Problem 8.1

Sunday, November 27, 2022 6:14 PM

8.1 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 *hw8_movies.txt*, *hw8_ids.txt*, and *hw8_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 (*Chappaquidick*) to most popular (*Inception*). Note how well these rankings do or do not corresponding to your individual preferences.

Ans: Here are the movies from least to most popular (i.e. in ascending order of mean popularity rating):

The movies from least to most popular are:

Chappaquidick
The_Last_Airbender
I_Feel_Pretty
Fifty_Shades_of_Grey
Fast_&_Furious:_Hobbs_&_Shaw
Hustlers
Magic_Mike
Bridemaids
World_War_Z
The_Shape_of_Water
Good_Boys
Prometheus
Pokemon_Detective_Pikachu
American_Hustle
Terminator:_Dark_Fate
The_Farewell
Man_of_Steel
Fast_Five
The_Hateful_Eight
Star_Wars:_The_Force_Awakens
The_Help
Rocketman
Drive
The_Girls_with_the_Dragon_Tattoo
Thor
Avengers:_Age_of_Ultron
Phantom_Thread
Us
The_Revenant
X-Men:_First_Class
Pitch_Perfect
Dunkirk
Ready_Player_One
Room
Jurassic_World
Mad_Max:_Fury_Road
Once_Upon_a_Time_in_Hollywood
Manchester_by_the_Sea
The_Perks_of_Being_a_Wallflower
Spiderman:_Far_From_Home
Her
Captain_America:_The_First_Avenger
Frozen

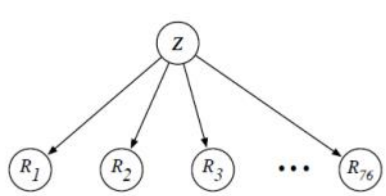
contd...

list contd. (after left list)

Hidden_Figures
The_Hunger_Games
Iron_Man_2
Les_Miserables
Toy_Story_3
Three_Billboards_Outside_Ebbing
Darkest_Hour
Ex_Machina
Gone_Girl
Black_Swan
12_Years_a_Slave
Avengers:_Endgame
The_Avengers
Midnight_in_Paris
The_Great_Gatsby
La_La_Land
Avengers:_Infinity_War
The_Theory_of_Everything
Now_You_See_Me
21_Jump_Street
Django_Unchained
The_Martian
Harry_Potter_and_the_Deathly_Hallows:_Part_1
Joker
Wolf_of_Wall_Street
The_Lion_King
Harry_Potter_and_the_Deathly_Hallows:_Part_2
Parasite
The_Social_Network
The_Dark_Knight_Rises
Shutter_Island
Interstellar
Inception

(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 $R_1, R_2, \ldots, R_{76}$ (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 student. Show that the likelihood of the t^{th} student's ratings is given by

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

Pf: we know $P(\{R_j = r_j^{(t)}\}_{j\in\Omega_t})$ = likelihood of tth student's ratings

$\| $ (Marginalization)

$$\sum_{i=1}^{k} P(Z=i) P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t} | Z=i) \overset{\text{product rule}}{=\!=} \sum_{i=1}^{k} P(Z=i, \{R_j = r_j^{(t)}\}_{j\in\Omega_t})$$

In the BN above Z d-separates R_j from R_i for $i \neq j$

$P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t} | Z=i) = \prod_{j\in\Omega_t} P(R_j=r_j^{(t)}|Z=i)$

$$\| \quad \sum_{i=1}^{k} P(Z=i) \prod_{j\in\Omega_t} P(R_j = r_j^{(t)} | Z=i) \quad \text{(proved)}.$$

(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

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

Pf:
from part (b) above, we have: $P(\{R_j = r_j^{(t)}\}_{j\in\Omega_t}) = \sum_{i=1}^{k} P(Z=i) \prod_{j\in\Omega_t} P(R_j=r_j^{(t)}|Z=i)$

& $P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t}|Z=i) = \prod_{j\in\Omega_t} P(R_j=r_j^{(t)}|Z=i)$

Then by Baye's rule:

$$P(Z=i \mid \{R_j=r_j^{(t)}\}_{j\in\Omega_t})$$

$\|$

$$P(Z=i) P(\{R_j=r_j^{(t)}\}_{j\in\Omega_t}|Z=i) \,/\, P(\{R_j = r_j^{(t)}\}_{j\in\Omega_t})$$

$\|$ (from above)

$$\frac{P(Z=i) \prod_{j\in\Omega_t} P(R_j = r_j^{(t)} | Z=i)}{\sum^{k} P(Z=i') \prod P(R_j=r_j^{(t)}|Z=i')} \quad \text{(proved)}.$$

$$\sum_{i'=1}^{k} P(Z=i') \prod_{j\in\Omega_t} P(R_j=r_j^{(t)}|Z=i')$$

(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

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

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

Pf: Recall the generally the M step CPT update looks like:

ML estimation for incomplete data

- **Notation**

  Nodes $X_1, X_2, \ldots, X_n$
  Examples $t = 1, 2, \ldots, T$
  Visible nodes $V_t = v_t$ for t^{th} example

- **EM algorithm**

  Initialize CPTs to nonzero values.
  Repeat until convergence:

  E-step — compute posterior probabilities.
  M-step — update CPTs:

  root nodes $\quad P(X_i=x) \longleftarrow \frac{1}{T}\sum_t P(X_i=x|V_t=v_t)$

  nodes with parents $\quad P(X_i=x|\text{pa}_i=\pi) \longleftarrow \frac{\sum_t P(X_i=x, \text{pa}_i=\pi|V_t=v_t)}{\sum_t P(\text{pa}_i=\pi|V_t=v_t)}$

(from lecture 12's slides)

So in this BN, the M step update looks like:

i) $P(Z=i) \leftarrow \frac{1}{T}\sum_{t=1}^{T} P(Z=i|\{R_j=r_j^{(t)}\}_{t\in\Omega_t}) = \frac{1}{T}\sum_{t=1}^{T}\rho_{it}$

(by defn of ρ_{it})

ii) $P(R_j=1|Z=i) \leftarrow \sum P(R_j=1, Z=i|\{R_j=r_j\}_{j\in\Omega_t})$

(by def^n of ρ_{it})

ii) $$P(R_j=1|Z=i) \leftarrow \frac{\sum_t P(R_j=1, Z=i \mid \{R_j=r_j\}_{j\in\Omega_t})}{\sum_t P(Z=i \mid \{R_j=r_j\}_{j\in\Omega_t})}$$

We note, the denominator here is $\sum_t \rho_{it}$ & the numerator is:

$$\sum_t P(R_j=1, Z=i \mid \{R_j=r_j\}_{j\in\Omega_t}) = \sum_{\substack{t \text{ s.t.} \\ j\in\Omega_t}} P(R_j=1, Z=i \mid \{R_j=r_j^{(t)}\}_{j\in\Omega_t})$$

$$+ \sum_{\substack{t \text{ s.t.} \\ j\notin\Omega_t}} P(R_j=1, Z=i \mid \{R_j=r_j^{(t)}\}_{j\in\Omega_t})$$

(& product rule) || (using indicator functions)

$$\sum_{\{t|j\in\Omega_t\}} I(r_j^{(t)}, 1)\, P(Z=i \mid \{R_j=r_j^{(t)}\}_{j\in\Omega_t})$$

$$+ \sum_{\{t|j\notin\Omega_t\}} P(R_j=1 \mid Z=i, \{R_j=r_j^{(t)}\}_{j\in\Omega_t})\, P(Z=i \mid \{R_j=r_j^{(t)}\}_{j\in\Omega_t})$$

(Z d-separates R_j from R_k for $k\neq j$) || (def^n of ρ_{it})

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

so we have the M-update as: (for node R_j)

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

(e) **Implementation**

Download the files *hw8_probZ_init.txt* and *hw8_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.

Ans: We fill out the table as (for outputs & source code, please see below)

Incomplete table

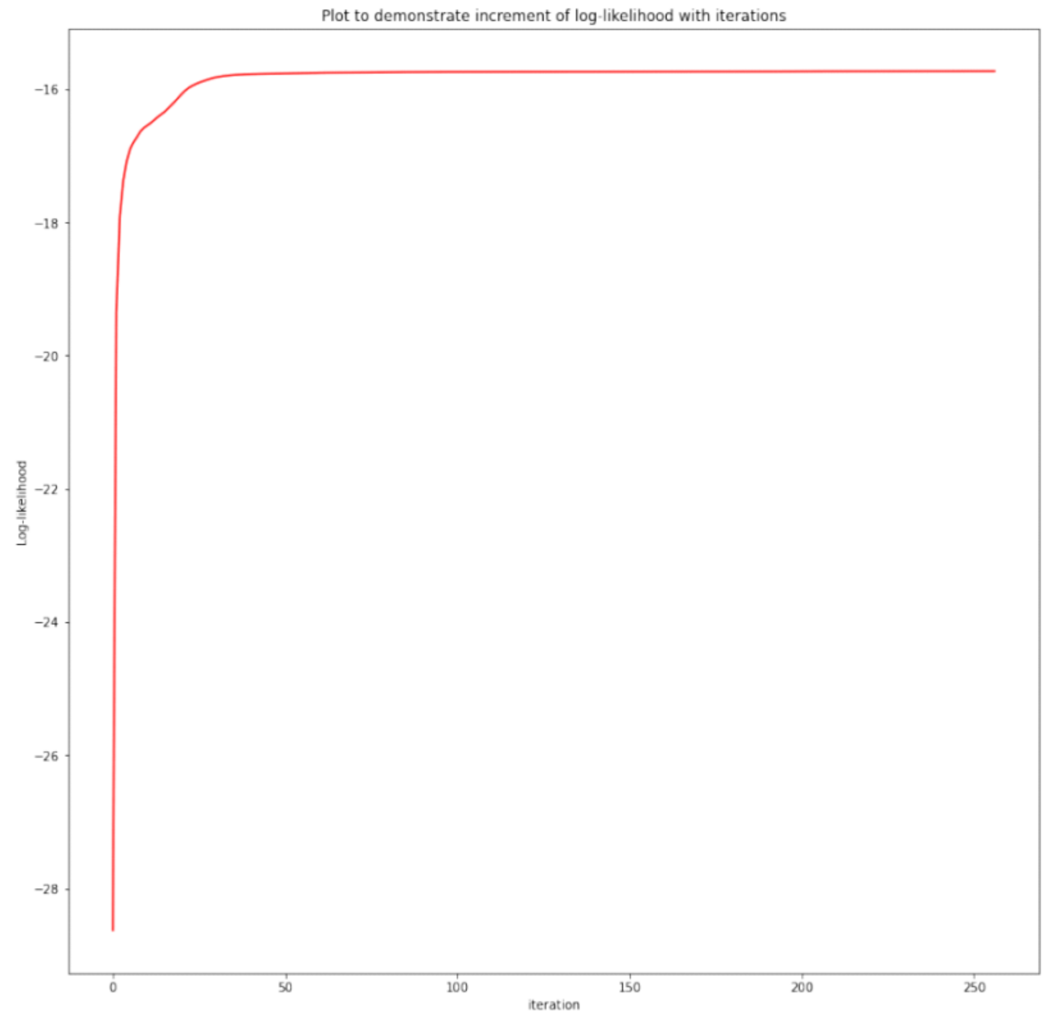
iteration	log-likelihood $\mathcal{L}$
0	-28.63
1	-19.35
2	
4	
8	
16	-16.29
32	
64	
128	
256	

$\Longrightarrow$ completed table

Completed table

iteration	log-likelihood $\mathcal{L}$
0	-28.627324487337628
1	-19.350314946503318
2	-17.909564818017916
4	-17.081155562337013
8	-16.629824767528117
16	-16.28782872191562
32	-15.801537953970273
64	-15.749887678844292
128	-15.735940712575662
256	-15.728520329683299

from the completed table we can see log-likelihoods are increasing fast at the top & then very slowly at the bottom. We confirm this observation by plotting the log-likelihood:



(f) Personal movie recommendations

Find your student PID in *hw8_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*:

$$P\left(R_\ell\!=\!1 \,\middle|\, \left\{R_j\!=\!r_j^{(t)}\right\}_{j\in\Omega_t}\right) \;=\; \sum_{i=1}^{k} P\left(Z\!=\!i \,\middle|\, \left\{R_j\!=\!r_j^{(t)}\right\}_{j\in\Omega_t}\right) P(R_\ell\!=\!1|Z\!=\!i) \quad \text{for } \ell \notin \Omega_t.$$

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.

Ans: Since I have watched ALL movies in the given movie list, there is no unseen movie for me to print as per recommendation here.

□ Upon asking this question on piazza, I was told the following:

Problem faced in question 1 part f

Problem faced in question 1 part f

Dear Instructors,

I was trying to code the question 1 part f, where I have to build a personal movie recommendation system on the movies that I 'have not' watched.

However, in the list of 76 movies, I have watched every single one of them (you can check for my PID: A53274333). I am really sorry if there was some instruction on putting some movies as '?' mandatorily, but in my data of rating, all movies are rated '0' and '1' and no movies are rated '?'. That is why this part f is giving me trivial output which is not indicative of the correctness of my code.

I will be highly obliged if you can kindly let me know what to do in this case. Can I please use someone else's PID ? If yes, can you suggest me one PID randomly ? I can take a screenshot of this piazza post and put in my solution to mention this exception.

Thank you
Soumya.

the instructors' answer, *where instructors collectively construct a single answer*

Wow, you have watched all of them? that's insane! Don't worry, we'll figure out a way to do this.

thanks! | 0 — Updated 6 days ago by Yufan Zhuang

followup discussions, *for lingering questions and comments*

Resolved Unresolved @592_f1

Yufan Zhuang 6 days ago
You can randomly pick a PID and then reason over that student's data. Try to discuss what movies you think they might like based on what they filled in, and what the model recommended. And please mention why you are doing this in your assignment. Let us know if you have any other questions!

good comment | 0

So I used a random no-generator in my code to randomly pick a PID (in the case shown here, the PID is: A59019917)

We note the following:

For PID A59019917, the list of recommended movies among the ones that the student has watched, are

The_Last_Airbender
Harry_Potter_and_the_Deathly_Hallows:_Part_1
Iron_Man_2
Toy_Story_3
Fast_Five
Thor
Captain_America:_The_First_Avenger
Harry_Potter_and_the_Deathly_Hallows:_Part_2
Prometheus
The_Avengers
The_Dark_Knight_Rises
The_Hunger_Games
Wolf_of_Wall_Street
The_Great_Gatsby
Frozen
Now_You_See_Me
World_War_Z
Interstellar
Mad_Max:_Fury_Road
Jurassic_World
Avengers:_Age_of_Ultron
Room
The_Martian
Avengers:_Infinity_War
Ready_Player_One
Avengers:_Endgame
The_Lion_King
Joker
Parasite
Pokemon_Detective_Pikachu
Terminator:_Dark_Fate

and

For PID A59019917, the list of not-recommended movies among the ones that the student has watched, movies are

The_Social_Network
X-Men:_First_Class
Fifty_Shades_of_Grey

9

La_La_Land
Spiderman:_Far_From_Home
Fast_&_Furious:_Hobbs_&_Shaw

← and, our code gives the following recommendation (along with probabilities) for unseen movies by the student with this PID: ⇓

this PID: ⇓

For PID A59019917, the list of unseen movies sorted by their expected ratings in ascending order

```
Chappaquidick            with probability                0.6213302052441917
Magic_Mike               with probability                0.6422617630243619
The_Hateful_Eight                with probability
0.7016135706894251
American_Hustle                  with probability
0.7045502369460868
The_Shape_of_Water               with probability
0.709683381852856
Bridemaids               with probability                0.7142694420643408
Star_Wars:_The_Force_Awakens             with probability
0.7336263473818456
Man_of_Steel             with probability                0.7403785362092967
Us               with probability                0.7410509024405416
Rocketman                with probability                0.7542065637770378
I_Feel_Pretty            with probability                0.7559210577238806
Once_Upon_a_Time_in_Hollywood            with probability
0.7596510691932159
Hustlers                 with probability                0.7648218353296454
Good_Boys                with probability                0.7913552362465591
Manchester_by_the_Sea            with probability
0.7929879515312204
The_Girls_with_the_Dragon_Tattoo                 with probability
0.7940073194473092
Dunkirk                  with probability                0.794897398744033
Pitch_Perfect            with probability                0.7961213640800816
The_Revenant             with probability                0.8043696029737456
The_Farewell             with probability                0.8061348172162325
Ex_Machina               with probability                0.8110077037153245
Drive            with probability                0.8135583125758297
Les_Miserables           with probability                0.8305900131230936
Three_Billboards_Outside_Ebbing                  with probability
0.8355696399463284
Her              with probability                0.8443174598327123
The_Help                 with probability                0.8569637223990642
```

```
Darkest_Hour              with probability                0.8578888043063
Phantom_Thread            with probability                0.8685114372164059
Black_Swan                with probability                0.8772510177813874
The_Perks_of_Being_a_Wallflower                    with probability
0.8828651401863068
12_Years_a_Slave                  with probability
0.8833275717449787
```

10

```
Gone_Girl                 with probability                0.8914308215264958
Hidden_Figures            with probability                0.8921955396178919
Midnight_in_Paris                 with probability
0.9086545662854848
21_Jump_Street            with probability                0.9171455074076733
Django_Unchained                  with probability
0.9179525125592574
The_Theory_of_Everything                  with probability
0.9410104515628016
Shutter_Island            with probability                0.9618589620996809
Inception                 with probability                0.9802567271065036
```

Compared to the list in part (a), I think this list here represents the student's personal taste slightly better. We can see quite a few up & down of rankings in this list compared to that of part a. Some movies of science-thriller genre seem to be of high probability for this student, for instance.

(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.

Ans: Please find the source code & outputs in the following:

Homework 8 problem 1

November 27, 2022

1 Source Code and outputs for Problem 8.1_Hw8_CSE 250A_Fall 2022

[1]:
```
# part a: calculating mean popularity rating

import numpy as np

# loading the hw8_ratings matrix: calling it 'rat':
rat1=np.loadtxt("hw8_ratings.txt", dtype=str, delimiter=',')
rat2=[line.replace('\t', ' ') for line in rat1]
rat3=[line.split() for line in rat2]
rat=np.array(rat3)
(studno, movieno)=np.shape(rat)  # records number of students, no. of movies
#print(rat)
movierec=np.zeros(movieno)    # vector for number of students who has␣
↪recommended, for each movie
movieseen=np.zeros(movieno)    # vector for number of students who has seen,␣
↪for each movie
meanpopmov=np.zeros(movieno)    # vector for mean popularity rating, for each␣
↪movie
for j in range(movieno):
    rec=0
    seen=0
    for i in range(studno):
        if rat[i,j]=='1':
            rec=rec+1
            seen=seen+1
        elif rat[i,j]=='0':
            seen=seen+1
    meanpopmov[j]=rec/seen  # stores mean popularity rating for jth movie

# loading the movie titles
mvtitles=np.loadtxt("hw8_movies.txt", dtype=str)

#printing movie titles from least popular to most popular
print(f"The movies from least to most popular are: \n")
for k in np.argsort(meanpopmov):
```

```
    print(mvtitles[k])     # prints the movie titles from least to most mean␣
↪popularity
```

```
The movies from least to most popular are:

Chappaquidick
The_Last_Airbender
I_Feel_Pretty
Fifty_Shades_of_Grey
Fast_&_Furious:_Hobbs_&_Shaw
Hustlers
Magic_Mike
Bridemaids
World_War_Z
The_Shape_of_Water
Good_Boys
Prometheus
Pokemon_Detective_Pikachu
American_Hustle
Terminator:_Dark_Fate
The_Farewell
Man_of_Steel
Fast_Five
The_Hateful_Eight
Star_Wars:_The_Force_Awakens
The_Help
Rocketman
Drive
The_Girls_with_the_Dragon_Tattoo
Thor
Avengers:_Age_of_Ultron
Phantom_Thread
Us
The_Revenant
X-Men:_First_Class
Pitch_Perfect
Dunkirk
Ready_Player_One
Room
Jurassic_World
Mad_Max:_Fury_Road
Once_Upon_a_Time_in_Hollywood
Manchester_by_the_Sea
The_Perks_of_Being_a_Wallflower
Spiderman:_Far_From_Home
Her
Captain_America:_The_First_Avenger
Frozen
```

```
Hidden_Figures
The_Hunger_Games
Iron_Man_2
Les_Miserables
Toy_Story_3
Three_Billboards_Outside_Ebbing
Darkest_Hour
Ex_Machina
Gone_Girl
Black_Swan
12_Years_a_Slave
Avengers:_Endgame
The_Avengers
Midnight_in_Paris
The_Great_Gatsby
La_La_Land
Avengers:_Infinity_War
The_Theory_of_Everything
Now_You_See_Me
21_Jump_Street
Django_Unchained
The_Martian
Harry_Potter_and_the_Deathly_Hallows:_Part_1
Joker
Wolf_of_Wall_Street
The_Lion_King
Harry_Potter_and_the_Deathly_Hallows:_Part_2
Parasite
The_Social_Network
The_Dark_Knight_Rises
Shutter_Island
Interstellar
Inception
```

[2]:
```
# part e: implementation

import matplotlib.pyplot as plt
import copy


def floatlistconvert(strlist):    #function to convert a character list to a␣
 ↪float list, if applicable
    floatlist=[float(stringel) for stringel in strlist]
    return floatlist

# creating the set of movies seen (hence rated) by t th students
Omega=[ [] for _ in range(studno) ]
```

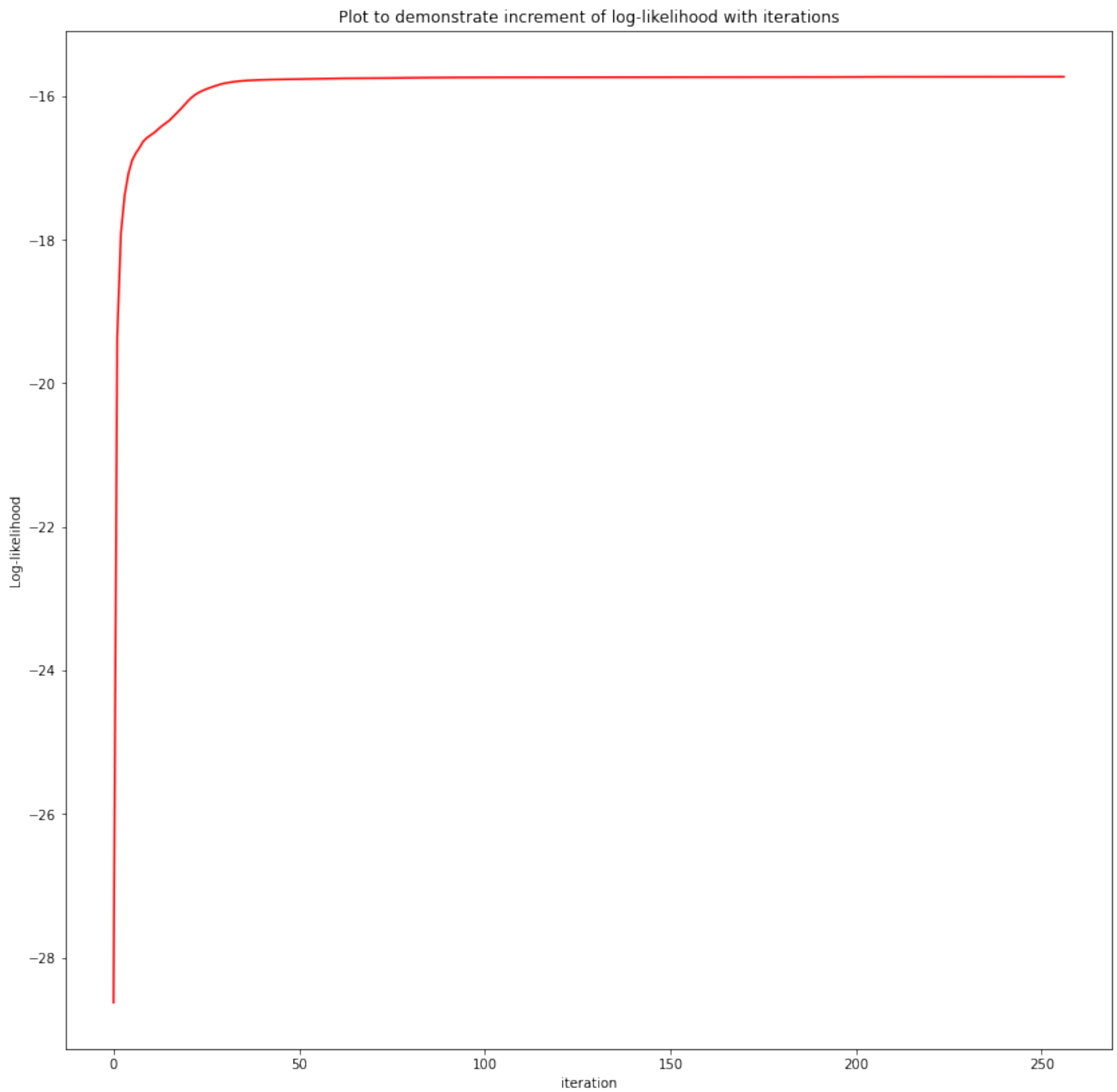
```
for t in range(studno):
    for i in range(movieno):
        if rat[t,i]=='1' or rat[t,i]=='0':
            Omega[t]=Omega[t]+[i]


#print(Omega)


# reading initial P(Z=i) and P(R_j=1|Z=i)
pzi=np.loadtxt("hw8_probZ_init.txt", dtype=float)  # note here first value of Z␣
 ↪is 0 (not 1), in this notation


pr1zi_1=np.loadtxt("hw8_probR_init.txt", dtype=str, delimiter=',')
pr1zi_2=[line.replace('\t', ' ') for line in pr1zi_1]
pr1zi_3=[line.split() for line in pr1zi_2]
pr1zi=np.array([floatlistconvert(item) for item in pr1zi_3])    # it is a␣
 ↪movieno. X k sized matrix
k=4
T=studno

def prodcondprob(t,i):
    prod=1
    for j in Omega[t]:
        if rat[t,j]=='1':
            prod=prod*pr1zi[j,i]
        elif rat[t,j]=='0':
            prod=prod*(1-pr1zi[j,i])
    return prod

def prj(t):
    s=0
    for i in range(k):
        p=pzi[i]*prodcondprob(t,i)
        s=s+p
    return s


def Estep(t):
    P=np.zeros(4)
    for i in range(k):
        P[i]=pzi[i]*prodcondprob(t,i)
    P=np.divide(P,prj(t))
    return P

# iteration 0
```

```python
L=np.zeros(257, dtype=float)

for t in range(T):
    L[0]=np.add(L[0],np.log(prj(t)))

L[0]=L[0]/T

print(f"For iteration 0, the log-likelihood is :{L[0]} \n")

# iteration 1 to 256
for itera in range(1, 257):
    r=np.zeros((k,T))   # r is the matrix rho here
    for t in range(T):
        r[:,t]=Estep(t)
    przcopy=copy.deepcopy(pr1zi)
    for i in range(k):
        pzi[i]=np.sum(r[i,:])/T
        for j in range(movieno):
            s=0
            for t in range(T):
                if j in Omega[t]:
                    if rat[t,j]=='1':
                        s=s+r[i,t]
                else:
                    s=s+r[i,t]*przcopy[j,i]
            pr1zi[j,i]=s/np.sum(r[i,:])

    for t in range(T):
        L[itera]=np.add(L[itera],np.log(prj(t)))

    L[itera]=L[itera]/T

for it in [1,2,4,8,16,32,64,128,256]:
    print(f"For iteration {it}, the log-likelihood is :{L[it]} \n")


# plot of log-likelihoods (with iterations) to see whether the increase or not:
itera=np.linspace(0, 256, num=257)

fig, ax = plt.subplots(figsize=(14, 14))
ax.plot(itera, L, color='r')

plt.xlabel("iteration")
plt.ylabel("Log-likelihood")
plt.title("Plot to demonstrate increment of log-likelihood with iterations")
```

```
For iteration 0, the log-likelihood is :-28.627324487337628
```

```
For iteration 1, the log-likelihood is :-19.350314946503318

For iteration 2, the log-likelihood is :-17.909564818017916

For iteration 4, the log-likelihood is :-17.081155562337013

For iteration 8, the log-likelihood is :-16.629824767528117

For iteration 16, the log-likelihood is :-16.28782872191562

For iteration 32, the log-likelihood is :-15.801537953970273

For iteration 64, the log-likelihood is :-15.749887678844292

For iteration 128, the log-likelihood is :-15.735940712575662

For iteration 256, the log-likelihood is :-15.728520329683299
```

[2]:
```
Text(0.5, 1.0, 'Plot to demonstrate increment of log-likelihood with
iterations')
```



[32]:
```
# part f: personal movie recommendation

import random

def Merge(dict_1, dict_2):           # creates a dictionary by merging two␣
↪dictionaries
        result = dict_1 | dict_2
        return result


######## loading student PID data
pids=np.loadtxt("hw8_ids.txt", dtype=str)
```

```
rand=random.randint(0, studno-1)   # selects a random number between all the␣
 ↪students

print(f"For my case (PID=A5327433), the output of the recommender system is␣
 ↪trivial because I have watched ALL the movies in the list.")
print(f"So as instructed in Piazza, I am randomly selecting (by a random number␣
 ↪generator) the PID {pids[rand]} from the list and recommending movies for␣
 ↪this id.")
#myindex=np.argwhere(pids=='A14763580')[0,0]    # finds out index/row number to␣
 ↪be found from the rating matrix using PID
myindex=np.argwhere(pids==pids[rand])[0,0]    # finds out index/row number to␣
 ↪be found from the rating matrix using PID
######## posterior probability of part(c)

postprob=Estep(myindex)
probdictnotseen={}
# expected ratings on movies I have not seen yet:
for l in range(movieno):
    if l not in Omega[myindex]:
        s=0
        for i in range(k):
            s=s+postprob[i]*pr1zi[l,i]    # s is the probability
        probdictnotseen=Merge(probdictnotseen,{l:s})

#print(probdictnotseen)

srteddictnotseen=dict(sorted(probdictnotseen.items(), key=lambda item:␣
 ↪item[1]))  # sorting the last dict above by values:ascending

#print(srteddictnotseen)

print(f"\n For PID {pids[myindex]}, the list of recommended movies among the␣
 ↪ones that the student has watched, are \n")
for l in range(movieno):
    if rat[myindex,l]=='1':
        print(mvtitles[l])

print(f"\n For PID {pids[myindex]}, the list of not-recommended movies among␣
 ↪the ones that the student has watched, movies are \n")
for l in range(movieno):
    if rat[myindex,l]=='0':
        print(mvtitles[l])

print(f"\n For PID {pids[myindex]}, the list of unseen movies sorted by their␣
 ↪expected ratings in ascending order\n")
for j in srteddictnotseen.keys():
```

```python
    print(f"{mvtitles[j]} \t \t with probability \t \t {probdictnotseen[j]}")
```

```
For my case (PID=A5327433), the output of the recommender system is trivial
because I have watched ALL the movies in the list.
So as instructed in Piazza, I am randomly selecting (by a random number
generator) the PID A59019917 from the list and recommending movies for this id.

 For PID A59019917, the list of recommended movies among the ones that the
student has watched, are

The_Last_Airbender
Harry_Potter_and_the_Deathly_Hallows:_Part_1
Iron_Man_2
Toy_Story_3
Fast_Five
Thor
Captain_America:_The_First_Avenger
Harry_Potter_and_the_Deathly_Hallows:_Part_2
Prometheus
The_Avengers
The_Dark_Knight_Rises
The_Hunger_Games
Wolf_of_Wall_Street
The_Great_Gatsby
Frozen
Now_You_See_Me
World_War_Z
Interstellar
Mad_Max:_Fury_Road
Jurassic_World
Avengers:_Age_of_Ultron
Room
The_Martian
Avengers:_Infinity_War
Ready_Player_One
Avengers:_Endgame
The_Lion_King
Joker
Parasite
Pokemon_Detective_Pikachu
Terminator:_Dark_Fate

 For PID A59019917, the list of not-recommended movies among the ones that the
student has watched, movies are

The_Social_Network
X-Men:_First_Class
Fifty_Shades_of_Grey
```

```
La_La_Land
Spiderman:_Far_From_Home
Fast_&_Furious:_Hobbs_&_Shaw

 For PID A59019917, the list of unseen movies sorted by their expected ratings
in ascending order

Chappaquidick            with probability                0.6213302052441917
Magic_Mike               with probability                0.6422617630243619
The_Hateful_Eight                with probability
0.7016135706894251
American_Hustle                  with probability
0.7045502369460868
The_Shape_of_Water               with probability
0.709683381852856
Bridemaids               with probability                0.7142694420643408
Star_Wars:_The_Force_Awakens             with probability
0.7336263473818456
Man_of_Steel             with probability                0.7403785362092967
Us               with probability                0.7410509024405416
Rocketman                with probability                0.7542065637770378
I_Feel_Pretty            with probability                0.7559210577238806
Once_Upon_a_Time_in_Hollywood            with probability
0.7596510691932159
Hustlers                 with probability                0.7648218353296454
Good_Boys                with probability                0.7913552362465591
Manchester_by_the_Sea            with probability
0.7929879515312204
The_Girls_with_the_Dragon_Tattoo                 with probability
0.7940073194473092
Dunkirk                  with probability                0.794897398744033
Pitch_Perfect            with probability                0.7961213640800816
The_Revenant             with probability                0.8043696029737456
The_Farewell             with probability                0.8061348172162325
Ex_Machina               with probability                0.8110077037153245
Drive            with probability                0.8135583125758297
Les_Miserables           with probability                0.8305900131230936
Three_Billboards_Outside_Ebbing                  with probability
0.8355696399463284
Her              with probability                0.8443174598327123
The_Help                 with probability                0.8569637223990642
Darkest_Hour             with probability                0.8578888043063
Phantom_Thread           with probability                0.8685114372164059
Black_Swan               with probability                0.8772510177813874
The_Perks_of_Being_a_Wallflower                  with probability
0.8828651401863068
12_Years_a_Slave                 with probability
0.8833275717449787
```

```
Gone_Girl               with probability                0.8914308215264958
Hidden_Figures          with probability                0.8921955396178919
Midnight_in_Paris               with probability
0.9086545662854848
21_Jump_Street          with probability                0.9171455074076733
Django_Unchained                with probability
0.9179525125592574
The_Theory_of_Everything                with probability
0.9410104515628016
Shutter_Island          with probability                0.9618589620996809
Inception               with probability                0.9802567271065036
```

[ ]: