBD model
AIC_nbd = (2*(len(candy.axes[1]))) - 2*(-soln_nbd.fun)
BIC_nbd = (len(candy.axes[1]))*np.log(sum(candy['People'])) - 2*(-soln_nbd.fun)

AIC_znbd = (2*(len(candy.axes[1])+1)) - 2*(-soln_znbd.fun)
BIC_znbd = (len(candy.axes[1])+1)*np.log(sum(candy['People'])) - 2*(-soln_znbd.fun)

print('Comparision Results of NBD and Zero Inflated NBD Models:')
print('1. Log-likelihood ratio: ',LLR2)
print('2. p-Value of the log-likelihood ratio test: ', p_value2)
print('3. AIC:')
print('   i) NBD Model: ',AIC_nbd)
print('   ii) Zero Inflated NBD Model: ',AIC_znbd)
print('4. BIC:')
print('   i) NBD Model: ',BIC_nbd)
print('   ii) Zero Inflated NBD Model: ',BIC_znbd)
```

```
Comparision Results of NBD and Zero Inflated NBD Models:
1. Log-likelihood ratio:  7.71621031340419
2. p-Value of the log-likelihood ratio test:  0.005472715692658836
3. AIC:
   i) NBD Model:  2284.0474924925184
   ii) Zero Inflated NBD Model:  2278.331282179114
4. BIC:
   i) NBD Model:  2292.292478111547
   ii) Zero Inflated NBD Model:  2290.6987606076573
```

1. p-Value < 0.001, so we can reject the null hypothesis that NBD Model is as good as Zero Inflated NBD Model.

2. AIC of Zero Inflated NBD model < AIC of NBD Model, and this shows that the Zero Inflated NBD model is better than the NBD Model as the model with small AIC is preferred.

3. BIC of Zero Inflated NBD model < BIC of NBD Model, and this shows that the Zero Inflated NBD model is better than the NBD Model as the model with small BIC is preferred.

```
##Comparing Zero Inflated NBD Model and Finite Mixture Model for 2 segments.

#Likelihood Ratio Test p-Value: 2(L1-L2)
LLR3 = 2*(-soln_fmm2.fun - (-soln_znbd.fun))

#p-Value of Log-Likelihood Ratio Test
p_value3 = 1-stats.chi2.cdf(LLR3,1)

#AIC and BIC of Poisson and NBD model
AIC_znbd = (2*(len(candy.axes[1])+1)) - 2*(-soln_znbd.fun)
BIC_znbd = (len(candy.axes[1])+1)*np.log(sum(candy['People'])) - 2*(-soln_znbd.fun)

AIC_fmm2 = (2*(len(candy.axes[1])+1)) - 2*(-soln_fmm2.fun)
BIC_fmm2 = ((len(candy.axes[1])+1)*np.log(sum(candy['People']))) - 2*(-soln_fmm2.fun)

print('Comparision Results of Zero Inflated NBD Model and Finite Mixture Model for 2 segments:')
print('1. Log-likelihood ratio: ',LLR3)
print('2. p-Value of the log-likelihood ratio test: ', p_value3)
print('3. AIC:')
print('   i) Zero Inflated NBD Model: ',AIC_znbd)
print('   ii) Finite Mixture Model for 2 segments: ',AIC_fmm2)
print('4. BIC:')
print('   i) Zero Inflated NBD Model: ',BIC_znbd)
print('   ii) Finite Mixture Model for 2 segments: ',BIC_fmm2)
```

```
Comparision Results of Zero Inflated NBD Model and Finite Mixture Model for 2 segments:
1. Log-likelihood ratio:  -105.33437226474598
2. p-Value of the log-likelihood ratio test:  1.0
3. AIC:
   i) Zero Inflated NBD Model:  2278.331282179114
   ii) Finite Mixture Model for 2 segments:  2383.66565444386
4. BIC:
   i) Zero Inflated NBD Model:  2290.6987606076573
   ii) Finite Mixture Model for 2 segments:  2396.0331328724033
```

1. p-Value > 0.001, so we can't reject the null hypothesis that Finite Mixture Model for 2 segment is as good as Zero Inflated NBD Model.

2. AIC of Zero Inflated NBD model < AIC of Finite Mixture Model for 2 segment, and this shows that the Zero Inflated NBD model is better than the Finite Mixture Model for 2 segment as the model with small AIC is preferred.

3. BIC of Zero Inflated NBD model < BIC of Finite Mixture Model for 2 segment, and this shows that the Zero Inflated NBD model is better than the Finite Mixture Model for 2 segment as the model with small BIC is preferred.

```
##Comparing Zero Inflated NBD Model and Finite Mixture Model for 3 segments.

#Likelihood Ratio Test p-Value: 2(L1-L2)
LLR4 = 2*(-soln_fmm3.fun - (-soln_znbd.fun))

#p-Value of Log-Likelihood Ratio Test
p_value4 = 1-stats.chi2.cdf(LLR4,1)

#AIC and BIC of Poisson and NBD model
AIC_znbd = (2*(len(candy.axes[1])+1)) - 2*(-soln_znbd.fun)
BIC_znbd = (len(candy.axes[1])+1)*np.log(sum(candy['People'])) - 2*(-soln_znbd.fun)

AIC_fmm3 = (2*(len(candy.axes[1])+3)) - 2*(-soln_fmm3.fun)
BIC_fmm3 = (len(candy.axes[1])+3)*np.log(sum(candy['People'])) - 2*(-soln_fmm3.fun)

print('Comparision Results of Zero Inflated NBD Model and Finite Mixture Model for 3 segments:')
print('1. Log-likelihood ratio: ',LLR4)
print('2. p-Value of the log-likelihood ratio test: ', p_value4)
print('3. AIC:')
print('   i) Zero Inflated NBD Model: ',AIC_znbd)
print('   ii) Finite Mixture Model for 3 segments: ',AIC_fmm3)
print('4. BIC:')
print('   i) Zero Inflated NBD Model: ',BIC_znbd)
print('   ii) Finite Mixture Model for 3 segments: ',BIC_fmm3)
```

```
Comparision Results of Zero Inflated NBD Model and Finite Mixture Model for 3 segments:
1. Log-likelihood ratio:  8.245313625048766
2. p-Value of the log-likelihood ratio test:  0.00408573488472086
3. AIC:
   i) Zero Inflated NBD Model:  2278.331282179114
   ii) Finite Mixture Model for 3 segments:  2274.0859685540654
4. BIC:
   i) Zero Inflated NBD Model:  2290.6987606076573
   ii) Finite Mixture Model for 3 segments:  2294.6984326016373
```

1. p-Value < 0.05, so we can reject the null hypothesis that Zero Inflated NBD Model is as good as Finite Mixture Model for 3 segment.

2. AIC of Zero Inflated NBD model > AIC of Finite Mixture Model for 3 segment, and this shows that the Finite Mixture Model for 3 segment is better than the Zero Inflated NBD model as the model with small AIC is preferred.

3. BIC of Zero Inflated NBD model > BIC of Finite Mixture Model for 3 segment, and this shows that the Finite Mixture Model for 3 segment is better than the Zero Inflated NBD model as the model with small BIC is preferred.

```
##Comparing Zero Inflated NBD Model and Finite Mixture Model for 4 segments.

#Likelihood Ratio Test p-Value: 2(L1-L2)
LLR5 = 2*(-soln_fmm4.fun - (-soln_fmm3.fun))

#p-Value of Log-Likelihood Ratio Test
p_value5 = 1-stats.chi2.cdf(LLR5,1)

#AIC and BIC of Poisson and NBD model
AIC_fmm3 = (2*(len(candy.axes[1])+3)) - 2*(-soln_fmm3.fun)
BIC_fmm3 = (len(candy.axes[1])+3)*np.log(sum(candy['People'])) - 2*(-soln_fmm3.fun)

AIC_fmm4 = (2*(len(candy.axes[1])+5)) - 2*(-soln_fmm4.fun)
BIC_fmm4 = ((len(candy.axes[1])+5)*np.log(sum(candy['People']))) - 2*(-soln_fmm4.fun)

print('Comparision Results of Zero Inflated NBD Model and Finite Mixture Model for 3 segments:')
print('1. Log-likelihood ratio: ',LLR5)
print('2. p-Value of the log-likelihood ratio test: ', p_value5)
print('3. AIC:')
print('   i) Finite Mixture Model for 3 segments: ',AIC_fmm3)
print('   ii) Finite Mixture Model for 4 segments: ',AIC_fmm4)
print('4. BIC:')
print('   i) Finite Mixture Model for 3 segments: ',BIC_fmm3)
print('   ii) Finite Mixture Model for 4 segments: ',BIC_fmm4)
```

```
Comparision Results of Zero Inflated NBD Model and Finite Mixture Model for 3 segments:
1. Log-likelihood ratio:  3.9447862157221607
2. p-Value of the log-likelihood ratio test:  0.04701682748121083
3. AIC:
   i) Finite Mixture Model for 3 segments:  2274.0859685540654
   ii) Finite Mixture Model for 4 segments:  2274.1411823383432
4. BIC:
   i) Finite Mixture Model for 3 segments:  2294.6984326016373
   ii) Finite Mixture Model for 4 segments:  2302.998632004944
```

1. p-Value < 0.05, so we can reject the null hypothesis that the Finite Mixture Model for 4 segment is as good as Finite Mixture Model for 3 segment.

2. AIC of Finite Mixture Model for 3 segment < AIC of Finite Mixture Model for 4 segment, and this shows that the Finite Mixture Model for 3 segment is better than the Finite Mixture Model for 4 segment as the model with small AIC is preferred.

3. BIC of Finite Mixture Model for 3 segment < BIC of Finite Mixture Model for 4 segment, and this shows that the Finite Mixture Model for 3 segment is better than the Finite Mixture Model for 4 segment as the model with small BIC is preferred.

Poisson model did not capture the zero packs purchased by people in the model. Furthermore, it did not account for the heterogeneity in the lambda. It is assuming that the rate parameter lambda is identical for everyone. To account that population is heterogeneous, NBD model is applied. The exposure rate lambda now has a gamma distribution which adds another parameter (n = shape parameter, alpha = scale parameter) to capture the unobserved heterogeneity. This results in a better model then before but still needs improvement to account for the zero purchases by the population.

To observe the zero's in the data, zero inflated model is applied. It views zero's coming from 2 sources. From a fraction π who will never purchase the candy and from a fraction (1 – π) who are likely to buy the candy but have not done so yet. AIC, BIC has lowered, and it gives a better Loglikelihood value then the previous models. It shows improvement then the previous models. To improve it further finite mixture models are applied with 2, 3 and 4 segments. Relaxing the assumption that lambda has a gamma distribution, instead discrete segments are assumed each with lamda. From among the segments, 3 gives the lowest AIC and BIC with 5 parameters. Finite mixture models are known for capturing the heterogeneity in population and by generalizing the distributional assumptions. It was able to do that on the candy dataset, by clustering it into segments and relaxing the assumption of lambda being a gamma distribution. In addition, with the parameter π a mixing proportion (re-parameterized for computational ease) and assuming λ for each segment Finite Mixture Model (3 segments) was able to make predictions better than all other models.

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, and

P(s = 1 | x = 5) =

```
#p(s = 1| x = 5) =
ss1 = (0.50272 *0.10095)/ ((0.50272 * 0.10095) + (0.24419 * 0) + (0.15139 * 0.11237) + (0.10170 * 0.00756))
ss1
#p (s = 2|x = 5) =
ss2 = (0.24419 *0)/ ((0.50272 * 0.10095) + (0.24419 * 0) + (0.15139 * 0.11237) + (0.10170 * 0.00756))
ss2
#p(s = 3| x = 5) =
ss3 = (0.15139 * 0.11237)/ ((0.50272 * 0.10095) + (0.24419 * 0) + (0.15139 * 0.11237) + (0.10170 * 0.00756))
ss3
#p (s = 4|x = 5) =
ss4 = (0.10170 * 0.00756)/ ((0.50272 * 0.10095) + (0.24419 * 0) + (0.15139 * 0.11237) + (0.10170 * 0.00756))
ss4
print('s =1 |x = 5:', ss1, 's =2 |x = 5:', ss2, 's =3 |x = 5:', ss3,'s =4 |x = 5:', ss4)
```

```
s =1 |x = 5: 0.7405441048752829 s =2 |x = 5: 0.0 s =3 |x = 5: 0.24823671318774657 s =4 |x = 5: 0.011219181936970577
```

```
#Expected purchases in 1 week

Pur1 = (3.002 * 0.740) + (0.205 * 0) + (7.418 * 0.248) + (12.872 * 0.0112)
print('Expected purchases in 1 week:',Pur1)

#Expected purchases in 5 weeks

Pur8 = 8 * Pur1
print('Expected purchases in 8 weeks:',Pur8)
```

```
Expected purchases in 1 week: 4.205310399999999
Expected purchases in 8 weeks: 33.642483199999994
```

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

```
#p(s = 1| x = 9)=
s1 = (0.50272 *0.00271)/ ((0.50272 * 0.00271) + (0.24419 * 0) + (0.15139 * 0.11252) + (0.10170 * 0.06867))
s1

#p (s = 2|x = 9) =
s2 = (0.24419 *0)/ ((0.50272 * 0.00271) + (0.24419 * 0) + (0.15139 * 0.11252) + (0.10170 * 0.06867))
s2

#p(s = 3| x = 9) =
s3 = (0.15139 * 0.11237)/ ((0.50272 * 0.10095) + (0.24419 * 0) + (0.15139 * 0.11237) + (0.10170 * 0.00756))
s3

