UNIVERSITY OF MICHIGAN
Department of Electrical Engineering and Computer Science
EECS 445 Introduction to Machine Learning
Winter 2019

Homework 4, Due: Tue. 04/16 at 11:59pm

Submission: Please upload your completed assignment by 11:59pm ET on Tuesday, April 16 to Gradescope. Include all code as an appendix following your write-up.

1 Expectation Maximization [15 pts]

Suppose we have three cities A, B and C each with a probability of having sunny weather, θ_A, θ_B and θ_C, unknown to us. To predict the weather of each city, we conduct an experiment. The experiment consists of N trials. For each trial, we choose one city from A, B and C uniformly at random, detect the weather of the chosen city over M days, and record the sequence of sunny / rainy. On the i-th trial, we define:

- $z^{(i)}$ is the city chosen for this trial, $z^{(i)} \in \{A, B, C\}$
- $x_j^{(i)}$ is the j-th detection for this trial, $x_j^{(i)} \in \{\texttt{S}, \texttt{R}\}$
- s_i is the number of sunny days recorded on this trial.

The data are generated as follows:

for trial $i \in \{1, \ldots, N\}$:
  $z^{(i)} \sim \text{Uniform}\{A, B, C\}$
  for detection $j \in \{1, \ldots, M\}$:
    $x_j^{(i)} \sim \text{Bernoulli}(\theta_{z^{(i)}})$, where $\theta_{z^{(i)}}$ is the probability of sunny.

The results of our experiment (with $N = 6$ trials and $M = 10$ detections per trial) are shown below.

Trial i	City $z^{(i)}$	Detection $\bar{x}^{(i)} = [x_1^{(i)}, \ldots, x_M^{(i)}]$
1	A	SSSSSSRRSS
2	B	SRRRRSRSRR
3	C	RSSRSRRSRS
4	B	RSRRRRRRRS
5	C	SRRSSRRSSS
6	A	SSSRRSSRSS

1.0 Visualizing Bayesian Networks

(a) (1 pt) **Draw** the associated Bayesian Network describing this generative model (using plate notation).

(b) (1 pt) **What** are the learnable parameters associated with this generative model?

(c) (1 pt) **What** are the latent variables of this generative model? **What** do they represent?

1.1 Complete-Data Case

In this section, we consider the scenario in which we know both the resulting sequences and which city was chosen on each trial.

(a) (2 pt) **Write down** an expression for the **complete-data log-likelihood** $\log p(\bar{x}^{(i)}, z^{(i)}; \bar{\theta})$ of a **single trial**, in terms of s_i, M, and $\theta_{z^{(i)}}$.

(b) (1 pt) Given the data from all N trials $\mathcal{X} = \{\bar{x}^{(i)}\}_{i=1}^{N}$ and $\mathcal{Z} = \{z^{(i)}\}_{i=1}^{N}$, **write down** an expression for the complete-data log-likelihood $\log p(\mathcal{X}, \mathcal{Z}; \bar{\theta})$.

(c) (2 pt) Using the complete-data log-likelihood, **derive** expressions for the maximum-likelihood estimates of θ_A, θ_B and θ_C and **report** estimates of θ_A, θ_B and θ_C for the sample data. In words, **describe** what these estimates correspond to.

1.2 Classification-Maximization

Now, assume the city indicators $z^{(i)}$ are unobserved (i.e., the city column in the table at the beginning of this problem is not available). That is, assume our experiment works as follows: you are a student stuck in BBB doing the 445 project. Your teammate decides to go to the three cities A, B and C with equal probability but he will not tell you which city he goes to. On each trial, he randomly chooses a city, detects its weather M times and records the sequence. The data are now "incomplete."

The **Classification-Maximization** algorithm estimates the model parameters by alternating between two steps: 1) Determining trial city assignment, 2) Maximizing likelihood. More precisely, it alternates between the following two steps after randomly initializing the parameters $\bar{\theta}^{(0)}$:

C-step: Compute $\hat{z}^{(i)}_{(t+1)} = \underset{k \in \{A,B,C\}}{\arg\max}\ p(z^{(i)} = k \mid \bar{x}^{(i)}; \bar{\theta}^{(t)})$ for $i = 1, \ldots, N$

(ties broken arbitrarily)

M-step: Update $\bar{\theta}^{(t+1)} = \underset{\bar{\theta}}{\arg\max}\, p(\bar{x}^{(i)}, \hat{z}^{(i)}_{(t+1)}; \bar{\theta})$

(maximum likelihood estimation given fixed city assignments)

For this section, your task is to derive a Classification-Maximization procedure for the city detection experiment.

