ISYE 6740 Homework 3

Nick DiNapoli, ndinapoli6@gatech.edu

Due: October 3, 2021

1 Conceptual questions

1.1 Based on the outline given in the lecture, show that the maximum likelihood estimate (MLE) for Gaussian mean and variance parameters are given by the two in Equation 1 respectively. Please make sure to complete details of the derivations.

$$\hat{\mu} = \frac{1}{m}\sum_{i=1}^{m} x^i, \quad \hat{\sigma}^2 = \frac{1}{m}\sum_{i=1}^{m}(x^i - \hat{\mu})^2 \tag{1}$$

The likelihood or PDF of the Gaussian distribution is given by,

$$p(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2\sigma^2}(x-\mu)^2} \tag{2}$$

and because of the monotone nature of the log function, the objective function can be taken as the log-likelihood because values that maximize the log-likelihood also maximize the likelihood. The objective function is now,

$$\begin{aligned} l(\mu,\sigma;D) &= log\prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2\sigma^2}(x^i-\mu)^2} \\ &= -\frac{m}{2}log(2\pi) - \frac{m}{2}log(\sigma^2) - \sum_{i=1}^{m}\frac{(x^i-\mu)^2}{2\sigma^2} \end{aligned} \tag{3}$$

taking note of the use of the product because points are i.i.d. and the product and sum can be interchanged because of the fact that, log(ab) = log(a) + log(b).

Now I aim to maximize l with respect to μ and σ by taking partial derivatives and setting them equal to zero as in Equation 4.

$$\frac{\partial l}{\partial \mu} = 0 \quad \frac{\partial l}{\partial \sigma^2} = 0 \tag{4}$$

First, I begin with μ and consider only terms in Equation 3 that depend on μ,

$$\begin{aligned} \frac{\partial l}{\partial \mu} &= -\frac{1}{2\sigma^2}\sum_{i=1}^{m}\frac{\partial}{\partial\mu}(x^i-\mu)^2 \\ 0 &= -\frac{1}{2\sigma^2}\sum_{i=1}^{m} -2(x^i-\mu) \end{aligned} \tag{5}$$

the -2 cancel and then multiply both sides by σ^2 and expand sum. Now I have,

$$0 = \sum_{i=1}^{m} x^i - \sum_{i=1}^{m}\mu \tag{6}$$

and notice that because μ does not depend on i, the second term becomes $m\mu$. Now rearrange terms,

$$\begin{aligned} m\mu &= \sum_{i=1}^{m} x^i \\ \mu &= \frac{1}{m}\sum_{i=1}^{m} x^i \quad \checkmark \end{aligned} \tag{7}$$

Next, I complete the same for σ^2,

$$\begin{aligned} \frac{\partial l}{\partial \sigma^2} &= -\frac{m}{2}\frac{\partial}{\partial \sigma^2} log(\sigma^2) - \sum_{i=1}^{m} \frac{\partial}{\partial \sigma^2}\frac{(x^i-\mu)^2}{2\sigma^2} \\ 0 &= -\frac{m}{2\sigma^2} - \sum_{i=1}^{m} \frac{-(x^i-\mu)^2}{2(\sigma^2)^2} \end{aligned} \tag{8}$$

and move the fist term to the left side of the equation and multiply both sides by $2\sigma^2$. Now I have,

$$\begin{aligned} m &= \frac{1}{\sigma^2}\sum_{i=1}^{m}(x^i-\mu)^2 \\ \sigma^2 &= \frac{1}{m}\sum_{i=1}^{m}(x^i-\mu)^2 \quad \checkmark \end{aligned} \tag{9}$$

1.2 Please give one advantage of KDE over histogram, and give one advantage of histogram over KDE.

KDE has an advantage over histogram because for histograms, the output depends on where one puts the bins and therefore the estimate can be quite noisy. In KDE, the kernel functions are placed at each data point and the density estimate is the superpositions of such kernels making for a much smoother estimate. However, histograms do have an advantage over KDE when it comes the the number parameters and memory. For histograms, these increase with $\frac{1}{\Delta}$ and $(\frac{1}{\Delta})^n$ respectively, whereas in KDE, these increase with m and mn. So if the number of data points is very large but n is modest, then KDE is not a good choice.

