1 K-means and K-medians

(b) All of the compressed images look at least slightly different from the original image. The differences are mainly in areas of high shading, where the compressed images lack detail. However, a quick glance at images compressed with 4 and 6 clusters is not enough to notice such shading differences. They both represent a significant amount of detail, such as the fur of the raccoon and most of the shadows on the grass.

(c) We will alternate between optimizing the cluster centers (with fixed clusters) and optimizing the clusters themselves (with fixed cluster centers). First fix K clusters, then we want to minimize

$$\mathcal{L}(c_1, \dots, c_K) = \sum_{i,j=1}^{N} \min_{k \in \{1,\dots,K\}} |x_{ij} - c_k|.$$

In order to minimize this, we will find its derivative, set it to 0, and solve for the c_k's. For any c_k, we have

$$\frac{d}{dc_k}\mathcal{L}(c_1, \dots, c_K) = \sum_{i \mid x_i \text{ in cluster k}} \text{sign}(x_{ij} - c_k) = 0.$$

If for some ij, $x_{ij} - c_k < 0$, then $\text{sign}(x_{ij} - c_k) = -1$, and the only way to get 0 in the sum over all ij is to have another point $x_{i'j'}$ satisfying $x_{i'j'} - c_k > 0$. This same reasoning holds in the reverse direction, so there must be an equal number of points to the left and right of c_k if it is to be optimal. If we choose the new c_k to be the median of the x_{ij} in cluster k, then we satisfy this condition.

Now fix $c_1, \dots, c_k$, and assign points to clusters 1 through K in order to minimize $|x_{ij} - c_k|$. This can be brute forced.

The full algorithm is summarized below.

Algorithm 1: K-medians

Assign K initial clusters randomly, and set each cluster's center c_k to be the median of all points in the cluster. **for** $t = 1$ *to* T **do**
  Assign each point x_{ij} to a cluster k by $k = \arg\min_{k \in \{1,\dots,K\}} |x_{ij} - c_k|$.
  for $k \in \{1, \dots, K\}$ **do**
    Set c_k to be the median of all points in cluster k.
  end
end

(d) The l_1 and l_2 distances are the same in 1 dimension, but the assignment of cluster centers is different in k-means vs k-medians, so the updates will end up being different. In general, the mean of a set of points is not the same as the median of the set, even if the points are only one dimensional.

Practically speaking, this difference is most pronounced when a set of points has outliers. Even if the original randomly assigned clusters are identical, updates in k-means will be more strongly affected by points near the edges of the dataset, while k-medians will care less about those points and more about the points near the center of each cluster.

(e) The result from k-medians is not exactly the same as the result from k-means (with $k = 2$). Not only is the final result different, but k-medians took 9 iterations to converge while k-means took only 3. However, both final products are strikingly similar.

Somewhat unsurprisingly (as k-medians is more resistant to outliers), the two grayscale values chosen by k-means are farther apart than the two values chosen by k-medians. This results in a less stark color contrast in the image created by k-medians, so the k-medians image seems a bit closer in terms of color to the original image.

2 Ridge Regression

(a) Since each $\varepsilon_i \sim \mathcal{N}(0,1)$, the labels y_i are generated by

$$Y \sim \mathcal{N}(X\beta + \beta_0, I),$$

where $Y(y_1, \dots, y_n)^T$, X is a matrix where every row is a data point $\mathbf{x}_i$, and $\beta = (\beta_1, \dots, \beta_p)^T$. Additionally, we overload the notation β_0 to mean a vector (with all entries the same) instead of a real number (in order to fit it more naturally into a vector-based formulation). The likelihood for the data based on this model is then

$$p(Y \mid \beta, X) = \frac{1}{(2\pi)^{p/2}} \exp\left(-\frac{1}{2}\|Y - X\beta - \beta_0\|_2^2\right).$$

But our generative process is $y_i = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j + \varepsilon_i$, so if we let $\varepsilon \doteq (\varepsilon_1, \dots, \varepsilon_n)^T$, then the likelihood of the data is

$$p(Y \mid \beta, X) = \frac{1}{(2\pi)^{p/2}} \exp\left(-\frac{1}{2}\|\varepsilon\|_2^2\right).$$

