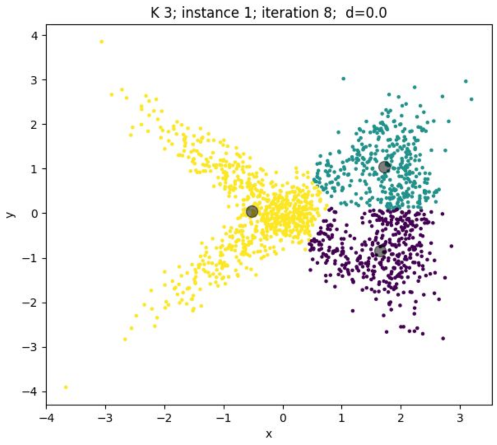
Carlos Miguel Sayao
CS446
A. Rhodes

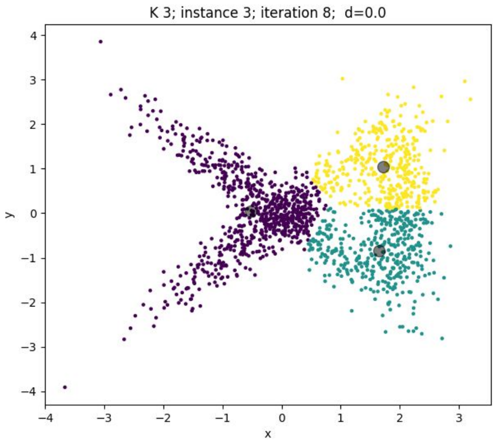
Program 1

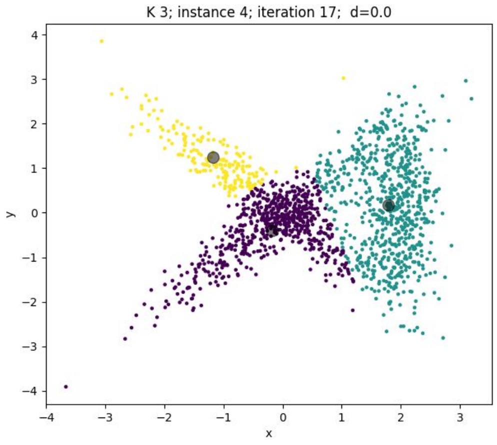
K-means: https://imgur.com/a/NxErzyE

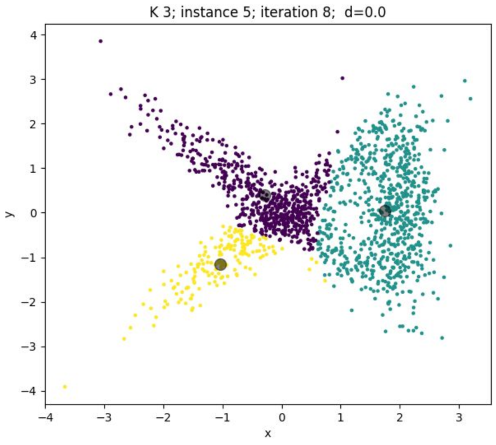
My k-means implementation was quite simple. I ran from 2 - 10 clusters, each 5 times, with stopping condition being when means don't change. When evaluating wcss for each clusters' 5 instances, I saved the means and covariances for the minimum wcss to be fed into the GMM (more on this later). K-means is the second one optimizes K-means correctly.

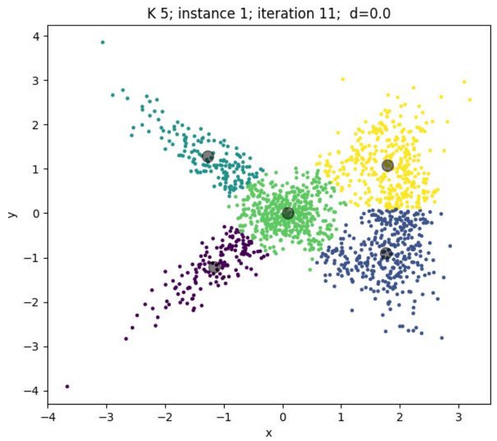
Below are 4 instances of the final centroids of K=3 and 5. The minimum wcss for 3 clusters was instance 3, for 5 clusters it was instance 2. You can see how these final centroids tended toward similar resting places.

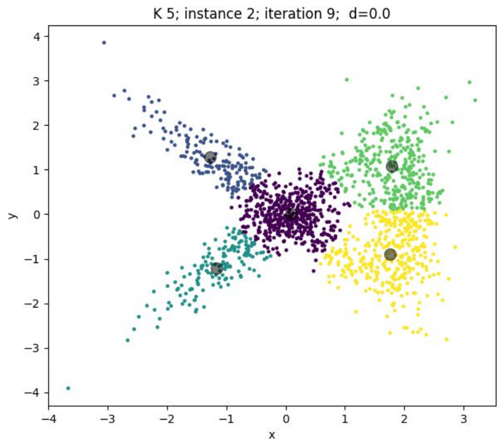

K 3; instance 1; iteration 8; d=0.0

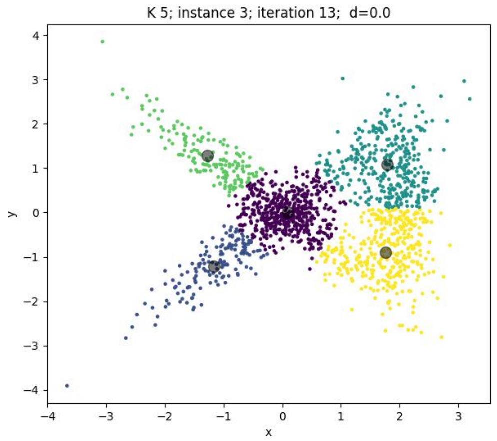

K 3; instance 3; iteration 8; d=0.0

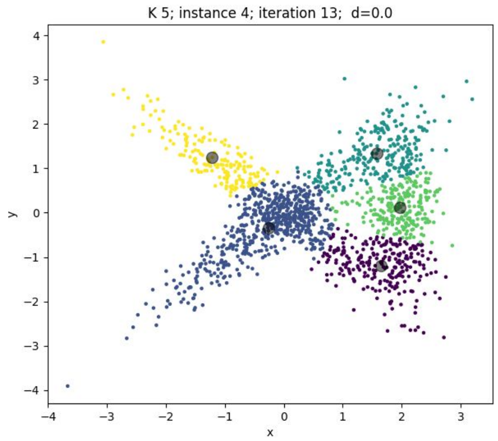

K 3; instance 4; iteration 17; d=0.0


K 3; instance 5; iteration 8; d=0.0


K 5; instance 1; iteration 11; d=0.0
y
4
3
2
1
0
−1
−2
−3
−4
x
−4
−3
−2
−1
0
1
2
3


K 5; instance 2; iteration 9; d=0.0
y
4
3
2
1
0
−1
−2
−3
−4
x
−4
−3
−2
−1
0
1
2
3


K 5; instance 3; iteration 13; d=0.0
y
4
3
2
1
0
−1
−2
−3
−4
x
−4
−3
−2
−1
0
1
2
3


K 5; instance 4; iteration 13; d=0.0
y
4
3
2
1
0
−1
−2
−3
−4
x
−4
−3
−2
−1
0
1
2
3

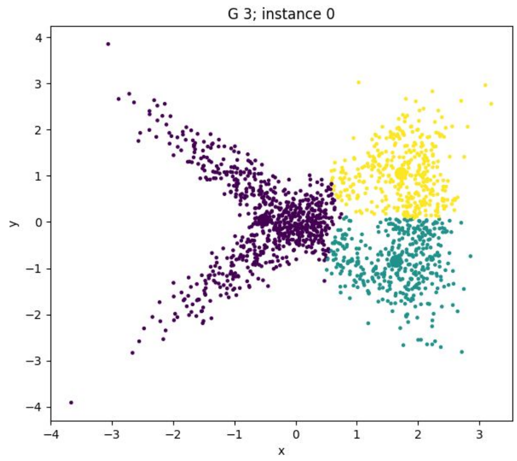
Gaussian Mixture Model: https://imgur.com/a/MD60fz6

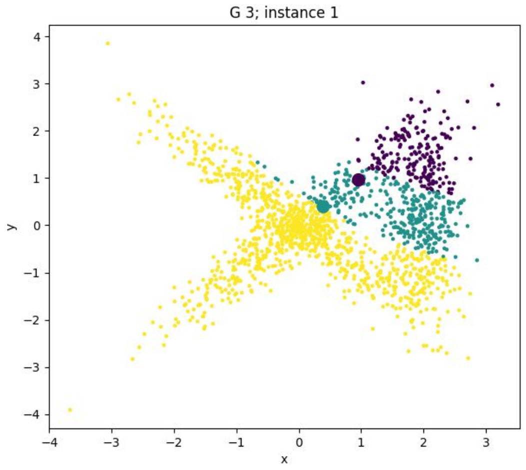
This one was much more difficult for me. I tried my best to understand the slides. Everytime I started to understand where to go next, I would confuse myself further, but overall, I think I coded most of the algorithm correctly. My confusion laid mainly in using the one-hot encoding inside gamma calculations. I wasn't sure whether or not those calculations should use the one-hot or not. I ended up not using one-hots for that.

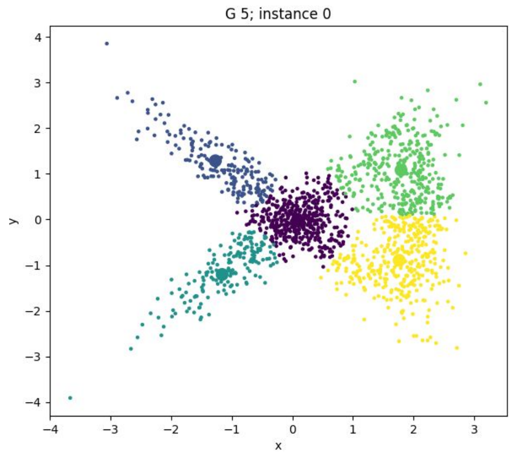
Another was using and calculating N_K and the mixing coefficients. From my understanding, the mixing coefficients were synonymous with prior distributions, essentially being how many data points belonged to each cluster. So I got the one-hots and hard clustered the most likely posterior membership to k for each n in data. N_K, in my understanding, was the sum of each posterior in k. So I coded both as such.

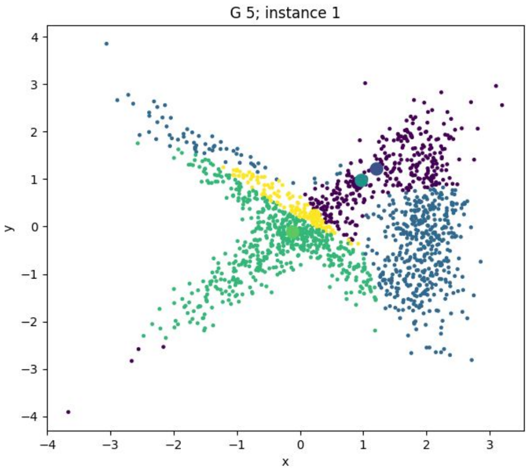
I then read in the saved means and covs from k-means, initialize means, k, covs, priors, and the data.

I calculate posteriors, then move on to the M step. This is where it goes wrong. For some reason after calculating every mean, the means end up on the same diagonal line. This can be seen below. On clusters 3 and 5 instance 0 and 1, after the first instance they end up on diagonal lines. This is consistent with all clusters I tested 2-10. I'm not sure what went wrong, but I needed to finish and turn something in.


G 3; instance 0

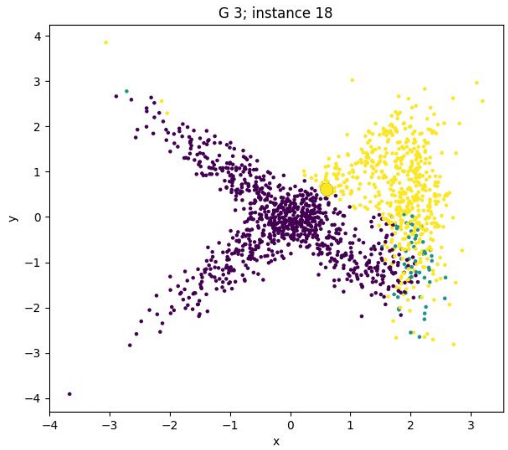

G 3; instance 1

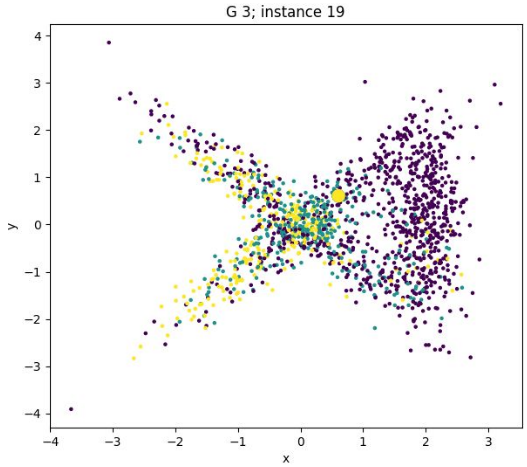

G 5; instance 0


G 5; instance 1

I coded my own Gaussian density computations correctly. I know this because replacing mine with scipy's stats multivariate function yielded the same results.

I started to think that maybe the one hot encoding was incorrect because there were never data points intermingled with other points. However, I ended up finding that as well.





You can see in the animated gif, that at times it seems that the points just switch between frames. I suspect the means may have no updated correctly?

I also didn’t end up using log likelihood because by this time I needed to wrap up. Next time, I need to collaborate with others rather than scouring the internet confusing myself further.