1.3 For EM algorithm for GMM, please show how to use Bayes rule to drive τ_k^i in closed-form expression.

Bayes' rule can be summarized as,

$$\begin{aligned} P(z|x) &= \frac{P(x|z)P(z)}{P(x)} \\ Posterior &= \frac{(likelihood) * (prior)}{(marginal\ distribution)} \end{aligned} \tag{10}$$

with English terms as the second equation in Equation 10. In the EM algorithm, τ_k^i, is the posterior distribution of a data point i belonging to the k component given an observation (proportion of each Gaussian). The easiest component to pick out in the GM algorithm is the prior probability, the marginal distribution of z, or $p(z) = \pi_z$. Next, I can take the likelihood of seeing point x^i given the k^{th} Gaussian distribution and the respective parameters, μ_k and Σ_k. This term can be written as, $\mathcal{N}(x^i|\mu_k, \Sigma_k)$. Lastly, I need the marginal distribution of x^i which is the summation over all $k's$ of the product of the prior and likelihood. Finally, putting this all together and following Equation 10, I get,

$$\tau_k^i = \frac{\pi_k \mathcal{N}(x^i|\mu_k, \Sigma_k)}{\sum_{k'=1}^{K} \pi_{k'} \mathcal{N}(x^i|\mu_{k'}, \Sigma_{k'})} \quad \checkmark \tag{11}$$

2 Density estimation: Psychological experiments

In this problem, I aim to answer the question of if there is a relationship between the size of two different brain regions in the brain, the amygdala and the anterior cingulate cortex (acc), and the political views of 90 university students. Additionally, I can try to determine if these regions are independent of one another. These brain regions are known to be involved in emotions and decision making. The political views of 90 university students are recorded on a 1-5 scale ranging from very conservative to very liberal, respectively. In this dataset, there are no 1 values however. The brain region data is taken as the residuals from predicted volumes after adjusting for height, sex, and other body-type variables.

2.1 1D Histogram and KDE

I first plot the 1D histograms and Gaussian KDE to estimate the distributions of the amygdala and acc. For this analysis, I bin the data such that the shapes of the distributions are well captured. I also use the same number of bins (10) for each distribution for comparison purposes. Figure 1 shows the histograms of the two brain regions and Figure 2 shows these histograms on the same axes over the same interval for the sake of comparison. It can be seen that both distributions are roughly 0-centered but the amygdala has a bit more variance.

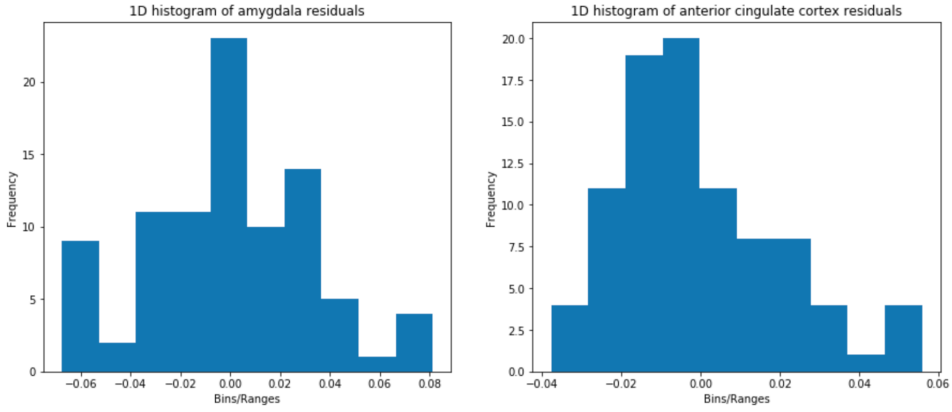


Figure 1: 1D histograms of each region using 10 bins.

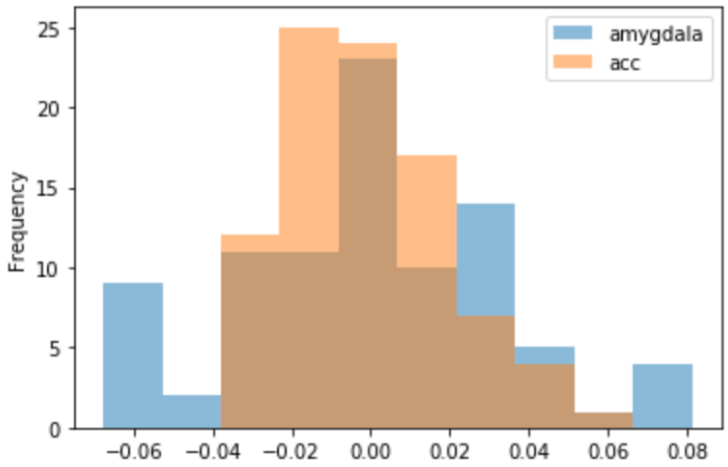


Figure 2: 1D histograms overlaid.