#p (s = 4|x = 9) =
s4 = (0.10170 * 0.00756)/ ((0.50272 * 0.10095) + (0.24419 * 0) + (0.15139 * 0.11237) + (0.10170 * 0.00756))
s4

print('s =1 |x = 9:', s1, 's =1 |x = 9:', s2, 's =1 |x = 9:', s3,'s =1 |x = 9:', s4)
```

```
s =1 |x = 9: 0.05367784331230815 s =1 |x = 9: 0.0 s =1 |x = 9: 0.24823671318774657 s =1 |x = 9: 0.011219181936970577
```

```
Partb = (3.002 * s1) + (0.205 * s2) + (7.418 * s3) + (12.872 * s4)
Partb
PartbPur8 = 8 * Partb
print('Expected purchases in 8 weeks:',PartbPur8)
```

```
Expected purchases in 8 weeks: 17.17579307154351
```

Part II: Analysis of New Data

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?

```
def LL_poi_reg(params,female, married, kids, prestige, menpubs, k):
    lambda0, beta1, beta2, beta3, beta4, beta5 = params
    lambdai = []
    prob_poi_reg = []
    ll_poi_reg = []
    for i in range(len(k)):
        lambdai.append(lambda0*(np.exp(np.dot([beta1,beta2,beta3,beta4,beta5],[female[i],married[i],kids[i],prestige[i],menpubs[i
        prob_poi_reg.append(((pow(lambdai[i],k[i])*np.exp(-lambdai[i]))/math.factorial(k[i])))
        ll_poi_reg.append(np.log(float(prob_poi_reg[i]), where=0<prob_poi_reg[i]))
    return ll_poi_reg
```

```
def NLL_poi_reg(params,female, married, kids, prestige, menpubs, k):
    return(-np.sum(LL_poi_reg(params,female, married, kids, prestige, menpubs, k)))
```

```
k = articles['articles']

female = articles['female']
married = articles['married']
kids = articles['kids']
prestige =articles['prestige']
menpubs = articles['menpubs']

params = np.array([1,0,0,0,0,0])
```

```
soln_poi_reg = minimize(NLL_poi_reg,
               args = (female, married, kids, prestige, menpubs, k),
               x0 = params,
               bounds = [(0.000001, None),(None,None),(None,None),(None,None),(None,None),(None,None)],
               tol = 1e-10,
               options = {'ftol':1e-8}
               )
```

```
print(soln_poi_reg)
print('\n')

print('The optimal Log-Likelihood Value:', -soln_poi_reg.fun)
print('The optimal value of lambda0:', soln_poi_reg.x[0])
print('Coefficients:')
coeff = pd.DataFrame()

coeff['Variables'] = articles.columns[1:]
coeff['Coefficients'] = soln_poi_reg.x[1:]
coeff
```

```
      fun: 1651.0563223252716
 hess_inv: <6x6 LbfgsInvHessProduct with dtype=float64>
      jac: array([ 0.02867182, -0.00964064,  0.0007276 , -0.02648903,  0.0620048 ,
       -0.28676368])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 371
      nit: 42
     njev: 53
   status: 0
  success: True
        x: array([ 1.35651257, -0.22470938,  0.1551679 , -0.18493418,  0.01278297,
        0.02554042])
```

```
The optimal Log-Likelihood Value: -1651.0563223252716
The optimal value of lambda0: 1.356512567384311
Coefficients:
```

	Variables	Coefficients
0	female	-0.224709
1	married	0.155168
2	kids	-0.184934
3	prestige	0.012783
4	menpubs	0.025540

Managerial Takeaway

Being a female has a higher impact on the number of publications by doctoral candidates, as being a female lowers the number of publications as compared to being a man. Having kids under the age of 5 also lowers the count of having number of publications. Being married shows a positive impact on number of publications followed by when mentors of those candidates also have publications and the prestige of the candidate's department. Managerially female's and candidates with kids should be provided resources to aid them so that they are able to focus more on research. It is recommended to provide programs to further the cause and presence of women in academia so that they are able to pursue their goals. Candidates with kids should be provided more sources and insurance so that they are able to engage more and keep on track with their research and projects. Programs like incentivized/subsidized child care, flex programs with added flexibility with timings. Mentors should also be made a focus by the departments. More engaged the mentor is more motivated will that person be to make other's pursue with their publications. Looking at the model more encouragement and focus on female candidates, candidates with kids and those who are married.

Interpretation of Coefficients

Female - If the doctoral candidate is a female, the difference in the logs of expected counts is expected to decrease by 0.22 times as compared to when the doctoral candidate is a male, given the other predictor variables in the model are held constant.

Married - If the doctoral candidate is married, the difference in the logs of expected counts is expected to increase by 0.155 times as compared to when the doctoral candidate is not married, given the other predictor variables in the model are held constant.

Kids – For a one unit change in variable kids(if doctoral candidates have kids under or equal to age 5), the difference in the logs of expected counts is expected to decrease by 0.18 times, given the other predictor variables in the model are held constant.

Prestige – For a one unit change in prestige of the candidate's department, the difference in the logs of expected counts is expected to increase by 0.012 times, given the other predictor variables in the model are held constant.

Mentorpubs – For a one unit change in number of publications by the candidate's mentor over the past 3 years, the difference in the logs of expected counts is expected to increase by 0.025 times, given the other predictor variables in the model are held constant.

These variables have more of an impact on the count of the number of books purchased then the variables region and household size as they are very low in magnitude.

Importance of variables:

female > kids > married > income > mentorpubs > prestige

2. Estimate all relevant parameters for NBD 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?

```
def LL_nbd_reg(params_nbd, female, married, kids, prestige, menpubs, k):
    exp_betas = []
    prob_nbd = []
    ll_nbd = []
    alpha, n, beta1, beta2, beta3, beta4, beta5 = params_nbd
    for i in range(len(k)):
        exp_betas.append(np.exp(np.dot([beta1,beta2,beta3,beta4,beta5],[female[i],married[i],kids[i],prestige[i],menpubs[i]])))
        p1_b = ((math.gamma(n+k[i])))/(math.gamma(n)*math.factorial(k[i]))
        p2_b = pow((alpha/(alpha+exp_betas[i])),n)
        p3_b = pow((exp_betas[i]/(alpha+exp_betas[i])),k[i])
        prob_nbd.append(p1_b*p2_b*p3_b)
        ll_nbd.append(np.log(float(prob_nbd[i])))
    return ll_nbd
```

```
#Converting

def NLL_nbd_reg(params_nbd, female, married, kids, prestige, menpubs, k):
    return(-np.sum(LL_nbd_reg(params_nbd,female, married, kids, prestige, menpubs, k)))
```

```
k = articles['articles']

female = articles['female']
married = articles['married']
kids = articles['kids']
prestige =articles['prestige']
menpubs = articles['menpubs']

params = np.array([1,1,0,0,0,0,0])
```

```
soln_nbd_reg = minimize(NLL_nbd_reg,
               args = (female, married, kids, prestige, menpubs, k),
               x0 = params,
               bounds = [(0.000001, None),(0.000001,None),(None,None),(None,None),(None,None),(None,None),(None,None)],
               tol = 1e-10,
               options = {'ftol':1e-8}
               )
```

```
print(soln_nbd_reg)
print('\n')

print('The optimal Log-Likelihood Value:', -soln_nbd_reg.fun)
print('The optimal value of alpha:', soln_nbd_reg.x[0])
print('The optimal value of n:', soln_nbd_reg.x[1])
print('Coefficients:')
coeff = pd.DataFrame()

coeff['Variables'] = articles.columns[1:]
coeff['Coefficients'] = soln_nbd_reg.x[2:]
coeff
```

```
      fun: 1560.9583480529882
 hess_inv: <7x7 LbfgsInvHessProduct with dtype=float64>
      jac: array([ 0.00654836, -0.00122782, -0.00416094, -0.03565219,  0.03467449,
       -0.08030838, -0.4871481 ])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 512
      nit: 54
     njev: 64
   status: 0
  success: True
        x: array([ 1.7524181 ,  2.2646854 , -0.2164132 ,  0.15021897, -0.17625251,
        0.01521719,  0.02907887])


The optimal Log-Likelihood Value: -1560.9583480529882
The optimal value of alpha: 1.7524181032416957
The optimal value of n: 2.2646853998933016
Coefficients:
```

	Variables	Coefficients
0	female	-0.216413
1	married	0.150219
2	kids	-0.176253
3	prestige	0.015217
4	menpubs	0.029079

Managerial Takeaway

Being a female has a higher impact on the number of publications by doctoral candidates, as being a female lowers the number of publications as compared to being a man. Having kids under the age of 5 also lowers the level of having number of publications. Being married shows a positive impact on number of publications followed by when mentors of those candidates also have publications and the prestige of the candidate's department. Managerially female's and candidates with kids should be provided resources to aid them so that they are able to focus more on research. Provide programs to further the cause and presence of women in academia so that they are able to pursue their goals. Candidates with kids should be provided more sources and insurance so that they are able to engage more and keep on track with their research and projects. Programs like incentivized/subsidized child care, flex programs with added flexibility with timings. Mentors should also be made a focus by the departments. More engaged the mentor is more motivated will that person be to make others pursue with their publications. This will ultimately higher the level of prestige of the respective department. Looking at the model more encouragement and focus on female candidates, candidates with kids and those who are married.

Interpretation of Coefficients

Female - If the doctoral candidate is a female, the difference in the logs of expected counts is expected to decrease by 0.21 times as compared to when the doctoral candidate is a male, given the other predictor variables in the model are held constant.

Married - If the doctoral candidate is married, the difference in the logs of expected counts is expected to increase by 0.15 times as compared to when the doctoral candidate is not married, given the other predictor variables in the model are held constant.

Kids – For a one unit change in variable kids(if doctoral candidates have kids under or equal to age 5), the difference in the logs of expected counts is expected to decrease by 0.17 times, given the other predictor variables in the model are held constant.

Prestige – For a one unit change in prestige of the candidate's department, the difference in the logs of expected counts is expected to increase by 0.015 times, given the other predictor variables in the model are held constant.

Mentorpubs – For a one unit change in number of publications by the candidate's mentor over the past 3 years, the difference in the logs of expected counts is expected to increase by 0.029 times, given the other predictor variables in the model are held constant.

These variables have more of an impact on the count of the number of books purchased then the variables region and household size as they are very low in magnitude.

Importance of variables:

female > kids > married > income > mentorpubs > prestige

3. In this question, you will apply the ideas learned in this course to build a model that you have not seen before – the Zero Inflated NBD Regression. First, recall that zero inflated models view 0s as coming from 2 sources - (i) from a fraction π who is 0 "by type" (in the context of this problem, these are candidates who will never publish), and (ii) from the remaining fraction (1 – π) who are likely to eventually become nonzero (these are candidates who will publish at some point, but have not done so yet). You can assume that the candidates in the latter group are distributed as a negative binomial (making the NBD regression appropriate for them). Explain the logic used in developing the model in detail. (hint: you do not need anything beyond what you have learned in the class to do this.)

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?

```
def LL_znbd_reg(params_znbd, female, married, kids, prestige, menpubs, k):
    exp_betas = []
    prob_znbd = []
    ll_znbd = []
    alpha, n, pi, beta1, beta2, beta3, beta4, beta5 = params_znbd
    for i in range(len(k)):
        exp_betas.append(np.exp(np.dot([beta1,beta2,beta3,beta4,beta5],[female[i],married[i],kids[i],prestige[i],menpubs[i]])))
        p1_b = ((math.gamma(n+k[i])))/(math.gamma(n)*math.factorial(k[i]))
        p2_b = pow((alpha/(alpha+exp_betas[i])),n)
        p3_b = pow((exp_betas[i]/(alpha+exp_betas[i])),k[i])
        if i==0:
            prob_znbd.append(pi+((1-pi)*(p1_b*p2_b*p3_b)))
        else:
            prob_znbd.append((1-pi)*(p1_b*p2_b*p3_b))
        ll_znbd.append(np.log(float(prob_znbd[i])))
    return ll_znbd
```

```
#Converting

def NLL_znbd_reg(params_znbd, female, married, kids, prestige, menpubs, k):
    return(-np.sum(LL_znbd_reg(params_znbd,female, married, kids, prestige, menpubs, k)))
```

```
k = articles['articles']

female = articles['female']
married = articles['married']
kids = articles['kids']
prestige =articles['prestige']
menpubs = articles['menpubs']

params = np.array([1,1,1,0,0,0,0,0])
```

```
soln_znbd_reg = minimize(NLL_znbd_reg,
               args = (female, married, kids, prestige, menpubs, k),
               x0 = params,
               bounds = [(0.000001, None),(0.000001,None),(0.000001,0.999999),(None,None),(None,None),(None,None),(None,None),(No
               tol = 1e-10,
               options = {'ftol':1e-8}
               )
```

```
print(soln_znbd_reg)
print('\n')

print('The optimal Log-Likelihood Value:', -soln_znbd_reg.fun)
print('The optimal value of alpha:', soln_znbd_reg.x[0])
print('The optimal value of n:', soln_znbd_reg.x[1])
print('Coefficients:')
coeff = pd.DataFrame()

coeff['Variables'] = articles.columns[1:]
coeff['Coefficients'] = soln_znbd_reg.x[3:]
coeff
```

```
      fun: 1560.959259792603
 hess_inv: <8x8 LbfgsInvHessProduct with dtype=float64>
      jac: array([-2.38742361e-03,  1.11413102e-02,  9.10999142e+02,  1.09139364e-03,
        1.68711267e-02,  5.90944183e-02,  3.95630195e-03, -6.60384103e-01])
  message: 'CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH'
     nfev: 810
      nit: 78
     njev: 90
   status: 0
  success: True
        x: array([ 1.75329008e+00,  2.26509723e+00,  1.00000000e-06, -2.16385434e-01,
        1.50481985e-01, -1.76277195e-01,  1.52848297e-02,  2.90733531e-02])


The optimal Log-Likelihood Value: -1560.959259792603
The optimal value of alpha: 1.7532900842597106
The optimal value of n: 2.265097226441735
Coefficients:
```

	Variables	Coefficients
0	female	-0.216385
1	married	0.150482
2	kids	-0.176277
3	prestige	0.015285
4	menpubs	0.029073

The logic behind this model is that to account for the zero's in the model, parameter pi was included which comes from 2 sources. Pi - those candidates who will never publish and (1 – pi) who are likely to eventually become nonzero (those candidates who will publish at some point but have not done so yet). To incorporate this parameter, it was multiplied with the probability of the negative binomial regression function. Further on to compute log-likelihood, this probability's log was taken and then sum to get the maximum log likelihood.

Managerial Takeaway

Being a female has a higher impact on the number of publications by doctoral candidates, as being a female lowers the number of publications as compared to being a man. Having kids under the age of 5 also lowers the level of having number of publications. Being married shows a positive impact on number of publications followed by when mentors of those candidates also have publications and the prestige of the candidate's department. Managerially female's and candidates with kids should be

provided resources to aid them so that they are able to focus more on research. Provide programs to further the cause and presence of women in academia so that they are able to pursue their goals. Candidates with kids should be provided more sources and insurance so that they are able to engage more and keep on track with their research and projects. Programs like incentivized/subsidized child care, flex programs with added flexibility with timings. Mentors should also be made a focus by the departments. More engaged the mentor is more motivated will that person be to make other's pursue with their publications. Looking at the model more encouragement and focus on female candidates, candidates with kids and those who are married.

Interpretation of Coefficients

Female - If the doctoral candidate is a female, the difference in the logs of expected counts is expected to decrease by 0.21 times as compared to when the doctoral candidate is a male, given the other predictor variables in the model are held constant.

Married - If the doctoral candidate is married, the difference in the logs of expected counts is expected to increase by 0.15 times as compared to when the doctoral candidate is not married, given the other predictor variables in the model are held constant.

Kids – For a one unit change in variable kids(if doctoral candidates have kids under or equal to age 5), the difference in the logs of expected counts is expected to decrease by 0.17 times, given the other predictor variables in the model are held constant.

Prestige – For a one unit change in prestige of the candidate's department, the difference in the logs of expected counts is expected to increase by 0.015 times, given the other predictor variables in the model are held constant.

Mentorpubs – For a one unit change in number of publications by the candidate's mentor over the past 3 years, the difference in the logs of expected counts is expected to increase by 0.02 times, given the other predictor variables in the model are held constant.

These variables have more of an impact on the count of the number of books purchased then the variables region and household size as they are very low in magnitude.

Importance of variables:

female > kids > married > income > mentorpubs > prestige

4. 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.

```
#Comparing Poisson Regression and NBD Regression

#Likelihood Ratio Test p-Value: 2(L1-L2)
LLR_reg1 = 2*(-soln_nbd_reg.fun - (-soln_poi_reg.fun))

#p-Value of Log-Likelihood Ratio Test
p_value_reg1 = 1-stats.chi2.cdf(LLR_reg1,1)

#AIC and BIC of Poisson and NBD model
AIC_poisson_reg = (2*(len(articles.axes[1]))) - 2*(-soln_poi_reg.fun)
BIC_poisson_reg = (len(articles.axes[1]))*np.log(len(articles)) - 2*(-soln_poi_reg.fun)

AIC_nbd_reg = (2*(len(articles.axes[1])+1)) - 2*(-soln_nbd_reg.fun)
BIC_nbd_reg = (len(articles.axes[1])+1)*np.log(len(articles)) - 2*(-soln_nbd_reg.fun)

print('Comparision Results of Poisson and NBD Regression:')
print('1. Log-likelihood ratio: ',LLR_reg1)
print('2. p-Value of the log-likelihood ratio test: ', p_value_reg1)
print('3. AIC:')
print('   i) Poisson Regression: ',AIC_poisson_reg)
print('   ii) NBD Regression: ',AIC_nbd_reg)
print('4. BIC:')
print('   i) Poisson Regression: ',BIC_poisson_reg)
print('   ii) NBD Regression: ',BIC_nbd_reg)
```

```
Comparision Results of Poisson and NBD Regression:
1. Log-likelihood ratio:  180.19594854456682
2. p-Value of the log-likelihood ratio test:  0.0
3. AIC:
   i) Poisson Regression:  3314.112644650543
   ii) NBD Regression:  3135.9166961059764
4. BIC:
   i) Poisson Regression:  3343.026189042196
   ii) NBD Regression:  3169.649164562905
```

1. p-Value < 0.001, so we can reject the null hypothesis that the Poisson Regression is as good as NBD Regression.

2. AIC of NBD Regression < AIC of Poisson Regression, and this shows that the NBD Regression is better than the Poisson Regression as the model with small AIC is preferred.

3. BIC of NBD Regression < BIC of Poisson Regression, and this shows that the NBD Regression is better than the Poisson Regression as the model with small BIC is preferred.

