CS181-S22

Assignment #5

Due: 11:59pm EST, April 8, 2021

Homework 5: EM with Mixtures, PCA, and Graphical Models

This homework assignment will have you work with EM for mixtures, PCA, and graphical models. We encourage you to read sections 9.4 and 8.2.5 of the course textbook.

Please type your solutions after the corresponding problems using this LaTeX template, and start each problem on a new page.

Please submit the **writeup PDF to the Gradescope assignment 'HW5'**. Remember to assign pages for each question.

Please submit your **LaTeX file and code files to the Gradescope assignment 'HW5 - Supplemental'**.

Problem 1 (Expectation-Maximization for Gamma Mixture Models, 25pts)

In this problem we will explore expectation-maximization for a Categorical-Gamma Mixture model.

Let us suppose the following generative story for an observation x: first one of K classes is randomly selected, and then the features x are sampled according to this class. If

$$z \sim \text{Categorical}(\boldsymbol{\theta})$$

indicates the selected class, then x is sampled according to the class or "component" distribution corresponding to z. (Here, $\boldsymbol{\theta}$ is the mixing proportion over the K components: $\sum_k \theta_k = 1$ and $\theta_k > 0$). In this problem, we assume these component distributions are gamma distributions with shared shape parameter but different rate parameters:

$$x|z \sim \text{Gamma}(\alpha, \beta_k).$$

In an unsupervised setting, we are only given a set of observables as our training dataset: $\mathcal{D} = \{x_n\}_{n=1}^N$. The EM algorithm allows us to learn the underlying generative process (the parameters $\boldsymbol{\theta}$ and $\{\beta_k\}$) despite not having the latent variables $\{z_n\}$ corresponding to our training data.

1. **Intractability of the Data Likelihood** We are generally interested in finding a set of parameters β_k that maximizes the likelihood of the observed data:

   $$\log p(\{x_n\}_{n=1}^N; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K).$$

   Expand the data likelihood to include the necessary sums over observations x_n and to marginalize out the latents $\mathbf{z}_n$. Why is optimizing this likelihood directly intractable?

2. **Complete Data Log Likelihood** The complete dataset $\mathcal{D} = \{(x_n, \mathbf{z}_n)\}_{n=1}^N$ includes latents $\mathbf{z}_n$. Write out the negative complete data log likelihood:

   $$\mathcal{L}(\boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) = -\log p(\mathcal{D}; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K).$$

   Apply the power trick and simplify your expression using indicator elements z_{nk}.[a] Notice that optimizing this loss is now computationally tractable if we know $\mathbf{z}_n$.

   (Continued on next page.)

[a] The "power trick" is used when terms in a PDF are raised to the power of indicator components of a one-hot vector. For example, it allows us to rewrite $p(\mathbf{z}_n; \boldsymbol{\theta}) = \prod_k \theta_k^{z_{nk}}$.

Problem 1 (cont.)

3. **Expectation Step** Our next step is to introduce a mathematical expression for $\mathbf{q}_n$, the posterior over the hidden component variables $\mathbf{z}_n$ conditioned on the observed data x_n with fixed parameters. That is:

$$\mathbf{q}_n = \begin{bmatrix} p(\mathbf{z}_n = \mathbf{C}_1 | x_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \\ \vdots \\ p(\mathbf{z}_n = \mathbf{C}_K | x_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \end{bmatrix}.$$

   Write down and simplify the expression for $\mathbf{q}_n$. Note that because the $\mathbf{q}_n$ represents the posterior over the hidden categorical variables $\mathbf{z}_n$, the components of vector $\mathbf{q}_n$ must sum to 1. The main work is to find an expression for $p(\mathbf{z}_n | x_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K)$ for any choice of $\mathbf{z}_n$; i.e., for any 1-hot encoded $\mathbf{z}_n$. With this, you can then construct the different components that make up the vector $\mathbf{q}_n$.

4. **Maximization Step** Using the $\mathbf{q}_n$ estimates from the Expectation Step, derive an update for maximizing the expected complete data log likelihood in terms of $\boldsymbol{\theta}$ and $\{\beta_k\}_{k=1}^K$.

   (a) Derive an expression for the expected complete data log likelihood using $\mathbf{q}_n$.

   (b) Find an expression for $\boldsymbol{\theta}$ that maximizes this expected complete data log likelihood. You may find it helpful to use Lagrange multipliers in order to enforce the constraint $\sum \theta_k = 1$. Why does this optimal $\boldsymbol{\theta}$ make intuitive sense?

