Russell Kunes
`russellkunes@college.harvard.edu`
CS181-S17

Assignment #4
Due: 5:00pm March 31, 2017

Collaborators: John Doe, Fred Doe

Homework 4: Clustering and EM

This homework assignment focuses on different unsupervised learning methods from a theoretical and practical standpoint. In Problem 1, you will explore Hierarchical Clustering and experiment with how the choice of distance metrics can alter the behavior of the algorithm. In Problem 2, you will derive from scratch the full expectation-maximization algorithm for fitting a simple topic model. In Problem 3, you will implement K-Means clustering on a dataset of handwritten images and analyze the latent structure learned by this algorithm.

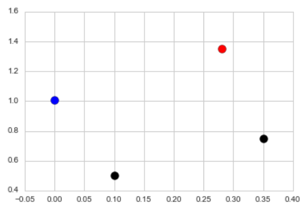
There is a mathematical component and a programming component to this homework. Please submit your PDF and Python files to Canvas, and push all of your work to your GitHub repository. If a question requires you to make any plots, please include those in the writeup.

Hierarchical Clustering [7 pts]

At each step of hierarchical clustering, the two most similar clusters are merged together. This step is repeated until there is one single group. We saw in class that hierarchical clustering will return a different result based on the pointwise-distance and cluster-distance that is is used. In this problem you will examine different choices of pointwise distance (specified through choice of norm) and cluster distance, and explore how these choices change how the HAC algorithm runs on a toy data set.

Problem 1

Consider the following four data points in $\mathbb{R}^2$, belonging to three clusters: the black cluster consisting of $\mathbf{x}_1 = (0.1, 0.5)$ and $\mathbf{x}_2 = (0.35, 0.75))$, the red cluster consisting of $\mathbf{x}_3 = (0.28, 1.35)$, and the blue cluster consisting of $\mathbf{x}_4 = (0, 1.01)$.



Different pointwise distances $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_p$ can be used. Recall the definition of the ℓ_1, ℓ_2, and ℓ_∞ norm:

$$\|\mathbf{x}\|_1 = \sum_{j=1}^{m} |x_i| \qquad \|\mathbf{x}\|_2 = \sqrt{\sum_{j=1}^{m} x_i^2} \qquad \|\mathbf{x}\|_\infty = \max_{j \in \{1,\ldots,m\}} |x_j|$$

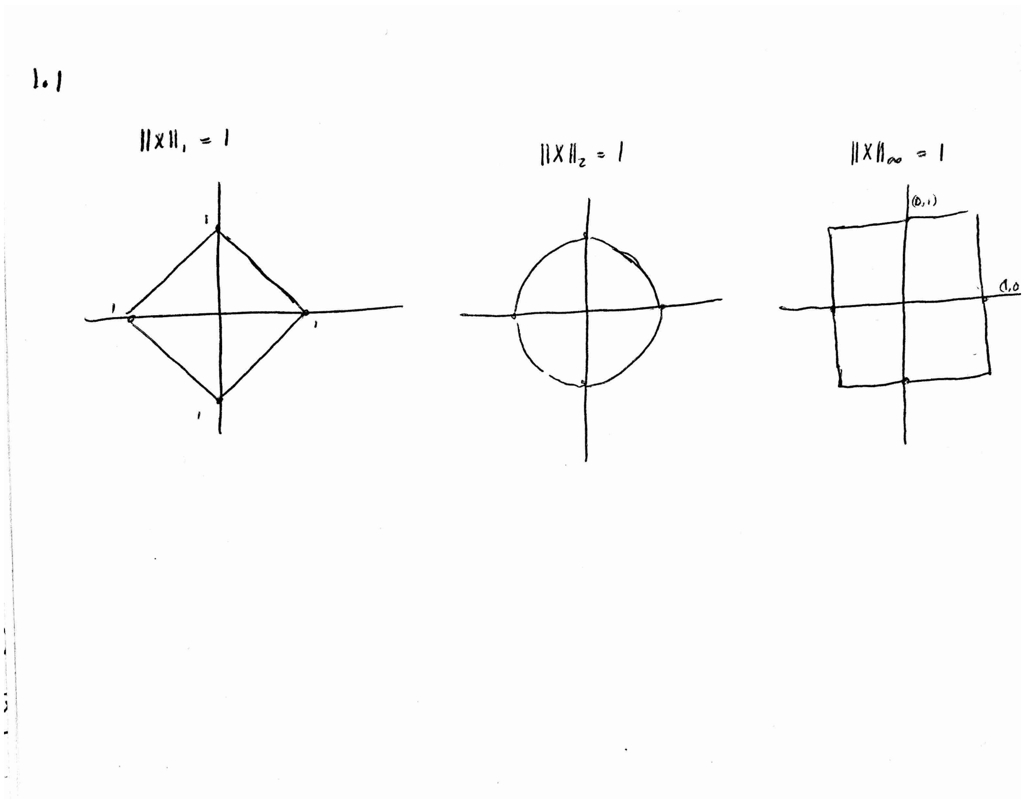
Also recall the definition of min-distance, max-distance, centroid-distance, and average-distance between two clusters (where $\boldsymbol{\mu}_G$ is the center of a cluster G):

$$\begin{aligned} d_{\min}(G, G') &= \min_{\mathbf{x} \in G, \mathbf{x}' \in G'} d(\mathbf{x}, \mathbf{x}') \\ d_{\max}(G, G') &= \max_{\mathbf{x} \in G, \mathbf{x}' \in G'} d(\mathbf{x}, \mathbf{x}') \\ d_{\text{centroid}}(G, G') &= d(\boldsymbol{\mu}_G, \boldsymbol{\mu}_{G'}) \\ d_{\text{avg}}(G, G') &= \frac{1}{|G||G'|} \sum_{\mathbf{x} \in G} \sum_{\mathbf{x}' \in G'} d(\mathbf{x}, \mathbf{x}') \end{aligned}$$