Next, I form the 1D Gaussian KDE of each distribution as a means to smooth the distributions and capture an even better estimate. I chose a kernel bandwidth equal to that using Silverman's rule in Equation 12. Figure 3 displays the KDE for both distributions over the same range. Now, my elementary analysis of the mean and variance of both distributions can be verified.

$$h \approx 1.06\hat{\sigma}m^{-1/5} \tag{12}$$

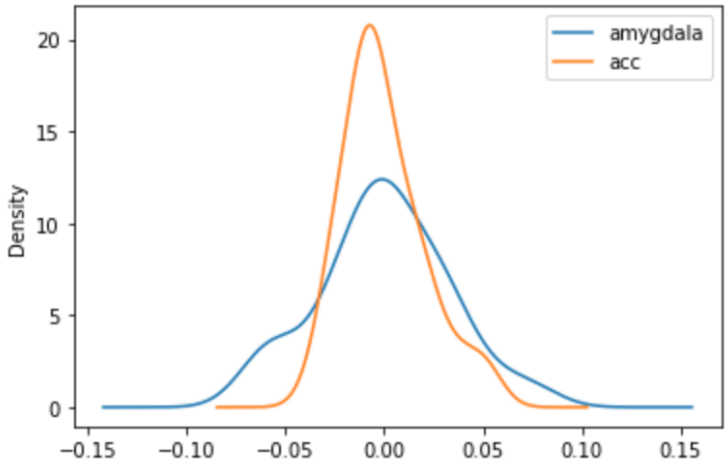


Figure 3: 1D KDE of both brain regions.

2.2 2D Histogram of brain regions

Also using a binning of 10 for comparison and data-capturing, I plot the residuals of each brain region against one another as a 2D histogram to get an even better feel for the data's shape in multiple dimensions. Figure 4 displays these results.

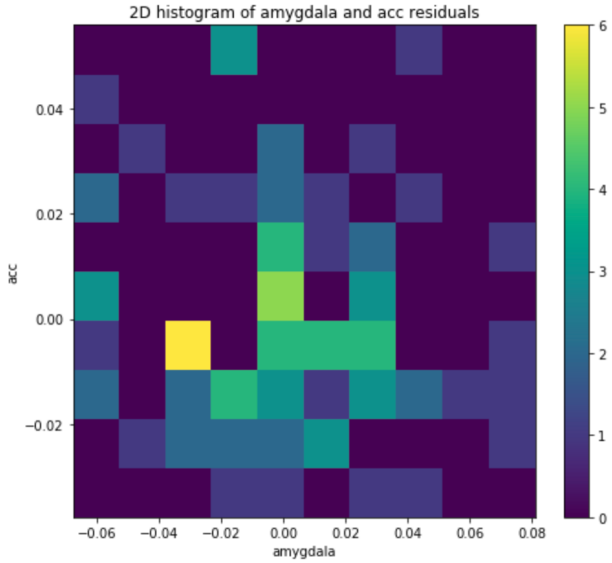


Figure 4: 2D histogram of brain regions

2.3 2D KDE of brain regions

I could also extend the kernel-density-estimation to 2 dimensions to obtain an even more enhanced view of what the brain residual data looks like in multiple dimensions. Figure 5 shows this 2D Gaussian KDE plot with contours as well as the original data points superimposed in red. This sort of visualization adds another layer to how the data is distributed. Again, as a sanity check, it can be seen that the data is more or less 0-centered and there is certainly more variance along the amygdala residual axis.

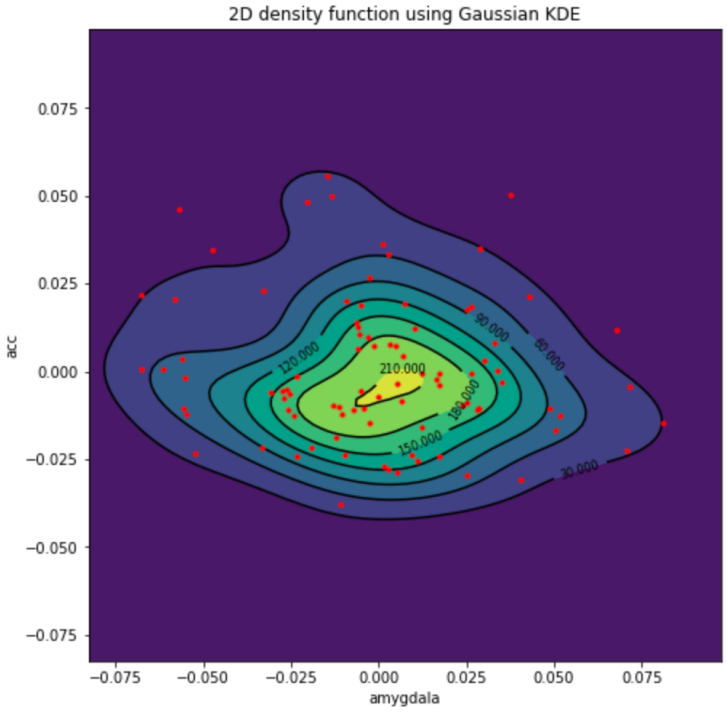


Figure 5: 2D histogram of brain regions

From this plot, there are even more takeaways, such as noticing that the distribution is uni-modal and there may be the existence of a few outliers. There exists about four points that lie in the darkest region of the plot (furthest from high-density regions) and could be considered outliers in this distribution. I can also answer the posed question of if these two variables are independent. Because the distribution is roughly symmetric and there are not drastic changes in the spacing of contour lines as well as having no clear trends along the diagonals of the 2D plot (positive/negative correlation), I determine these variables are in fact independent.