(b) The posterior for β is

$$\begin{aligned} p(\beta \mid Y, X) &= \frac{1}{Z} p(Y \mid \beta, X) p(\beta) \\ &= \frac{1}{Z} \frac{1}{(2\pi)^{p/2}} \exp\left(-\frac{1}{2}\|\varepsilon\|_2^2\right) \frac{\lambda^{p/2}}{(2\pi)^{p/2}} \exp\left(-\frac{\lambda}{2}\|\beta\|_2^2\right) \\ &= \frac{1}{Z} \frac{\lambda^{p/2}}{(2\pi)^p} \exp\left(-\frac{1}{2}\|\varepsilon\|_2^2 - \frac{\lambda}{2}\|\beta\|_2^2\right), \end{aligned}$$

where Z is a normalization constant

$$\int_\beta p(Y \mid \beta, X) p(\beta)\, d\beta.$$

We could also write this as

$$p(\beta \mid Y, X) = \frac{1}{Z} \frac{\lambda^{p/2}}{(2\pi)^p} \exp\left(-\frac{1}{2}\left(\|Y - X\beta - \beta_0\|_2^2 + \lambda\|\beta\|_2^2\right)\right).$$

We can put this in a more interpretable form, though. We can rewrite the exponentiated term to show that the posterior of β is in fact another Gaussian. We assume for the moment that our intercept term is 0 (if not, we can simply shift the data so that the intercept term becomes 0). Then ridge regression has the closed form solution

$$\beta^* = (X^T X + \lambda I) X^T Y.$$

Then we have

$$\begin{aligned} \|Y - X\beta\|^2 + \lambda\|\beta\|^2 &= (Y - X\beta)^T (Y - X\beta) + \lambda\beta^T\beta \\ &= (Y^T - \beta^T X^T)(Y - X\beta) + \lambda\beta^T\beta \\ &= Y^T Y - Y^T X\beta - \beta^T X^T Y + \beta^T X^T X\beta + \lambda\beta^T\beta \\ &= Y^T Y - \beta^{*T}(X^T X + \lambda I)\beta - \beta^T (X^T X + \lambda I)\beta^* + \beta^T (X^T X + \lambda I)\beta \\ &= (\beta^T - \beta^{*T})(X^T X + \lambda I)(\beta - \beta^*) + [\text{stuff}], \end{aligned}$$

where [stuff] contains terms that do not depend on β. Thus we have

$$\begin{aligned} p(\beta \mid Y, X) &= \frac{1}{Z} \frac{\lambda^{p/2}}{(2\pi)^p} \exp\left(-\frac{1}{2}(\beta - \beta^*)^T (X^T X + \lambda I)(\beta - \beta^*)\right) \exp([\text{stuff}]) \\ &\propto \mathcal{N}(\beta - \beta^*, (X^T X + \lambda I)^{-1}). \end{aligned}$$

From this we can see that as we have higher variance (i.e. lower λ), the variance of the prior distribution grows.

(c) Consider a dataset with two-dimensional points that either habe label 0 or label 1. Suppose that those points corresponding to label 0 are near $(0, 0)$ and that those corresponding to label 1 are near $(1/1000, 1000)$. Since our model has the form $\beta_1 x_{i_1} + \beta_2 x_{i_2} + \beta_0$, we expect β_1 to be large and β_2 to be small.

Since we are sampling β_1 and β_2 from the same Gaussian distribution, and since their values are probably going to be far apart, this means the distribution we're sampling from has large variance. In terms of the notation from the problem, our β distribution has low λ.

This means that for any given β, its posterior (which, since $p = 2$ in this case, includes a $\lambda^{p/2} = \lambda$ term), will be low. This means our model will never be very sure of the labels it generates. So our model could find the optimal β, but we would have no idea if we were only looking at the posterior of β since it would always appear bad.

Another interpretation is through the lens of regularization. The negative log likelihood of posterior would have two β-dependent terms. The first (the "loss" term) would not be affected by λ, but as $\lambda \to 0$, the second term approaches ∞. This term represents regularization, so we can think of having unnormalized data masking the true performance of our model. Since there is so much regularization error, it is difficult to tell if our model actually optimizes its objective well.

If we were to normalize the data first, β_1 and β_2 will be much closer together, meaning λ is higher, so we can achieve larger posteriors for reasonable β. That way, we can more accurately determine the performance of our model by looking at the posterior.

