CSE 150A/250A. Assignment 7

7.1 EM algorithm for binary matrix completion

In this problem you will use the EM algorithm to build a simple movie recommendation system. Download the files *hw7_movies.txt*, *hw7_ids.txt*, and *hw7_ratings.txt*. The last of these files contains a matrix of zeros, ones, and missing elements denoted by question marks. The $\langle i, j \rangle^{\text{th}}$ element in this matrix contains the i^{th} student's rating of the j^{th} movie, according to the following key:

1	recommended,
0	not recommend,
?	not seen.

(a) **Sanity check**

Compute the mean popularity rating of each movie, given by the simple ratio

$$\frac{\text{number of students who recommended the movie}}{\text{number of students who saw the movie}},$$

and sort the movies by this ratio. Print out the movie titles from least popular to most popular along with their mean popularity ratings. Note how well these rankings do or do not corresponding to your individual preferences.

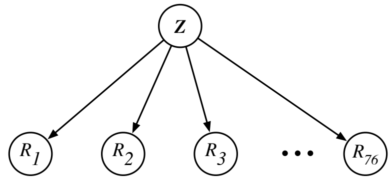
output is too long to screenshot, see below:
0.418848167539267: Justice League
0.48484848484848486, The Last Airbender
0.5137614678899083, Batman v Superman: Dawn of Justice
0.521551724137931, Suicide Squad
0.5797665369649806, Ant-Man and the Wasp
0.5862068965517241, Solo
0.589041095890411, The Shape of Water
0.5897435897435898, Star Wars: The Last Jedi
0.6037735849056604, Terminator Genisys
0.6092436974789915, Wonder Woman
0.6127450980392157, Furious 7
0.6373626373626373, It
0.6422018348623854, Star Trek Beyond
0.6604938271604939, World War Z
0.6666666666666666, Man of Steel
0.6774193548387096, Jumanji: Welcome to the Jungle
0.6774193548387096, Oceans 8
0.6863905325443787, Rogue One
0.6870229007633588, Mad Max: Fury Road

0.6956521739130435, Fantastic Beasts and Where To Find Them
0.7063492063492064, The Lego Movie
0.7156862745098039, Ex Machina
0.7218045112781954, Tron
0.7222222222222222, Venom
0.7260726072607261, Jurassic World
0.7644230769230769, Star Wars: The Phantom Menace
0.7703180212014135, Thor: Ragnarok
0.7711864406779662, Frozen
0.7899686520376176, Iron Man 3
0.7907801418439716, Guardians of the Galaxy Vol. 2
0.7949790794979079, Moana
0.8018018018018018, Zootopia
0.8092105263157895, The Greatest Showman
0.8178913738019169, Captain America: Civil War
0.821656050955414, Get Out
0.8266666666666667, Blade Runner 2049
0.8338557993730408, Black Panther
0.833976833976834, Deadpool 2
0.8403361344537815, Coco
0.8449367088607594, The Hunger Games
0.8454935622317596, La La Land
0.8488372093023255, Logan
0.8503937007874016, 2001: A Space Odyssey
0.8551532033426184, The Avengers
0.8595505617977528, Terminator 2
0.8619631901840491, Harry Potter and the Deathly Hallows: Part 2
0.8733624454148472, Mission: Impossible - Fallout
0.8823529411764706, Avengers: Infinity War
0.8884297520661157, The Martian
0.8923076923076924, Doctor Strange
0.9015873015873016, Guardians of the Galaxy
0.9032258064516129, The Lord of the Rings: The Fellowship of the Ring
0.9056603773584906, The Imitation Game
0.9073359073359073, The Wolf of Wall Street
0.9202453987730062, WALL-E
0.9206349206349206, Jurassic Park (1993)
0.9397163120567376, Inception
0.9397163120567376, The Dark Knight
0.9421768707482994, The Matrix
0.9464285714285714, Interstellar

(b) **Likelihood**

Now you will learn a naive Bayes model of these movie ratings, represented by the belief network shown below, with hidden variable $Z \in \{1, 2, \ldots, k\}$ and partially observed binary variables

$R_1, R_2, \ldots, R_{60}$ (corresponding to movie ratings).



This model assumes that there are k different types of movie-goers, and that the i^{th} type of movie-goer—who represents a fraction $P(Z=i)$ of the overall population—likes the j^{th} movie with conditional probability $P(R_j=1|Z=i)$. Let Ω_t denote the set of movies seen (and hence rated) by the t^{th} student. Show that the likelihood of the t^{th} student's ratings is given by

$$P\left(\left\{R_j=r_j^{(t)}\right\}_{j\in\Omega_t}\right) \;=\; \sum_{i=1}^{k} P(Z=i) \prod_{j\in\Omega_t} P\left(R_j=r_j^{(t)}\middle|\, Z=i\right).$$

Marginalize $P(R_j=r_j^{(t)})$
$\sum_{i=1}^{k} P(Z=i)\; P(R_j=r_j^{(t)}|Z=i)$ All R_j are conditionally independent given Z
$\sum_{i=1}^{k} P(Z=i)\; \prod_{j\in\Omega_t} P(R_j=r_j^{(t)}|Z=i)$

(c) **E-step**

The E-step of this model is to compute, for each student, the posterior probability that he or she corresponds to a particular type of movie-goer. Show that

$$P\left(Z=i\,\middle|\left\{R_j=r_j^{(t)}\right\}_{j\in\Omega_t}\right) \;=\; \frac{P(Z=i)\prod_{j\in\Omega_t} P\left(R_j=r_j^{(t)}\middle|\, Z=i\right)}{\sum_{i'=1}^{k} P(Z=i')\prod_{j\in\Omega_t} P\left(R_j=r_j^{(t)}\middle|\, Z=i'\right)}.$$

Bayes theorem
$\frac{P(Z=i)P(R_j=r_j^{(t)}|Z=i)}{P(R_j=r_j^{(t)}|Z=i')}$ marginalize denominator
$\frac{P(Z=i)P(R_j=r_j^{(t)}|Z=i)}{\sum_{i'=1}^{k} P(Z=i')P(R_j=r_j^{(t)}|Z=i')}$ conditional independence of R_j, same as (b)
$\frac{P(Z=i)\prod_{j\in\Omega_t} P(R_j=r_j^{(t)}|Z=i)}{\sum_{i'=1}^{k} P(Z=i')\prod_{j\in\Omega_t} P(R_j=r_j^{(t)}|Z=i')}$

(d) **M-step**

The M-step of the model is to re-estimate the probabilities $P(Z=i)$ and $P(R_j=1|Z=i)$ that define the CPTs of the belief network. As shorthand, let

$$\rho_{it} \;=\; P\left(Z=i\,\middle|\left\{R_j=r_j^{(t)}\right\}_{j\in\Omega_t}\right)$$

denote the probabilities computed in the E-step of the algorithm. Also, let T denote the number of students. Show that the EM updates are given by

$$P(Z=i) \;\leftarrow\; \frac{1}{T}\sum_{t=1}^{T}\rho_{it},$$

$$P(R_j=1|Z=i) \;\leftarrow\; \frac{\sum_{\{t|j\in\Omega_t\}}\rho_{it}\, I\Big(r_j^{(t)},1\Big) + \sum_{\{t|j\notin\Omega_t\}}\rho_{it}\, P(R_j=1|Z=i)}{\sum_{t=1}^{T}\rho_{it}}.$$

General form for root nodes is $\frac{count(X_i=x|V_t=v_t)}{T}$, in this case Z_i is X_i and R_j is V_t
$P(Z=i) = \frac{1}{T}\sum_{t=1}^{T}\rho_{it}$

General form for child nodes is $\frac{P(X_i,pa_i=\pi|V_t=v_t)}{\sum P(pa_i=\pi|V_t=v_t)}$
denominator is Z_i, and Z_i is known
$\frac{P(Z_i,R_j=1|R_j)}{\sum_{t=1}^{T}\rho_{it}}$