(a) (2 pt) **Write down** an expression for $p(z^{(i)} = k \mid \bar{x}^{(i)}; \bar{\theta}^{(t)})$, the probability that trial i in city $k \in \{A, B, C\}$ given the data $\bar{x}^{(i)}$ and the current parameter estimate $\bar{\theta}^{(t)}$. Express your answer in terms of s_i, M and the model parameters $\theta_A^{(t)}, \theta_B^{(t)}, \theta_C^{(t)}$.

(b) (2 pt) Perform **three** iterations of Classification-Maximization for the sample data (either with code or by hand) and report your values of $p(z^{(i)} = k \mid \bar{x}^{(i)}, \bar{\theta}^{(t)})$, $\hat{z}_{(t)}^{(i)}$, and $\bar{\theta}^{(t)}$ for each step. Initialize $\bar{\theta}^{(0)} = (0.7, 0.4, 0.6)$. In the case of a tie, choose the **lexicographically smaller** city, i.e., $A < B < C$.

1.3 Expectation-Maximization

As you may have noticed, Classification-Maximization struggles to make a decision when there is more than one plausible assignment for an observed sequence, and has no way of expressing the level of confidence in which city was chosen for a sequence. The **Expectation-Maximization** algorithm overcomes this limitation. Instead of using fixed assignments, it computes the probability $p(z^{(i)} = k \mid \bar{x}^{(i)}; \theta)$ that each point $\bar{x}^{(i)}$ belongs to each distribution $z^{(i)} = k \in \{A, B, C\}$.

In discussion and in the above questions, we saw it was possible to write the complete-data log-likelihood with indicator variables that indicate which point belongs to which distribution. Now we can replace the indicator variables in the log-likelihood function with the probabilities above. This is analogous to going from *hard-assignments* to *soft-assignments*.

In this case, the Expectation-Maximization algorithm repeats the following steps until convergence.

E-Step. Evaluate $p(z^{(i)} = k \mid \bar{x}^{(i)}; \bar{\theta}^{(t)})$ for each $i = 1, \ldots, N$ and each $k \in \{A, B, C\}$

M-Step. Update $\bar{\theta}^{(t+1)} =$
$$\arg\max_{\bar{\theta}} \sum_{i=1}^{N} \sum_{k \in \{A,B,C\}} p(z^{(i)} = k \mid \bar{x}^{(i)}; \bar{\theta}^{(t)}) \log \left(\tfrac{1}{3} (\theta_k)^{s_i} (1 - \theta_k)^{M - s_i} \right)$$

Now, you will derive an instance of the Expectation-Maximization algorithm for our city detection experiment.

(a) (2 pt) Using the equations above, **show** that the updates for θ_k is as follows:

$$\theta_k^{(t+1)} = \frac{\sum_{i=1}^{N} p(z^{(i)} = k \mid \bar{x}^{(i)}; \theta^{(t)}) \times s_i}{M \times \sum_{i=1}^{N} p(z^{(i)} = k \mid \bar{x}^{(i)}; \theta^{(t)})}$$

(b) (1 pt) **Describe** in words how the parameters above are updated; that is, what are these equations doing? Do they correspond to your intuition?

2 EM for Literary Creation [5 pts]

As we saw above, the EM algorithm is a useful method for estimating the parameters of a distribution. How can we apply this algorithm to creating literature? Suppose we are interested in learning the underlying generative model for news articles.

We hypothesize that each word in an article is generated based on the previous word. However, since we have limited data for estimating the model, we assume the current word is assigned to a cluster (denoted by $z \in \{1, \ldots, k\}$), which then determines the next word. In other words, given w_i (current word), we will generate w_j (next word), according to:

$$P(w_j|w_i) = \sum_{z=1}^{k} P(w_j|z)P(z|w_i) = \sum_{z=1}^{k} \theta_{w_j|z}\phi_{z|w_i}$$

where $P(w_j|z) = \theta_{w_j|z}$ and $\sum_{w\in\mathcal{W}} \theta_{w|z} = 1$ for any cluster $z = 1, ..., k$. Similarly, $P(z|w_i) = \phi_{z|w_i}$ and $\sum_{z=1}^{k} \phi_{z|w_i} = 1$ for any current word $w_i \in \mathcal{W}$. Here $\mathcal{W}$ denotes the set of possible words. This means that the probability of a word given the previous word is independent given cluster z. Our model is parameterized by two probability tables, $\theta_{w|z}$ and $\phi_{z|w}$.

