Q1. EM algorithm for binary matrix completion:

(a) Sanity check:
Mean popularity = #Recommended / #Saw
The movies sorted from least to most popular are :

Fifty_Shades_of_Grey	0.331
The_Last_Airbender	0.357
Magic_Mike	0.525
Prometheus	0.544
Bridemaids	0.556
World_War_Z	0.590
Man_of_Steel	0.592
Mad_Max:_Fury_Road	0.633
Drive	0.636
Thor	0.665
Pitch_Perfect	0.667
The_Hunger_Games	0.671
Fast_Five	0.678
The_Hateful_Eight	0.683
Iron_Man_2	0.692
The_Perks_of_Being_a_Wallflower	0.695
American_Hustle	0.716
The_Help	0.725
Avengers:_Age_of_Ultron	0.730
21_Jump_Street	0.736
Captain_America:_The_First_Avenger	0.736
Les_Miserables	0.745
Star_Wars:_The_Force_Awakens	0.749
Jurassic_World	0.754
The_Great_Gatsby	0.755
X-Men:_First_Class	0.764
The_Revenant	0.767
Her	0.779
Ex_Machina	0.786
Room	0.786
Django_Unchained	0.789
The_Girls_with_the_Dragon_Tattoo	0.789
Frozen	0.790
Midnight_in_Paris	0.798
The_Avengers	0.801
Wolf_of_Wall_Street	0.804
Harry_Potter_and_the_Deathly_Hallows:_Part_1	0.804
Black_Swan	0.810
Toy_Story_3	0.810
Harry_Potter_and_the_Deathly_Hallows:_Part_2	0.832
Gone_Girl	0.839
The_Theory_of_Everything	0.841
12_Years_a_Slave	0.841
Now_You_See_Me	0.861
The_Social_Network	0.894
The_Martian	0.901
Shutter_Island	0.908
Interstellar	0.914
The_Dark_Knight_Rises	0.926
Inception	0.996

For the most part the sorted movie list does correspond to my preferences, as I really liked the movies near the far-end of the list like Martian, Inception, The dark knight trilogy etc. Also it is interesting to note that the most popular movies are directed by Christopher Nolan.

(e)

The log-likelihood increases at each iteration, verifying the convergence.

Iteration	Log-likelihhod (L)
0	-23.6819
1	-14.3421 (Got -15.9532)
2	-13.4962
4	-12.2653
8	-11.9739
16	-11.6822 (Got -11.5974)
32	-11.2251
64	-10.7815

(f)

The predictions for movies I am yet to see are (sorted by expectation) are:

Thor
Harry_Potter_and_the_Deathly_Hallows:_Part_1
Room
American_Hustle
The_Girls_with_the_Dragon_Tattoo
Toy_Story_3
The_Great_Gatsby
Frozen
Bridemaids
Drive
The_Help
Midnight_in_Paris
Pitch_Perfect
Les_Miserables
Magic_Mike

The list seems accurate as I just saw Thor this weekend!

(g) Source code attached

Source code:

```
# coding: utf-8

# In[133]:

import numpy as np
from collections import defaultdict
import math


# In[12]:

with open('data/hw8_movieTitles.txt','r') as f:
    movies=f.read().split()
with open('data/hw8_studentPIDs.txt','r') as f:
    students=f.read().split()


# In[151]:

ratings=[]
count=0
with open('data/hw8_ratings.txt','r') as f:
    temp=f.read().split('\n')
    for i in temp:
        i=i.replace('?','-1')
        temp1 = i.strip().split(' ')
        temp1=[int(j) for j in temp1]
        ratings.append(temp1)
lratings=ratings
ratings=np.matrix(ratings)


# In[153]:

vis = defaultdict(list)
for i,j in enumerate(lratings):
    vis[i]=[]
    for k in range(len(j)):
        if j[k]==1 or j[k]==0:
            vis[i].append(k)


# In[110]:

ls=len(students)
lm=len(movies)
moviepop=[]
for i in range(lm):
    temp=ratings[:,i].flatten().tolist()[0]
    moviepop.append(temp.count(1)*1.0/(temp.count(1)+temp.count(0)))
moviespopular=sorted(zip(moviepop,movies),key=lambda i:i[0])
for i in moviespopular:
    print '{0:45s} {1:.3f}'.format(i[1],i[0])


# In[119]:

priorZ=[]
with open('data/hw8_probZ_init.txt','r') as f:
    temp=f.read().strip().split('\n')
    for i in temp:
        priorZ.append(float(i))


# In[171]:

rgz=[]
with open('data/hw8_probRgivenZ_init.txt','r') as f:
    temp=f.read().strip().split('\n')
    for i in temp:
        rgz.append([float(k) for k in i.split()])
#rgz=np.matrix(rgz)


# In[176]:

pzt=defaultdict()
for i in range(len(students)):
    pzt[i]=defaultdict()
    for j in range(len(priorZ)):
        pzt[i][j]=priorZ[j]

rgzd=defaultdict()
for i in range(len(movies)):
    rgzd[i]=defaultdict()
    for j in range(len(priorZ)):
        rgzd[i][j]=rgz[i][j]


# In[184]:

priorZ={0:0.25,1:0.25,2:0.25,3:0.25}
```

```
# In[195]:

def estep(priorZ,pzt,rgzd,vis):
    for i in vis:
        temp1=0
        for j in priorZ:
            temp=priorZ[j]
            for k in vis[i]:
                temp*=rgzd[i][k]
            pzt[i][j]=temp
            temp1+=temp
        for i in pzt:
            for j in pzt[i]:
                pzt[i][j]/=temp1
    return priorZ,pzt,rgzd,vis

def mstep(priorZ,pzt,rgzd,vis):
    for i in priorZ:
        temp=0
        for k in pzt:
            temp+=pzt[k][i]
        priorZ[i]=temp/len(students)
    for i in rgzd:
        for k in rgzd[i]:
            tmp=0
            for p in range(len(movies)):
                if p in viz[i]:
                    if ratings[i][p]==1:
                        tmp+=pzt[k][i]
                    else:
                        tmp+=pzt[k][i]*rgzd[p][k]
            rgzd[i][k]=temp/len(students)/priorZ[k]
    return priorZ,pzt,rgzd,vis

def ll(priorZ,pzt,rgzd,vis):
    temp2=0
    for i in vis:
        temp1=0
        for j in priorZ:
            temp=priorZ[j]
            for k in vis[i]:
                temp*=rgzd[i][k]
            temp1+=temp
        temp2+=math.log(temp1)
    return temp2/len(students)


# In[196]:

for ite in range(64):
    priorZ,pzt,rgzd,vis=estep(priorZ,pzt,rgzd,vis)
    priorZ,pzt,rgzd,vis=mstep(priorZ,pzt,rgzd,vis)
    print ll(priorZ,pzt,rgzd,vis)


# In[ ]:
```