2.4 Conditional distributions

Now I wish to consider the political orientations of the students to better answer the questions previously posed. I now plot the conditional distributions i.e. p(amygdala|orientation = c) for c = 2, ...,5 of amygdala and acc using KDE with a kernel bandwidth using Equation 12. Although I could generate these plots separately, it is more informative to visualize the conditional distributions on the same scale for each c. Figures 5-8 show exactly this.

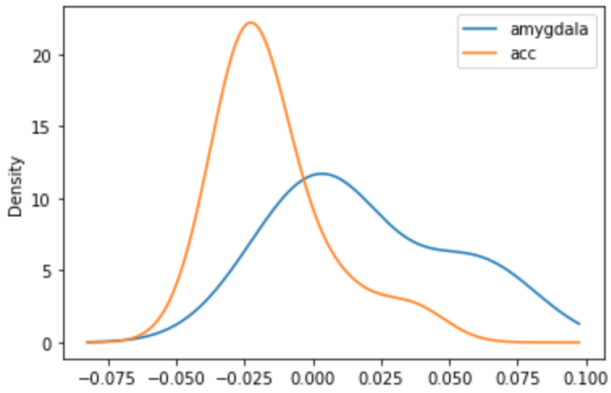


Figure 6: Conditional distributions when c = 2.

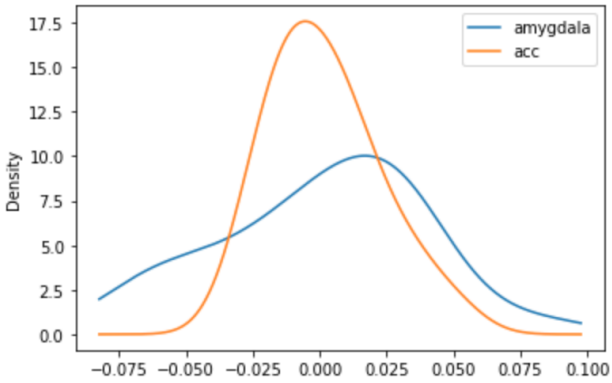


Figure 7: Conditional distributions when c = 3.

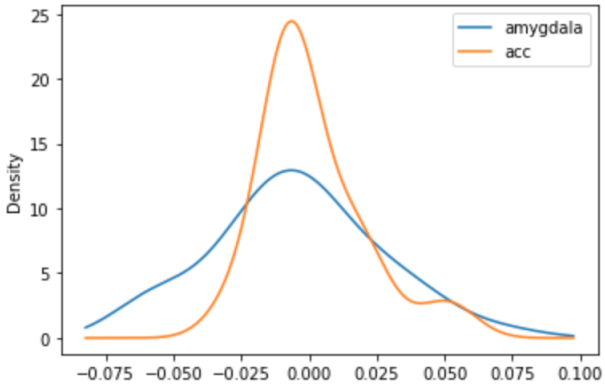


Figure 8: Conditional distributions when c = 4.

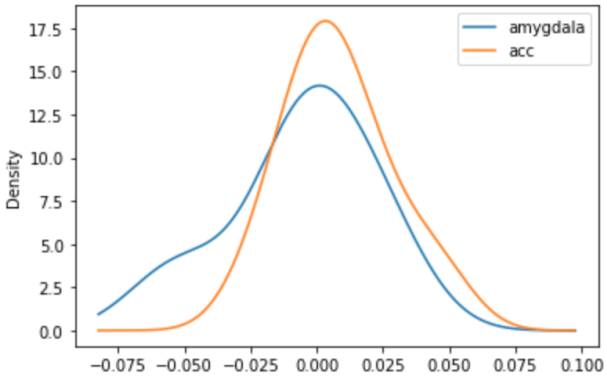


Figure 9: Conditional distributions when c = 5.

