David Hughes
davidralphhughes@college.harvard.edu
CS181-S18

Assignment #4
Due: 11:59pm March 30, 2018

Collaborators: Alexander Munoz

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 Gaussian mixture 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.

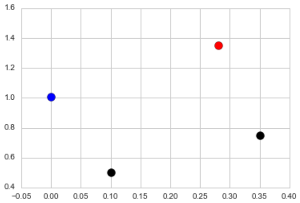
There is a mathematical component and a programming component to this homework. Please submit your PDF and Python files to Canvas, and push all of your work to your GitHub repository. If a question requires you to make any plots, please include those in the writeup.

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:

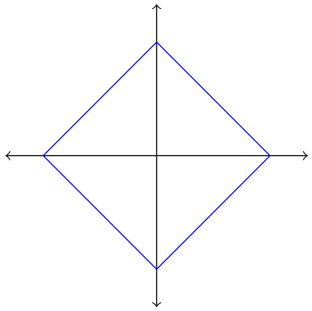
$$\|\mathbf{x}\|_1 = \sum_{j=1}^{m} |x_i| \qquad \|\mathbf{x}\|_2 = \sqrt{\sum_{j=1}^{m} x_i^2} \qquad \|\mathbf{x}\|_\infty = \max_{j \in \{1,\ldots,m\}} |x_j|$$

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):

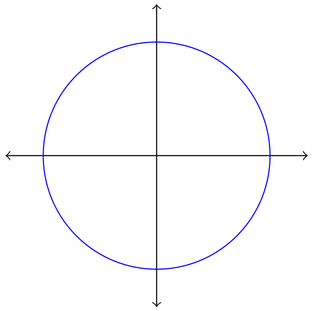
$$\begin{aligned}
d_{\min}(G, G') &= \min_{\mathbf{x} \in G, \mathbf{x}' \in G'} d(\mathbf{x}, \mathbf{x}') \\
d_{\max}(G, G') &= \max_{\mathbf{x} \in G, \mathbf{x}' \in G'} d(\mathbf{x}, \mathbf{x}') \\
d_{\text{centroid}}(G, G') &= d(\boldsymbol{\mu}_G, \boldsymbol{\mu}_{G'}) \\
d_{\text{avg}}(G, G')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

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. We have provided some code to make this easier for you. You are not required to use it.

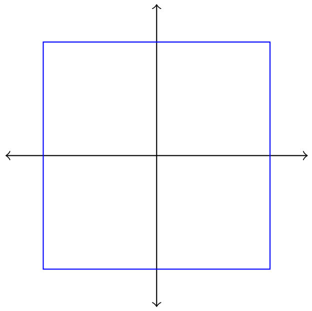
Solution

1. 

ℓ_1



ℓ_2



ℓ_∞

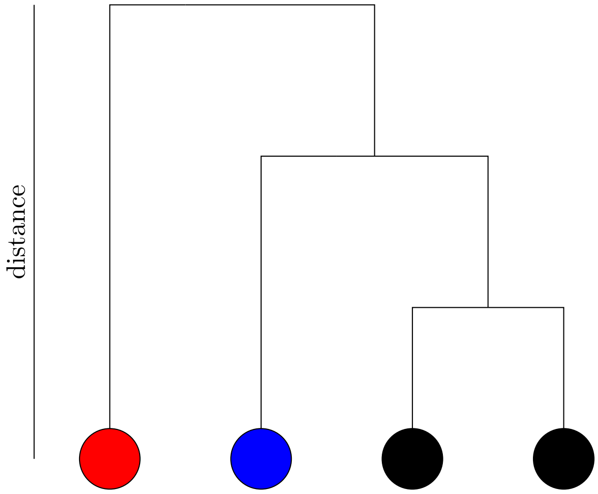
Each intersection with the x or y axis occurs one unit away from the origin.

2.

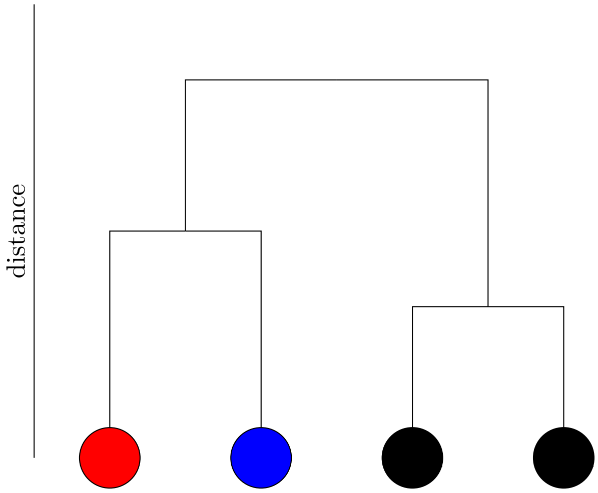
Distance metric	ℓ_1	ℓ_2	ℓ_∞
d_{min}	black-blue	black-blue	red-blue
d_{max}	black-blue	red-blue	red-blue
d_{avg}	black-blue	red-blue	red-blue
$d_{centroid}$	black-blue	red-blue	red-blue

3.

Using d_{min} :



Using $d_{max}, d_{avg}, d_{centroid}$:



Expectation-Maximization for Gaussian Mixture Models [7pts]

In this problem we will explore expectation-maximization for the Gaussian Mixture model. Each observation $\mathbf{x}_i$ is a vector in $\mathbb{R}^D$. We posit that each observation comes from *one* mixture component. For this problem, we will assume there are c components. Each component $k \in \{1, \dots, c\}$ will be associated with a mean vector $\mu_k \in R^D$ and a covariance Σ_k. Finally let the (unknown) overall mixing proportion of the components be $\boldsymbol{\theta} \in [0,1]^c$, where $\sum_{k=1}^{c} \theta_k = 1$.

Our generative model is that each of the n observations comes from a single component. We encode observation i's component-assignment as a one-hot vector $\mathbf{z}_i \in \{0,1\}^c$ over components. This one-hot vector is drawn from $\boldsymbol{\theta}$; then, $\mathbf{x}_i$ is drawn from $N(\mu_{z_i}, \Sigma_{z_i})$. Formally documents are generated in two steps:

$$\begin{aligned} \mathbf{z}_i &\sim \text{Categorical}(\boldsymbol{\theta}) \\ \mathbf{x}_i &\sim N(\mu_{z_i}, \Sigma_{z_i}) \end{aligned}$$

Problem 2

1. **Intractability of the Data Likelihood** Let ϕ_k represent all the parameters associated with a component (μ_k, Σ_k). We are generally interested in finding a set of parameters ϕ_k that maximize the data likelihood $\log p(\{x_i\}|\{phi_k\})$. Expand the data likelihood to include the necessary sums over observations x_i and latents z_i. Why is optimizing this loss directly intractable?

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.

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

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 $p(\mathbf{z}_i|\mathbf{x}_i; \boldsymbol{\theta}, \{\mu_k, \Sigma_k\}_{k=1}^c)$.

   - 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 $\{\mu_k, \Sigma_k\}_{k=1}^c$.

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 $\{\mu_k, \Sigma_k\}_{k=1}^c$.

   - 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 (μ_k, Σ_k)'s that maximizes the expected complete-data log likelihood.

5. Finally, compare this EM approach to the generative model for classification in Homework 2. How are the computations similar? Different?

Solution

1.

$$\begin{aligned}\log p(x|\phi) &= \log \sum_z p(x, z|\phi) \\ &= \sum_{n=1}^{N} \log \sum_z p(x_n, z_n|\phi) \\ &= \sum_{n=1}^{N} \log \sum_z p(x_n, z_n|\phi) + \log p(z_n|\phi) \\ &= \sum_{n=1}^{N} N(x_n|\mu_{z_n}, \Sigma_{z_n}) + \log \sum_z \mathrm{Cat}(z_n|\phi) \\ &= \sum_{n=1}^{N} N(x_n|\mu_{z_n}, \Sigma_{z_n}) + \log \sum_z \theta_{z_n}\end{aligned}$$

This is intractable because we have a sum inside the log, therefore there is no closed expression for the MLE.