(a) (2 pt) Suppose we observe $w_1 =$ "machine" (the current word) and $w_2 =$ "learning" (the next word). **Derive** an expression for the posterior probability of the current cluster $z = 1$ in terms of θ and ϕ given this information.

(b) (2 pt) We will use the EM algorithm to maximize the log-likelihood of generating the data. In our case, based on a sequence of words $w_1, ..., w_T$ (i.e., an article), we aim to predict the underlying model generating the article. **Derive** an expression for the **log**-likelihood of words $w_2, ..., w_T$ given w_1 in terms of the parameters θ and ϕ.

(c) (1 pt) In the EM algorithm, the M-step updates parameters θ and ϕ based on estimates of the posterior evaluated in the E-step. The M-step update for θ is:

$$\theta_{w|z} = \frac{\sum_{t=1}^{T-1} p(z_t = z|w_t, w_{t+1})[[w_{t+1} == w]]}{\sum_{w'\in\mathcal{W}} \sum_{t=1}^{T-1} p(z_t = z|w_t, w_{t+1})[[w_{t+1} == w']]}$$

Explain how this equation updates $\theta_{w|z}$ in words. **Does** it correspond to your intuition?

3 Soft Clustering using GMM [7 pts]

When fitting a model, adding more parameters can possibly increase the likelihood of data, but doing so may result in overfitting. In order to control the model size, we use the Bayesian Information Criteria (BIC):

$$\text{BIC} = m \ln N - 2 \ln \hat{\mathcal{L}}$$

where m is the number of independent parameters, N is the number of data points, and $\hat{\mathcal{L}}$ is the maximized likelihood of data. We aim to minimize the BIC.

For this problem, you will implement the EM algorithm for GMM and interpret the results. We will use the iris classification dataset, for which you may view details here: https://archive.ics.uci.edu/ml/datasets/iris.

We have provided you with skeleton code including functions to access the data. You will need to modify `gmm.py` and `run.py` as noted in the comments with `TODO` tags. If you encounter an error claiming that the `load_iris()` function was not found, please follow the instructions in 3.1 to update `scikit-learn`. Include answers to the following questions in your write-up.

(a) (1 pt) Note that the code assumes all K Gaussians in the mixture to have a shared spherical diagonal covariance matrix $\sigma^2\mathbf{I}$. The likelihood of a point $\bar{x} \in \mathbb{R}^d$ in this case is:

$$p(\bar{x}|\theta) = \sum_{k=1}^{K} \gamma_k \mathcal{N}(\bar{x}; \bar{\mu}^{(k)}, \sigma^2\mathbf{I})$$

Write down the number of independent parameters corresponding to the mixture model.

(b) (3 pt) Implement the missing portions of `gmm.py`. The parts which you need to complete are the estimation functions for cluster probabilities for every point (`zk` matrix in code), GMM parameters and the Bayesian Information criteria (BIC).

Here, z_{ik} is the probability that the ith data point belongs to cluster k.

You will use these update rules in your implementation in `gmm.py`.

$$\gamma_k = \frac{1}{N}\sum_{i=1}^{N} z_{ik}$$

$$\bar{\mu}^{(k)} = \frac{1}{\sum_{i=1}^{N} z_{ik}}\sum_{i=1}^{N} z_{ik}\bar{x}^{(i)}$$

$$\sigma^2 = \frac{1}{Nd}\sum_{i=1}^{N}\sum_{k=1}^{K} z_{ik}(\bar{x}^{(i)} - \bar{\mu}^{(k)})^T(\bar{x}^{(i)} - \bar{\mu}^{(k)})$$

Note: You should first obtain the log-likelihood, and then obtain the likelihood/probability from that quantity. We encourage you to think about why it is that we might want to calculate likelihood in this way (this is more about issues with computers rather than with Machine Learning), but we won't ask you for this explanation on the write up.
Important implementation detail: Whenever you are taking the log of the sum of exponentials, you could run into issues due to the limited precision of computers. To obtain stable results, you should use the logsumexp trick. We imported the `logsumexp` function from the `scipy` package for you, which you should use to replace any log of the sum of exponentials. Refer to https://en.wikipedia.org/wiki/LogSumExp or https://www.xarg.org/2016/06/the-log-sum-exp-trick-in-machine-learning/ for more details.

(c) (2 pt) In the `run.py` script, you have to train the GMM using integer values of hyperparameter k between 2 and 8 inclusive. Since EM is dependent on initialization, for each cluster size, you will have to run the `gmm` function multiple times to find the best BIC value which is achievable. While comparing results of various runs for each number of clusters, you'll have to choose the *best* according to the BIC value. Explain (1) how you will compare two BIC values (lower vs. higher), and (2) how you will choose the clusters. Be sure to explain your reasoning for these choices.

