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 receive no points for code not included below.

Solution

1. Each data point is independent, and taking the logarithm turns products into sums, so

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

   By law of total probability,

$$\sum_{n=1}^N \log p(x_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) = \sum_{n=1}^N \log \left(\sum_{k=1}^K p(x_n, \mathbf{z}_n = \mathbf{C}_k; \boldsymbol{\theta}, \beta_k) \right).$$

   This is intractable since the summation inside the logarithm makes it impossible for us to get a closed form analytical solution for the MLE.

2. First we'll use the fact that the logarithm turns products into sums to expand the expression:

$$-\log p(\mathcal{D}; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) = -\sum_{n=1}^N \log p(x_n, \mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K).$$

Next we'll use conditioning to write this as

$$-\sum_{n=1}^N \log \big(p(x_n|\mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) \cdot p(\mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K)\big) = -\sum_{n=1}^N \log p(x_n|\mathbf{z}_n; \{\beta_k\}_{k=1}^K) + \log p(\mathbf{z}_n; \boldsymbol{\theta}).$$

Next we'll use the power trick to write $p(x_n|\mathbf{z}_n; \{\beta_k\}_{k=1}^K)$ as $\prod_k f_k(x_n; \beta_k)^{z_{nk}}$ where f_k denotes the pdf of the Gamma distribution with shape parameter α and rate parameter β_k. Similarly, we can write $p(\mathbf{z}_n; \boldsymbol{\theta})$ as $\prod_k \theta_k^{z_{nk}}$. Plugging into our expression and utilizing log rules yields

$$-\sum_{n=1}^N \log\left(\prod_k f_k(x_n; \beta_k)^{z_{nk}}\right) + \log\left(\prod_k \theta_k^{z_{nk}}\right) = -\sum_{n=1}^N \sum_{k=1}^K z_{nk} \log f_k(x_n; \beta_k) + z_{nk} \log \theta_k.$$

Finally, we can plug in the Gamma pdf $f_k(x_n; \beta_k) = \frac{\beta_k^\alpha}{\Gamma(\alpha)} x_n^{\alpha-1} e^{-\beta_k x_n}$ to get

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

and then use log rules to get

$$\mathcal{L}(\boldsymbol{\theta}, \{\beta_k\}_{k=1}^K) = -\sum_{n=1}^N \sum_{k=1}^K z_{nk}\big(\alpha \log \beta_k - \log \Gamma(\alpha) + (\alpha - 1) \log x_n - \beta_k x_n\big) + z_{nk} \log \theta_k.$$

3. By Bayes' rule,

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

Plugging in θ_k and the Gamma pdf yields

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

We want to scale q_{nk} such that $\sum_k q_{nk} = 1$ for all n, so we'll simply divide our expression by the sum of this value over k. This yields

$$q_{nk} = \frac{\theta_k \frac{\beta_k^\alpha}{\Gamma(\alpha)} x_n^{\alpha-1} e^{-\beta_k x_n}}{\sum_{j=1}^K \theta_j \frac{\beta_j^\alpha}{\Gamma(\alpha)} x_n^{\alpha-1} e^{-\beta_j x_n}} = \frac{\frac{1}{\Gamma(\alpha)} x_n^{\alpha-1}\big(\theta_k \beta_k^\alpha e^{-\beta_k x_n}\big)}{\frac{1}{\Gamma(\alpha)} x_n^{\alpha-1}\big(\sum_{j=1}^K \theta_j \beta_j^\alpha e^{-\beta_j x_n}\big)} = \frac{\theta_k \beta_k^\alpha e^{-\beta_k x_n}}{\sum_{j=1}^K \theta_j \beta_j^\alpha e^{-\beta_j x_n}}.$$

4. (a) By definition, the expected complete data log likelihood is

$$\mathrm{E}_{\mathbf{z}|x}[\log p(\{x_n, \mathbf{z}_n\}_{n=1}^N; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K)].$$

By log rules and linearity of expectation, this is equal to

$$\sum_{n=1}^N \mathrm{E}_{\mathbf{z}_n|x_n}[\log p(x_n, \mathbf{z}_n; \boldsymbol{\theta}, \{\beta_k\}_{k=1}^K)].$$

Expanding the expectation and plugging in $p(\mathbf{z}_n = \mathbf{C}_k|x_n) = q_{nk}$ yields

$$\sum_{n=1}^{N}\sum_{k=1}^{K} p(\mathbf{z}_n = \mathbf{C}_k|x_n) \log p(x_n, \mathbf{z}_n = \mathbf{C}_k; \boldsymbol{\theta}, \beta_k) = \sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk} \log p(x_n, \mathbf{z}_n = \mathbf{C}_k; \boldsymbol{\theta}, \beta_k).$$