The conditional distributions of the amygdala and acc residuals certainly vary across the different values for c (political orientations). There are several trends present from visualizing these distributions. First, I look intra-orientation distribution trends. The most notable conclusion I can draw is that for more conservative students, the amygdala residuals tend to be higher than those of the acc. For more liberal students, the brain region distributions seem to be relatively similar. I can draw the conclusion that if the acc is small compared to the amgydala, a student is more likely to be conservative. As for comparing the same brain region across different orientations, there seems to be no consistent trend for the amygdala but the acc residuals certainly appear to get larger with increasing c. Therefore as the acc volume increases, it does appear that a student is more likely to be increasingly liberal. These two results together do suggest that brain structure and political orientation are correlated in some way. A deeper analysis may need to be done however to prove a causal relationship. As a way to quantify the above analysis, I document the conditional sample means for the two variables in Table 1.

Brain region	c = 2	c = 3	c = 4	c = 5
amygdala	0.01906154	0.0005875	-0.00471951	-0.00569167
acc	-0.01476923	0.00167083	0.00130976	0.00814167

Table 1: Conditional sample means for amygdala and acc.

It is very informative to quantify the above results because there is a clear trend present that one could have missed. Not only does the political orientation generally increase (more liberal) from an increase in acc but it is also true that as the amygdala size increases, there is a clear decrease in c (more conservative). It is extremely interesting to see these different results.

2.5 Conditional joint distributions

As the final part of my analysis, I plot the conditional joint distributions p(amygdala,acc|orientation = c) for c = 2, ...,5 using 2D KDE and Silverman's rule once again as a selection of kernel bandwidth. I display these plots as contour plots with the data superimposed in red as seen in Figures 10-13.

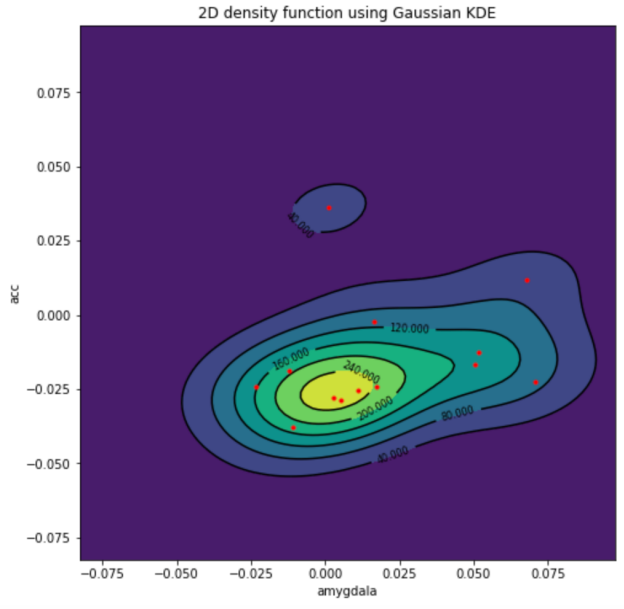


Figure 10: Conditional joint distributions when c = 2.

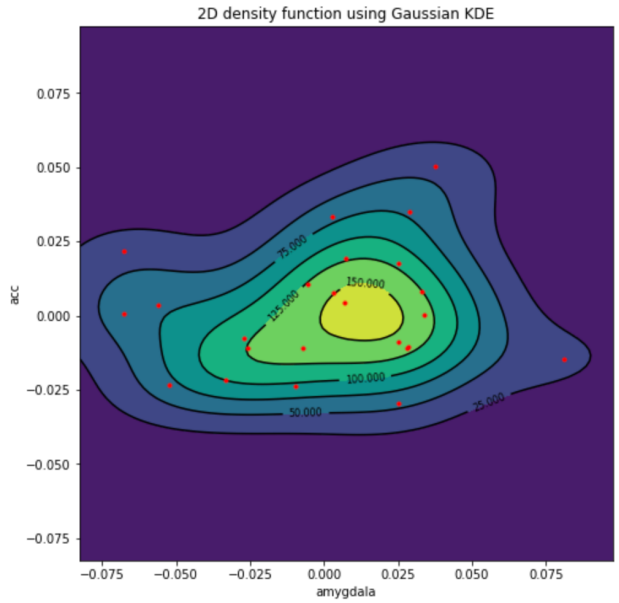


Figure 11: Conditional joint distributions when c = 3.

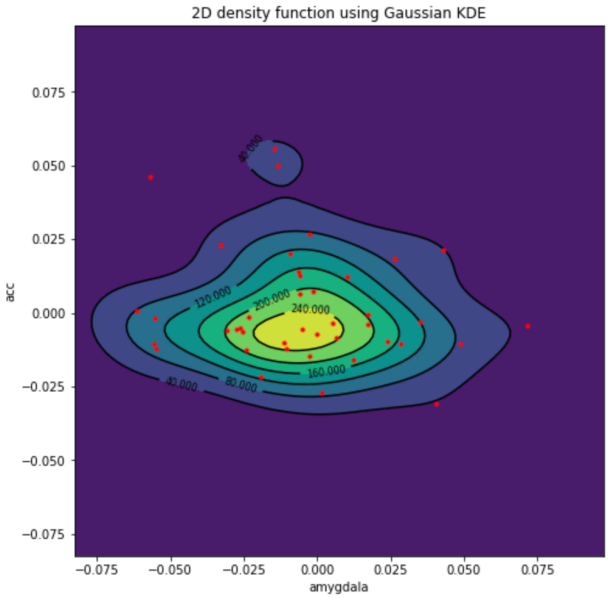


Figure 12: Conditional joint distributions when c = 4.