(d) (1 pt) Show the plot of the Bayesian Information Criteria (BIC) as a function of the number of clusters K. Describe the shape of the graph and write down your interpretation of the trend. Given this plot of BIC values, what is the number of clusters you'll choose? Include the plot in your writeup as well.
Note: Significant variation in this plot is expected due to the random initialization.

3.1 Code Prerequisites

- If you are currently using `scikit-learn` version 0.19.1 (or later), you should be sufficiently up-to-date. You can verify this by running `conda list scikit-learn`
- If not, please run `conda update scikit-learn` to update

4 Bayesian Networks [5 pts]

Instead of using HMM's, we will be looking at an alternative Bayesian Network that tries to solve the same problem of hidden variables and observations. The network is as follows:

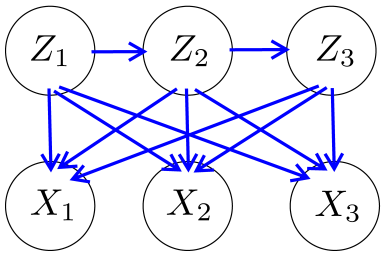

Figure 2: Alternative to HMM

(a) (2 pt) Check the following independence conditions of the graph:

- Are X_1, X_2, and X_3 all independent of each other given Z_1 and Z_2 (i.e., $X_i \perp\!\!\!\perp X_j | \{Z_1, Z_2\}$ for all i and j, where $i \neq j$)?
- Is Z_3 independent of Z_1 given Z_2?

(b) (1 pt) Write down the factored joint probability distribution associated with the Bayesian Network described in part a.

(c) (1 pt) If the random variables are all binary, how many parameters are there in the joint probability distribution implied by your graph?

(d) (1 pt) Suppose we learned the parameters for the above model, in addition to the parameters for an HMM (in which the Z's represent hidden states, and the X's represent observations). Applied to the training data both models achieve the same log-likelihood. Which model should we use? Briefly explain your answer.

5 Another Bayesian Network [3 pts]

Consider the following Bayesian Network.

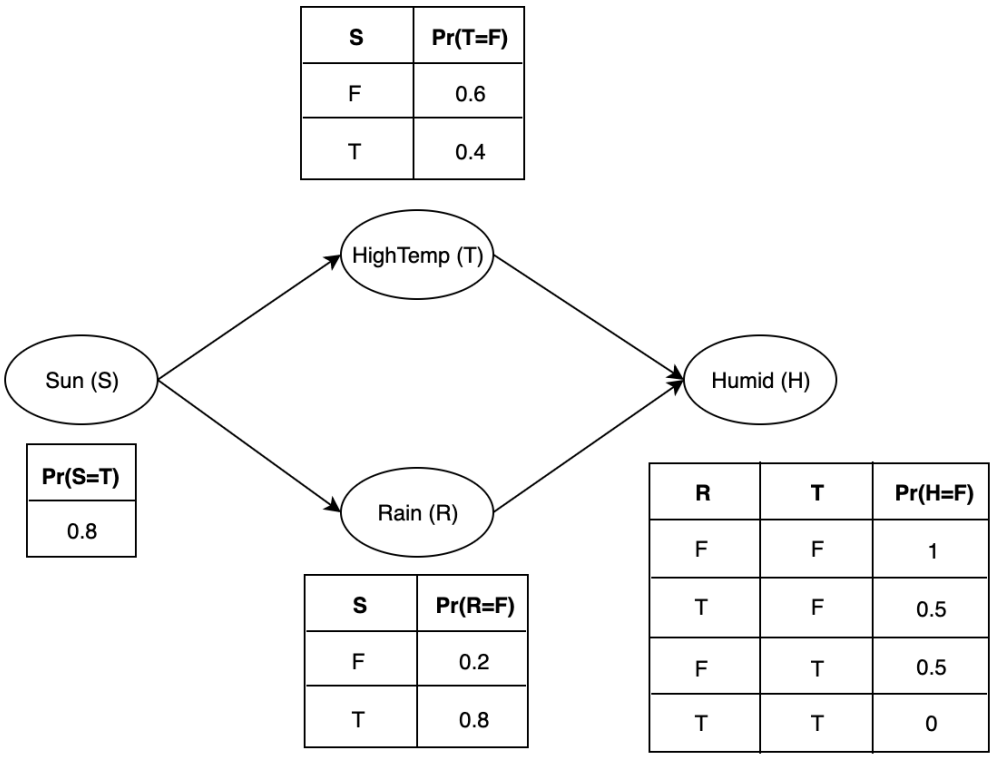

Figure 3: Bayesian network for humidity

(a) (1 pt) Write down the factored joint probability distribution associated with this Bayesian Network.