Rewriting the probability with conditioning yields

$$\sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk} \log p(x_n|\mathbf{z}_n = \mathbf{C}_k; \beta_k) + q_{nk} \log p(\mathbf{z}_n = \mathbf{C}_k; \boldsymbol{\theta}).$$

(b) To solve for an optimal $\boldsymbol{\theta}$, we will add in the constraint $\sum_k \theta_k = 1$ with a Lagrange multiplier:

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

Taking the derivative with respect to θ_k yields

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

since $p(x_n|\mathbf{z}_n = \mathbf{C}_k; \beta_k)$ does not involve $\boldsymbol{\theta}$ and $p(\mathbf{z}_n = \mathbf{C}_k; \boldsymbol{\theta}) = \theta_k$. Setting equal to 0 and rearranging yields

$$\theta_k = \frac{\sum_{n=1}^{N} q_{nK}}{-\lambda}.$$

Thus θ_k must be proportional to $\sum_{n=1}^{N} q_{nk}$, and since θ_k must sum to 1, we can write

$$\theta_k = \frac{\theta_k}{1} = \frac{\theta_k}{\sum_{j=1}^{K}\theta_j} = \frac{\frac{1}{-\lambda}\sum_{n=1}^{N} q_{nk}}{\sum_{j=1}^{K}\frac{1}{-\lambda}\sum_{n=1}^{N} q_{nj}} = \frac{\frac{1}{-\lambda}\sum_{n=1}^{N} q_{nk}}{\frac{1}{-\lambda}\sum_{j=1}^{K}\sum_{n=1}^{N} q_{nj}} = \frac{\sum_{n=1}^{N} q_{nk}}{\sum_{j=1}^{K}\sum_{n=1}^{N} q_{nj}}.$$

It follows that the optimal update is

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

Note that the middle equality holds since $\sum_{j=1}^{K} q_{nj} = 1$ by our definition in part 3. The expression for θ_k^* intuitively makes sense since this represents the average of q_{nk} over all of the data, and q_{nk} is the estimated likelihood that the hidden variable $\mathbf{z}_n$ is in class k.

(c) We start with

$$\sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk} \log p(x_n|\mathbf{z}_n = \mathbf{C}_k; \beta_k) + q_{nk} \log p(\mathbf{z}_n = \mathbf{C}_k; \boldsymbol{\theta}).$$

Note that we can drop the second term since it does not depend on β_k, and then we can expand the Gamma pdf $p(x_n|\mathbf{z}_n = \mathbf{C}_k; \beta_k)$:

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

Taking the derivative with respect to β_k yields

$$\sum_{n=1}^{N} q_{nk}\Big(\frac{\alpha}{\beta_k} - x_n\Big).$$

Setting equal to 0 and rearranging yields

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

The gamma distribution with shape parameter α and rate parameter β has mean $\frac{\alpha}{\beta}$, and

$$\widehat{x} = \frac{\sum_{n=1}^{N} q_{nk}x_n}{\sum_{n=1}^{N} q_{nk}}$$

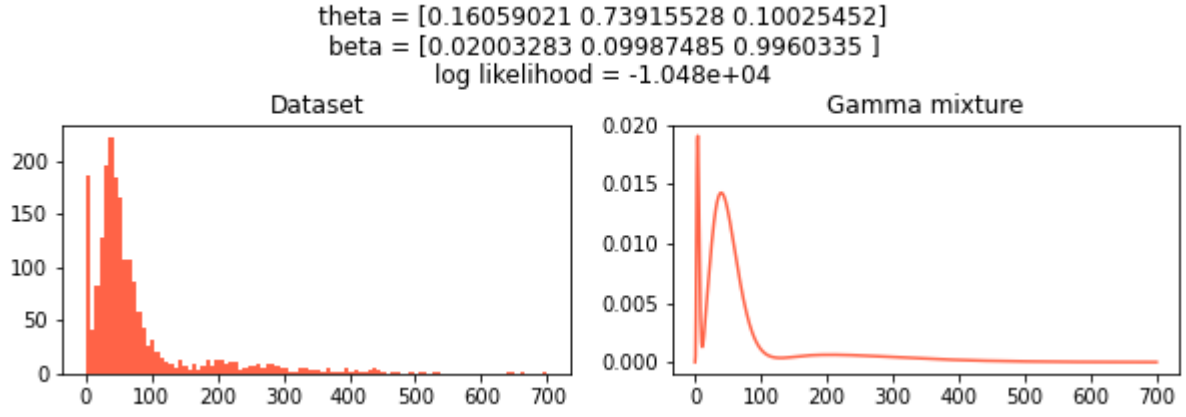
is sort of like a weighted sample average since q_{nk} is an estimate of the probability that the hidden variable $\mathbf{z}_n$ is in class k. Thus we expect

$$\widehat{x} \approx \frac{\alpha}{\beta_k} \implies \beta_k = \frac{\alpha}{\widehat{x}},$$

and this is our expression for β_k^*, so the optimal update makes intuitive sense.

5. Yes, if we look back to part 2, our loss function is just the expression from part 4.a but negated and with z_{nk} (indicator variables for $\mathbf{z}_n = \mathbf{C}_k$) instead of q_{nk} (estimated probabilities for $\mathbf{z}_n = \mathbf{C}_k$). This makes sense since we're trying to minimize loss in part 2 and trying to maximize expected complete data log likelihood in part 4. It follows that the derivation will be exactly the same, and we can just use z_{nk} instead of q_{nk} in our expressions for the optimal θ_k and β_k. This is because we treated q_{nk} as a constant — like z_{nk} — in the maximization step even though it was found using β_k and $\boldsymbol{\theta}$ in the expectation step. Note that we won't have to do multiple update steps in this case since $\mathbf{z}_n$ are true labels rather than estimates like $\mathbf{q}_n$. Additionally note that in our derivation of θ_k^* in part 4.b, we made use of the fact that $\sum_{j=1}^{K} q_{nj} = 1$, but this will also be true for $\sum_{j=1}^{K} z_{nj}$ since $\mathbf{z}_n$ is a one hot vector.

6. Plot:



Code:

```
    def e_step(x, alpha, theta, betas):
        q = theta * (betas ** alpha) * np.exp(-betas * x)
        return q / np.sum(q, axis = 1, keepdims=True)

    def m_step(x, q, alpha):
        theta_hat = np.mean(q, axis = 0)
        beta_hats = alpha * np.sum(q, axis=0) / np.sum(q * x, axis = 0)
        return theta_hat, beta_hats

    def gamma_pdf(x, alpha, betas):
        return (betas ** alpha) * (x ** (alpha - 1)) * np.exp(-betas * x) / factorial(
                                                        alpha - 1)

    def log_px(x, alpha, theta, betas):
        q = e_step(x, alpha, theta, betas)
        return np.sum(q * np.log(gamma_pdf(x, alpha, betas) * theta), axis = 1)

    def log_likelihood(x, alpha, theta, betas):
        return np.sum(log_px(x, alpha, theta, betas))

    def run_em(x, alpha, theta, betas, iterations=1000):
        for _ in range(iterations):
            q = e_step(x, alpha, theta, betas)
            theta, betas = m_step(x, q, alpha)
        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 receive no points for using third-party PCA implementations (i.e. `scikit-learn`).

You will receive 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 principal components. How do the principal 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

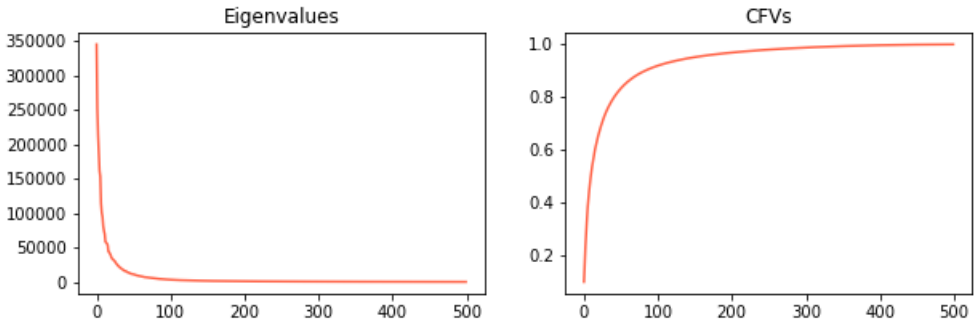
3. Compute the reconstruction error on the data set using the mean image of the dataset. Then compute the reconstruction error using the first 10 principal components. How do these errors compare to the final objective loss achieved by using K-means on the dataset? Discuss any similarities and differences.

   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 principal 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

1. Plots:

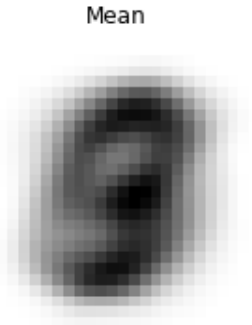


Code:

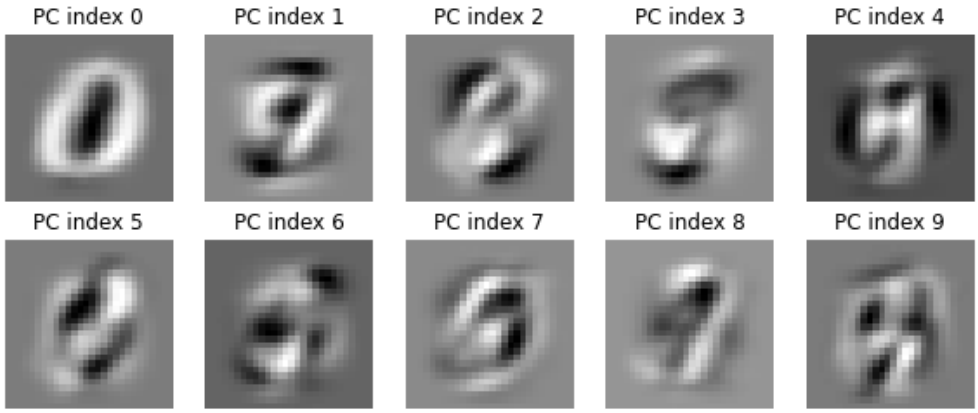
```
def pca(x, n_comps=500):
    svd = np.linalg.svd(x - np.mean(x, axis = 0)) # zero out x before svd
    top_eigvals = svd[1][:num_comps] ** 2 / x.shape[0]
    top_pcomps = svd[2][:num_comps]
    return top_eigvals, top_pcomps

def calc_cfvs(eigvals):
    cumsum = np.cumsum(eigvals)
    return cumsum / cumsum[-1]

def calc_errs(x, pcomps):
    zero_x = x - np.mean(x, axis = 0)
    err_mean = np.sum(zero_x ** 2) / x.shape[0]
    err_pcomp = np.sum((zero_x - (zero_x @ pcomps[:10].T @ pcomps[:10])) ** 2) / x.
                                                    shape[0]
    return err_mean, err_pcomp
```

The variance from the first 500 components is 3433955.8. The cumulative proportion of variance is steadily increasing until around $k = 50$ at which point it begins to slow down before almost completely leveling off at $k = 400$.

2. The mean image of the dataset is:



The images corresponding to the first 10 principal components are:



Both the cluster centers from **standardized** K-means and the principal component images have greyed out edges with dark patches in the middle and the actual digits being a chalky white. Most of the principal component images look mangled and blurry and, they don't resemble any one digit unlike the cluster centers. This can be explained by the fact that principal components aren't meant to individually represent digits; rather a linear combination of the components should come to close to representing each image. On the other hand, the cluster centers are meant to be representative of that particular cluster which usually corresponded to one or two numbers.

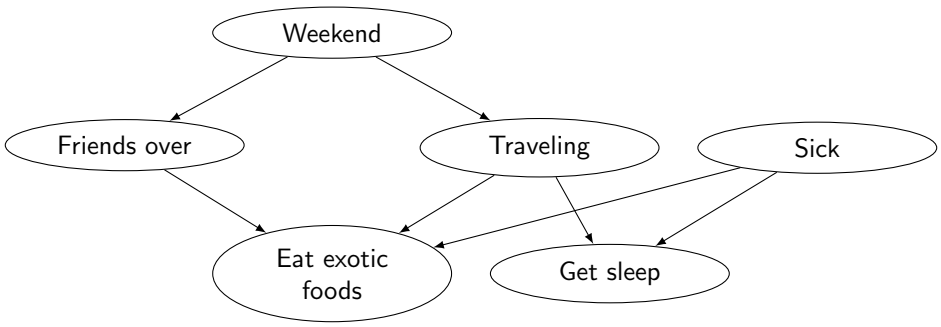
3. The reconstruction error using the mean image is 3.436021e+06, and the reconstruction error using the first 10 principal components is 1.731316e+06. On the other hand, the final K-means objective loss is 1.27e+10, and dividing by the size of the large dataset (5000) yields a mean error of 2.54e+06. Thus K-means, reconstruction with just the mean image, and reconstruction with the first 10 principal components all have errors with the same order of magnitude. In addition, K-means is better than reconstruction with just the mean but worse than reconstruction with first 10 principal components.

4. Yes, it would change the reconstruction error from part 3. The original complete set of principal components form an orthonormal basis by definition, and all of those properties still hold since rotation does not change the magnitude of a vector ($(RU_i)^T(RU_i)$ is still 1), and rotation does not change the angle between two vectors ($(RU_i)^T(RU_j)$ is still 0 for $i \neq j$). Similarly, the rotated vectors must still be linearly independent because rotation matrices are invertible, and if there exists a linear combination of the rotated vectors that equals the vector 0, then we can apply R^{-1} to the linear combination to get a linear combination of the original vectors that equals the vector 0, and this is a contradiction since the original vectors were linearly independent. Thus the rotated vectors still form an orthonormal basis.

   Since the rotated components still form a basis, they are still able to perfectly reconstruct each image if they are used as a complete set. However, if we use a partial set of rotated components, then we won't get the same performance as the same partial set of original components since rotation has changed the direction of each component. The directions of the original components were chosen such that they explained as much variance as possible, and they were ordered by how much variance they explained. This interpretation is no longer true after the rotation since the directions and therefore the rankings have changed.

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.

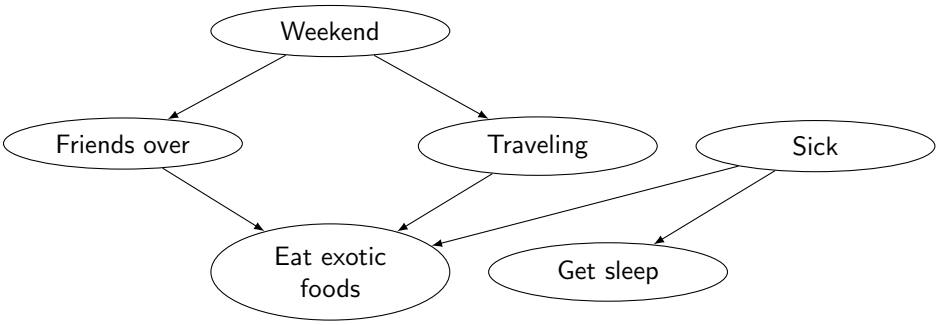
Example Answer: NO. The path from Friends over – Weekend – Traveling is not blocked following the d-separation rules as we do not observe Weekend. Thus, the two are not independent.

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, Weekend and Get Sleep are not independent since we haven't observed Traveling, and the path between the two is not blocked. Intuitively, if it's the weekend, then there's a chance you're traveling, which means there's a chance that you're going to get sleep.
2. Yes, Sick and Weekend are independent since we haven't observed the common descendant Get Sleep, and the path between the two is blocked. Intuitively, if we know that the person has gotten sleep, then either it is the weekend (and they're traveling) or they're sick. However, if we don't know about them getting sleep, then there's no connection we can draw between it being the weekend and the person being sick.
3. No, Sick and Friends Over are not independent since we've observed the common descendant Eat Exotic Foods, and the path between the two is not blocked. Intuitively, if we know the person has eaten exotic foods, then they either have friends over or they're traveling or they're sick. Thus if we know about whether they have friends over, then that tells us something about how likely they are to be sick and vice versa.
4. No, Friends Over and Get Sleep are not independent since we haven't observed the common ancestor Weekend, and the path between the two is not blocked. Intuitively, having friends over and getting sleep can both be explained by it being the weekend, so knowing about one tells us about the likelihood of the other via this explanation. However, if we already know that it is the weekend, then knowing about one doesn't give us any new information.
5. Yes, Friends Over and Get Sleep are independent given Traveling since observing Traveling blocks the path between the two. Intuitively, if we're getting sleep or having friends over, then there's a chance that it's the weekend, and observing one tells us about the other. However, having friends over and traveling can also be explained by it being the weekend. Since we already know that the person's traveling, whether or not they got sleep tells us nothing about the chance they have friends over. Similarly having friends over has no effect on whether they slept since we know about whether they're traveling, and the latter sort of overrides the information provided by finding out about whether they have friends over.
6. The modified network is:



7. Yes, Friends Over and Get Sleep are independent since the path between them is blocked. The path would no longer blocked if we observed Eat Exotic Foods.

Name

Christy Jestin

Collaborators and Resources

I worked with David Qian and used Google for syntax.

Calibration

I spent about 16 hours.