   (c) Find an expression for β_k that maximizes the expected complete data log likelihood. Why does this optimal β_k make intuitive sense?

5. Suppose that this had been a classification problem. That is, you were provided the "true" components $\mathbf{z}_n$ for each observation x_n, and you were going to perform the classification by inverting the provided generative model (i.e. now you're predicting $\mathbf{z}_n$ given x_n). Could you reuse any of your derivations above to estimate the parameters of the model?

6. Finally, implement your solution in `p1.ipynb` and attach the final plot below.

   You will recieve no points for code not included below.

Solution

1. The likelihood expands to

$$\begin{aligned}\log p(\{x_n\}_{n=1}^N; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) &= \log \prod_{n=1}^N p(x_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \\ &= \log \prod_{n=1}^N \sum_{k=1}^K p(x_n, z_{nk}; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \\ &= \sum_{n=1}^N \log \left[\sum_{k=1}^K p(x_n, z_{nk}; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \right].\end{aligned}$$

Optimizing this likelihood directly is intractable because of the summation over the K classes inside of the logarithm, which precludes an analytical solution.

2. The negative complete data log likelihood expands to

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) &= -\log p(\mathcal{D}; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \\ &= -\sum_{n=1}^N \log p(x_n, \mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \\ &= -\sum_{n=1}^N \log[p(x_n | \mathbf{z}_n; \{\beta_k\}_{k=1}^K) p(\mathbf{z}_n; \boldsymbol{\theta})] \\ &= -\sum_{n=1}^N \log p(x_n | \mathbf{z}_n; \{\beta_k\}_{k=1}^K) + \log p(\mathbf{z}_n; \boldsymbol{\theta}).\end{aligned}$$

Applying the power trick,

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) &= -\sum_{n=1}^N \left[\log \prod_{k=1}^K p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K)^{z_{nk}} + \log \prod_{k=1}^K \theta_k^{z_{nk}} \right] \\ &= -\sum_{n=1}^N \sum_{k=1}^K z_{nk} \log p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K) + z_{nk} \log \theta_k.\end{aligned}$$

3. Using Bayes' Rule and the power trick,

$$\begin{aligned}p(\mathbf{z}_n | x_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) &= \frac{\prod_{k=1}^K \left(p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K) \theta_k \right)^{z_{nk}}}{\sum_{k=1}^K p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K) \theta_k} \\ &= \frac{\prod_{k=1}^K (p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K))^{z_{nk}} \prod_{k=1}^K (\theta_k)^{z_{nk}}}{\sum_{k=1}^K p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K) \theta_k}.\end{aligned}$$

The soft assignment value for a particular class k is

$$\begin{aligned}q_{nk} &= p(\mathbf{z}_n = \mathbf{C}_k | x_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \\ &= \frac{p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K) p(\mathbf{z}_n = \mathbf{C}_k; \boldsymbol{\theta})}{\sum_{k=1}^K p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K) p(\mathbf{z}_n = \mathbf{C}_k; \boldsymbol{\theta})} \\ &= \frac{p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K) \theta_k}{\sum_{k=1}^K p(x_n | \mathbf{z}_n = \mathbf{C}_k; \{\beta_k\}_{k=1}^K) \theta_k}.\end{aligned}$$

Therefore,

$$\mathbf{q}_n = \begin{bmatrix} \frac{p(x_n|\mathbf{z}_n=\mathbf{C}_1;\{\beta_k\}_{k=1}^K)\theta_1}{\sum_{k=1}^K p(x_n|\mathbf{z}_n=\mathbf{C}_k;\{\beta_k\}_{k=1}^K)\theta_k} \\ \vdots \\ \frac{p(x_n|\mathbf{z}_n=\mathbf{C}_K;\{\beta_k\}_{k=1}^K)\theta_K}{\sum_{k=1}^K p(x_n|\mathbf{z}_n=\mathbf{C}_k;\{\beta_k\}_{k=1}^K)\theta_k} \end{bmatrix}.$$

