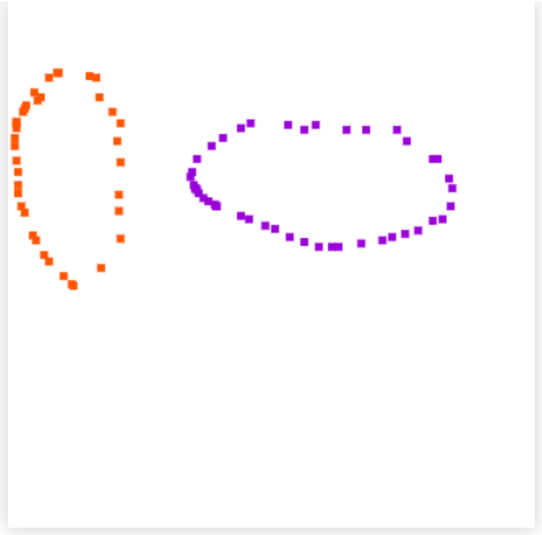
Free response questions (2 points, 1 point per question)

1. What happens as beta in soft K-Means approaches infinity? How does this relate to regular K-Means?

   As beta increases, the distribution tightens. Low distances (near 0) get really big, while high distances go to 0.

   As the stiffness β goes to ∞, the soft K-means algorithm becomes identical to the original hard K-means algorithm, except for the way in which means with no assigned points behave.

2. Draw a data distribution using ml-playground that would be better modeled using a GMM x than with soft K-Means. Explain why this is so. Include the plot you made in ML playground. (Hint: think about the covariance matrix)



In soft k-means only calculates conventional Euclidean distance.

In GMM, GMM takes variance into consideration when it calculates the measurement.

The d by d covariance matrix Σ describes the shape and orientation of an ellipse.

Therefore, in this situation GMM is better than soft k-means.

Comparing approaches without labels (1 point)

Report the performance of each of these algorithms using the Adjusted Mutual Information Score (implemented in `code/metrics.py` for you). Which algorithm performed best?

	GMM(full)	GMM(diag)	GMM(spherical)	kmeans
1	0.512485	0.3240455	0.521409856	0.493902
2	0.510578	0.3224389	0.502543411	0.513603
3	0.494216	0.3184612	0.516150044	0.494286
4	0.449505	0.3171988	0.494704804	0.503314
5	0.483748	0.3638246	0.502514924	0.512329
6	0.49028	0.3505009	0.518049635	0.507092
7	0.483553	0.307492	0.510547434	0.486224
8	0.489556	0.3473698	0.510674998	0.492881
9	0.496614	0.3864949	0.500764433	0.4897
10	0.481441	0.3426201	0.516759387	0.506864
Average	0.489198	0.3380447	0.509411893	0.50002

In average GMM with spherical is a little bit better than Kmeans.

Comparing approaches with labels (1 point)

Since we actually *do* know the labels of the handwritten digits, we can also consider the accuracy of these unsupervised approaches. For each cluster, find the most common label of all examples in that cluster. Call that the label of the cluster. Find the proportion of images whose label matches their cluster label. That's the accuracy. Report the performance of each of these algorithms using this measure. Which algorithm performed best? Is this the same one that did best with Adjusted Mutual Information Score?

	GMM(diag)	GMM(spherical)	kmeans
1	0.560874	0.604148	0.598655
2	0.553587	0.557623	0.570179
3	0.502018	0.582623	0.600785
4	0.49204	0.59417	0.577691
5	0.518049	0.584193	0.605157
6	0.48565	0.586771	0.604036
7	0.495628	0.61435	0.602691

8	0.503251	0.584529	0.571525
9	0.490135	0.610426	0.595628
10	0.494283	0.597534	0.598879
average	0.509552	0.591637	0.592522

In average, kmeans is better than GMM.

Although in the result of AMIS, GMM is a little bit better than kmeans, in the result of accuracy kmeans has a better result than GMM.