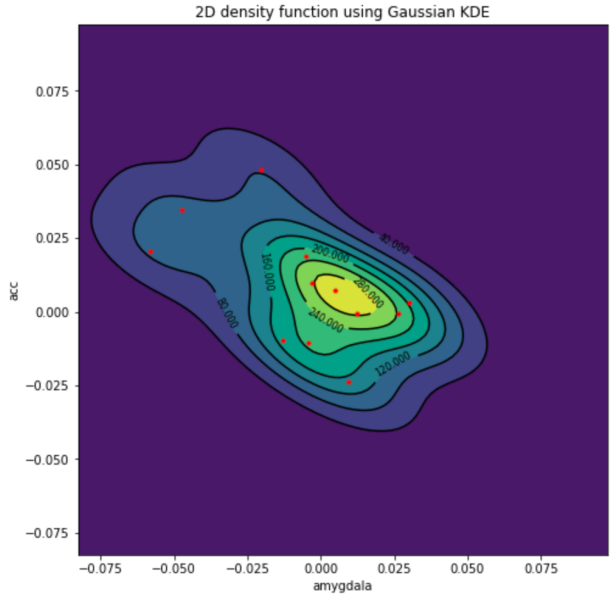


Figure 13: Conditional joint distributions when c = 5.

The conditional joint distributions could be exactly what I needed to conclude my analysis from the previous section and to more soundly make the determination that there is an apparent difference between brain structure and political view. For the less extreme political views (c = 3 and c = 4) there are no clear trends to be seen from the conditional joint distributions because the contours seem to be more or less symmetric with relatively equal spacing, similar to that of Figure 5. However, from Figures 10 and 13 (c = 2 and c = 5) the more extreme political views, it is clear that there is some correlation. I say this because there is a trend along the diagonals of the plots suggesting that it is more apparent at extreme views that there is correlation between brain region size and political view. In fact, the trends are the exact same as in Table 1, for c = 2 the high density region falls below the acc = 0.0 line and for c = 5 the high density region is above the acc = 0.0 line. Interestingly, the conditional joint distributions don' show a clear pattern

when analyzing where the high density regions of the amygdala lie. This suggests that it is imperative to complete this sort of comprehensive exploratory data analysis when extracting causal relationships. In this analysis I see slightly different results when introducing different combinations of variables and responses.

3 Implementing EM for MNIST dataset

I now implement the expectation-maximization (EM) algorithm for fitting a Gaussian mixture model (GMM) for the classical MNIST hand-written digits dataset. For this section I only consider two different digits, "2" and "6" and therefore aim to fit the model with C = 2. There are 1990, 28x28 pixel images and I first pre-process the data by using PCA to reduce the dimensionality of the data (combined) to 4. Therefore I am completing EM for the projected data.

3.1 Expressions for the E-step and M-step

First, I aim to write down the steps and expressions for the EM algorithm which is as follows:

1. Associate the i^{th} data and each component with a τ_k^i
2. Initialize (π_k, μ_k, Σ_k), k = 1.... K
3. Iterate until convergence:
   (a) E-step
      i. Update τ_k^i given current (π_k, μ_k, Σ_k)
   (b) M-step
      i. Update (π_k, μ_k, Σ_k) given τ_k^i

First, I take a detailed look at the E-step,

$$\tau_k^i = p(z_k^i = 1|D, \mu, \Sigma) = \frac{\pi_k \mathcal{N}(x^i|\mu_k, \Sigma_k)}{\sum_{k'=1}^{K} \pi_{k'} \mathcal{N}(x^i|\mu_{k'}, \Sigma_{k'})} \tag{13}$$

where,

$$\mathcal{N}(X|\mu_k, \Sigma_k) = \frac{1}{|\Sigma|^{\frac{1}{2}}(2\pi)^{\frac{n}{2}}} e^{\frac{-1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)} \tag{14}$$

and realizing that $(2\pi)^{n/2}$ cancels in the numerator and denominator leads to:

$$\tau_k^i = \frac{\frac{\pi_k}{|\Sigma_k|^{\frac{1}{2}}} e^{\frac{-1}{2}(x_i-\mu_k)^T \Sigma_k^{-1}(x^i-\mu_k)}}{\sum_{k'=1}^{K} \frac{\pi_{k'}}{|\Sigma_{k'}|^{\frac{1}{2}}} e^{\frac{-1}{2}(x_i-\mu_{k'})^T \Sigma_{k'}^{-1}(x^i-\mu_{k'})}} \tag{15}$$

Next, I take a detailed look at the closed form expressions of the M-step:

$$\begin{aligned} \pi_k &= \frac{1}{m} \sum_i \tau_k^i \\ \mu_k &= \frac{\sum_i \tau_k^i x^i}{\sum_i \tau_k^i} \\ \Sigma_k &= \frac{\sum_i \tau_k^i (x^i - \mu_k)(x^i - \mu_k)^T}{\sum_i \tau_k^i} \end{aligned} \tag{16}$$

3.2 Implementing the EM algorithm

Following the steps in the previous section, I implemented the EM algorithm from scratch. I used the following initializations: π_k are random numbers normalized to 1, μ_k is a random Gaussian vector with zero mean, and Σ_k in the following way,

$$\begin{aligned} \Sigma_1 &= S_1 S_1^T + I_n \\ \Sigma_2 &= S_2 S_2^T + I_n \end{aligned} \tag{17}$$

where S_1 and S_2 are Gaussian random matrices and I_n is the identity matrix, all of size n-by-n. To test the algorithm's convergence, I plot the log-likelihood against the number of iterations. Figure 14 shows these results.

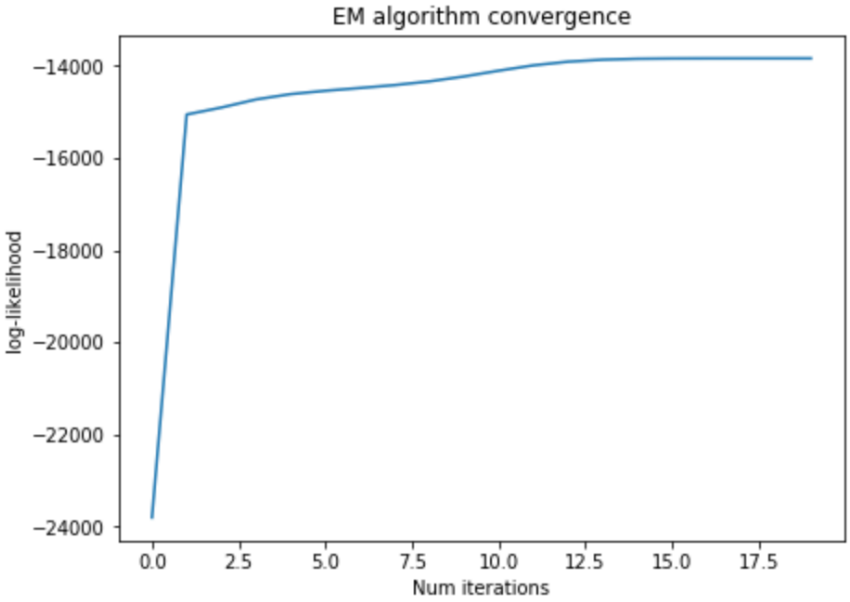


Figure 14: Testing the convergence of the EM algorithm.

The algorithm converges quite quickly, consistently in under 20 iterations after running it several times. The algorithm also improves (increase in likelihood) quite rapidly after the first iteration once the parameters are updated from the random initializations. As expected, the log-likelihood is monotonically increasing as it converges to a local maximum.

3.3 Reporting fitted GMM model

After the EM algorithm terminates, I can go one step further by analyzing the fitted parameters. After completion, the priors or π_k's become, $\pi_1 = 0.51234537$ and $\pi_2 = 0.48765463$. As expected, the weights of each component still sum to 1 after the data is seen and the GMM is fitted. I can visualize the mean of each component by using the eigendecomposition and mapping them back into the original data space. Figure 15 shows the mean of each component mapped back to the original space.

