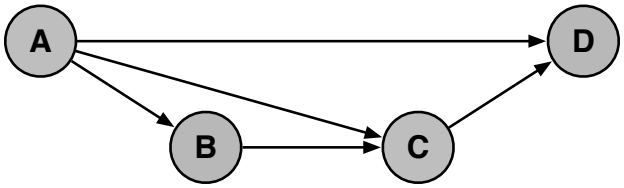
CSE 150. Assignment 5 *Fall 2018*

Out: *Tue Nov 6*
Due: *Tue Nov 13 (beginning of class)*

5.1 Maximum likelihood estimation



(a) **Complete data**

Consider a complete data set of *i.i.d.* examples $\{a_t, b_t, c_t, d_t\}_{t=1}^{T}$ drawn from the joint distribution of the above belief network. Compute the maximum likelihood estimates of the conditional probability tables (CPTs) shown below for this data set. Express your answers in terms of equality-testing functions, such as:

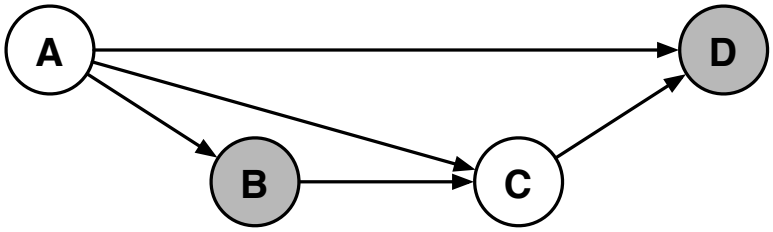
$$I(a, a_t) = \begin{cases} 1 \text{ if } a = a_t, \\ 0 \text{ if } a \neq a_t. \end{cases}$$

For example, in terms of this function, the maximum likelihood estimate for the CPT at node A is given by $P(A = a) = \frac{1}{T} \sum_{t=1}^{T} I(a, a_t)$. Complete the numerators and denominators in the below expressions.

$$P(B{=}b|A{=}a) \;=\; \frac{\qquad\qquad}{\qquad\qquad}$$

$$P(C{=}c|A{=}a, B{=}b) \;=\; \frac{\qquad\qquad}{\qquad\qquad}$$

$$P(D{=}d|A{=}a, C{=}c) \;=\; \frac{\qquad\qquad}{\qquad\qquad}$$



(b) **Posterior probability**

Consider the belief network shown above, with observed nodes B and D and hidden nodes A and C. Compute the posterior probability $P(a, c|b, d)$ in terms of the CPTs of the belief network—that is, in terms of $P(a)$, $P(b|a)$, $P(c|a, b)$ and $P(d|a, c)$.

(c) **Posterior probability**

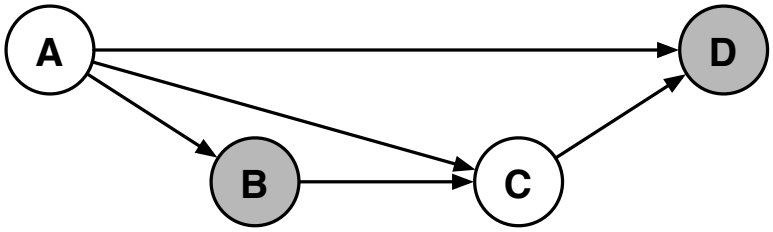
Compute the posterior probabilities $P(a|b, d)$ and $P(c|b, d)$ in terms of your answer from part (b). In other words, in this problem, you may assume that $P(a, c|b, d)$ is given.

(d) **Log-likelihood**

Consider a partially complete data set of *i.i.d.* examples $\{b_t, d_t\}_{t=1}^{T}$ drawn from the joint distribution of the above belief network. The log-likelihood of the data set is given by:

$$\mathcal{L} = \sum_t \log P(B = b_t, D = d_t).$$

Compute this log-likelihood in terms of the CPTs of the belief network. You may re-use work from earlier parts of the problem.



(e) **EM algorithm**

The posterior probabilities from part (b) can be used by an EM algorithm to estimate CPTs that maximize the log-likelihood from part (c). Complete the numerator and denominator in the below expressions for the EM update rules. Simplify your answers as much as possible, expressing them in terms of the posterior probabilities $P(a, c|b_t, d_t)$, $P(a|b_t, d_t)$, and $P(c|b_t, d_t)$, as well as the functions $I(b, b_t)$, and $I(d, d_t)$.

$$P(A=a) \leftarrow \text{———————————}$$

$$P(B=b|A=a) \leftarrow \text{———————————}$$

$$P(C=c|A=a, B=b) \leftarrow \text{———————————}$$

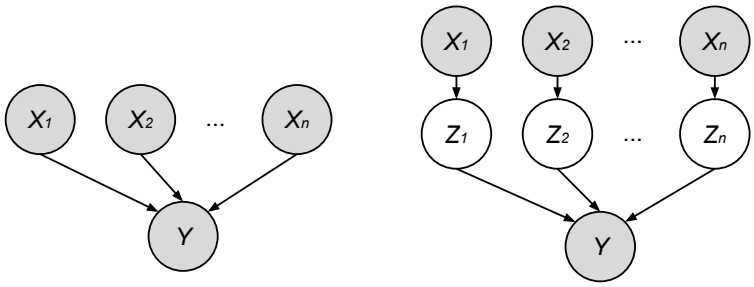
$$P(D=d|A=a, C=c) \leftarrow \text{———————————}$$

5.2 EM algorithm for noisy-OR

In this problem, we will see how the EM algorithm can be used to learn the parameters of a noisy-OR model. Part (a) is related to the derivation of the EM update rules for the noisy-OR model. Parts (b) through (d) will ask you to write code to learn the parameters of a noisy-OR model based on a particular data set.

(a) **Equivalence of models**

The following diagrams depict two variations of the noisy-OR model:



Model A **Model B**

For this part of the problem, we will use the notation $P_A(\cdots)$ and $P_B(\cdots)$ to distinguish between probabilities computed using Model A and Model B, respectively.

Model A is a standard noisy-OR model in which the CPT for Y is

$$P_A(Y = 1|\vec{X} = \vec{x}) = 1 - \prod_{i=1}^{n}(1 - p_i)^{x_i}.$$

Model B is a variation of the noisy-OR model in which we've inserted a layer of hidden variables $Z_1, \ldots, Z_n$, with the following CPTs:

$$P_B(Z_i = 1|X_i = x_i) = \begin{cases} p_i, & \text{if } x_i = 1 \\ 0, & \text{if } x_i = 0 \end{cases}$$

$$P_B(Y = 1|\vec{Z} = \vec{z}) = \begin{cases} 1, & \text{if } Z_i = 1 \text{ for any } i \\ 0, & \text{if } Z_i = 0 \text{ for all } i \end{cases}$$

In contrast to Model A, the Y node in Model B is a deterministic OR, since it will be 1 if and only if *any* of the Z_i's are 1. We can view the Z_i nodes as being a sort of "noisy copy" of the corresponding X_i nodes: if $X_i = 0$, then Z_i is guaranteed to be 0 as well, but if $X_i = 1$, then Z_i has probability p_i of being 1.

Note also that both models are defined in terms of parameters p_i for $i \in \{1, \ldots, n\}$.

To prove that the Y nodes in both models are equivalent, it is enough to show that

$$P_A(Y = 0|\vec{X} = \vec{x}) = P_B(Y = 0|\vec{X} = \vec{x}).$$

The following equations are an incomplete proof of this fact. To complete this proof, provide a brief justification for each of the steps labeled (i) through (iv).

$$\begin{aligned} P_B(Y = 0|\vec{X} = \vec{x}) &= \sum_{\vec{z}} P_B(Y = 0, \vec{Z} = \vec{z} \,|\, \vec{X} = \vec{x}) && \text{(i)} \\ &= \sum_{\vec{z}} \left[P_B(Y = 0 \,|\, \vec{Z} = \vec{z}) \prod_{i=1}^{n} P_B(Z_i = z_i \,|\, X_i = x_i) \right] && \text{(ii)} \\ &= \prod_{i=1}^{n} P_B(Z_i = 0 \,|\, X_i = x_i) && \text{(iii)} \\ &= \prod_{i=1}^{n} (1 - p_i)^{x_i} && \text{(iv)} \end{aligned}$$

(Because of this result, we can simply use the notation $P(\cdots)$ for probabilities in the remaining parts of this problem, since we have shown that the the two models $P_A(\cdots)$ and $P_B(\cdots)$ agree.)

(b) **EM Implementation: Per-iteration statistics**

Suppose we are given a data set of T samples $\{(\vec{x}_t, y_t)\}_{t=1}^{T}$ from Model A.

If we try to apply standard maximum-likelihood techniques to estimate the values of the parameters p_i based on Model A, we will find that there is no closed-form solution.

However, since part (a) showed that Model A and Model B are equivalent, we can instead view the data as a partially-observed data set corresponding to Model B. This allows us to estimate the values of p_i using the EM algorithm.

Noting that p_i corresponds to $P(Z_i = 1|X_i = 1)$, which is one of the CPT entries for Model B, we can write the EM update rule as:

$$\begin{aligned} p_i &\leftarrow \frac{\widehat{\text{count}}(Z_i = 1, X_i = 1)}{\widehat{\text{count}}(X_i = 1)} \\ &= \frac{\sum_t P(Z_i = 1, X_i = 1|\vec{X} = \vec{x}_t, Y = y_t)}{\sum_t P(X_i = 1|\vec{X} = \vec{x}_t, Y = y_t)} \end{aligned}$$

Then, as shown in class, the EM update rule for noisy-OR can be simplified to the following:

$$p_i \leftarrow \frac{1}{T_i} \sum_{t=1}^{T} \frac{y_t x_{it} p_i}{1 - \prod_{j=1}^{n} (1 - p_j)^{x_{jt}}},$$

where $T_i = \sum_t x_{it}$ is the number of observations where $X_i = 1$.

Next, you will use the EM algorithm for estimating the noisy-OR parameters p_i. Download the data files *hw5_noisyOr_x.txt* and *hw5_noisyOr_y.txt* for this homework assignment. The data set has $T = 267$ examples over $n = 23$ inputs.[1]

For a data set $\{(\vec{x}_t, y_t)\}_{t=1}^{T}$, the normalized log (conditional) likelihood is given by:

$$\mathcal{L} = \frac{1}{T} \sum_{t=1}^{T} \log P(Y = y_t | X = \vec{x}_t).$$

In your code, you should initialize all $p_i = 0.05$ and perform 512 iterations of the EM algorithm using the update rule shown above. At each iteration, compute the log conditional likelihood shown above. (If you have implemented the EM algorithm correctly, this log conditional likelihood will always increase from one iteration to the next.)

Also compute the number of mistakes made by the model at each iteration; a mistake occurs either

- when $y_t = 0$ and $P(Y = 1 | \vec{X} = \vec{x}_t) \geq 0.5$ (indicating a *false positive*), or
- when $y_t = 1$ and $P(Y = 1 | \vec{X} = \vec{x}_t) < 0.5$ (indicating a *false negative*).

The number of mistakes should generally decrease as the model is trained, though it is not guaranteed to do so at each iteration.

For this part, turn in a completed version of the following table:

iteration	number of mistakes M	log conditional likelihood $\mathcal{L}$
0	175	-0.9581
1	56	
2		-0.4082
4		
8		
16		
32		
64		
128	36	
256		
512		-0.3100

You should use the already completed entries of this table to check your work. As always you may program in the language of your choice.

(c) **EM Implementation: Estimated values for p_i**

Produce a table or bar plot of your final estimated values for p_i.

(d) **EM Implementation: Source code**

Include your source code for this problem as part of your Gradescope submission.

[1] For those interested, more information about this data set is available here:
`http://archive.ics.uci.edu/ml/machine-learning-databases/spect/SPECT.names`

Debugging hints: If you are having trouble getting your results to match starting at iteration 1, try printing out the values of p_1 and p_{23} after the first few iterations. After iteration 1, you should find that $p_1 \approx 0.11655$ and $p_{23} \approx 0.15310$. If you find that your value of p_1 matches but p_{23} does not, this is likely a sign that you are using the updated values for p_i too early. To solve this, you should save the new values for p_i in temporary variables until $p_1, p_2, \ldots, p_{23}$ have all been computed; then, overwrite the old values of p_i all at once.

5.3 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 *hw5_movieTitles_fa18.txt* and *hw5_movieRatings_fa18.txt*. The second file contains a 133×60 matrix of zeros, ones, and missing elements denoted by question marks. The $(i, j)^{\text{th}}$ 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 recommended,
?	not seen.

(a) **Sanity check**

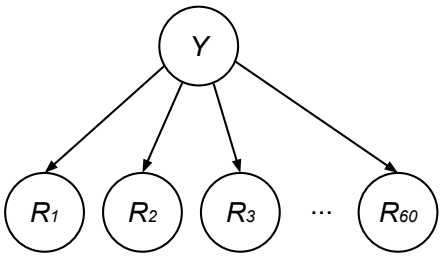
Compute the mean popularity rating of each movie, given by the simple ratio

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

and sort the movies by this ratio. Print out the movie titles from least popular (*The Last Airbender*) to most popular (*Doctor Strange*).

(b) **Likelihood**

Now you will learn a naive Bayes model of these movie ratings, represented by the belief network shown below, with hidden variable $Y \in \{1, 2, \ldots, k\}$ 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 $P(Y=i)$ of the overall population—likes the j^{th} movie with conditional probability $P(R_j=1|Y=i)$. 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

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

(c) **E-step**

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

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

(d) **M-step**

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

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

denote the probabilities computed in the E-step of the algorithm. Also, let $T = 133$ denote the number of students. Show that the EM updates are given by

$$\begin{aligned} P(Y=i) \;&\leftarrow\; \frac{1}{T}\sum_{t=1}^{T}\rho_{it}, \\ P(R_j=1|Y=i) \;&\leftarrow\; \frac{\sum_{t:j\in\Omega_t}\rho_{it}\,I\left(r_j^{(t)},1\right)+\sum_{t:j\notin\Omega_t}\rho_{it}\,P(R_j=1|Y=i)}{\sum_{t=1}^{T}\rho_{it}}. \end{aligned}$$

Here, $\sum_{t:j\in\Omega_t}$ denotes a sum over all students t that rated movie j. Similarly, $\sum_{t:j\notin\Omega_t}$ denotes a sum over all students t that did *not* rate movie j.

(e) **Implementation**

Download the files *hw5_probType_init_fa18.txt* and *hw5_probRatingGivenType_init_fa18.txt*, and use them to initialize the probabilities $P(Y=i)$ and $P(R_j=1|Y=i)$ for a model with $k=4$ types of movie-goers. Run 128 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. Your log-likelihood should increase (i.e., become less negative) at each iteration. Fill in the following table, using the already provided entries to check your work:

iteration	log-likelihood $\mathcal{L}$
0	-27.9848
1	-15.5730
2	
4	
8	
16	-12.0195
32	
64	
128	

Note: Choosing $k = 4$ is somewhat arbitrary, but it is based on assuming that there are a small number of fundamentally different movie-goer types. Moreover, the initial values for $P(R_j = 1 | Y = i)$ in *hw5_probRatingGivenType_init_fa18.txt* have been chosen randomly. If you are interested in exploring further after completing this assignment, feel free to try out other initializations and other values of k.

(f) **Personal categorization**

Using the formula from part (c) along with the parameters you estimated in part (e), evaluate the probability distribution over movie-goer types (i.e., over Y) given your own movie ratings. Include either a table or a plot of $P(Y = i \,|\, \text{your ratings})$ for $i \in \{1, \ldots, k\}$.

What value of $i \in \{1, \ldots, k\}$ maximizes the value of $P(Y = i \,|\, \text{your ratings})$? (In other words, which of the k types of movie-goers do you most closely match?)

Note: The movie ratings dataset does not include names or PIDs, so you should hard-code your own ratings based on the results that you saved when you took the survey, or based on your own memory.

(g) **Personal movie recommendations**

Next, compute your *expected* ratings on the movies *you haven't yet seen*. You can do this by applying the following formula for each movie ℓ that you haven't seen:

$$P\left(R_\ell = 1 \mid \text{your ratings}\right) = \sum_{i=1}^{k} P\left(Y = i \mid \text{your ratings}\right) P(R_\ell = 1 | Y = i)$$

Print out the list of these (unseen) movies 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).

(h) **Source code**

Include your source code for this problem as part of your Gradescope submission. As usual, you may program in the language of your choice.