3 Neural Networks

We start by computing $\langle \phi(x_1), \phi(x_2) \rangle$ when each x_j and s_j is fixed. This comes out to

$$\begin{aligned}
\langle \phi(x_1), \phi(x_2) \rangle &= \text{tr}(\phi(x_1)^T \phi(x_2)) \\
&= \text{tr}\left(\begin{bmatrix} | & & | & & | \\ \cdots & s_j x_1 \mathbb{I}_{[w_1^T x_1 \geq 0]} & \cdots \\ | & & | & & | \end{bmatrix} \begin{bmatrix} — & \vdots & — \\ — & s_j x_2^T \mathbb{I}_{[w_j^T x_2 \geq 0]} & — \\ — & \vdots & — \end{bmatrix} \right) \\
&= \frac{1}{m} \left(s_1^2 \mathbb{I}_{[w_1^T x_1 \geq 0]} \mathbb{I}_{[w_1^T x_2 \geq 0]} + \cdots + s_1^2 \mathbb{I}_{[w_m^T x_1 \geq 0]} \mathbb{I}_{[w_m^T x_2 \geq 0]} \right) x_1^T x_2.
\end{aligned}$$

Then by linearity of expectation, the expected value of this quantity over all w_j and s_j is

$$\mathbb{E}\left[\text{tr}(\phi(x_1)^T \phi(x_2)) \right] = \frac{1}{m} \left(\mathbb{E}[s_1^2] \mathbb{E}[\mathbb{I}_{[w_1^T x_1 \geq 0]}] \mathbb{E}[\mathbb{I}_{[w_1^T x_2 \geq 0]}] + \cdots + \mathbb{E}[s_m^2] \mathbb{E}[\mathbb{I}_{[w_m^T x_1 \geq 0]}] \mathbb{E}[\mathbb{I}_{[w_m^T x_2 \geq 0]}] \right) x_1^T x_2.$$

Since each s_j is identically distributed, $\mathbb{E}[s_j^2] = \mathbb{E}[s_1^2]$ for all j. Additionally, the expectation of an indicator function is just the probability of the indicator function being 1, so we can rewrite this as

$$= \frac{1}{m} \mathbb{E}[s_1^2] \left(\mathbb{P}(w_1^T x_1 \geq 0) \mathbb{P}(w_1^T x_2 \geq 0) + \cdots + \mathbb{P}(w_m^T x_1 \geq 0) \mathbb{P}(w_m^T x_2 \geq 0) \right) x_1^T x_2.$$

Now we are given

$$\mathbb{P}_w \left(\arccos \frac{w^T x_1}{\|w\|_2 \|x_1\|_2} \leq \frac{\pi}{2}, \arccos \frac{w^T x_2}{\|w\|_2 \|x_2\|_2} \leq \frac{\pi}{2} \right) = \frac{\pi - \arccos \frac{x_1^T x_2}{\|x_1\|_2 \|x_2\|_2}}{2\pi}.$$

Denote this probability by $\rho_w(x_1, x_2)$, then we can rewrite our inner product expectation in terms of $\rho_{w_1}, \ldots, \rho_{w_m}$. Since each w_j is independent,

$$\mathbb{P}(w_j^T x_1 \geq 0) \mathbb{P}(w_j^T x_2 \geq 0) = \mathbb{P}(w_j^T x_1 \geq 0, w_j^T x_2 \geq 0)$$

for all j. Since $w_j^T x \geq 0$ only when the angle between w and x is less than $\pi/2$, we can rewrite our inner product expectation as

$$\begin{aligned}
\mathbb{E}\left[\text{tr}(\phi(x_1)^T \phi(x_2)) \right] &= \frac{1}{m} \mathbb{E}[s_1^2] \left(\rho_{w_1}(x_1, x_2) + \cdots + \rho_{w_m}(x_1, x_2) \right) x_1^T x_2 \\
&= \frac{1}{m} \mathbb{E}[s_1^2] m \frac{\pi - \arccos \frac{x_1^T x_2}{\|x_1\|_2 \|x_2\|_2}}{2\pi} x_1^T x_2 \\
&= \mathbb{E}[s_1^2] \frac{\pi - \arccos \frac{x_1^T x_2}{\|x_1\|_2 \|x_2\|_2}}{2\pi} x_1^T x_2.
\end{aligned}$$

Finally, since s_1 is sampled from Unif$\{-1, 1\}$,

$$\mathbb{E}[s_1^2] = 1^2 \mathbb{P}(s_1 = 1) + (-1)^2 \mathbb{P}(s_1 = -1) = 1.$$

This means we can simplify our inner product expectation to just

$$\mathbb{E}\left[\text{tr}(\phi(x_1)^T \phi(x_2)) \right] = x_1^T x_2 \frac{\pi - \arccos \frac{x_1^T x_2}{\|x_1\|_2 \|x_2\|_2}}{2\pi},$$

as desired.