```
#Comparing NBD Regression and Zero-Inflated NBD Regression

#Likelihood Ratio Test p-Value: 2(L1-L2)
LLR_reg2 = 2*(-soln_znbd_reg.fun - (-soln_nbd_reg.fun))

#p-Value of Log-Likelihood Ratio Test
p_value_reg2 = 1-stats.chi2.cdf(LLR_reg2,1)

#AIC and BIC of Poisson and NBD model
AIC_nbd_reg = (2*(len(articles.axes[1])+1)) - 2*(-soln_nbd_reg.fun)
BIC_nbd_reg = (len(articles.axes[1])+1)*np.log(len(articles)) - 2*(-soln_nbd_reg.fun)

AIC_znbd_reg = (2*(len(articles.axes[1])+2)) - 2*(-soln_znbd_reg.fun)
BIC_znbd_reg = (len(articles.axes[1])+2)*np.log(len(articles)) - 2*(-soln_znbd_reg.fun)

print('Comparision Results of NBD and Zero Inflated NBD Regression:')
print('1. Log-likelihood ratio: ',LLR_reg2)
print('2. p-Value of the log-likelihood ratio test: ', p_value_reg2)
print('3. AIC:')
print('   i) NBD Regression: ',AIC_nbd_reg)
print('   ii) Zero Inflated NBD Regression: ',AIC_znbd_reg)
print('4. BIC:')
print('   i) NBD Regression: ',BIC_nbd_reg)
print('   ii) Zero Inflated NBD Regression: ',BIC_znbd_reg)
```

```
Comparision Results of NBD and Zero Inflated NBD Regression:
1. Log-likelihood ratio:  -0.001823479229642544
2. p-Value of the log-likelihood ratio test:  1.0
3. AIC:
   i) NBD Regression:  3135.9166961059764
   ii) Zero Inflated NBD Regression:  3137.918519585206
4. BIC:
   i) NBD Regression:  3169.649164562905
   ii) Zero Inflated NBD Regression:  3176.46991210741
```

1. p-Value > 0.001, so we can't reject the null hypothesis that the NBD Regression is as good as Zero Inflated NBD Regression.

2. AIC of NBD Regression < AIC of Zero Inflated NBD Regression, and this shows that the NBD Regression is better than the Zero Inflated NBD Regression as the model with small AIC is preferred.

3. BIC of NBD Regression < BIC of Zero Inflated NBD Regression, and this shows that the NBD Regression is better than the Zero Inflated NBD Regression as the model with small BIC is preferred.

Poisson regression is not capturing the underlying distribution of the number of publications by doctoral candidates. It is not adjusting the variance independently from the mean as Poisson distribution assumes that the mean and variance are the same. It is assuming that the rate parameter lambda is identical for everyone. When NBD Regression is applied it accounts for the heterogeneity in the lambda. The exposure rate lambda now has a gamma distribution which adds another parameter (n = shape parameter, alpha = scale parameter) to capture the heterogeneity (it adjusts the variance independently from the mean). This results in a better model then before but still ‘can’ need improvement to account for the zero’s in the data. When zero inflated NBD Regression is applied; It views zero’s coming from 2

sources. Pi - those candidates who will never publish and (1 – pi) who are likely to eventually become nonzero (those candidates who will publish at some point but have not done so yet).

For this case, log-likelihood for NBD Regression and Zero-inflated NBD Regression are pretty similar. There is a small difference between the values of AIC/BIC of the models. After looking at the p-value and the log-likelihood ratio, NBD regression seems to have performed better with a lower AIC and BIC. Even though zero inflated capture's the zeros in the model, it all depends on the kind of data at hand and here NBD regression performed better.

As with project 01, briefly summarize what you learned from project 02. Remember – this is an open-ended question, so please include anything you found worthwhile.

This project provided a very succinct view into the issues provided by the questions by applying the appropriate models and then comparing those models by their respective comparison methods and tools like log-likelihood ratio, AIC/BIC and log-likelihood values. It provides a clear window of what changes to make in the model, and what the model outputs as result. Model selection holds great importance as the appropriate selection of model for the given data at hand will reduce operational costs and produce results which would be more time efficient.