1. Draw the 2D unit sphere for each norm, defined as $\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^2 : \|\mathbf{x}\| = 1\}$. Feel free to do it by hand, take a picture and include it in your pdf.
2. For each norm ($\ell_1, \ell_2, \ell_\infty$) and each clustering distance, specify which two clusters would be the first to merge.
3. Draw the complete dendrograms showing the order of agglomerations for the ℓ_2 norm and each of the clustering distances.

Solution

1.



2. **L1 Norm:** First computing the distances between points, we have:

$$d(x_3, x_4) = 0.62$$

$$d(x_3, x_1) = 1.03$$

$$d(x_4, x_1) = 0.61$$

$$d(x_3, x_2) = 0.67$$

$$d(x_4, x_2) = 0.61$$

Inspecting these we have that for $\mathbf{d_{min}}$, (x_1, x_2) and x_4 merge first, with $\mathbf{d_{min}} = 0.61$. For the other distance metrics (x_1, x_2) and x_4 merge first, with $d = 0.61$. **L2 Norm:** First computing distances:

$$d(x_3, x_4) = 0.4404$$

$$d(x_3, x_1) = .8689$$

$$d(x_4, x_1) = 0.5197$$

$$d(x_3, x_2) = 0.6041$$

$$d(x_4, x_2) = 0.4360$$

Inspecting these we have that for $\mathbf{d_{min}}$, (x_1, x_2) and x_4 merge first, with $\mathbf{d_{min}} = 0.4360$. For $\mathbf{d_{max}}$, $\mathbf{d_{centroid}}$, $\mathbf{d_{avg}}$, x_3 and x_4 merge first with distance of .4404. For the L_∞ norm, we have distances of:

$$d(x_3, x_4) = 0.34$$

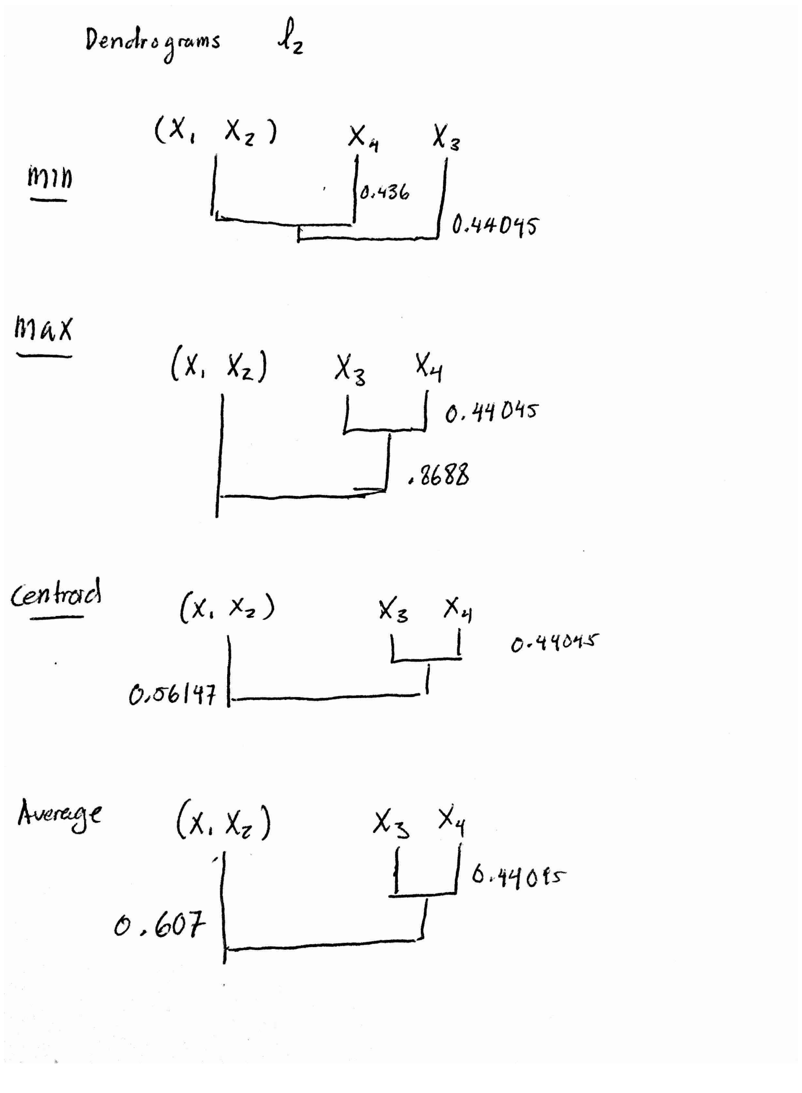
$$d(x_3, x_1) = .85$$

$$d(x_4, x_1) = 0.51$$

$$d(x_3, x_2) = 0.6$$

$$d(x_4, x_2) = 0.35$$

For **min** and **max**, x_3 and x_4 merge first,with distance 0.34. The **centroid** and **avg** distances also have x_3 and x_4 merge first. 3.