For numerator $P(Z_i, R_j = 1|R_j)$
The count must be summed over "known" data (1 or 0) separately from unknown data
For $P(Z_i, R_j = 1|R_j = 1)$ ($r_j^{(t)} = 1$), this is equivalent to $P(Z_i|R_j = r_j^{(t)})$
For $P(Z_i, R_j = 1|R_j = 0)$ ($r_j^{(t)} = 0$), this is equivalent to 0 since $R_j \neq 1$ if R_j is already given as 0
This can be expressed as an indicator function multiplied by $P(Z=i|\{R_j=r_j^{(t)}\}_{j\in\Omega_t})$, or ρ_{it}
This gives us the $\sum_{\{t|j\in\Omega_t\}}\rho_{it}\, I\Big(r_j^{(t)},1\Big)$ term

To solve for $P(Z=i, R_j = 1|\{R_j=r_j^{(t)}\}_{j\in\Omega_t})$ when $j\notin\Omega_t$
$P(Z=i, R_j = 1|\{R_j=r_j^{(t)}\}_{j\in\Omega_t})$
$\frac{P(Z=i,R_j=1,\{R_j=r_j^{(t)}\}_{j\in\Omega_t})}{P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t})}$
Product rule on the joint in the numerator
$\frac{P(R_j=1)P(Z=i|R_j=1)P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t}|Z=i,R_j=1)}{P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t})}$
Since $j\notin\Omega_t$ R_j is conditionally independent of $\{R_j=r_j^{(t)}\}_{j\in\Omega_t}$ given Z
$\frac{P(R_j=1)P(Z=i|R_j=1)P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t}|Z=i)}{P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t})}$
multiply by $P(Z = i)$ in numerator and denominator
$\frac{P(Z=i)P(R_j=1)P(Z=i|R_j=1)P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t}|Z=i)}{P(Z=i)P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t})}$
separate terms
$\frac{P(R_j=1)P(Z=i|R_j=1)}{P(Z=i)}\frac{P(Z=i)P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t}|Z=i)}{P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t})}$
This simplifies to $\sum_{\{t|j\notin\Omega_t\}}\rho_{it}\, P(R_j=1|Z=i)$

(e) **Implementation**

Download the files *hw7_probZ_init.txt* and *hw7_probR_init.txt*, and use them to initialize the probabilities $P(Z=i)$ and $P(R_j=1|Z=i)$ for a model with $k=4$ types[1] of movie-goers. Run 256 iterations of the EM algorithm, computing the (normalized) log-likelihood

$$\mathcal{L} \;=\; \frac{1}{T}\sum_{t=1}^{T} \log P\left(\left\{R_j = r_j^{(t)}\right\}_{j\in\Omega_t}\right)$$

at each iteration. Does your log-likelihood increase (i.e., become less negative) at each iteration? Fill in a completed version of the following table, using the already provided entries to check your work. Note, there will be some small variance across correct implementations. We have reported four significant figures – a precision we have determined to be mostly reproducible. However, if you're getting only three significant figures of agreement, that is not necessarily indicative of a problem.

iteration	log-likelihood $\mathcal{L}$
0	-28.0893
1	-16.4466
2	-14.5124
4	-13.5682
28	-13.1919
16	-13.0204
32	-12.9766
64	-12.9514
128	-12.9482
256	-12.9470

[1]There is nothing special about these initial values or the choice of $k=4$; feel free to experiment with other choices.

```
0 | -28.089322318978795
1 | -16.446667908107614
2 | -14.512415968857413
4 | -13.568215594235635
8 | -13.191755726801526
16 | -13.020405719830022
32 | -12.976600080083852
64 | -12.9513657787 4896
128 | -12.948196484339462
256 | -12.947037962691367
```

(f) **Personal movie recommendations**

Find your student PID in *hw7_ids.txt* to determine the row of the ratings matrix that stores your personal data. Compute the posterior probability in part (c) for this row from your trained model, and then compute your *expected* ratings on the movies *you haven't yet seen*:

$$P\left(R_\ell\!=\!1\middle|\left\{R_j\!=\!r_j^{(t)}\right\}_{j\in\Omega_t}\right) \;=\; \sum_{i=1}^{k} P\left(Z\!=\!i\middle|\left\{R_j\!=\!r_j^{(t)}\right\}_{j\in\Omega_t}\right) P(R_\ell\!=\!1|Z\!=\!i) \quad \text{for } \ell\not\in\Omega_t.$$

Print out the list of these (unseen) movie sorted by their expected ratings. Does this list seem to reflect your personal tastes better than the list in part (a)? Hopefully it does (although our data set is obviously *far* smaller and more incomplete than the data sets at companies like Netflix or Amazon).

Note: if you didn't complete the survey in time, then you will need to hard-code your ratings in order to answer this question.

Some of the Marvel movies seem to have higher probabilities, which makes sense since my small data set was mostly marvel/disney, and I agree I would probably hate suicide squad and the last airbender. I don't know much about inception or the martian I have no idea why they're so high.

Too long to screenshot again, formatted in the order they are listed in hw7_movies-1.txt with the index corresponding to the item's location in the hw7_movies-1.txt list, skipping over movies that I have watched

1: Inception, 0.9997567025320717
2: The Last Airbender, 0.2600053880296673
3: The Hunger Games, 0.9145701722936624
4: The Wolf of Wall Street, 0.9546728568445593
5: World War Z, 0.6888361620190133
6: Interstellar, 0.9810256788699457
7: The Martian, 0.9967108891605855
8: Iron Man 3, 0.6333353115045937
9: La La Land, 0.9399348473802427
12: The Dark Knight, 0.9997476660546349
14: The Matrix, 0.999893168553906
15: Star Trek Beyond, 0.8610846342364251
16: Jurassic World, 0.6229704770647546
17: Jurassic Park (1993), 0.9996965031979234
18: Deadpool 2, 0.8255112116349292
19: Guardians of the Galaxy, 0.9346410399612107
20: Mission: Impossible - Fallout, 0.8703848026624152
21: Guardians of the Galaxy Vol. 2, 0.7238968609850688
23: Tron, 0.8434474222063633
24: Star Wars: The Phantom Menace, 0.8050825928295468
26: Man of Steel, 0.669808533345263
27: Get Out, 0.9560229264548108
28: Suicide Squad, 0.3067678992285917
29: The Shape of Water, 0.8706683821027354
30: The Avengers, 0.870972148305148
31: Mad Max: Fury Road, 0.8467078035069898
33: The Imitation Game, 0.9586486108732233
34: Ex Machina, 0.7441969703722172
36: Blade Runner 2049, 0.9275954527198084
37: Terminator Genisys, 0.3518702911434544
38: Terminator 2, 0.9771741763694934
41: Ant-Man and the Wasp, 0.30868291625182237
42: Venom, 0.6605660055650291
43: Oceans 8, 0.6719806277990718
44: The Greatest Showman, 0.9304129449957286
46: Harry Potter and the Deathly Hallows: Part 2, 0.855078767708039
48: Logan, 0.9668309553731811
49: Jumanji: Welcome to the Jungle, 0.6117763899044222
50: It, 0.7247106636141912
51: Justice League, 0.053693183658078576
53: Rogue One, 0.799735407140897
54: Solo, 0.4840112061124336
55: Captain America: Civil War, 0.8037399425051308
56: Batman v Superman: Dawn of Justice, 0.23441502576778778
59: Furious 7, 0.4636523881946322

(g) **Source code**

Turn in a copy of your source code for all parts of this problem. As usual, you may program in the language of your choice.

