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

Problem 1.1

We want to expand the data likelihood to include the sums over observations x_n and marginalize out the latents $\mathbf{z}_n$.

$$\begin{aligned}
p(\{x_n\}; \boldsymbol{\theta}, \{\beta_k\}) &= \prod_{n=1}^{N} \sum_{k=1}^{K} p(x_n, \mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}) \\
\log(p(\{x_n\}; \boldsymbol{\theta}, \{\beta_k\})) &= \sum_{n=1}^{N} \log\left(\sum_{k=1}^{K} p(x_n, \mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k)\}\right) \\
&= \sum_{n=1}^{N} \log\left(\sum_{k=1}^{K} p(x_n|\mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}) p(\mathbf{z}_n|\boldsymbol{\theta}, \{\beta_k\})\right) \\
&= \sum_{n=1}^{N} \log\left(\sum_{k=1}^{K} \text{Gamma}(x_n|\alpha, \beta_k)\theta_k\right)
\end{aligned}$$

Optimizing this likelihood is directly intractable because of the summation within the log.

Problem 1.2

We want to write out the negative complete data log likelihood and apply the power trick to simplify our expression using indicator elements z_{nk}.

$$\begin{aligned}
p(\{x_n\}, \{\mathbf{z}_n\}; \boldsymbol{\theta}, \{\beta_k\}) &= \prod_{n=1}^{N} p(x_n, \mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}) \\
&= \prod_{n=1}^{N} p(x_n|\mathbf{z}_n; \{\beta_k\}) p(\mathbf{z}_n; \boldsymbol{\theta}) \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} (\text{Gamma}(x_n|\alpha, \beta_k)\theta_k)^{z_{nk}} \\
-\log p(\{x_n\}, \{\mathbf{z}_n\}; \boldsymbol{\theta}, \{\beta_k\}) &= -\sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \log \text{Gamma}(x_n|\alpha, \beta_k) + z_{nk} \log \theta_k
\end{aligned}$$

And so we have written the complete negative log likelihood.

Problem 1.3

We want to write down and simplify the expression for $\mathbf{q}_n$. Let us work with a single element q_{nk} of $\mathbf{q}_n$ and then assemble these elements into a vector at the end.

$$\begin{aligned}
q_{nk} &= p(\mathbf{z}_n = C_k | x_n; \boldsymbol{\theta}, \beta_k) \\
&= p(x_n | \mathbf{z_n} = C_k; \beta_k) p(\mathbf{z}_n = C_k; \boldsymbol{\theta}) \\
&= \theta_k \text{Gamma}(x_n|\alpha, \beta_k)
\end{aligned}$$

We also need to make sure to normalize the elements of $\mathbf{q}_n$ since we know that $\sum_{k=1}^K q_{nk} = 1$. And so we have the following expression for $\mathbf{q}_n$.

$$\mathbf{q}_n = \begin{bmatrix} \theta_1 \text{Gamma}(x_n|\alpha, \beta_1) / \left(\sum_{k=1}^K \theta_k \text{Gamma}(x_n|\alpha, \beta_k)\right) \\ \vdots \\ \theta_K \text{Gamma}(x_n|\alpha, \beta_K) / \left(\sum_{k=1}^K \theta_k \text{Gamma}(x_n|\alpha, \beta_k)\right) \end{bmatrix}$$

Problem 1.4(a)

We want to derive an expression for the expected complete data log likelihood using $\mathbf{q}_n$.

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

This uses $q_{nk} = p(\mathbf{z}_n = C_k|x_n)$ to simplify notation. And so we are done.

Problem 1.4(b)

We want to find an expression for $\boldsymbol{\theta}$ that maximizes this expected complete data log likelihood. We will use Lagrange multipliers and enforce the constraint $\sum_{i=1}^K \theta_k = 1$.