Topic Modeling [15 pts]

In this problem we will explore a simplified version of topic modeling in which each document has just a *single* topic. For this problem, we will assume there are c topics. Each topic $k \in \{1, \dots, c\}$ will be associated with a vector $\boldsymbol{\beta}_k \in [0,1]^{|\mathcal{V}|}$ describing a distribution over the vocabulary $\mathcal{V}$ with $\sum_{j=1}^{|\mathcal{V}|} \beta_{k,j} = 1$.

Each document $\mathbf{x}_i$ will represented as a bag-of-words $\mathbf{x}_i \in (\mathbb{Z}^{\geq 0})^{|\mathcal{V}|}$, where $x_{i,j}$ is a non-negative integer representing the number of times word j appeared in document i. Document i has n_i word tokens in total. Finally let the (unknown) overall mixing proportion of topics be $\boldsymbol{\theta} \in [0,1]^c$, where $\sum_{k=1}^{c} \theta_k = 1$.

Our generative model is that each of the n documents has a single topic. We encode this topic as a one-hot vector $\mathbf{z}_i \in \{0,1\}^c$ over topics. This one-hot vector is drawn from $\boldsymbol{\theta}$; then, each of the words is drawn from $\boldsymbol{\beta}_{z_i}$. Formally documents are generated in two steps:

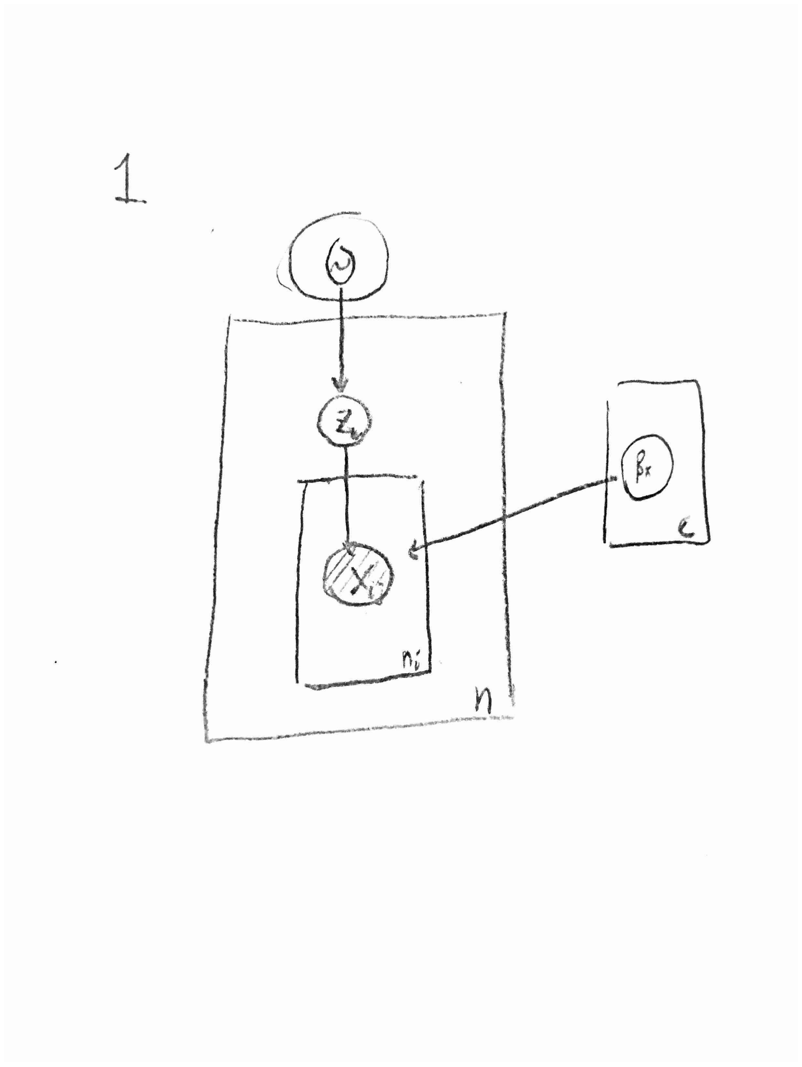
$$
\begin{aligned}
\mathbf{z}_i &\sim \text{Categorical}(\boldsymbol{\theta}) \\
\mathbf{x}_i &\sim \text{Multinomial}(\boldsymbol{\beta}_{\mathbf{z}_i})
\end{aligned}
$$

Problem 2

1. Draw the graphical model representation of this problem. Be sure to use the plate notation to indicate repeated random variables and gray nodes to indicate observed variables.
2. **Complete-Data Log Likelihood** Define the complete data for this problem to be $D = \{(\mathbf{x}_i, \mathbf{z}_i)\}_{i=1}^n$.
   - Write out the complete-data (negative) log likelihood.
   $$\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c) = -\ln p(D \,|\, \boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c).$$
   - Explain in one sentence why we cannot directly optimize this loss function.
3. **Expectation Step** Our next step is to introduce a mathematical expression for $\mathbf{q}_i$, the posterior over the hidden topic variables $\mathbf{z}_i$ conditioned on the observed data $\mathbf{x}_i$ with fixed parameters, i.e $p(\mathbf{z}_i | \mathbf{x}_i; \boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c)$.
   - Write down and simplify the expression for $\mathbf{q}_i$.
   - Give an algorithm for calculating $\mathbf{q}_i$ for all i, given the observed data $\{\mathbf{x}_i\}_{i=1}^n$ and settings of the parameters $\boldsymbol{\theta}$ and $\{\boldsymbol{\beta}_k\}_{k=1}^c$.
4. **Maximization Step** Using the $\mathbf{q}_i$ estimates from the Expectation Step, derive an update for maximizing the expected complete data log likelihood in terms of $\boldsymbol{\theta}$ and $\{\boldsymbol{\beta}_k\}_{k=1}^c$.
   - Derive an expression for the expected complete-data log likelihood in terms of $\mathbf{q}_i$.
   - 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 force the constraint $\sum \theta_k = 1$. Why does this optimized $\boldsymbol{\theta}$ make intuitive sense?
   - Apply a similar argument to find the value of the $\boldsymbol{\beta}_k$'s that maximizes the expected complete-data log likelihood.

Solution

1.



2.

$$\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c}) = -\ln p(D \,|\, \boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c})$$

$$\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c}) = -\ln \prod_{i=1}^{n} p(\mathbf{z}_i, \mathbf{x}_i \,|\, \boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c}) = -\ln \prod_{i=1}^{n} p(\mathbf{z}_i \,|\, \boldsymbol{\theta}) p(\mathbf{x}_i \,|\, \mathbf{z}_i, \{\boldsymbol{\beta}_k\}_{k=1}^{c})$$

$$\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c}) = -\ln \prod_{i=1}^{n} [\prod_{k=1}^{c} \theta_k^{z_{ik}} \prod_{k=1}^{c} (\prod_{j=1}^{|V|} \beta_{k,j}^{x_{i,j}})^{z_{ik}}]$$

$$\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c}) = -\sum_{i=1}^{n} \sum_{k=1}^{c} z_{i,k} \ln \theta_k - \sum_{i=1}^{n} \sum_{k=1}^{c} \sum_{j=1}^{|V|} z_{i,k} x_{i,j} \ln \beta_{k,j}$$

We can't directly optimize this loss function because the z variable is unknown; instead we have to optimize the marginal log likelihood for the x variables.

3. Using Bayes rule:

$$\mathbf{q}_i = p(\mathbf{z}_i \,|\, \mathbf{x}_i; \theta, \{\boldsymbol{\beta}_k\}_{k=1}^c)$$

And

$$q_{ik} = p(\mathbf{z}_i = C_k \,|\, \mathbf{x}_i; \theta, \{\boldsymbol{\beta}_k\}_{k=1}^c) \propto p(\mathbf{x}_i \,|\, \mathbf{z}_i = C_k; \theta, \{\boldsymbol{\beta}_k\}_{k=1}^c) p(\mathbf{z}_i = C_k \,|\, \theta)$$

So

$$q_{ik} \propto \theta_k \prod_{j=1}^{|V|} \beta_{k,j}^{x_{i,j}}$$

Also we know $\sum_k q_{ik} = 1$. So we must have

$$q_{ik} = \frac{\theta_k \prod_{j=1}^{|V|} \beta_{k,j}^{x_{i,j}}}{\sum_{k'=1} \theta_{k'} \prod_{j=1}^{|V|} \beta_{k',j}^{x_{i,j}}}$$

Taking these quantities as a vector, we have $\mathbf{q}_i$. To compute this, we coompute the product in the numerator for each B_k and θ_k, in $c|V|$ steps, and store these quantities. Then we take the sum of all the quantities, to normalize to vector that sums to 1.

4. The negative log likelihood is:

$$\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c) = -\sum_{i=1}^n \sum_{k=1}^c z_{i,k} \ln \theta_k - \sum_{i=1}^n \sum_{k=1}^c \sum_{j=1}^{|V|} z_{i,k} x_{i,j} \ln \beta_{k,j}$$

We also have

$$E(z_{i,k}) = p(z_{i,k} = 1) = q_{i,k}$$

As $z_{i,k}$ is an indicator random variable. Taking the expectation:

$$E(\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c)) = -\sum_{i=1}^n \sum_{k=1}^c q_{i,k} \ln \theta_k - \sum_{i=1}^n \sum_{k=1}^c \sum_{j=1}^{|V|} q_{i,k} x_{i,j} \ln \beta_{k,j}$$