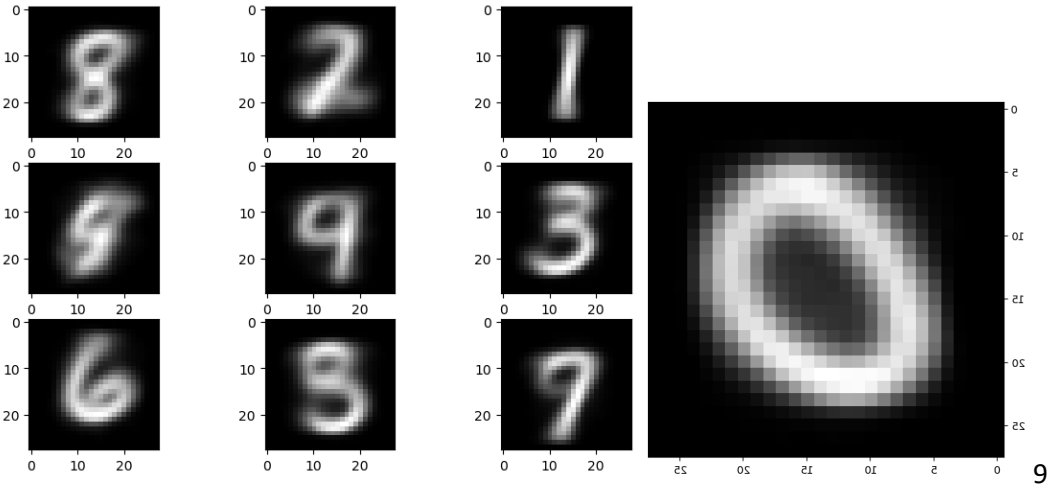
Visualization (1 point)

Visualizing examples within each cluster can help us understand what our clustering algorithms have learned. Here are two ways that a cluster can be visualized:

1. Find the mean of all examples belonging to a cluster.
2. Find the mean of all examples belonging to a cluster, then find the nearest example to the mean (i.e., the nearest neighbor).

For the best performing algorithm according to Adjusted Mutual Information Score, perform both of these visualization techniques on all 10 clusters. Show us the results of the visualization. Note that you will have to reshape your features to 28x28 images to see the results. Use euclidean distance to determine the nearest neighbor.

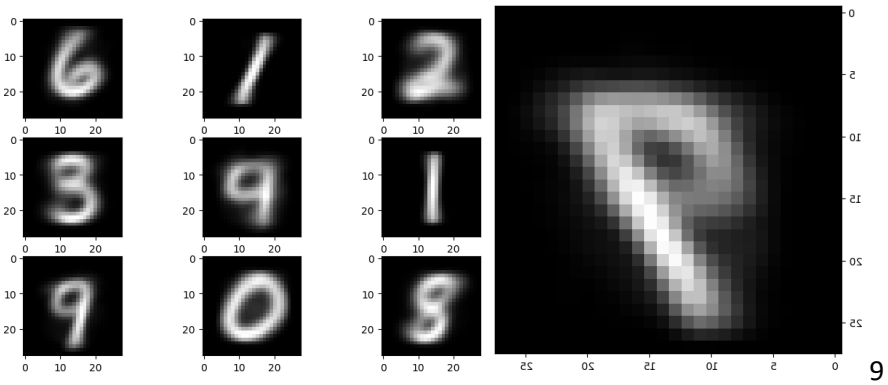
What visual differences do you notice between the images generated from each of the visualization techniques? What artifacts do you notice in first technique? What do you think is the causing these artifacts?

1. 

0	1	2

3	4	5
6	7	8

With accuracy 0.18475336322869956

2. 

