Homework 4:
Principal component analysis

In this homework, you will be asked to find the top k principal components, for different values of k. By this, we mean, given a dataset $x_1, \ldots, x_n$, the k *orthonormal* vectors $v_1, \ldots, v_k$ that maximize the objective

$$\sum_{j=1}^{k} \sum_{i=1}^{n} \langle v_j, x_i \rangle^2 \ .$$

Recall, as discussed in lecture, that these vectors may not be unique, but they will be for the datasets we consider here. You may use the Python `sklearn` package to compute these components.

Problem 1: PCA for mixtures of 2 Gaussians [28 points]

We saw in class that PCA can be a very useful tool for learning *mixture models*. In this homework problem, we will explore this in more detail.

A mixture of k (spherical) Gaussians (a.k.a. a k-Gaussian mixture model, or k-GMM for short) is a distribution over $\mathbb{R}^d$ specified by k mean vectors $\mu_1, \ldots, \mu_k \in \mathbb{R}^d$, and k mixing weights $w_1, \ldots, w_k$ which satisfy that $w_i \geq 0$ for all $i = 1, \ldots, k$, and $\sum_{i=1}^{k} w_i = 1$. In other words, the mixing weights specify a probability distribution over k elements. To sample from a k-GMM, we perform the following 2-step procedure:

1. Choose j from $\{1, \ldots, k\}$ with probability w_i, and then
2. Sample a point $X \sim \mathcal{N}(\mu_j, I)$.

We call $\mathcal{N}(\mu_j, I)$ the *j-th component* of the mixture. For most of the problem, we will consider the $k = 2$ setting of this problem, so our distribution will be a mixture of two Gaussians. We'll also assume that the mixing weights are *uniform*, so $w_j = 1/k$ for all $j = 1, \ldots, k$, or for the case of $k = 2$, $w_1 = w_2 = 1/2$.

There are two very related problems we will consider in this setting:

1. **Clustering:** Given a set of points $X_1, \ldots, X_n$ from a k-GMM, for all (or almost all) i, recover which component X_i was sampled from.
2. **Parameter learning:** Given a set of points $X_1, \ldots, X_n$ from a k-GMM, learn the means $\mu_1, \ldots, \mu_k$.

(a) [2 points] Consider first what happens in the 1-dimensional setting. Let $\mu_1 = a$ and $\mu_2 = -a$ for some value $a \in \mathbb{R}$. Suppose I am given n samples $X_1, \ldots, X_n$ from a uniform mixture of two Gaussians with means μ_1 and μ_2. Give a simple clustering algorithm for this setting, and argue informally that the clustering algorithm you have is optimal. Let $n = 100$, and for $a \in \{0.1, 1, 10, 100\}$, plot the fraction of mis-clustered points as a function of a.

(b) [2 points] For the same setting, give a simple estimator for $|a|^2$ as a function of $|X_i|^2$, for $i = 1, \ldots, n$, and briefly explain why your estimator should be correct. (Note that in this case, recovering $|a|^2$ is more or less equivalent to parameter recovery.) Let $n = 100$, and for $a \in \{0.1, 0.2, \ldots, 3.9, 4.0\}$, plot the error of your estimator as a function of a. What trends do you observe, and why?

We'll now move on to the significantly more challenging high-dimensional setting. For the rest of the problem, let $d = 1000$, and let v be a vector whose coordinates are $\pm 1/\sqrt{d}$ uniformly at random. Note that v is always a unit vector, i.e. $\|v\|_2 = 1$. Then, for some scale parameter $a \geq 0$, let $\mu_1 = av$ and $\mu_2 = -av$.

(c) [4 points] A natural attempt at generalizing the 1D clustering algorithm above is to perform the 1D algorithm along every coordinate, and to somehow agglomerate the results to generate an overall clustering. Informally explain why this should be hard to do when a is small. Based on your simulation results from part (a), is there a value of a at which this should start working?

(d) [4 points] Now let's turn to the task of parameter recovery, or actually, the even simpler task of trying to guess a. A natural attempt to generalize the 1D algorithm from part (b) is to come up with an estimate of $|a|^2$ as a function of $\|X_i\|_2^2$, for $i = 1, \ldots, n$. Let $n = 4000$, and for $a \in \{0.1, 0.2, \ldots, 3.9, 4.0\}$, plot the error of your estimator as a function of a. How does this compare to the error in part (b)? How does it scale as you vary d? Can you explain why?

(e) [4 points] Finally, let's try using PCA to do better! For the same setting as in paragraph (d), run PCA to output the top principal component of the dataset. Then, run the 1D estimator from part (b) on the dataset projected onto this principal component. Produce two plots: (1) plot the squared inner product between the principal component and v as a function of a, and (2) plot the error of the 1D estimator as a function of a. What trends do you see? How does this compare to the estimator from part (d)?

(f) [4 points] How would you expect the graphs from part (e) to change as (1) we fix d and let $n \to \infty$, vs (2) we fix n and let $d \to \infty$? Can you briefly explain why?

(g) [4 points] How might you extend this algorithm for clustering a mixture of 3 spherical Gaussians? What are the key components that would need to change? Can you offer some plausible generalizations of the techniques covered here that would do so?

(h) [4 points] Now, consider a setting where, before we see the data, an adversary can do the following procedure to any data point: if X_i is sampled from $\mathcal{N}(\mu_j, I)$, it can move any coordinate of X_i closer to the corresponding coordinate of μ_j, by any arbitrary amount it wants. So, for instance, it could make $X_i = \mu_j$ for all of the samples coming from the j-th component. Now, suppose that v can be an arbitrary unit vector, and let $\mu_j = \pm av$, as before, for $j = 1, 2$. How does the behavior of PCA change, as a function of a?

Problem 3: PCA for genetic data [26 points]

The file `pca-data.txt` on the course webpage contains data from the 100 genomes project. Each of the lines in the file represents an individual. The first three columns contain: an individual's unique identifiers, his or her biological sex (1=male, 2=female), and the population to which they belong. The encodings for these populations can also be found in the course webpage. The subsequent columns of each line are a sample of the nucleobases from that individual's genome.

Convert the file into a matrix as follows. Let d be the number of columns, and n be the number of rows. For every column $j = 1, \ldots, d$, let η_j denote the most common nucleotide in that column. Call that the *mode* for column j. Construct an $n \times d$ matrix X as follows. For individual i, let $X_{ij} = 0$ if individual i has

η_j in position j, and 1 otherwise. In other words, individual i has a 1 at column j if they have a mutation at position j.

Recall that if we are going to perform PCA on a real-world dataset of vectors $x_1, \ldots, x_n \in \mathbb{R}^d$, then we typically want to first *center* then, i.e. replace them with $x_1 - \mu, x_2 - \mu, \ldots, x_n - \mu$, where

$$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i \, .$$

For the rest of this problem, work on a centered version of the dataset X.

(a) [4 points] We will first examine the first 2 principal components of X. These components contain lots of information about our data set. Create a scatter plot with each of the 995 rows of X projected onto the first two principal components. In other words, the horizontal axis should be v_1 and the vertical axis v_2, and each individual should be projected onto the subspace spanned by v_1 and v_2. Your plot should use a different color for each population and include a legend. (Recall that the population data occurs in the third column.)

(b) [6 points] In two sentences, list 1 or 2 basic facts about the plot created in part (b). Can you interpret the first two principal components? What aspects of the data do the first two principal components capture?

(c) [5 points] We will now examine the third principal component of X. Create another scatter plot with each individual projected onto the subspace spanned by the first and third principal components. After plotting, play with different labeling schemes (with labels derived from the meta-data) to explain the clusters that you see. Your plot must include a legend.

(d) [5 points] In one sentence, what information does the third principal component capture?

(e) [6 points] As noted previously, the top 3 principal components of X are all unique. How would you add an additional row to the matrix to make the first principal component *not* unique? For this additional row, you don't need to be restricted to a $\{0, 1\}$ vector. More generally, assume that the first $k + 1$ principal components are all unique. How would you add an row to make the k-th principal component not unique? Again, this row doesn't need to be $\{0, 1\}$-valued.