Introducing a lagrange multiplier, λ, and taking a derivative with respect to θ_k and setting to 0:

$$\frac{\partial}{\partial \theta_k} E(\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c)) = \frac{\partial}{\partial \theta_k} [-\sum_{i=1}^n \sum_{k=1}^c q_{i,k} \ln \theta_k + \lambda(\sum_k \theta_k - 1)]$$

$$0 = -\sum_{i=1}^n \frac{q_{i,k}}{\theta_k} + \lambda$$

$$\theta_k = \sum_{i=1}^n \frac{q_{i,k}}{\lambda}$$

Taking a derivative with respect to λ and setting to 0 gives $\sum_k \theta_k = 1$. So

$$\sum_k^c \theta_k = \theta_k = \sum_k^c \sum_{i=1}^n \frac{q_{i,k}}{\lambda} = 1$$

$$\lambda = \sum_{k}^{c} \sum_{i=1}^{n} q_{i,k}$$

$$\hat{\theta_k} = \frac{\sum_{i=1}^{n} q_{i,k}}{\sum_{k}^{c} \sum_{i=1}^{n} q_{i,k}} = \frac{\sum_{i=1}^{n} q_{i,k}}{n}$$

θ_k is the overall distribution of the topics. This value makes sense since it is the average probability of being in topic k over the documents.
To solve for MLE for B_k, add a lagrange multiplier λ term to enforce that the β's sum to 1:

$$E(\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c})) = -\sum_{i=1}^{n} \sum_{k=1}^{c} q_{i,k} \ln \theta_k - \sum_{i=1}^{n} \sum_{k=1}^{c} \sum_{j=1}^{|V|} q_{i,k} x_{i,j} \ln \beta_{k,j} + \lambda(\sum_{j=1}^{|V|} \beta_{k,j} - 1)$$

Taking a derivative with respect to $\beta_{j,k}$ and setting to 0

$$\frac{\partial}{\partial \beta_{k,j}} E(\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c})) = \frac{\partial}{\partial \beta_{k,j}} [-\sum_{i=1}^{n} \sum_{k=1}^{c} \sum_{j=1}^{|V|} q_{i,k} x_{i,j} \ln \beta_{k,j} + \lambda(\sum_{j=1}^{|V|} \beta_{k,j} - 1)]$$

$$\frac{\partial}{\partial \beta_{k,j}} E(\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c})) = \sum_{i=1}^{n} \frac{q_{i,k} x_{i,j}}{\beta_{k,j}} - \lambda$$

$$\beta_{k,j} = \sum_{i=1}^{n} \frac{q_{i,k} x_{i,j}}{\lambda}$$

We also have

$$\sum_{j=1}^{|V|} \sum_{i=1}^{n} q_{i,k} x_{i,j} = \lambda$$

So,

$$\beta_{k,j} = \frac{\sum_{i=1}^{n} q_{i,k} x_{i,j}}{\sum_{j=1}^{|V|} \sum_{i=1}^{n} q_{i,k} x_{i,j}} = \frac{\sum_{i=1}^{n} q_{i,k} x_{i,j}}{\sum_{i=1}^{n} q_{i,k} n_i}$$

So,

$$\hat{\boldsymbol{\beta}_k} = \frac{\sum_{i=1}^{n} q_{i,k} \mathbf{x}_i}{\sum_{i=1}^{n} q_{i,k} n_i}$$

This makes intuitive sense because the numerator sums up the total probability weight of x_is in class k, and the denominator normalizes this to sum to 1. For example, if $q_{i,k}$ was one hot, we would have the sum of the $\mathbf{x}_i$ that are in class k, divided by the total number of words for class k.

K-Means [15 pts]

For this problem you will implement K-Means clustering from scratch. Using `numpy` is fine, but don't use a third-party machine learning implementation like `scikit-learn`. You will then apply this approach to clustering of image data.

We have provided you with the MNIST dataset, a collection of handwritten digits used as a benchmark of image recogntion (you can learn more about the data set at `http://yann.lecun.com/exdb/mnist/`). The MNIST task is widely used in supervised learning, and modern algorithms with neural networks do very well on this task.

Here we will use MNIST unsupervised learning. You have been given representations of 6000 MNIST images, each of which are 28×28 greyscale handwritten digits. Your job is to implement K-means clustering on MNIST, and to test whether this relatively simple algorithm can cluster similar-looking images together.

Problem 3

The given code loads the images into your environment as a 6000x28x28 array.

- Implement K-means clustering from different random initializations and for several values of K using the ℓ_2 norm as your distance metric. (You should feel free to explore other metrics than the ℓ_2 norm, but this is strictly optional.) Compare the K-means objective for different values of K and across random initializations.
- For three different values of K, and a couple of random restarts for each, show the mean images for each cluster (i.e., for the cluster prototypes), as well as the images for a few representative images for each cluster. You should explain how you selected these representative images. To render an image, use the pyplot `imshow` function.
- Are the results wildly different for different restarts and/or different values of K? For one of your runs, plot the K-means objective function as a function of iteration and verify that it never increases.

As in past problem sets, please include your plots in this document. (There may be several plots for this problem, so feel free to take up multiple pages.)