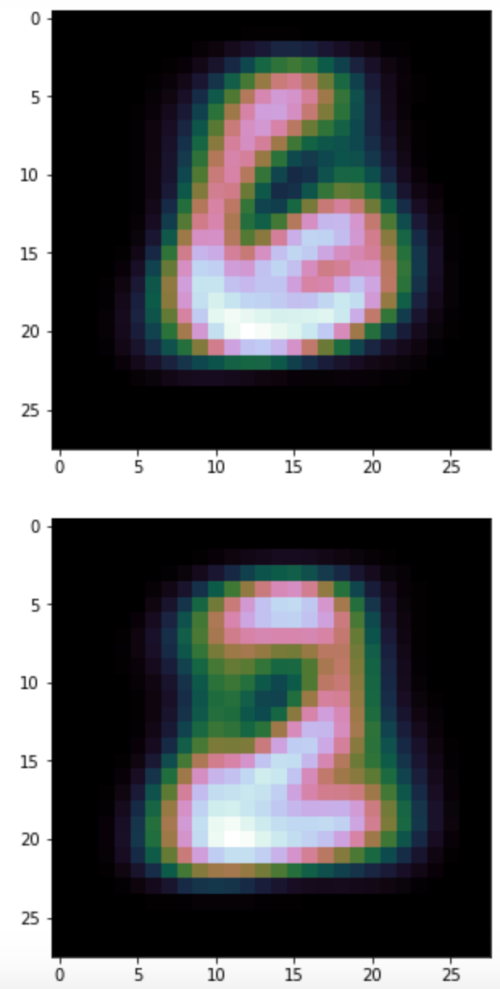
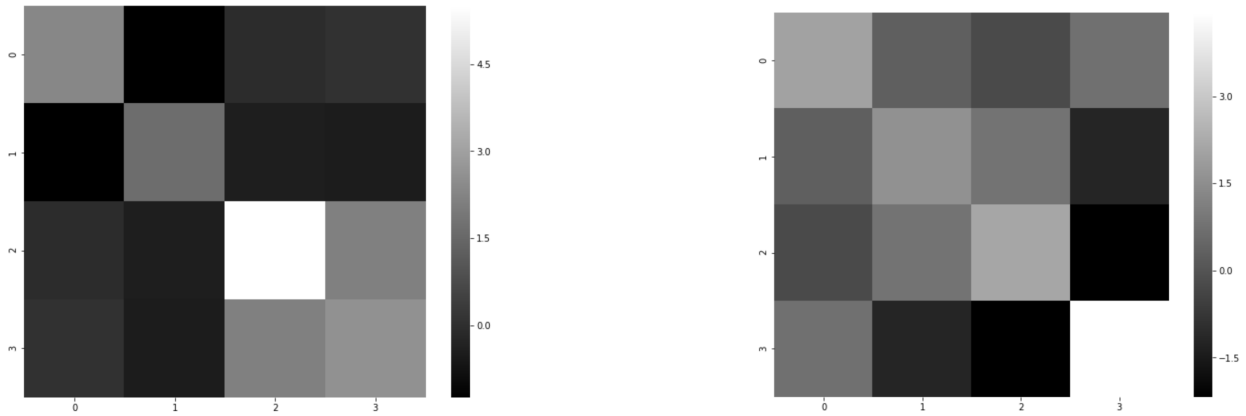


Figure 15: The top image represents the mean of the "6" component and the bottom image the "2" component.

Here I see the expected result as the mean components are mapped back into the original space. The digits appears as a bunch of handwritten digits morphed together with some added noise due to the fact that the handwritten digits are unique and that the dimensionality of the data was reduced before fitting the model. However, it is a great sanity check to see that each component quite clearly represents a different digit. Lastly, I also visualize the fitted covariance matrices by showing a heatmap of them. Each heatmap is 4 x 4 as I reduced the dimension of the data to n = 4 prior to running the algorithm. Figure 16 shows the heatmap of each component.



(a) Covariance matrix of "6" component.

(b) Covariance matrix of "2" component.

Figure 16: A visualization of the covariance matrices as a heatmap.

3.4 Mis-classification rates of GMM vs. k-means

After algorithm completion, as well looking at the parameters in the previous section, I can also use the τ_k^i to infer labels for each data point. I can then use the ground truth values provided to obtain a quantitative result for how well the GMM performed on the data. Additionally, I can compare the GMM to the k-means clustering approach to see which method might be most useful when analyzing this type of data or even each particular digit for that matter. The τ_k for each data point represents the proportion of each Gaussian (or class) it belongs to. In this way, for each point I compared the values for each component and assigned the point to the class "2" or "6" depending on which value was greater. I have summarized the mis-classification rates in Table 2.

Algorithm	digit	mis-classification rate	overall accuracy
GMM + EM	2	0.06298	0.963
GMM + EM	6	0.00835	0.963
k-means	2	0.06202	0.892
k-means	6	0.08038	0.892

Table 2: Performances comparisons between GMM and k-means.

The results above are quite interesting in the sense that overall, the GMM out performs the k-means clustering algorithm but if we take a deeper look, this mis-classification rate for the "2" digit is nearly identical. In contrast, the mis-classification rate for the digit "6" when comparing the the two algorithm, shows that the GMM out-performs kmeans by a significant margin. Because of these facts, it is clear that for this particular use-case the GMM + EM is a better choice.

References

[1] https://stackoverflow.com/questions/30145957/plotting-2d-kernel-density-estimation-with-python

[2] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.gaussian_kde.html