Code uses breaklines for lines that are too long to fit the width of the page

```
#!/usr/bin/env python
# coding: utf-8


import numpy as np


ratings = []
for i in open('hw7_ratings-1.txt', 'rt'):
    ratings.append(np.array(i.split()))
ratings = np.array(ratings)
movies = []
for i in open('hw7_movies-1.txt', 'rt'):
    movies.append(i.strip())



probs = []
for i in open('hw7_probR_init.txt', 'rt'):
    probs.append(np.array(i.split()).astype(float))
probs = np.array(probs)
pids = []
for i in open("hw7_ids-2.txt"):
    pids.append(i.strip())


# part a
ratings_1 = np.array([sum('1' == i) for i in ratings.T])
ratings_0 = np.array([sum('0' == i) for i in ratings.T])
ratings_q = np.array([sum('?' == i) for i in ratings.T])
r = ratings_1 / (ratings_1 + ratings_0)
movie_mean = [[r[i], movies[i]] for i in range(len(r))]
movie_mean.sort()
# produces latex code for me so I don't have to screenshot:
#for i in range(len(movie_mean)):
#    print(str(movie_mean[i][0]) + ", " + movie_mean[i][1] +
 ↪  "\\\\")
```

```
movie_mean



# part e
def log_likelihood(d, z, probs):
    L = 0
    for t in range(len(d)):
        r_t = ratings[t]
        r_j = 0
        for i in range(4):
            prod = 1
            for j in range(60):
                if r_t[j] == '?':
                    continue
                elif r_t[j] == '1':
                    prod *= probs[j, i]
                elif r_t[j] == '0':
                    prod *= (1 - probs[j, i])
            r_j += z[i] * prod
        L += np.log(r_j)
    return L / (len(d))

# m step
# me on my way to invent the most suboptimal code
def update(d, z, probs):
    # T x k matrix of all rho_it
    pits = []
    for t in range(len(d)):
        r_t = ratings[t]
        for_t = []
        for i in range(4):
            prod = 1
            for j in range(60):
                if r_t[j] == '?':
                    continue
                elif r_t[j] == '1':
                    prod *= probs[j, i]
                elif r_t[j] == '0':
                    prod *= (1 - probs[j, i])
            for_t.append(z[i] * prod)
        pits.append(np.array(for_t) / sum(for_t))
    pits = np.array(pits)
    # all summation pit for all i
    summation_pit = np.sum(pits, axis = 0)
```

```python
    # divide by T
    z = summation_pit / len(d)
    rj_z = []
    for j in range(60):
        nums = []
        for i in range(4):
            numer = 0
            denom = 0
            for t in range(len(d)):
                r_t = ratings[t]
                denom += pits[t, i]
                if r_t[j] == '0':
                    continue
                elif r_t[j] == '?':
                    numer += pits[t, i] * probs[j, i]
                elif r_t[j] == '1':
                    numer += pits[t, i]
            nums.append(numer / denom)
        rj_z.append(np.array(nums))
    #print(summation_pit)
    return z, np.array(rj_z)


z = [0.25, 0.25, 0.25, 0.25]
p = probs
print('0 | ' + str(log_likelihood(ratings, [0.25, 0.25, 0.25,
 ↪  0.25], probs)))
for i in range(1, 257):
    z, p = update(ratings, z, p)
    if i in [2 ** i for i in range(0, 9)]:
        print(str(i) + " | " + str(log_likelihood(ratings, z, p)))
#b
#a, b


r_t = ratings[pids.index('A18083527')]


for r in range(len(r_t)):
    zis = []
    rl_zs = []
    for i in range(4):
        prod = 1
        for j in range(60):
```

```
            if r_t[j] == '?':
                continue
            elif r_t[j] == '1':
                prod *= probs[j, i]
            elif r_t[j] == '0':
                prod *= (1 - probs[j, i])
        zis.append(z[i] * prod)
        rl_zs.append(p[r, i])
    if r_t[r] == "?":
        # formatted to
        print(str(r) + ": " + movies[r]  + ", " +
         ↪  str(sum((np.array(zis) / sum(zis)) * rl_zs)) + "\\\\")
```