Solution

Trying different values for K and tracking the K-means objective value, for different types of random initialization:

1) **Uniform(0,5)** initialization:

$$K = 5 : 9933626.8408$$

$$K = 10 : 9345477.84417$$

$$K = 15 : 9015388.60795$$

$$K = 20 : 8794630.07115$$

$$K = 25 : 8618765.40819$$

2) **Exponential(1)** initialization:

$$K = 5 : 9964434.44138$$

$$K = 10 : 9356472.53912$$

$$K = 15 : 9044526.41185$$

$$K = 20 : 8803002.36919$$

3) **Exponential(i)** initialization with $i \in [1, 28]$:

$$K = 5 : 9964750.39713$$

$$K = 10 : 9356246.4973$$

$$K = 15 : 9027478.82675$$

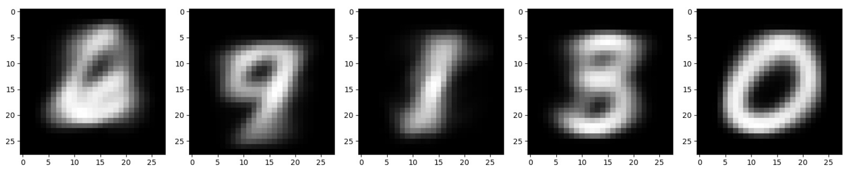
$$K = 20 : 8785608.68617$$

As expected, as K goes up the objective function goes down, since every data point gets closer to its assigned cluster. There isn't much noticable difference between the different types of random initialization.

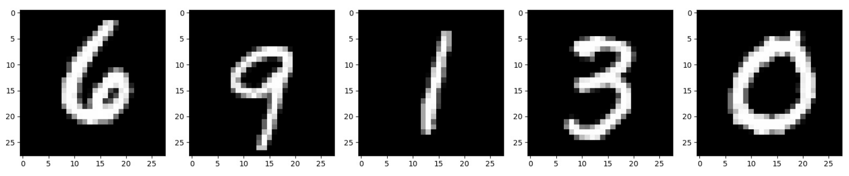
For K=5,K=10, and K=15, I got the mean cluster images, and the images for the most representative data point. The most representative was defined by closest to the cluster center in Euclidean distance.

K=5

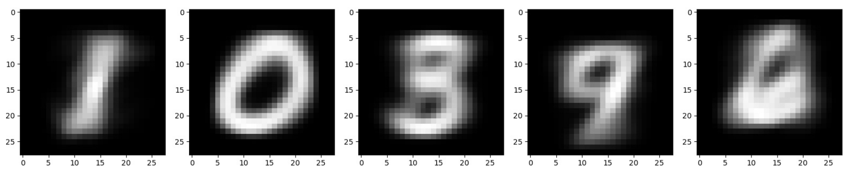
Mean Images 1 (K=5)



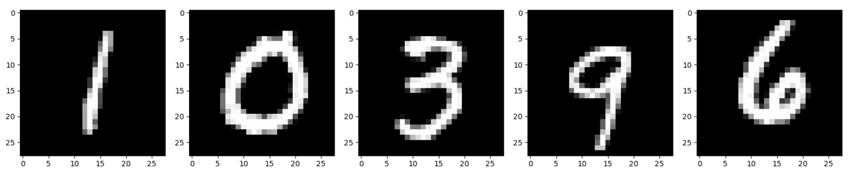
Representative Images 1 (K=5)



Mean Images 2 (K=5)

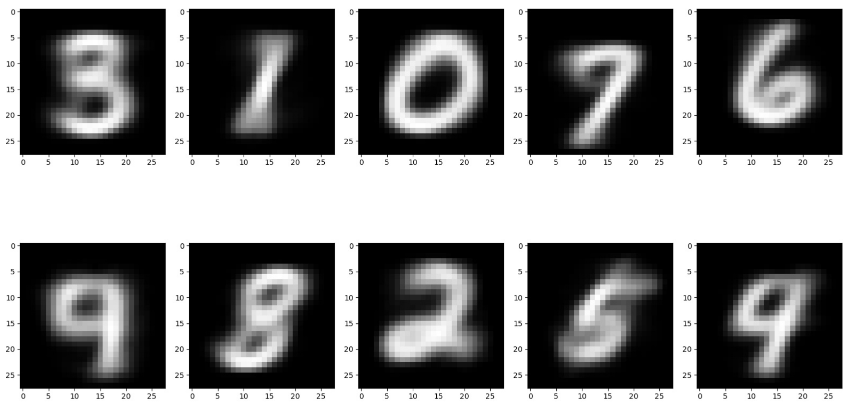


Representative Images 2 (K=5)

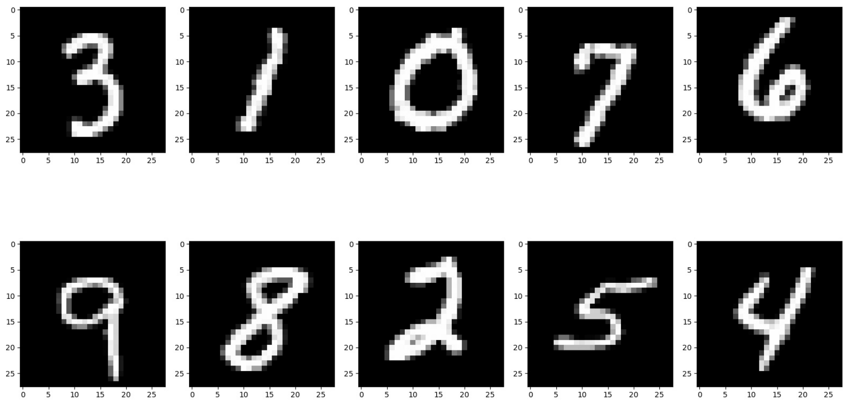


K=10

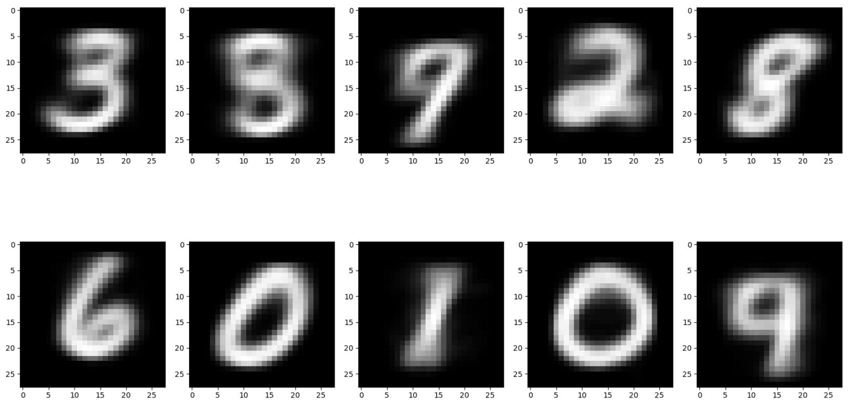
Mean Images 1 (K=10)



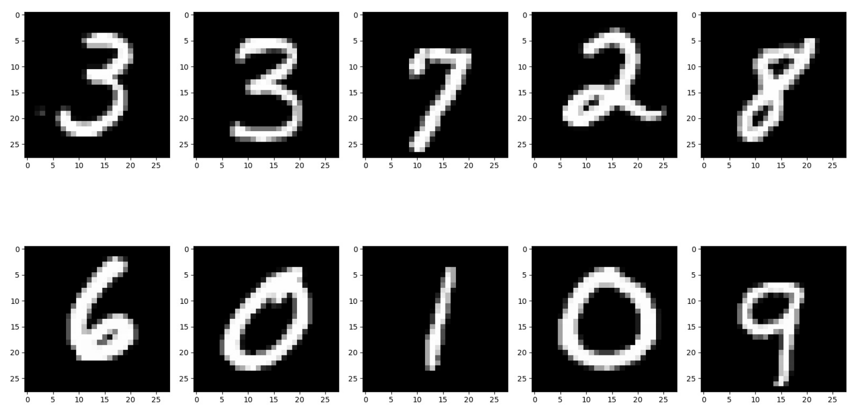
Representative Images 1 (K=10)



Mean Images 2 (K=10)

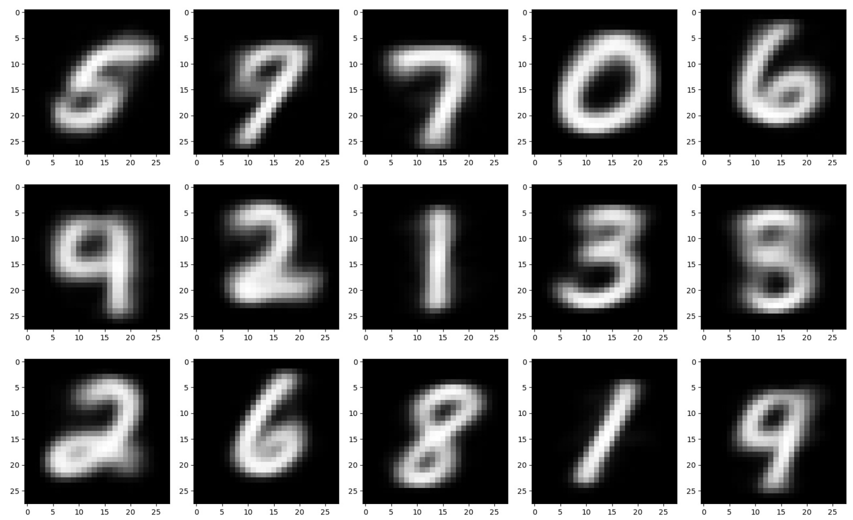


Representative Images 2 (K=10)