4 Expectation Maximization

(a) Let $\ell(\lambda)$ denote the log likelihood of the data given λ, then since each data point is sampled i.i.d. from the Poisson distribution, we have

$$\begin{aligned}
\ell(\lambda) &= \log \mathbb{P}(X_1, \ldots, X_n = x_1, \ldots, x_n \mid \lambda) \\
&= \log \prod_{i=1}^{100} \mathbb{P}(X_i = x_i \mid \lambda) \\
&= \sum_{i=1}^{100} \log \mathbb{P}(X_i = x_i \mid \lambda) \\
&= \sum_{i=1}^{100} (-\lambda + x_i \log(\lambda) - \log(x_i!)) \\
&= -100\lambda + \left(\log \lambda \sum_{i=1}^{100} x_i \right) - \left(\sum_{i=1}^{100} \log(x_i!) \right).
\end{aligned}$$

Setting the derivative to zero yields

$$\begin{aligned}
\frac{d}{d\lambda}\ell = -100 + \frac{1}{\lambda} \sum_{i=1}^{100} x_i = 0 \\
\lambda = \frac{1}{100} \sum_{i=1}^{100} x_i.
\end{aligned}$$

thus $\hat{\lambda}$ is the mean of the sampled x_i's.

(b) The likelihood of observing outcome x_i given that person i is of type $Z_i = k$ is

$$\mathbb{P}\left(X_i = x_i \mid Z_i = k, \lambda\right) = (2-k)\mathbb{I}_{[x_i=0]} + (k-1)\frac{e^{-\lambda}\lambda^{x_i}}{x_i!}.$$

Alternatively, we can simply state

$$\mathbb{P}\left(X_i = x_i \mid Z_i = 1, \lambda\right) = \mathbb{I}_{[x_i=0]}$$

and

$$\mathbb{P}\left(X_i = x_i \mid Z_i = 2, \lambda\right) = \frac{e^{-\lambda}\lambda^{x_i}}{x_i!}.$$

(c) The likelihood of observing outcome x_i is then

$$\begin{aligned}
\mathbb{P}(X_i = x_i \mid \lambda) &= \sum_k \mathbb{P}(X_i = x_i \mid Z_i = k, \lambda)\mathbb{P}(Z_i = k \mid \lambda) \\
&= \mathbb{P}(X_i = x_i \mid Z_i = 1, \lambda)w + \mathbb{P}(X_i = x_i \mid Z_i = 2, \lambda)(1-w) \\
&= \mathbb{I}_{[x_i=0]}w + \frac{e^{-\lambda}\lambda^{x_i}}{x_i!}(1-w).
\end{aligned}$$

To make notation simpler later on, we now define

$$\begin{aligned}
\rho_{i,1} &\doteq \mathbb{P}(X_i = x_i \mid Z_i = 1, \lambda)\mathbb{P}(Z_i = 1 \mid \lambda) \\
&= \mathbb{I}_{[x_i=0]}w
\end{aligned}$$

and

$$\begin{aligned}\rho_{i,2} &\doteq \mathbb{P}(X_i = x_i \mid Z_i = 2, \lambda)\mathbb{P}(Z_i = 2 \mid \lambda) \\ &= \frac{e^{-\lambda}\lambda^{x_i}}{x_i!}(1-w),\end{aligned}$$

so that $\mathbb{P}(X_i = x_i \mid \lambda) = \rho_{i,1} + \rho_{i,2}$.

(d) When $k = 1$, we have

$$\begin{aligned}\gamma_{i,1} &= \mathbb{P}(Z_i = k \mid X_i = x_i, \lambda) \\ &= \frac{\mathbb{P}(X_i = x_i \mid Z_i = 1, \lambda)\mathbb{P}(Z_i = 1, \lambda)}{\mathbb{P}(X_i = x_i \mid \lambda)} \\ &= \frac{\rho_{i,1}}{\rho_{i,1} + \rho_{i,2}}.\end{aligned}$$

Similarly, for $k = 2$, we have

$$\begin{aligned}\gamma_{i,2} &= \frac{\mathbb{P}(X_i = x_i \mid Z_i = 1, \lambda)\mathbb{P}(Z_i = 1, \lambda)}{\mathbb{P}(X_i = x_i \mid \lambda)} \\ &= \frac{\rho_{i,2}}{\rho_{i,1} + \rho_{i,2}}.\end{aligned}$$

(e) The derivative of A with respect to λ is

$$\frac{\partial}{\partial\lambda}A(w,\lambda) = \sum_{i=1}^{N}\sum_{k=1}^{K}\gamma_{i,k}\frac{\partial_\lambda\mathbb{P}(X_i = x_i, Z_i = k \mid \lambda)}{\mathbb{P}(X_i = x_i, Z_i = k \mid \lambda)}.$$