Let us set up a Lagrange multiplier function L.

$$L(\{x_n\}, \lambda) = \left[\sum_{n=1}^N \sum_{k=1}^K q_{nk} \log p(x_n|\mathbf{z}_n; \{\beta_k\}) + q_{nk} \log p(\mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}\})\right] - \lambda\left(\left[\sum_{k=1}^K \theta_k\right] - 1\right)$$

Now take the derivative with respect to λ, set the LHS to 0, and rearrange.

$$\nabla_\lambda L(\{x_n\}, \lambda) = 1 - \sum_{k=1}^K \theta_k$$

$$\sum_{k=1}^K \theta_k = 1$$

Then we take the derivative with respect to θ_k.

$$\nabla_{\theta_k} L(\{x_n\}, \lambda) = \left[\sum_{n=1}^N \frac{q_{nk}}{\theta_k}\right] - \lambda$$

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

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

Subbing this into our expression above, we get the following value of λ.

$$\begin{aligned}
\sum_{k=1}^{K} \theta_k &= 1 \\
\sum_{k=1}^{K} \frac{\sum_{n=1}^{N} q_{nk}}{\lambda} &= 1 \\
\lambda &= \sum_{k=1}^{K}\sum_{n=1}^{N} q_{nk} \\
\lambda &= N
\end{aligned}$$

Finally, we sub this into our expression for θ_k.

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

And so we have found an expression for $\boldsymbol{\theta}$ (a vector of these θ_k for each k) that maximizes the expected complete data log likelihood.

This expression makes sense because it is essentially the average probability for each class across the data set. Since θ_k is the probability for a given class because our latent variables are categorical, then this is appropriate.

Problem 1.4(c)

We want to find an expression for β_k that maximizes the expected complete data log likelihood. Let E be our complete data log likelihood.

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

We use the product rule to differentiate the Gamma PDF.

$$\begin{aligned}
\text{Gamma}(x_n|\alpha,\beta_k) &= \frac{x_n^{\alpha-1}}{\Gamma(\alpha)}\beta_k^{\alpha}e^{-\beta_k x_n} \\
\nabla_{\beta_k}\text{Gamma}(x_n|\alpha,\beta_k) &= \frac{x_n^{\alpha-1}}{\Gamma(\alpha)}\left[\alpha\beta_k^{\alpha-1}e^{-\beta_k x_n} - x\beta_k^{\alpha}e^{-\beta_k x_n}\right] \\
&= \frac{x_n^{\alpha-1}}{\Gamma(\alpha)}\left[\beta_k^{\alpha-1}e^{-\beta_k x_n}(\alpha - x_n\beta_k)\right]
\end{aligned}$$

Then we can plug this back in, simplify, set the LHS to 0, and solve for β_k.

$$\begin{aligned}
\nabla_{\beta_k} E &= \sum_{n=1}^{N} q_{nk} \left[\frac{\beta_k^{\alpha-1} e^{-\beta_k x_n} (\alpha - x_n \beta_k)}{\beta_k^{\alpha} e^{-\beta_k x}} \right] \\
&= \sum_{n=1}^{N} q_{nk} \left[\frac{\alpha - x_n \beta_k}{\beta_k} \right] \\
&= \sum_{n=1}^{N} q_{nk} \left[\frac{\alpha}{\beta_k} - x_n \right] \\
0 &= \sum_{n=1}^{N} q_{nk} \frac{\alpha}{\beta_k} - \sum_{n=1}^{N} q_{nk} x_n \\
\sum_{n=1}^{N} q_{nk} x_n &= \frac{\alpha}{\beta_k} \sum_{n=1}^{N} q_{nk} \\
\beta_k &= \frac{\alpha \sum_{n=1}^{N} q_{nk}}{\sum_{n=1}^{N} q_{nk} x_n}
\end{aligned}$$

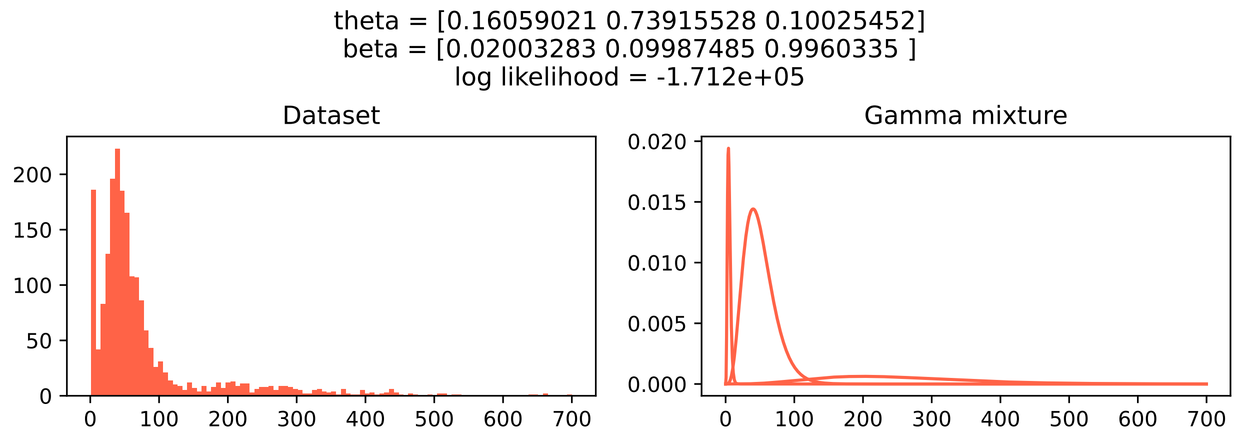
To see that this is intuitive, note that we can rearrange the equation above to be in the form $\frac{\sum_{n=1}^{N} q_{nk} x_n}{\sum_{n=1}^{N} q_{nk} q_{nk}} = \alpha/\beta_k$. We know that the mean of a Gamma distribution is given by α/β. And so the above value of β_k is the value corresponding to the most likely mean of the data for our (where each data point is weighted by the certainty of a data point x_n being a member of class C_k).

Problem 1.5

If we are provided the true components $\mathbf{z}_n$ for each observation and we were inverting the generative model, then we could reuse all of our derivations to estimate the model parameters except those from the expectation step. The expectation step attempts to find a soft assignment of the data to each class; however, if we are given $\mathbf{z}_n$, then there is no need to complete this step because we already know the assignment. We may still reuse the derivations in the maximization step since these will find the most likely model for our data.

Problem 1.6

Plot:



Code:

```
def e_step(theta, betas):
    q = np.multiply(gamma.pdf(x, alpha, scale=(1 / betas)), theta)
    q /= np.sum(q, axis=1)[:,None]
    return q


def m_step(q):
    qsum = np.sum(q, axis=0)
    theta_hat = qsum / x.shape[0]
    beta_hats = (qsum * alpha) / (q.T@x).flatten()
    return theta_hat, beta_hats


def log_px(x, theta, betas):
    p = np.log(np.multiply(gamma.pdf(x, alpha, scale=(1 / betas)), theta))
    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.

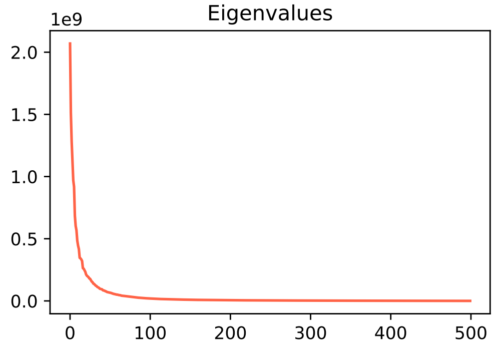
1. Compute the PCA. Plot the eigenvalues corresponding to the most significant 500 components in order from most significant to least. Make another plot that describes the cumulative proportion of variance explained by the first k most significant components for values of k from 1 through 500. How much 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.

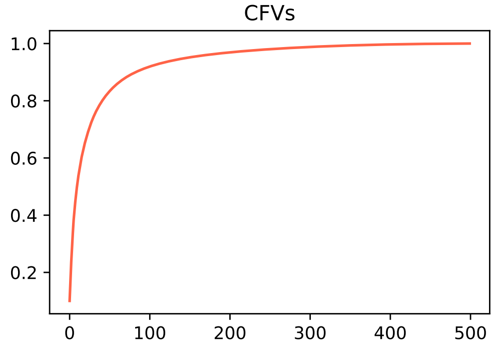
   Reminder: Center the data before performing PCA
3. Compute the reconstruction error on the data set using the mean image of the dataset. Then compute the reconstruction error using the first 10 principal components. How do these errors compare to the final objective loss achieved by using K-means on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

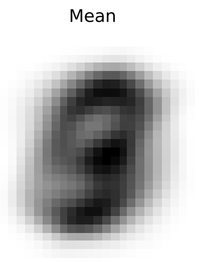
   For consistency in grading, define the reconstruction error as the squared L2 norm averaged over all data points.
4. Suppose you took the original matrix of principle components that you found U and multiplied it by some rotation matrix R. Would that change the quality of the reconstruction error in the last problem? The interpretation of the components? Why or why not?

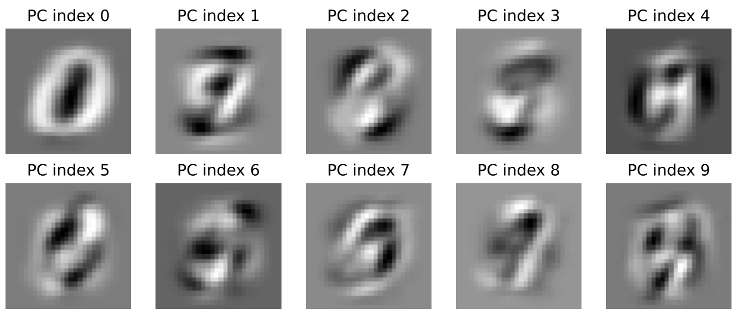
Solution

Plots:









Code:

```
def pca(x, n_comps=500):
    xm = x - x.mean(axis=0)

    U, S, V = np.linalg.svd(xm, full_matrices=False)

    top_eigvals = (S ** 2)[:n_comps]
    top_pcomps = V.T[:,:n_comps].T

    return top_eigvals, top_pcomps


def calc_cfvs(eigvals):
    return np.cumsum(eigvals) / np.sum(eigvals)


def calc_errs(x, pcomps):
    xm = x - x.mean(axis=0)
    err_mean = np.mean(np.linalg.norm(xm, axis=1) ** 2)

    pcs = pcomps[:n_comps]
    err_pcomp = np.mean(np.linalg.norm(xm - xm@pcs.T@pcs, axis=1) ** 2)

    return err_mean, err_pcomp
```

Problem 2.1

Nearly all of the variance is explained by the first 500 components. The cumulative proportion of variance explained increases at a decreasing rate as k increases, reaching a "hump" at around $k = 50$ and quickly plateauing.

Problem 2.2

The PC images are very different than the cluster centers from k-means on the un-standardized data set (from Homework 4 Problem 2). Where the un-standardized k-means images are clear numbers, the PC images are blurred and with the exception of PC 0 do not look like numbers.

Interestingly, the PC images look somewhat similar to the cluster centers from k-means when the data is standardized: the edges are mostly grey and most of the variance occurs in the center. However, whereas the k-means cluster centers generally like distinct and clear numbers, the PC images do not and look like an assembly of white and black splotches (again with the exception of PC zero).

Problem 2.3

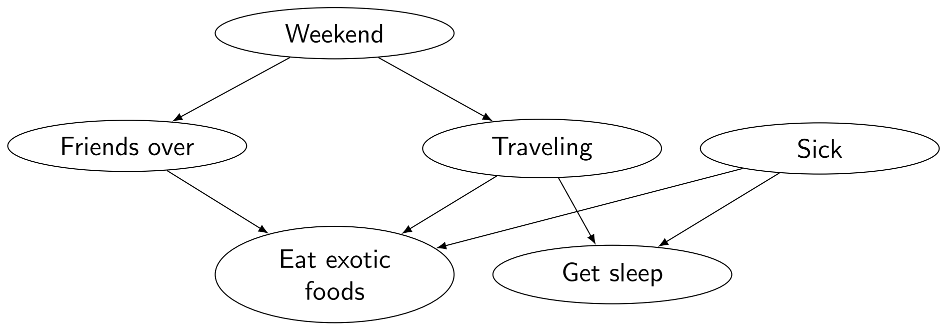
The reconstruction error using the mean was $3,436,024$ and the reconstruction error using the mean and the top 10 principal components was $1,731,315$. This is around half of the k-means mean objective loss when the data was not standardized, which (from Homework 4 Problem 2) was around $2,550,000$. These reconstruction errors are all roughly similar, with the mean and top 10 principal components performing best.

Problem 2.4

Multiplying the matrix of principle components U by some rotation matrix R would change the reconstruction error. The principal components are in the axes of the highest variance in the data: by rotating the principal components, they would no longer represent the axes of highest variance (the one exception being the rare instance where they were rotated orthogonally), thus changing the interpretation of the components as well, and so would yield completely different reconstructed images.

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.

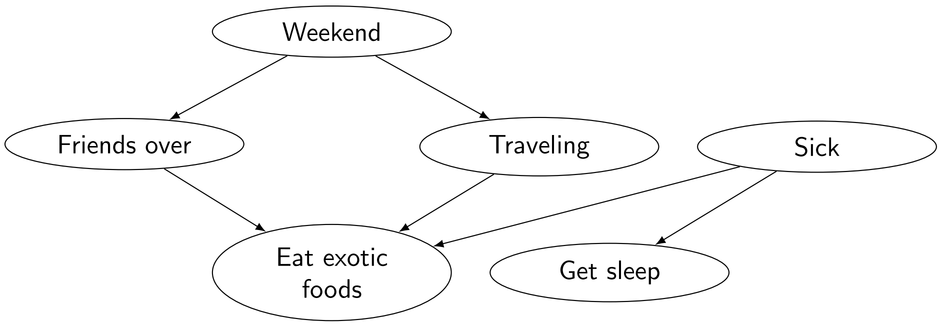
Example Answer: NO. The path from Friends over – Weekend – Traveling is not blocked following the d-separation rules as we do not observe Weekend. Thus, the two are not independent.

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. NO. The path from Weekend – Traveling – Get sleep is not blocked following d-separation rules as we do not observe Traveling, and so they are not independent. Intuitively, if do not observe Traveling, then whether or not it is the Weekend will affect whether or not we got sleep.
2. YES.
3. NO. By d-separation rule 4, since eating exotic food is a child of both Friends over and Sick and we observed Eat exotic food, then Friends over and Sick are not independent. Intuitively, observing eating exotic food gives us information about both whether or not we had friends over or were sick, so they are no longer independent.
4. NO. By d-separation rule 3, since Weekend was not observed, Friends over is a child of weekend, and Get sleep is a descendant of Weekend, then they are not independent. Intuitively, if we observed eating exotic foods then we have information about having friends over and being sick, and since being sick affects our sleep then they are not independent.
5. YES.
6. The modified network:



7. YES, Friends over is independent to Get sleep in this modified network. Observing Eat exotic foods would cause them to be no longer independent. This is because by d-separation rule 4, observing Eat exotic foods would cause Friends over and Sick to no longer be independent, and since Get sleep is a child of Sick then it would also not be independent of Friends over.

Name

Collaborators and Resources

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

Calibration

Approximately how long did this homework take you to complete (in hours)? 25