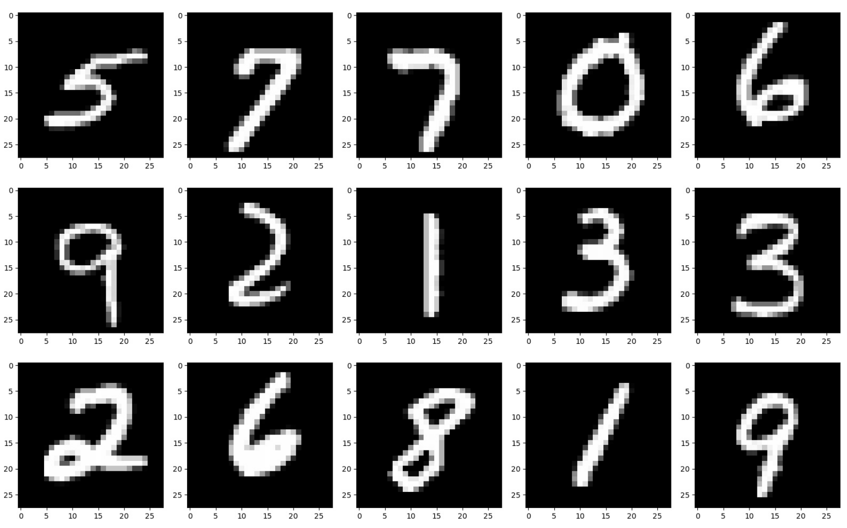


K=15

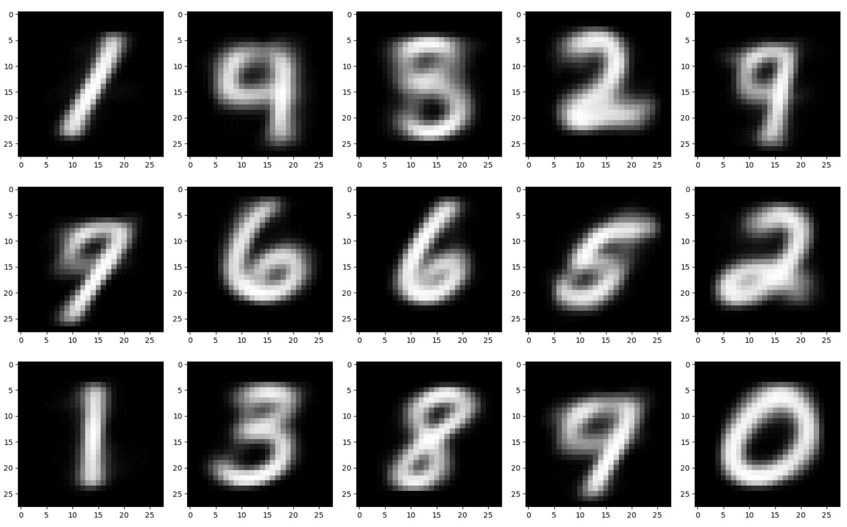
Mean Images 1 (K=15)



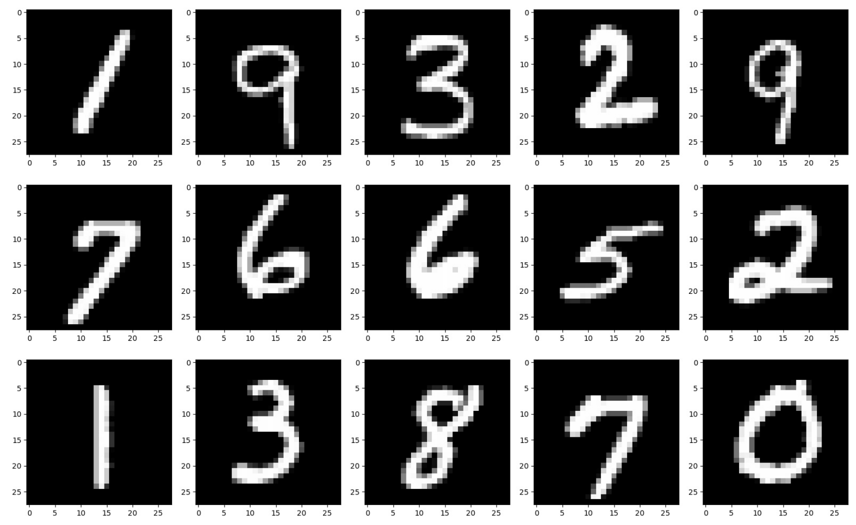
Representative Images 1 (K=15)



Mean Images 2 (K=15)

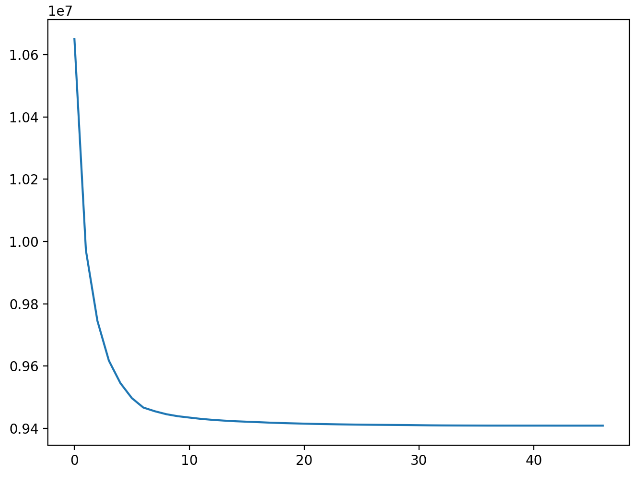


Representative Images 2 (K=15)



The results tend to vary a lot for each random restart. For K=5, we get very blurry means, and for higher values of K, we get less blurry means. There tends to be more variation for the higher values of K for what clusters we get.

Loss Function plotted over iterations



The loss function goes down monotonically, as expected.

Problem 4 (Calibration, 1pt)

Approximately how long did this homework take you to complete? 10 hours