(b) (2 pt) Suppose we observe that it is humid (i.e., H=T), what is the probability that it is raining?

6 Collaborative Filtering [10 pts]

We can think of collaborative filtering as matrix factorization. Any $n \times m$ matrix X can be approximated by a lower rank matrix UV^T where U is $n \times d$ and V is $m \times d$ and $d \leq \min\{m, n\}$. Let $\bar{u}^{(i)} \in \mathbb{R}^d$ be the i^{th} row vector of U and $\bar{v}^{(j)}$ be the j^{th} row vector of V. Then $[UV^T]_{ij} = \bar{u}^{(i)} \cdot \bar{v}^{(j)}$. We want to use this low rank matrix to approximate observed binary labels in Y. Let $Y_{ij} \in \{-1, 1\}$ if the (i, j) entry is observed, and $Y_{ij} = 0$ otherwise. Then we would like to find U and V such that:

$$Y_{ij}[UV^T]_{ij} = Y_{ij}(\bar{u}^{(i)} \cdot \bar{v}^{(j)}) > 0 \text{ whenever } Y_{ij} \neq 0$$

Note that this is trivially possible if $X = UV^T$ is full rank (each element can be chosen independently from each other). The smaller d we choose, the fewer adjustable parameters we have in U and V. Therefore, d controls the complexity of the solution that satisfies the constraints.

It is often advantageous to potentially avoid satisfying all the constraints, e.g., if some of the labels may be mistakes. We might also suspect that we gain something by turning the estimation problem into a maximum

margin problem, for $\bar{u}^{(i)}$'s given $\{\bar{v}^{(j)}\}$ and for $\bar{v}^{(j)}$'s given $\{\bar{u}^{(i)}\}$. The formulation with slack is given by:

$$\min_{U,V,\xi} \quad \sum_{i=1}^{n} \frac{1}{2}\|\bar{u}^{(i)}\|^2 + \sum_{j=1}^{m} \frac{1}{2}\|\bar{v}^{(j)}\|^2 + C \sum_{i,j:Y_{ij}\neq 0} \xi_{ij}$$

$$\text{subject to} \quad Y_{ij}(\bar{u}^{(i)} \cdot \bar{v}^{(j)}) \geq 1 - \xi_{ij},$$

$$\xi_{ij} \geq 0 \text{ for all } (i,j) \text{ where } Y_{ij} \neq 0$$

(a) (2 pt) If we fix U, i.e., fix all $\bar{u}^{(1)}, ..., \bar{u}^{(n)}$ the optimization problem reduces to m independent optimization problems. Define these optimization problems for $\bar{v}^{(j)}$ where $j = 1, ..., m$. What do these optimization problems correspond to?

(b) (1 pt) If we fix V, what does the optimization problem reduce to?

(c) (1 pt) Suppose we have no observations in the i^{th} row of Y. In other words, $Y_{ij} = 0$ for all $j = 1, ..., m$. If we fix U, will adding such an empty row affect what we get for $\bar{v}^{(j)}$?

(d) (2 pt) Suppose there's only one observation in the i^{th} row. In other words, $Y_{ij} = 1$ for exactly one j (the rest are zeros). Assume no slack ($C = \infty$). Solve for $\bar{u}^{(i)}$ in terms of $\bar{v}^{(j)}$ Note that the resulting $\bar{u}^{(i)}$ specifies a maximum margin boundary through the origin. (Hint: try solving this geometrically.)

(e) (4 pt) Consider a simple two-user and two-movie example and let the observed rating matrix and our initial guess for V be given by:

$$Y = \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \quad V = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Assume no slack variables for this example.

i. Solve by hand $\hat{U} = [\bar{u}^{(1)}, \bar{u}^{(2)}]^T$, given the initial V. Given this V and your solution $\hat{U}$, can you predict Y_{22}?

ii. Solve by hand $\hat{V} = [\bar{v}^{(1)}, \bar{v}^{(2)}]^T$, given your solution $\hat{U}$. Can you predict Y_{22} now? If so, what is the resulting prediction?

REMEMBER to submit your completed assignment to Gradescope by 11:59pm ET on Tuesday, April 16. Include all code as an appendix following your write-up.