4. (a) The expected complete data log likelihood is

$$\begin{aligned}
&\mathbb{E}_{\{\mathbf{z}_n\}_{n=1}^N|\{x_n\}_{n=1}^N}\left[\log p(\mathcal{D};\boldsymbol{\theta},\{\beta_k\}_{k=1}^K)\right] \\
&= \mathbb{E}_{\{\mathbf{z}_n\}_{n=1}^N|\{x_n\}_{n=1}^N}\left[\sum_{n=1}^N \log p(x_n|\mathbf{z}_n;\{\beta_k\}_{k=1}^K) + \log p(\mathbf{z}_n;\boldsymbol{\theta})\right] \\
&= \sum_{n=1}^N \mathbb{E}_{\mathbf{z}_n|x_n}\left[\log p(x_n|\mathbf{z}_n;\{\beta_k\}_{k=1}^K) + \log p(\mathbf{z}_n;\boldsymbol{\theta})\right] \\
&= \sum_{n=1}^N\sum_{k=1}^K p(\mathbf{z}_n = \mathbf{C}_k|x_n;\boldsymbol{\theta},\{\beta_k\}_{k=1}^K)(\log p(x_n|\mathbf{z}_n = \mathbf{C}_k;\{\beta_k\}_{k=1}^K) + \log p(\mathbf{z}_n = \mathbf{C}_k;\boldsymbol{\theta})) \\
&= \sum_{n=1}^N\sum_{k=1}^K q_{nk}\log p(x_n|\mathbf{z}_n = \mathbf{C}_k;\{\beta_k\}_{k=1}^K) + q_{nk}\log\theta_k,
\end{aligned}$$

where q_{nk} is the probability computed in the E-step.

(b) Use Lagrange multipliers on the expected complete data log likelihood from the previous part with the constraint $\sum_k \theta_k - 1 = 0$. The Lagrangian is

$$\sum_{n=1}^N\sum_{k=1}^K \left(q_{nk}\log p(x_n|\mathbf{z}_n = \mathbf{C}_k;\{\beta_k\}_{k=1}^K) + q_{nk}\log\theta_k\right) - \lambda\left(\sum_{k=1}^K \theta_k - 1\right).$$

Optimizing by differentiating with respect to θ_k and setting equal to zero, noting that we treat the q_{nk} as fixed in the M-step:

$$\begin{aligned}
\sum_{n=1}^N \frac{q_{nk}}{\theta_k} - \lambda &= 0 \\
\theta_k &= \frac{\sum_{n=1}^N q_{nk}}{\lambda}.
\end{aligned}$$

Summing over k, recalling that $\sum_k \theta_k = 1$,

$$\sum_{k=1}^K \theta_k = \frac{\sum_{k=1}^K\sum_{n=1}^N q_{nk}}{\lambda} = 1 \Rightarrow \lambda = \sum_{k=1}^K\sum_{n=1}^N q_{nk} = \sum_{n=1}^N\sum_{k=1}^K q_{nk}.$$

Substituting this expression for λ back into our expression for θ_k,

$$\theta_k = \frac{\sum_{n=1}^N q_{nk}}{\sum_{n=1}^N\sum_{k=1}^K q_{nk}} = \frac{\sum_{n=1}^N q_{nk}}{N}.$$

Therefore the optimal $\boldsymbol{\theta}$ is

$$\left(\frac{\sum_{n=1}^N q_{n1}}{N},\ldots,\frac{\sum_{n=1}^N q_{nK}}{N}\right).$$

This makes intuitive sense because the optimal θ_k for each class is proportional to the sum of all of the data points' soft assignment probabilities for that class.

(c) Substituting the gamma PDF into the expected complete data log likelihood gives

$$\sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk}\log\frac{(x_n)^{\alpha-1}e^{-\beta_k x_n}(\beta_k)^{\alpha}}{\Gamma(\alpha)}+q_{nk}\log\theta_k$$

$$\sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk}\left((\alpha-1)\log x_n-\beta_k x_n+\alpha\log\beta_k-\log\Gamma(\alpha)\right)+q_{nk}\log\theta_k$$

Optimizing by differentiating with respect to β_k and setting equal to zero, we have

$$\sum_{n=1}^{N} -q_{nk}x_n+q_{nk}\frac{\alpha}{\beta_k}=0$$

$$\beta_k=\alpha\cdot\frac{\sum_{n=1}^{N}q_{nk}}{\sum_{n=1}^{N}q_{nk}x_n}.$$

This optimal β_k makes intuitive sense because it is α times the reciprocal of the weighted average of all of the x_n, where each x_n is weighted by how likely it currently is to belong to class k.

5. Yes, we could reuse the derivated parameter estimates by substituting $\mathbf{q}_n$ with $\mathbf{z}_n$ for each data point, making $q_{nk}=1$ for the true class k and $q_{nk}=0$ for all other classes. This corresponds to predicting the true class with probability 1. This substitution gives the following MLE parameters:

$$\theta_k=\frac{\sum_{n=1}^{N}q_{nk}}{N}=\frac{\sum_{n=1}^{N}z_{nk}}{N},$$

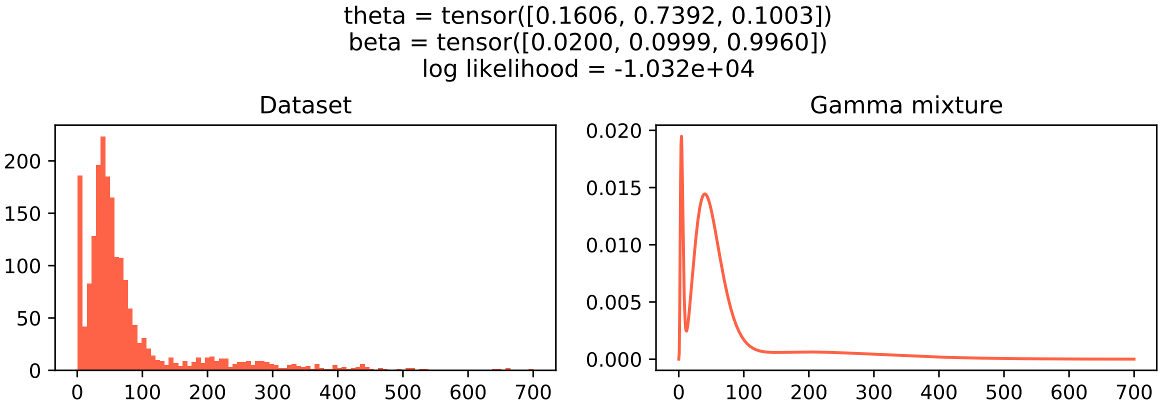
which is the empirical proportion of data points in class k, and

$$\beta_k=\alpha\cdot\frac{\sum_{n=1}^{N}q_{nk}}{\sum_{n=1}^{N}q_{nk}x_n}=\alpha\cdot\frac{\sum_{n=1}^{N}z_{nk}}{\sum_{n=1}^{N}z_{nk}x_n}.$$

which is α times the reciprocal of the empirical mean of data points in class k. These parameters can be used to approximate the joint distribution $p(x_n,\mathbf{z}_n;\boldsymbol{\theta},\{\beta_k\}_{k=1}^{K})$, which can in turn be used to predict $\mathbf{z}_n$ given x_n using Bayes' rule, as we did in classification:

$$p(\mathbf{z}_n=\mathbf{C}_k|x_n;\boldsymbol{\theta},\{\beta_k\}_{k=1}^{K})=\frac{p(x_n,\mathbf{z}_n;\boldsymbol{\theta},\{\beta_k\}_{k=1}^{K})}{p(x_n;\boldsymbol{\theta},\{\beta_k\}_{k=1}^{K})}\propto p(x_n|\mathbf{z}_n=\mathbf{C}_k;\{\beta_k\}_{k=1}^{K})p(\mathbf{z}_n=\mathbf{C}_k;\boldsymbol{\theta}).$$

6. Plot:



```
theta = tensor([0.1606, 0.7392, 0.1003])
beta = tensor([0.0200, 0.0999, 0.9960])
log likelihood = -1.032e+04
```

Code:

```
def e_step(theta, betas):
    log_q = ds.gamma.Gamma(alpha, betas).log_prob(x) + torch.log(theta)
    q = torch.exp(log_q - log_q.logsumexp(dim=1, keepdim=True))
    return q


def m_step(q):
    q_sums = torch.sum(q, 0)
    theta_hat = q_sums / q.shape[0]
    beta_hats = alpha * q_sums / torch.sum(q * x, 0)
    return theta_hat, beta_hats


def log_px(x, theta, betas):
    non_log_ps = (ds.gamma.Gamma(alpha, betas).log_prob(x) + torch.log(theta)).exp()
    p = torch.log(non_log_ps.sum(1))
    return p


def run_em(theta, betas, iterations=1000):
    for _ in range(iterations):
        q = e_step(theta, betas)
        theta, betas = m_step(q)
    return theta, betas
```

Problem 2 (PCA, 15 pts)

For this problem you will implement PCA from scratch on the first 6000 images of the MNIST dataset. Your job is to apply PCA on MNIST and discuss what kind of structure is found. Implement your solution in `p2.ipynb` and attach the final plots below.

You will recieve no points for using third-party PCA implementations (i.e. `scikit-learn`).

You will recieve no points for code not included below.

