Assignment 8, Amogha Sekhar, A53301791

8.1a) Sanity Check

In [2]:
```
import numpy as np
import pprint
```

In [3]:
```
def readTitles():
    return [line.rstrip() for line in open("hw8_movies.txt").readlines()]

def readPIDs():
    return [line.rstrip() for line in open("hw8_ids.txt").readlines()]

def readRatings():
    return [[int(x) if x != '?' else -1 for x in line.rstrip().split(" ")]
            for line in open("hw8_ratings.txt").readlines()]

def orderByPopularity():
    meanratingofmovies = [[el for el in x if el != -1] for x in zip(*rating
    meanratingofmovies = [sum(x)/len(x) for x in meanratingofmovies]

    movieswithrating = [[movie, rating] for (movie, rating)
                        in zip(movies, meanratingofmovies)]
    movieswithrating.sort(key=lambda x: x[1])
    return movieswithrating

movies = readTitles()
students = readPIDs()
ratings = readRatings()
movieratings = orderByPopularity()
```

In [4]:
```
pprint.pprint(movieratings)
```

```
[['The_Last_Airbender', 0.2112676056338028],
 ['Fifty_Shades_of_Grey', 0.36],
 ['I_Feel_Pretty', 0.38095238095238093],
 ['Magic_Mike', 0.42857142857142855],
 ['Man_of_Steel', 0.5257731958762887],
 ['The_Shape_of_Water', 0.5616438356164384],
 ['World_War_Z', 0.5617977528089888],
 ['Hustlers', 0.5652173913043478],
 ['Prometheus', 0.5844155844155844],
 ['Fast_Five', 0.625],
 ['American_Hustle', 0.6274509803921569],
 ['Jurassic_World', 0.6287425149700598],
 ['Once_Upon_a_Time_in_Hollywood', 0.6290322580645161],
 ['Pitch_Perfect', 0.6428571428571429],
 ['Fast_&_Furious:_Hobbs_&_Shaw', 0.6551724137931034],
 ['Star_Wars:_The_Force_Awakens', 0.66],
 ['Pokemon_Detective_Pikachu', 0.6637168141592921],
 ['Phantom_Thread', 0.6666666666666666],
 ['The_Hunger_Games', 0.6717948717948717],
 ['Manchester_by_the_Sea', 0.6808510638297872],
 ['Avengers:_Age_of_Ultron', 0.6878612716763006],
 ['Rocketman', 0.6896551724137931],
 ['Mad_Max:_Fury_Road', 0.6944444444444444],
 ['Us', 0.6981132075471698],
 ['Bridemaids', 0.7],
 ['The_Farewell', 0.7],
 ['Chappaquidick', 0.7058823529411765],
 ['Good_Boys', 0.7142857142857143],
 ['Terminator:_Dark_Fate', 0.723404255319149],
 ['Thor', 0.7262569832402235],
 ['The_Perks_of_Being_a_Wallflower', 0.7313432835820896],
 ['The_Revenant', 0.7341772151898734],
 ['The_Hateful_Eight', 0.7377049180327869],
 ['The_Great_Gatsby', 0.7407407407407407],
 ['Dunkirk', 0.7452830188679245],
 ['Darkest_Hour', 0.7608695652173914],
 ['Toy_Story_3', 0.7631578947368421],
 ['Captain_America:_The_First_Avenger', 0.7653631284916201],
 ['Midnight_in_Paris', 0.7692307692307693],
 ['X-Men:_First_Class', 0.7733333333333333],
 ['Drive', 0.775],
 ['Frozen', 0.7758620689655172],
 ['La_La_Land', 0.7832167832167832],
 ['Her', 0.7906976744186046],
 ['Ex_Machina', 0.7906976744186046],
 ['Spiderman:_Far_From_Home', 0.7947019867549668],
 ['21_Jump_Street', 0.7954545454545454],
 ['Iron_Man_2', 0.8020304568527918],
 ['Harry_Potter_and_the_Deathly_Hallows:_Part_1', 0.8038277511961722],
 ['Room', 0.8103448275862069],
 ['Wolf_of_Wall_Street', 0.8136645962732919],
 ['Harry_Potter_and_the_Deathly_Hallows:_Part_2', 0.8186274509803921],
 ['Gone_Girl', 0.8207547169811321],
 ['The_Social_Network', 0.8244274809160306],
 ['The_Lion_King', 0.8313953488372093],
```

```
['Joker', 0.8333333333333334],
['Now_You_See_Me', 0.8376623376623377],
['The_Girls_with_the_Dragon_Tattoo', 0.8392857142857143],
['Parasite', 0.8448275862068966],
['Les_Miserables', 0.8470588235294118],
['Avengers:_Endgame', 0.851063829787234],
['Ready_Player_One', 0.8571428571428571],
['The_Avengers', 0.8599033816425121],
['12_Years_a_Slave', 0.8627450980392157],
['The_Help', 0.8666666666666667],
['Avengers:_Infinity_War', 0.8688524590163934],
['Hidden_Figures', 0.875],
['Shutter_Island', 0.8793103448275862],
['Black_Swan', 0.8910891089108911],
['The_Theory_of_Everything', 0.8962264150943396],
['The_Dark_Knight_Rises', 0.9010989010989011],
['The_Martian', 0.9014084507042254],
['Django_Unchained', 0.9042553191489362],
['Three_Billboards_Outside_Ebbing', 0.9076923076923077],
['Interstellar', 0.9346733668341709],
['Inception', 0.9858490566037735]]
```

8.1e) Implementation

In [6]:
```
def initz():
    return [float(line.rstrip()) for line
            in open("hw8_probZ_init.txt").readlines()]

def initrgivenz():
    return [[float(el) for el in line.rstrip().split()]
            for line in open("hw8_probR_init.txt").readlines()]

z = initz()
rgivenz = initrgivenz()

def Estep(z, rgivenz, ratings):
    rho = []*len(ratings)
    for t in range(len(ratings)):
        rho.append([1]*len(z))

    for t in range(len(ratings)):
        den = 0
        for i in range(len(z)):
            rho[t][i] *= z[i]
            for j in range(len(rgivenz)):
                if ratings[t][j] == 1:
                    rho[t][i] *= rgivenz[j][i]
                elif ratings[t][j] == 0:
                    rho[t][i] *= (1 - rgivenz[j][i])

            den += rho[t][i]

        for i in range(len(z)):
            rho[t][i] /= den

    return rho

def Mstep(z, rgivenz, ratings, rho):
    samplecount = len(ratings)
    newz = [0]*len(z)
    newrgivenz = []*len(rgivenz)
    for j in range(len(rgivenz)):
        newrgivenz.append([0]*len(z))

    for i in range(len(z)):
        newz[i] = sum([x[i] for x in rho])/samplecount

    for t in range(samplecount):
        for i in range(len(z)):
            for j in range(len(rgivenz)):
                if ratings[t][j] != -1:
                    if ratings[t][j] == 1:
                        newrgivenz[j][i] += rho[t][i]
                    else:
                        newrgivenz[j][i] += rho[t][i] * rgivenz[j][i]

    for j in range(len(rgivenz)):
        for i in range(len(z)):
            newrgivenz[j][i] /= sum([x[i] for x in rho])
```

```
    return (newz, newrgivenz)


def likelihood(z, rgivenz, ratings):
    samplecount = len(ratings)
    ll = 0
    for t in range(len(ratings)):
        val = 0
        for i in range(len(z)):
            prob = 1
            for j in range(len(rgivenz)):
                if ratings[t][j] != -1:
                    if ratings[t][j] == 1:
                        prob *= rgivenz[j][i]
                    else:
                        prob *= (1 - rgivenz[j][i])

            val += prob*z[i]
        ll += np.log(val)/samplecount
    return ll


def EM(iters, z, rgivenz, ratings):
    for i in range(iters):
        ll = likelihood(z, rgivenz, ratings)
        if i in [0,1,2,4,8,16,32,64, 128, 256]:
            print("Log Likelihood at Iteration {}: {}"
                  .format(i, likelihood(z, rgivenz, ratings)))
        rho = Estep(z, rgivenz, ratings)
#         pprint.pprint(rho)
        z, rgivenz = Mstep(z, rgivenz, ratings, rho)
    print("Log Likelihood at Iteration {}: {}"
          .format(i+1, likelihood(z, rgivenz, ratings)))
    return (rho, z, rgivenz)

rho, z, rgivenz = EM(256, z, rgivenz, ratings)
```

```
Log Likelihood at Iteration 0: -29.32759381861126
Log Likelihood at Iteration 1: -18.13928372437196
Log Likelihood at Iteration 2: -16.17129923219528
Log Likelihood at Iteration 4: -14.941642713922834
Log Likelihood at Iteration 8: -14.21071932394613
Log Likelihood at Iteration 16: -13.858051333076753
Log Likelihood at Iteration 32: -13.763965178021754
Log Likelihood at Iteration 64: -13.739830925803794
Log Likelihood at Iteration 128: -13.73771683485948
Log Likelihood at Iteration 256: -13.73749791054687
```

8.1f) Personal Movie Recommendation

In [7]:
```
myratings = ratings[students.index("A53301791")]
myratingindex = students.index("A53301791")
expected_ratings = [0]*len(myratings)
for j in range(len(myratings)):
    for i in range(len(z)):
        if myratings[j] == -1:
            expected_ratings[j] += rho[myratingindex][i] * rgivenz[j][i]

suggested_movies = [[i,val] for i,val
                    in enumerate(expected_ratings) if val > 0]
suggested_movies.sort(key=lambda x: x[1], reverse=True)
[movies[i] for i, val in suggested_movies]
```

Out[7]:
```
['Hidden_Figures',
 'The_Hateful_Eight',
 'The_Farewell',
 'Shutter_Island',
 'Django_Unchained',
 'The_Martian',
 'Ready_Player_One',
 '12_Years_a_Slave',
 'The_Help',
 'Drive',
 'Her',
 'Les_Miserables',
 'Chappaquidick',
 'Three_Billboards_Outside_Ebbing',
 'The_Girls_with_the_Dragon_Tattoo',
 'X-Men:_First_Class',
 'Toy_Story_3',
 'Parasite',
 'Darkest_Hour',
 'Rocketman',
 'Room',
 'Mad_Max:_Fury_Road',
 'Midnight_in_Paris',
 'Star_Wars:_The_Force_Awakens',
 'Prometheus',
 'Phantom_Thread',
 'Terminator:_Dark_Fate',
 'The_Hunger_Games',
 'Manchester_by_the_Sea',
 'American_Hustle',
 'Ex_Machina',
 'Once_Upon_a_Time_in_Hollywood',
 'Us',
 'Man_of_Steel',
 'World_War_Z',
 'Hustlers',
 'Good_Boys',
 'The_Shape_of_Water',
 'I_Feel_Pretty',
 'Magic_Mike',
 'The_Last_Airbender']
```

In [ ]: