ECE 174 Mini Project #1 Report

Benjamin Crawford
A15048805

Problem 1:

a.) Code is complete and attached.

b.) The training error of the One vs All Classifier was: 2.365%
The training error of the One vs One Classifier was: 1.077%
The training confusion matrix for the One vs All Classifier is shown below:

	0	1	2	3	4	5	6	7	8	9	Total
0	964	0	17	6	0	18	14	7	17	10	1053
1	1	1095	45	30	17	10	8	24	83	11	1324
2	1	7	794	25	14	7	14	5	10	8	885
3	5	4	31	880	2	88	0	5	33	17	1065
4	3	2	22	9	861	15	6	41	18	39	1016
5	8	3	1	12	9	613	17	0	33	1	697
6	10	0	40	10	6	24	947	1	19	1	1058
7	0	1	10	20	5	4	0	909	2	72	1023
8	8	13	28	13	7	65	8	3	708	9	862
9	1	2	3	27	59	19	0	75	21	810	1017
Total	1001	1127	991	1032	980	863	1014	1070	944	978	10000

The training confusion matrix for the One vs One Classifier is shown below:

	0	1	2	3	4	5	6	7	8	9	Total
0	984	0	8	2	2	6	2	2	7	6	1019
1	0	1110	18	13	3	9	2	12	34	1	1202
2	0	4	908	15	2	5	4	11	13	4	966
3	2	3	6	946	0	29	0	0	21	14	1021
4	2	1	16	1	934	7	3	20	8	19	1011
5	3	3	2	22	3	776	13	1	19	1	843
6	4	0	10	3	6	15	981	0	14	0	1033
7	1	0	6	13	2	0	0	985	3	18	1028
8	5	3	17	10	0	10	8	1	819	4	877
9	0	3	0	7	28	6	1	38	6	911	1000
Total	1001	1127	991	1032	980	863	1014	1070	944	978	10000

c.) The testing error of the One vs All Classifier was: 14.67%
The testing error of the One vs One Classifier was: 7.11%
The testing confusion matrix for the One vs All Classifier is shown below:

	0	1	2	3	4	5	6	7	8	9	Total
0	941	0	18	4	1	22	17	5	17	18	1043
1	0	1107	57	15	22	16	9	39	55	10	1330
2	1	2	814	24	6	4	10	17	10	4	892
3	3	3	28	891	2	85	0	8	31	14	1065
4	1	1	16	2	868	18	19	19	25	73	1042
5	9	1	0	13	3	620	19	0	44	0	709
6	14	5	38	10	9	21	875	2	17	1	992
7	1	2	19	23	1	13	0	880	13	79	1031
8	8	14	38	18	15	69	9	3	739	12	925
9	2	0	4	10	55	24	0	55	23	798	971
Total	980	1135	1032	1010	982	892	958	1028	974	1009	10000

The testing confusion matrix for the One vs One Classifier is shown below:

	0	1	2	3	4	5	6	7	8	9	Total
0	961	0	9	6	1	13	8	1	8	6	1013
1	0	1124	17	4	2	7	6	16	17	5	1198
2	5	3	927	12	5	2	7	18	9	1	989
3	1	3	13	931	1	33	0	2	20	10	1014
4	0	0	10	2	929	10	7	12	10	30	1010
5	2	1	5	20	2	785	14	0	32	9	870
6	6	1	17	3	8	16	914	0	11	0	976
7	3	1	7	8	4	2	0	951	10	23	1009
8	1	2	26	18	3	20	2	3	847	5	927
9	1	0	1	6	27	4	0	25	10	920	994
Total	980	1135	1032	1010	982	892	958	1028	974	1009	10000

(Note: I had my code output my confusion matrices as .CSV files and then I cleaned them up a bit for the report. I hope this is ok. Also I put the predicted values on the rows and the true labels on the columns, this doesn't change the meaning but it is the reverse of the book.)

Because the testing and training error rates are relatively close we can assume that both of the models generalized fairly well on the data both had a similar scaling of error.
For both the One vs All and One vs One it seems that 2 and 5 are very hard to identify as they had a very high error rate. The two most commonly confused digits were 5's that the model mistook for 3's which kind of makes sense as they are visually somewhat similar. The easiest

digits to recognize were 0 and 1 maybe this is because they look somewhat distinct from the others or in the case of 1 because it had the most examples. The One vs One was able to achieve insanely low error rates on zero and one in particular.

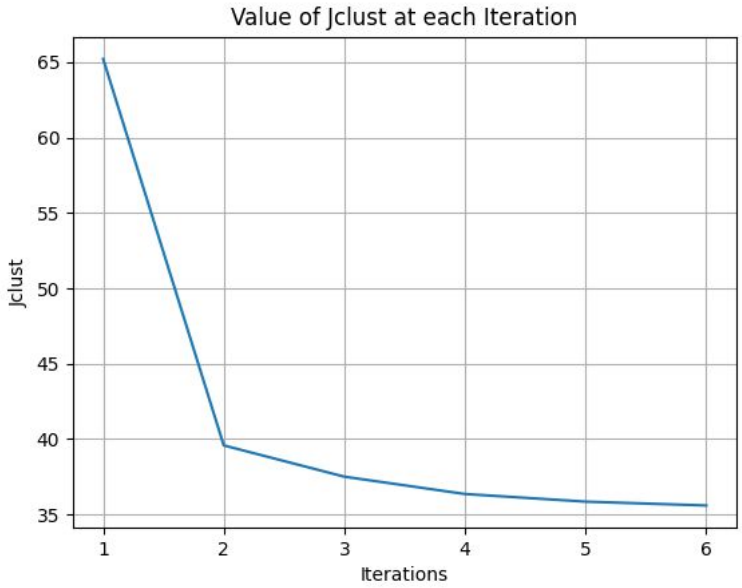
d.) The One vs One Classifier did much better than the One vs All Classifier. I think this is because each of the binary classifiers inside are being trained on a more specific dataset which could help with getting better accuracy. This is corroborated by the fact that a single binary One v One Classifier seems to do better than a single binary One vs All classifier. Furthermore because each example is going through more classifiers there may be an opportunity for them to catch the others mistakes.

Problem 2:

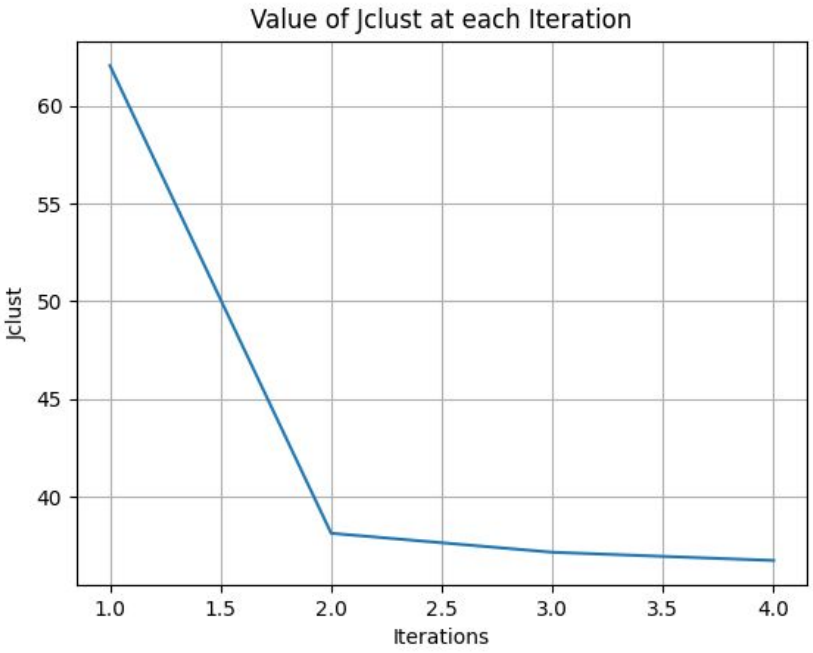
For this problem I used the method of randomly assigning the cluster representatives as the initial conditions.

First Trial - K= 20 P = 30:

Plot of Jclust for minimum run:

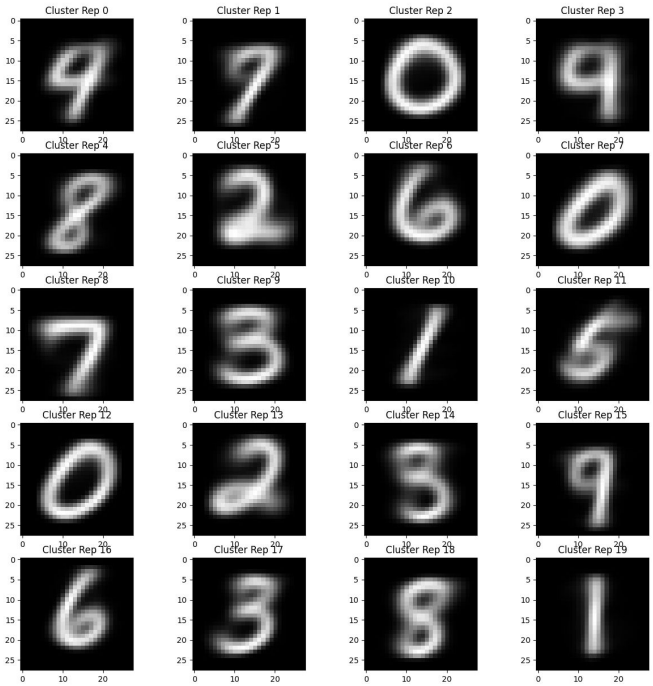


Plot of Jclust for maximum run:

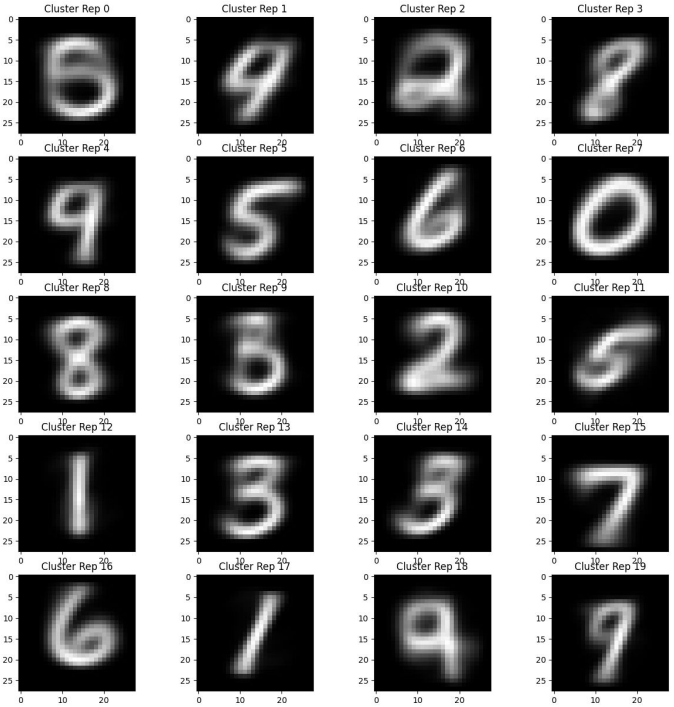


Comments: The first thing to note is that the two runs had very similar Jclust values, I think the minimum was 36 and the maximum was around 37. However it seems that the run that minimized Jclust ran for two more iterations which could have potentially been the defining factor that brought this run down.

Plot of the cluster representatives for the minimum run:



Plot of the cluster representatives for the maximum run:



Comments: These plots are very telling as we can see that the run that better minimized Jclust have much clearer digits that are much closer to those contained in the original data set. Another thing that is interesting is that neither plot contains a distinct 4. Which will become apparent as we look at some of the members of these clusters.

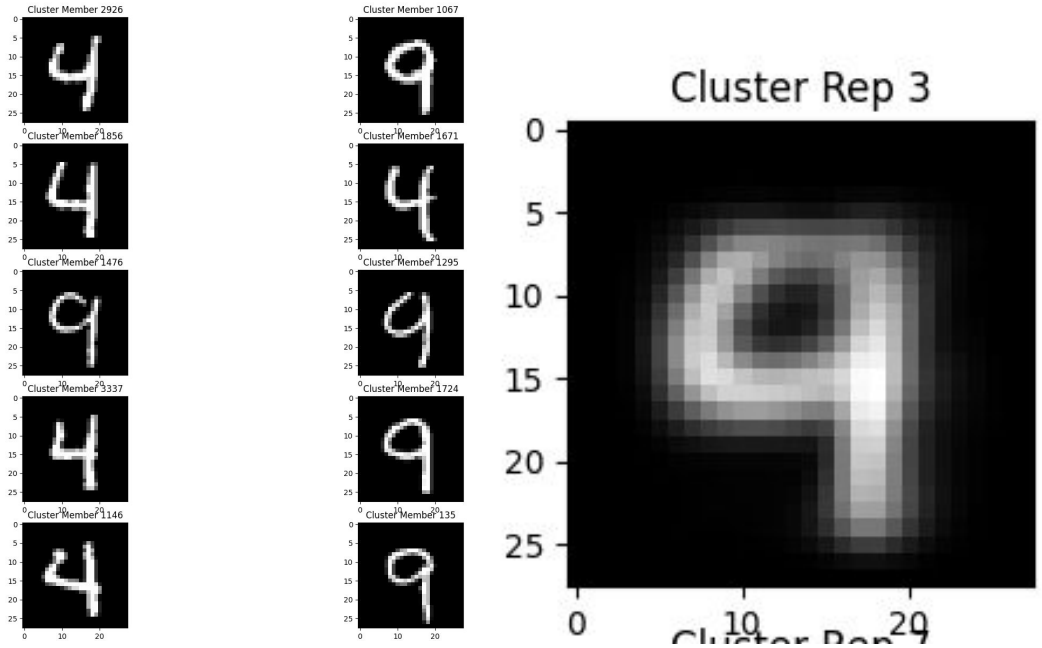
Table of the digits that the clusters ten closest members represent for the minimum run:

Cluster Number	Representative Digit	Closest Member Digits
0	9?	[9,4,4,9,9,4,9,4,4,4]
1	7?	All 7s
2	0	All 0s
3	9?	[4,4,9,4,4,9,4,9,9,9]
4	8	All 8s
5	2	All 2s
6	6	All 6s
7	0	All 0s
8	7	All 7s
9	3	All 3s
10	1	All 1s
11	5	[6,5,5,5,5,5,5,6,5,6]
12	0	All 0s
13	2	All 2s
14	3?	[3,3,5,5,3,3,3,3,3,5]
15	9	[9,7,9,9,7,9,7,9,9,9]
16	6	All 6s
17	3	All 3s
18	8	[8,5,8,8,8,8,8,8,8,8]
19	1	All 1s

Table of the digits that the clusters ten closest members represent for the maximum run:

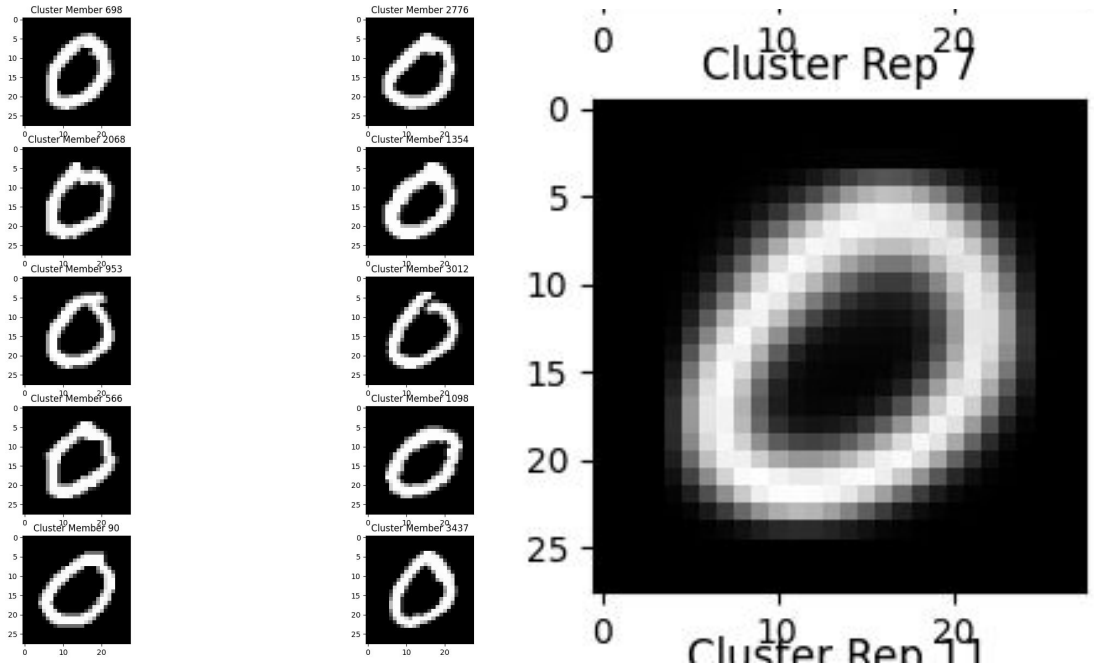
Cluster Number	Representative Digit	Closest Member Digits
0	5?	All 5s
1	4?	[4,4,9,4,4,9,9,4,4,9]
2	2	All 2s
3	8	[7,8,1,1,8,8,9,8,8,8]
4	9	[9,4,4,9,9,9,9,4,4,9]
5	5	All 5s
6	6	All 6s
7	0	All 0s
8	8	All 8s
9	3?	[3,3,3,3,5,5,5,3,5,3]
10	2	All 2s
11	5	All 5s
12	1	All 1s
13	3	All 3s
14	3?	[3,3,3,5,3,3,3,3,3,3]
15	7	All 7s
16	6	All 6s
17	1	All 1s
18	4?	[4,4,9,4,9,4,4,4,4,9]
19	9	[7,9,7,9,7,9,7,7,9,7]

For both of the runs 4 and 9 are by far the most confused digits here is the plot of the closest members cluster 3 of the min run:

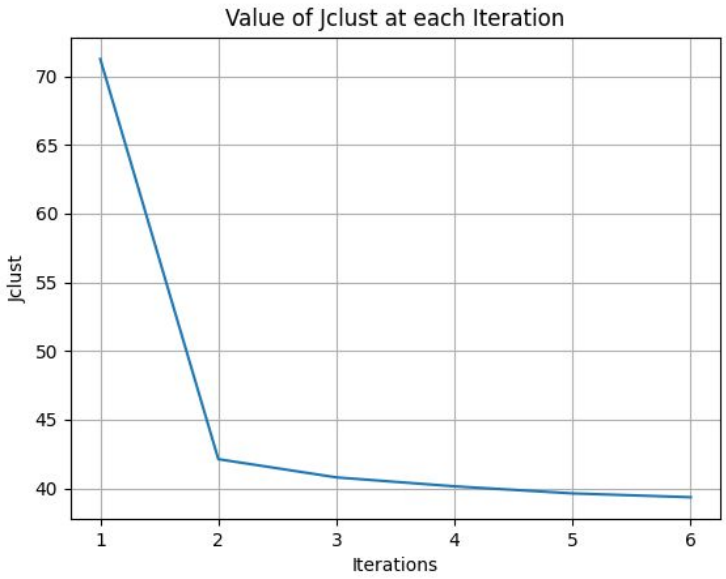


This makes a lot of sense as visually they are very similar and therefore their vector representations would likely be similar as well.

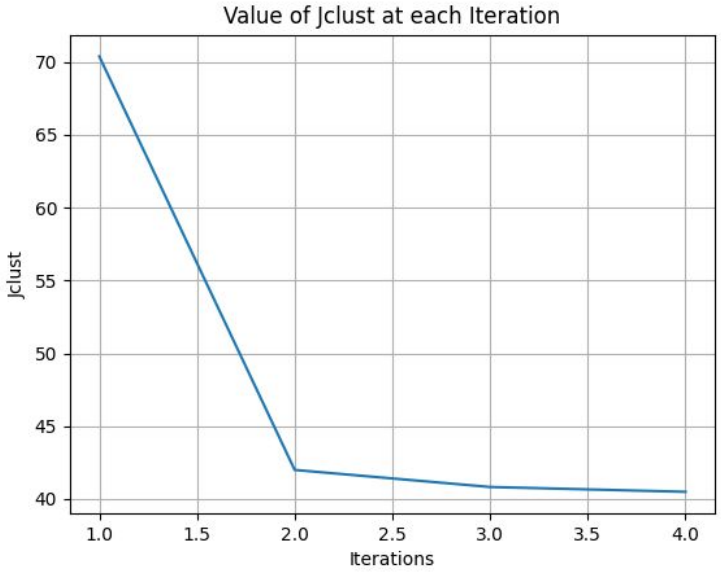
The clusters representing 0s however were very consistent for instance here is the plot of cluster 7 of the max run:



Second Trial - K = 10 P = 20:
Plot of Jclust for minimum run:

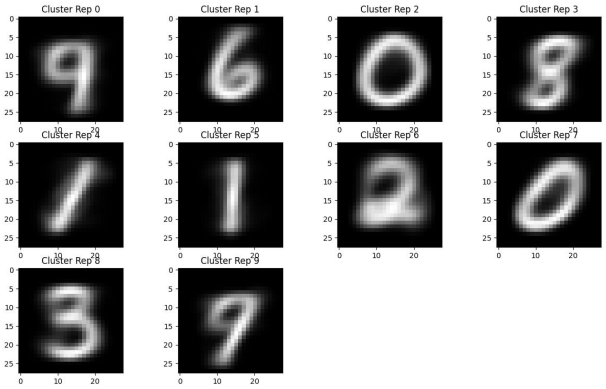


Plot of Jclust for maximum run:

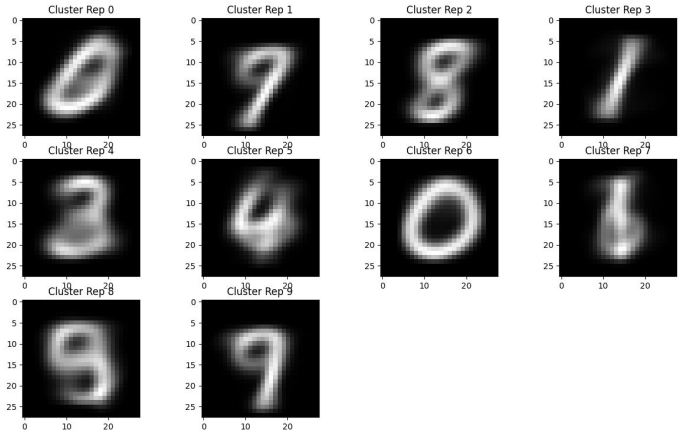


Comments:Similar to the runs in the first trial the run that converged to a lower value of Jclust ran for 2 more iterations. Also important to note that these had higher Jclust values than the run with K equal to 20. This is not necessarily bad if the clusters better represent individual digits this model could be considered to be more general.

Plot of the cluster representatives for the minimum run:



Plot of the cluster representatives for the maximum run:



Comments: Similar to the previous trials the run with the lower Jclust has clearer digits that are very close to actual numbers. The max run has some digits that are really illegible which indicates that the clusters are not specific to any digits and we can expect to see a lot of misclassified digits in those clusters. Something that is interesting is the slanted one vs straight one in the minimum run and the two clusters representing 0. I wonder why it ignored some numbers entirely in favor of making two clusters of 1s and 0s. After looking at the cluster members there are some interesting things going on here. First thing that is interesting to me is that there are much more consistent groups of closest members in this trial however since not every digit is represented there must be a ton of digits randomly strewn through these clusters. Another thing that is interesting is my assumption that there would be more errors in the clusters that I couldn't read wasn't necessarily true as the closest members were very consistent in those.

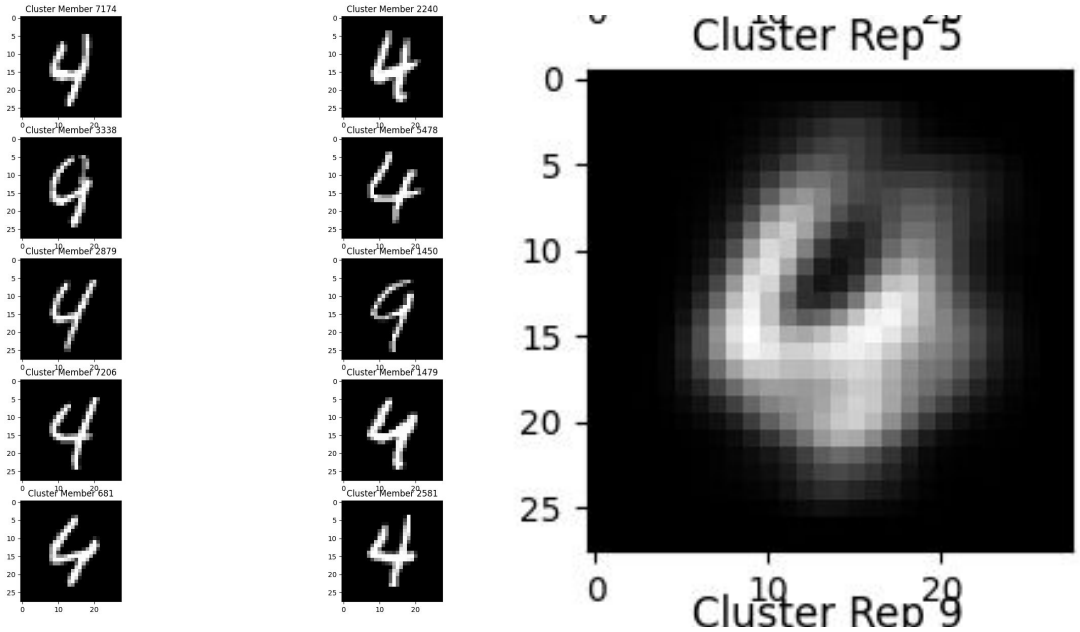
Table of the digits that the clusters ten closest members represent for the minimum run:

Cluster Number	Representative Digit	Closest Member Digits
0	9	All 9s
1	6	All 6s
2	0	All 0s
3	8	All 8s
4	1	All 1s
5	1	All 1s
6	2	All 2s
7	0	All 0s
8	3	All 3s
9	9	[7,7,9,7,9,7,7,7,7,7]

Table of the digits that the clusters ten closest members represent for the maximum run:

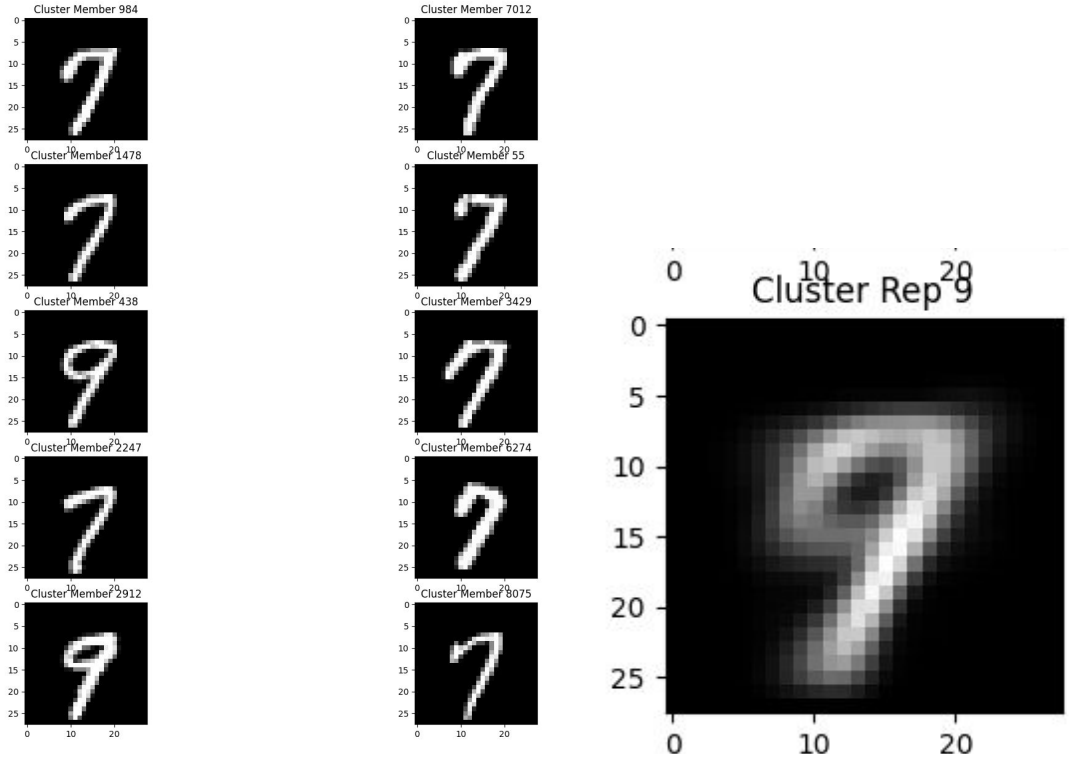
Cluster Number	Representative Digit	Closest Member Digits
0	0	All 0s
1	7? Originally thought 9.	All 7s
2	8	All 8s
3	1	All 1s
4	3?	All 3s
5	4	[4,9,4,4,4,4,4,9,4,4]
6	2	All 0s
7	1?	All 1s
8	8?	[9,4?,9,9,3?,9,9,9?,9,9]
9	9? Must actually be 7.	All 7s

For the max run the most confused digits were once again 4 and 9 here is the plot of cluster 8:

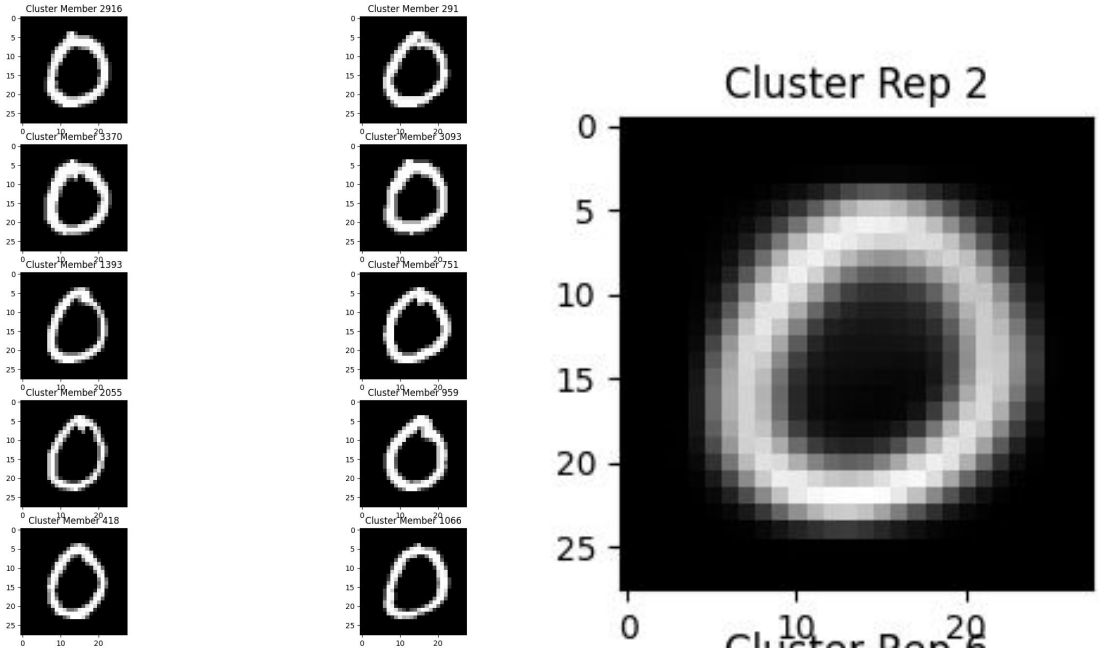


With this amorphous blob being their representative though it makes sense how the two could get confused.

For the min run the only non consistent cluster was number 9 that represented 7s and 9s:

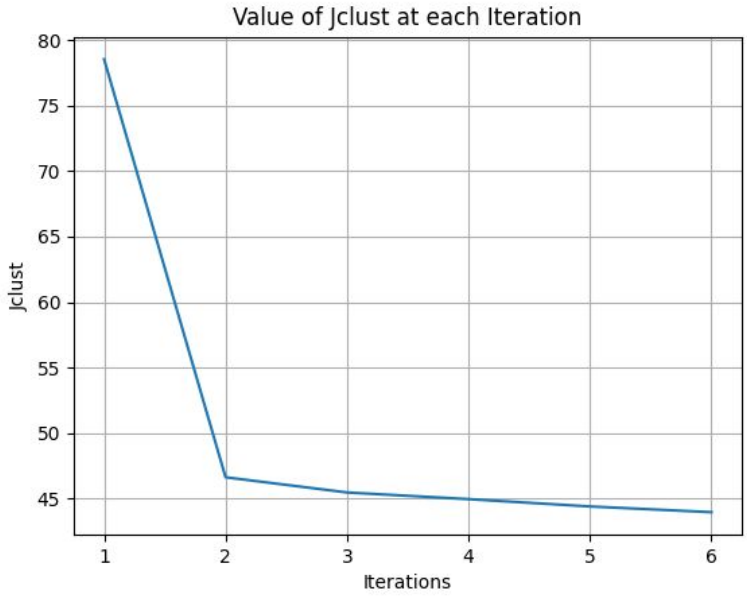


The clusters representing 0s however were once again very consistent for instance here is the plot of cluster 2 of the min run:

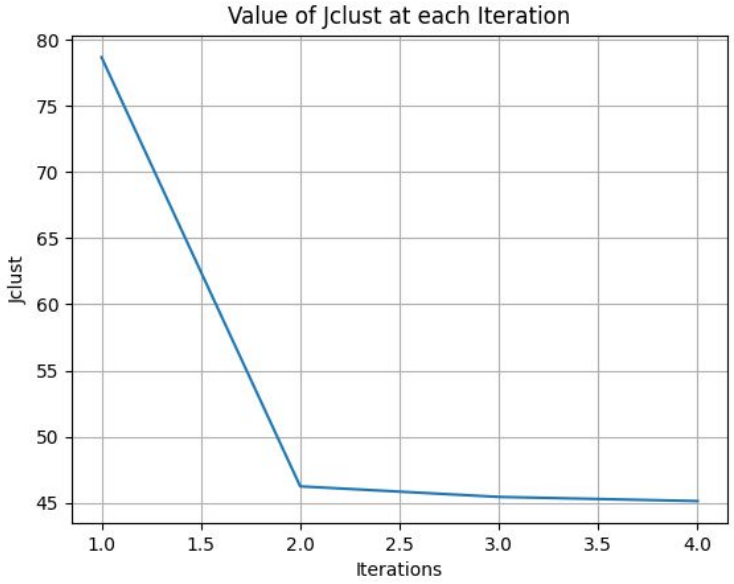


Third Trial - K = 5 P = 10:

Plot of Jclust for minimum run:

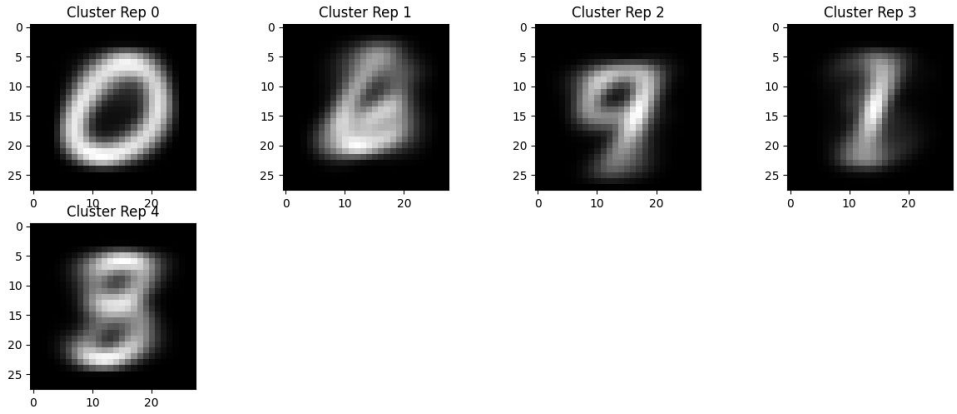


Plot of Jclust for maximum run:

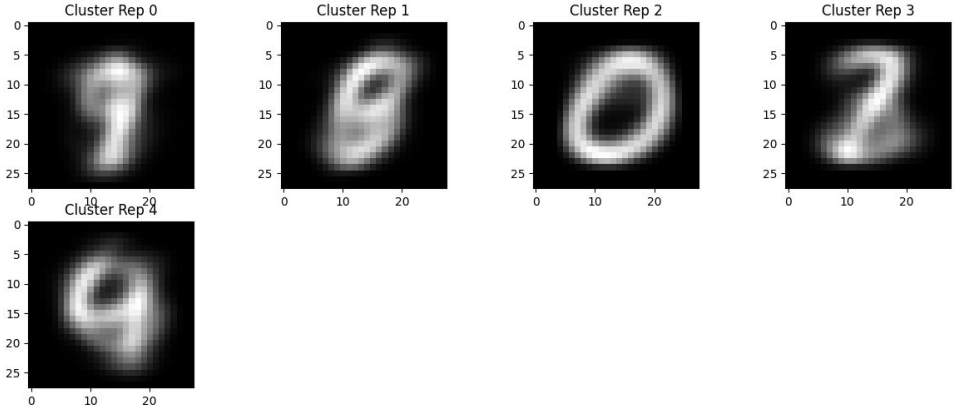


Comments:Similar to the runs in the previous two trials trial the run that converged to a lower value of Jclust ran for 2 more iterations. Also important to note that these had much higher Jclust values than the run with K equal to 20 or even the one with K equal to 10. This likely will correlate to a much poorer classification of digits.

Plot of the cluster representatives for the minimum run:



Plot of the cluster representatives for the maximum run:



Comments: As in the previous two runs the minimum run representatives are clearer although that isn’t saying much as both sets are very hard to read. This is to be expected when the number of clusters is much lower than the number of real world categories as each cluster necessarily will have to represent more than one digit.

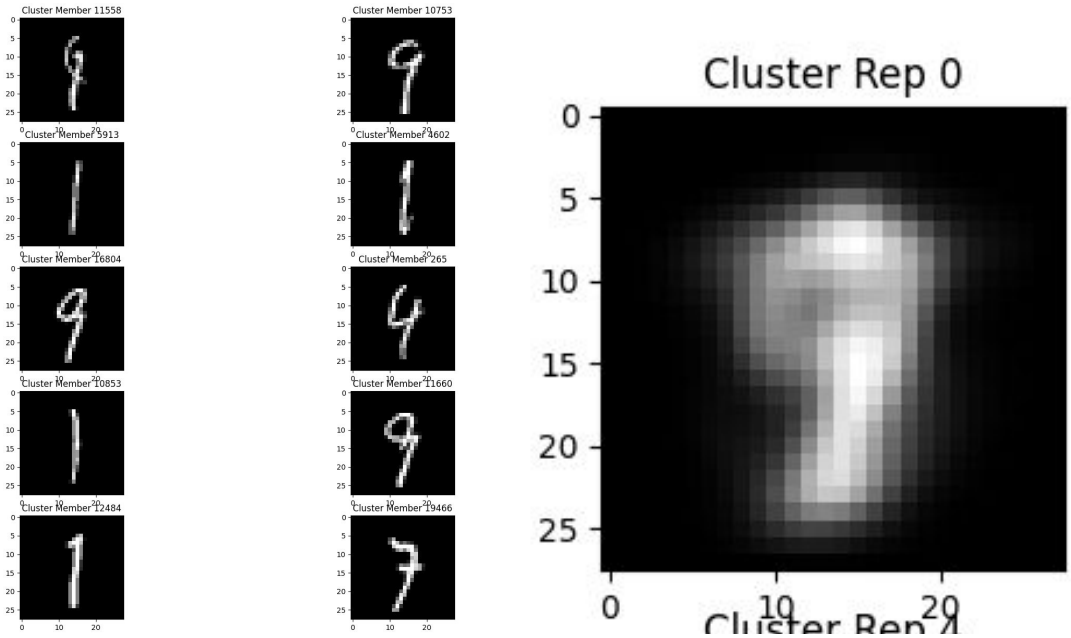
Table of the digits that the clusters ten closest members represent for the minimum run:

Cluster Number	Representative Digit	Closest Member Digits
0	0	All 0s
1	6?	All 6s
2	9?	All 9s
3	1?	All 1s
4	8?	[3,8,3,3,3,3,3,3,3,8]

Table of the digits that the clusters ten closest members represent for the maximum run:

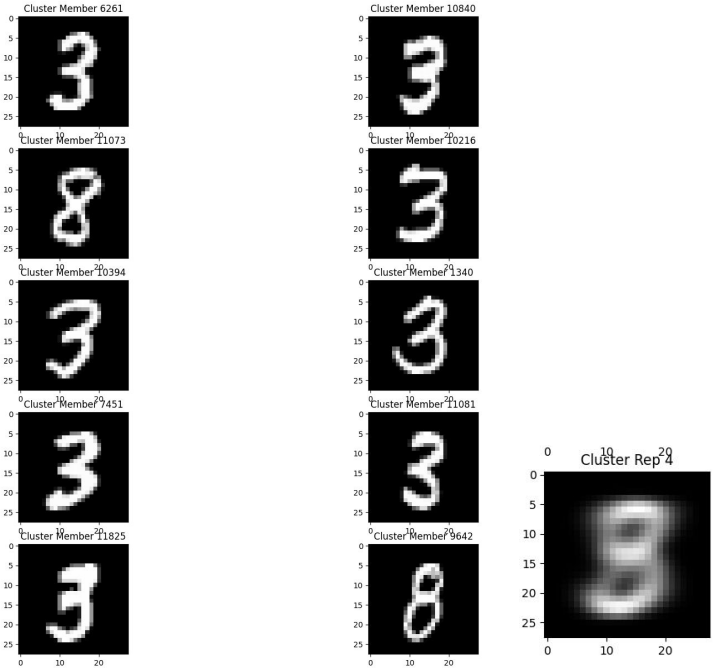
Cluster Number	Representative Digit	Closest Member Digits
0	9?	[9,1,9,1,1,9,1,4,9,7]
1	8?	[9,9,9,9,8,9,9,9,8,9]
2	0?	All 0s
3	1? Originally thought 7.	[1,1,1,1,1,2?,1,2,1,1]
4	4? Originally thought 9.	[4,4,4,4,4,4,4,9,4,4]

For the max run there were many misclassifications I think the most interesting is cluster 0 where it was a free for all for all skinny straight numbers:

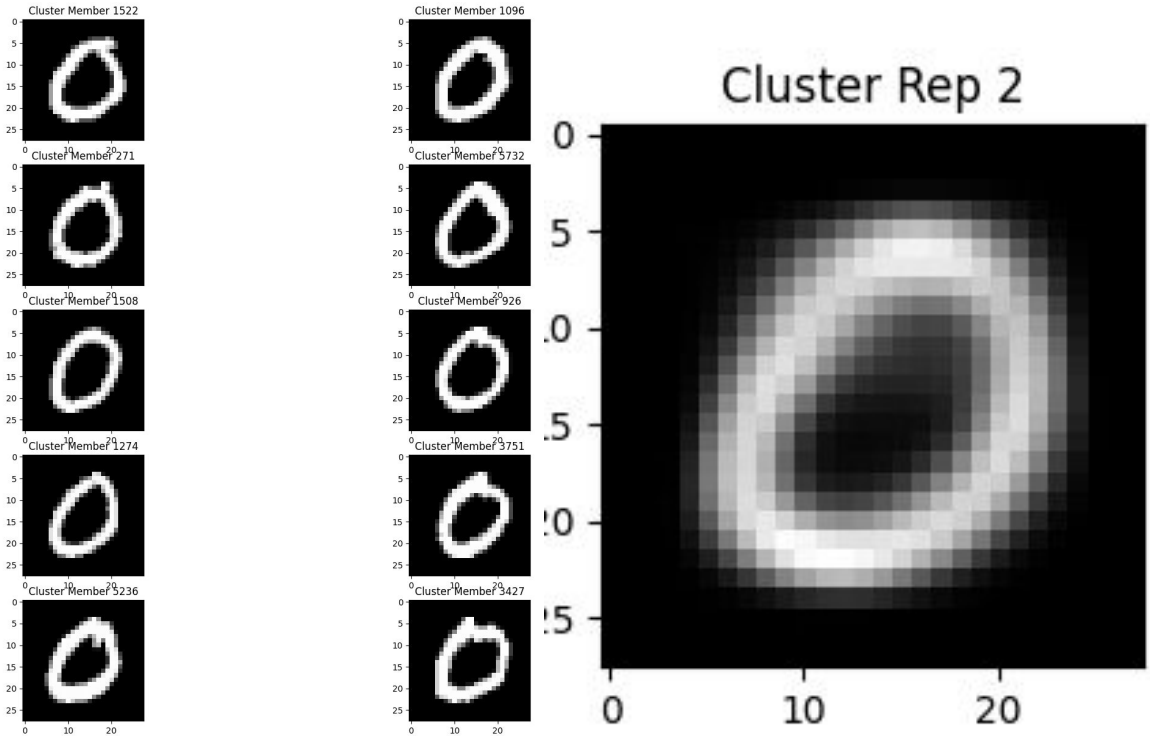


I think this is interesting because it points to this idea that when there aren't enough clusters for every category they begin to just represent the features of the initial categories for instance this one is the skinny numbers.

For the min run the only non consistent cluster was number 4 that represented 3s and 8s:



The clusters representing 0s however were once again very consistent for instance here is the plot of cluster 2 of the max run:



Comparing the results of the supervised binary classifier and this unsupervised classifier is kind of a difficult question as they seem to me to serve different purposes and have different use cases. In the case of this dataset where it's well labelled and the categories are easily defined it makes sense that the supervised binary classifier will crush the k-means algorithm in accuracy and efficiency. However, in a situation where the data is not properly labelled or the groups are not so well defined k means is the obvious choice. That strength of k-means is also a failure in some ways because the clusters will still have to be labelled afterwards. However in terms of a cool factor I think k means is much cooler because of its ability to classify irregardless of what is actually in the image, it's such a general utility.