But noting that

$$\mathbb{P}(X_i = x_i, Z_i = k \mid \lambda) = \mathbb{P}(X_i \mid Z_i = k, \lambda)\mathbb{P}(Z_i = k \mid \lambda),$$

the derivative becomes

$$\frac{\partial}{\partial\lambda}A(w,\lambda) = \sum_{i=1}^{N}\left[\gamma_{i,1}\frac{\partial_\lambda\rho_{i,1}}{\rho_{i,1}} + \gamma_{i,2}\frac{\partial_\lambda\rho_{i,2}}{\rho_{i,2}}\right].$$

Since $\rho_{i,1}$ is independent of λ, the second fraction is always 0. Since $\partial_\lambda\rho_{i,2} = \frac{e^{-\lambda}\lambda^{x_i}}{x_i!}(1-w)\left(\frac{x_i}{\lambda} - 1\right) = \rho_{i,2}\left(\frac{x_i}{\lambda} - 1\right)$, this simplifies to

$$= \sum_{i=1}^{N}\left(\gamma_{i,2}\frac{x_i}{\lambda} - 1\right).$$

Setting this to 0 and re-arranging gives

$$\boxed{\lambda = \frac{\sum_{i=1}^{N}\gamma_{i,2}x_i}{\sum_{i=1}^{N}\gamma_{i,2}}.}$$

Comparison: This is a generalization of the case from part **(a)**. When there was no latent variable, $\gamma_{i,k}$ would always be 1, in which case we recover

$$\lambda = \frac{\sum_{i=1}^{N}x_i}{N},$$

the mean of the sampled x_i (this is what we derived in part **(a)**. The new form of λ that we just derived in the case where there *is* a latent variable is just a weighted mean of the x_i's, which depends on the mixture weight w.

(f) Similarly, the derivative of A with respect to w can be written in terms of the derivatives of $\rho_{i,1}$ and $\rho_{i,2}$, so we compute them to be

$$\begin{aligned}\partial_w \rho_{i,1} &= \mathbb{I}_{[x_i=0]} \\ \partial_w \rho_{i,2} &= -\frac{e^{-\lambda}\lambda^{x_i}}{x_i!}.\end{aligned}$$

Then we have

$$\begin{aligned}\frac{\partial}{\partial w} A(w,\lambda) &= \sum_{i=1}^{N}\sum_{k=1}^{K} \gamma_{i,k}\frac{\partial_w \rho_{i,k}}{\rho_{i,k}} \\ &= \sum_{i=1}^{N}\left[\gamma_{i,1}\frac{1}{w} - \gamma_{i,2}\frac{1}{1-w}\right],\end{aligned}$$

and setting this to 0 (and assuming we don't have the trivial cases $w = 0$ or $w = 1$) yields

$$\begin{aligned}\sum_{i=1}^{N}\left[\gamma_{i,1}(1-w) - \gamma_{i,2}w\right] &= 0 \\ \frac{1-w}{w} &= \frac{\sum_{i=1}^{N}\gamma_{i,2}}{\sum_{i=1}^{N}\gamma_{i,1}} \\ \frac{1}{w} - 1 &= \frac{\sum_{i=1}^{N}\gamma_{i,2}}{\sum_{i=1}^{N}\gamma_{i,1}} \\ \frac{1}{w} &= \frac{\sum_{i=1}^{N}(\gamma_{i,1}+\gamma_{i,2})}{\sum_{i=1}^{N}\gamma_{i,1}}.\end{aligned}$$

Noting that $\gamma_{i,1} + \gamma_{i,2} = \frac{\rho_{i,1}}{\rho_{i,1}+\rho_{i,2}} + \frac{\rho_{i,1}}{\rho_{i,1}+\rho_{i,2}} = 1$, this simplifies to

$$\boxed{w = \frac{\sum_{i=1}^{N}\gamma_{i,1}}{N}.}$$

Comparison: This is the same as the case of Gaussian mixture models from class. The only difference is semantic: in our case, we need only one paramter w to keep track of the probabilities of both latent classes, whereas with the Gaussian mixture models we were considering multiple mixture weights.

The formula derived in class was

$$w_{k'} = \frac{\sum_{i=1}^{N}\gamma_{i,k'}}{N},$$

which is a straightforward generaliztion of the formula we just derived here when we need more than just a single w to keep track of the weights for all our latent classes.