2.

$$\begin{aligned}\log p(x, z|\phi) &= \sum_{n=1}^{N} \log p(x_n, z_n|\phi) \\ &= \sum_{n=1}^{N} \log p(x_n, z_n|\phi) + \log p(z_n|\phi) \\ &= \sum_{n=1}^{N} \log N(x_n|\mu_{z_n}, \Sigma_{z_n}) + \log \mathrm{Cat}(z_n|\phi) \\ &= \sum_{n=1}^{N} \log N(x_n|\mu_{z_n}, \Sigma_{z_n}) + \log \theta_{z_n} \\ &= \sum_{n=1}^{N} \sum_{k=1}^{c} z_{n,k} \log N(x_n|\mu_k, \Sigma_k) + z_{n,k} \log \theta_k\end{aligned}$$

3.

$$\begin{aligned}q_i &= p(z_i|x_i, \mu, \Sigma, \theta) \\ &\propto p(x_i|z_i, \mu, \Sigma, \theta)p(z_i|\mu, \Sigma, \theta) \\ &= N(x_i|\mu_{z_i}, \Sigma_{z_i})\theta\end{aligned}$$

Therefore, to find the probability $q_{i,k}$ (the probability that datapoint i comes from cluster k, we multiply the given prior θ_k, times the liklihood of the point being in that Gaussian cluster $N(x_i|\mu_k, \Sigma_k)$). We can look over the data algorithmically and do these calculations easily.

4.

$$\begin{aligned}
E_z[\mathcal{L}_{\text{complete}}(\phi)] &= E_z[\log p(x_n, z_n|\phi)] \\
&= E_z\Bigg[\sum_{n=1}^{N}\sum_{k=1}^{c} z_{n,k}\log\theta_k + z_{n,k}\log N(x_n|\mu_k,\Sigma_k)\Bigg] \\
&= \sum_{n=1}^{N}\sum_{k=1}^{c}(q_{n,k}\log\theta_k + q_{n,k}\log N(x_n|\mu_k,\Sigma_k)) \\
&\to \sum_{n=1}^{N}\sum_{k=1}^{c}(q_{n,k}\log\theta_k + q_{n,k}\log N(x_n|\mu_k,\Sigma_k)) - \lambda\Bigg(\sum_{k}\theta_k - 1\Bigg) \\
\frac{dE_z}{d\phi_k} &= \sum_{n=1}^{N}\frac{q_{n,k}}{\theta_k} - \lambda = 0 \\
\frac{dE_z}{d\lambda} &= \sum_{n=1}^{N}\theta_k = 1 \\
\lambda &= \frac{1}{c}\sum_{n=1}^{N}\sum k = 1^c\frac{q_{n,k}}{\theta_k} = N \\
\hat{\theta}_k &= \frac{\sum_n q_{n,k}}{N}
\end{aligned}$$

5. The MLEs from T2 are similar to these. Instead of averaging over supervised classifications in T2, we average over $q_{n,k}$ from the E-step.

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

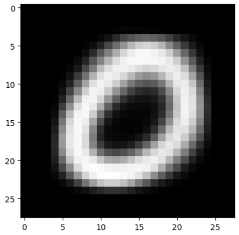
The given code loads the images into your environment as a 6000x28x28 array.

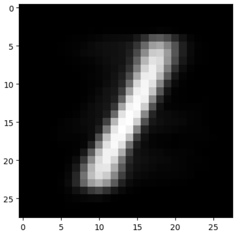
- Implement K-means clustering from different random initializations and for several values of K using the ℓ_2 norm as your distance metric. (You should feel free to explore other metrics than the ℓ_2 norm, but this is strictly optional.) Compare the K-means objective for different values of K and across 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.

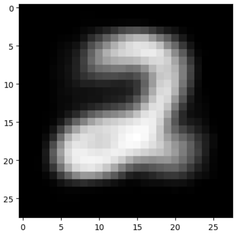
As in past problem sets, please include your plots in this document. (There may be several plots for this problem, so feel free to take up multiple pages.)

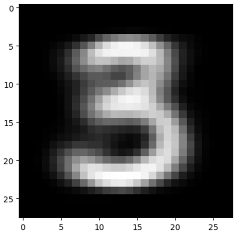
Solution

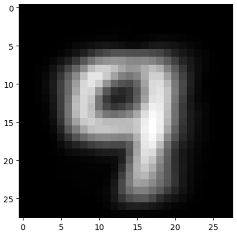
1. The Kmeans objectives vary based on initialization and value of K, and loss goes down for larger values of K.
2. Here are the images for $K = 10$. The conclusion is that higher K makes centroid images more crisp, and can encapsulate differences in writing (such as the cross on the 7). There are some errors, but this is the best I could do.

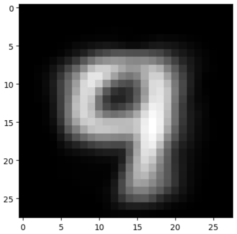


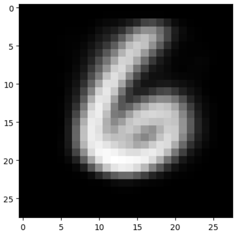

0
5
10
15
20
25
0
5
10
15
20
25

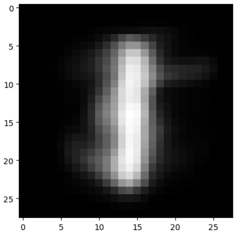

0
5
10
15
20
25
0
5
10
15
20
25

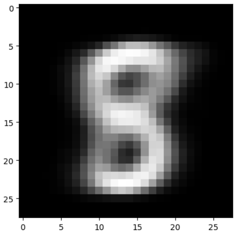

0
5
10
15
20
25
0
5
10
15
20
25

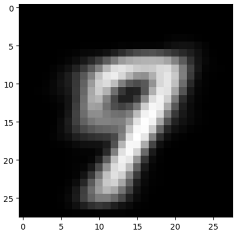

0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25

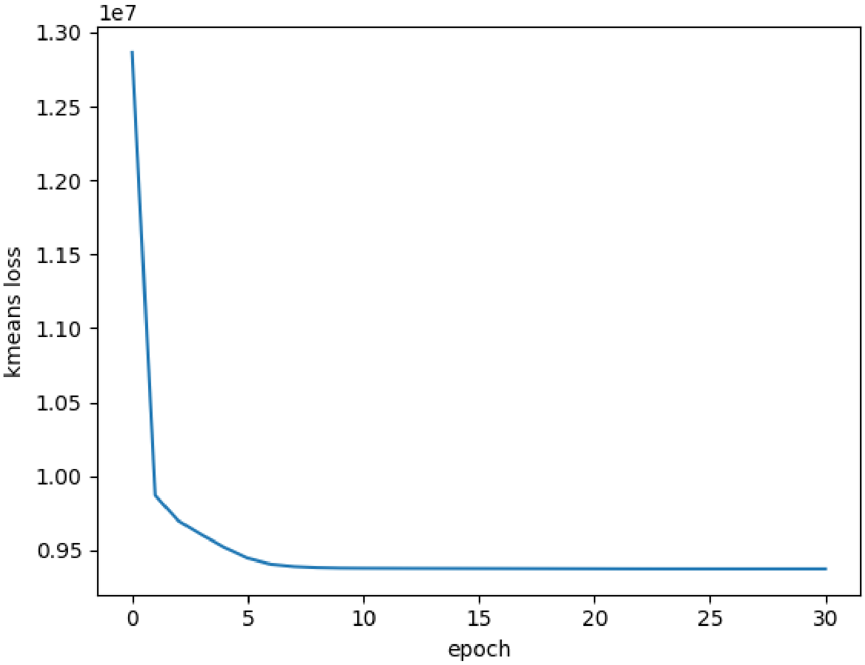

0
5
10
15
20
25
0
5
10
15
20
25







3. Loss does not increase at any iteration.



Problem 4 (Calibration, 1pt)

Approximately how long did this homework take you to complete? 15 hours