1. Compute the PCA. Plot the eigenvalues corresponding to the most significant 500 components in order from most significant to least. Make another plot that describes the cumulative proportion of variance explained by the first k most significant components for values of k from 1 through 500. How much variance is explained by the first 500 components? Describe how the cumulative proportion of variance explained changes with k. Include this plot below.

2. Plot the mean image of the dataset and plot an image corresponding to each of the first 10 principle components. How do the principle component images compare to the cluster centers from K-means? Discuss any similarities and differences. Include these two plots below.

   Reminder: Center the data before performing PCA

3. Compute the reconstruction error on the data set using the mean image of the dataset. Then compute the reconstruction error using the first 10 principal components. How do these errors compare to the final objective loss achieved by using K-means on the dataset? Discuss any similarities and differences.

   For consistency in grading, define the reconstruction error as the squared L2 norm averaged over all data points.

4. Suppose you took the original matrix of principle components that you found U and multiplied it by some rotation matrix R. Would that change the quality of the reconstruction error in the last problem? The interpretation of the components? Why or why not?

Solution

Plots:

Code:

```
def pca(x, n_comps=500):
    top_eigvals = ’not implemented’
    top_pcomps = ’not implemented’
    return top_eigvals, top_pcomps


def calc_cfvs(eigvals):
    cum_frac_vars = ’not implemented’
    return cum_frac_vars


def calc_errs(x, pcomps):
    err_mean = ’not implemented’
    err_pcomp = ’not implemented’
    return err_mean, err_pcomp
```

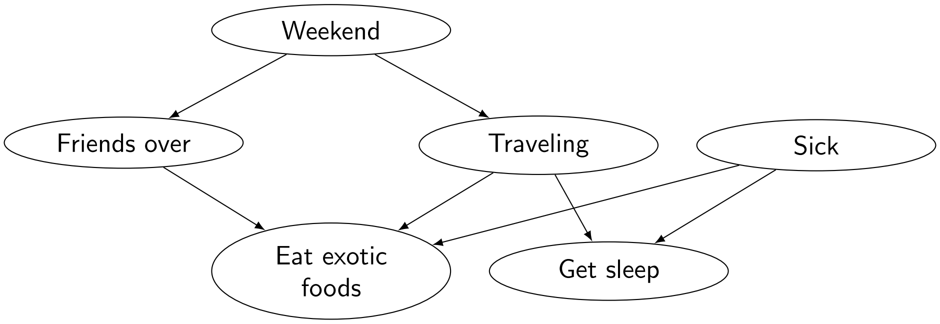
1.

2.

3.

Problem 3 (Bayesian Networks, 10 pts)

In this problem we explore the conditional independence properties of a Bayesian Network. Consider the following Bayesian network representing a fictitious person's activities. Each random variable is binary (true/false).



The random variables are:

- Weekend: Is it the weekend?
- Friends over: Does the person have friends over?
- Traveling: Is the person traveling?
- Sick: Is the person sick?
- Eat exotic foods: Is the person eating exotic foods?
- Get Sleep: Is the person getting sleep?

For the following questions, $A \perp B$ means that events A and B are independent and $A \perp B|C$ means that events A and B are independent conditioned on C.

Use the concept of d-separation to answer the questions and show your work (i.e., state what the blocking path(s) is/are and what nodes block the path; or explain why each path is not blocked).

Example Question: Is Friends over $\perp$ Traveling? If NO, give intuition for why.

Example Answer: NO. The path from Friends over – Weekend – Traveling is not blocked following the d-separation rules as we do not observe Weekend. Thus, the two are not independent.

Actual Questions:

1. Is Weekend $\perp$ Get Sleep? If NO, give intuition for why.
2. Is Sick $\perp$ Weekend? If NO, give intuition for why.
3. Is Sick $\perp$ Friends over | Eat exotic foods? If NO, give intuition for why.
4. Is Friends over $\perp$ Get Sleep? If NO, give intuition for why.
5. Is Friends over $\perp$ Get Sleep | Traveling? If NO, give intuition for why.
6. Suppose the person stops traveling in ways that affect their sleep patterns. Travel still affects whether they eat exotic foods. Draw the modified network. (Feel free to reference the handout file for the commands for displaying the new network in LaTeX).
7. For this modified network, is Friends over $\perp$ Get Sleep? If NO, give an intuition why. If YES, describe what observations (if any) would cause them to no longer be independent.

Solution

1.
2.
3.
4.
5.
6.
7.

Name

Collaborators and Resources

Whom did you work with, and did you use any resources beyond cs181-textbook and your notes?

Calibration

Approximately how long did this homework take you to complete (in hours)?