0	1	2
3	4	5
6	7	8

With accuracy 0.32466367713004485

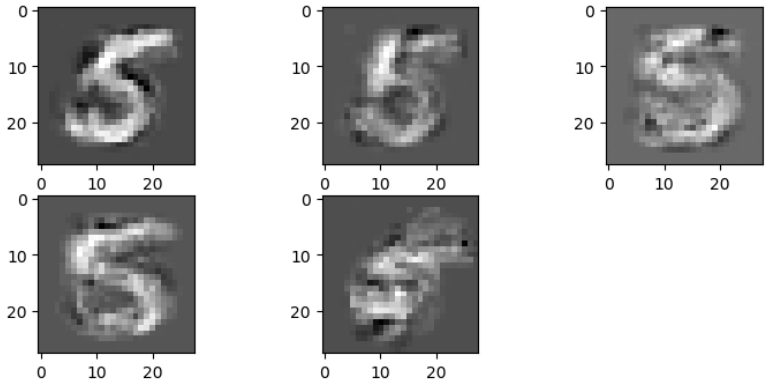
The difference between method 1 and 2 is that even the accuracy is low, second method still has chance to produce the correct answer. However, the first method hardly can produce the right one except that cluster has a lot of correct samples.

First method, each digit is not clear. Sometimes, we even can't tell which digit it is. Sometimes, we can say it belongs to either this digit or the other digit(ex. 1 or 2). The reason is that in first method we average all the samples in the cluster. With too many wrong predictions, the mean value of all samples in the cluster produces an unclear answer.

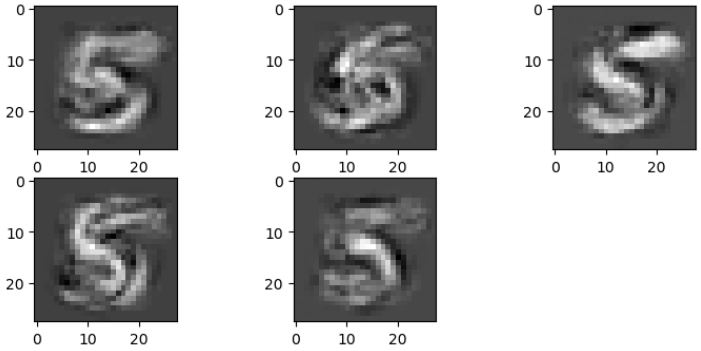
Although both methods' accuracy is low, the second way has higher chance to recognize the digit correctly. The reason is that it uses the nearest neighbor of mean, and then it has higher chance can ignore the noise if noise is not too large.

Generating handwritten digits (1 point - bonus)

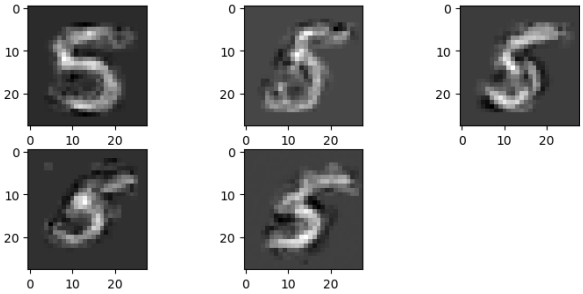
1



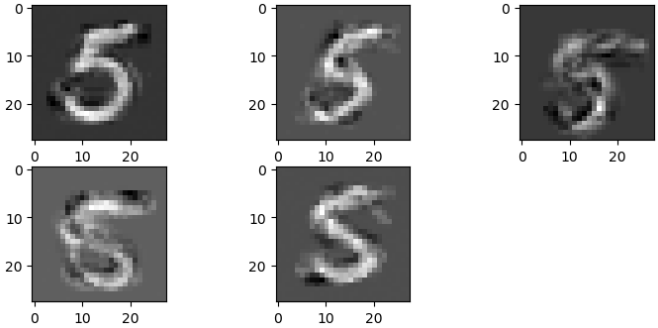
4



10



20



With larger of the number of components, the sample will become clearer. I choose digit 5 as my class. Although when number of components = 4 is worse than number of components = 1, number of components = 10 and number of components = 20 are much better than number of components = 1 and number of components = 4.