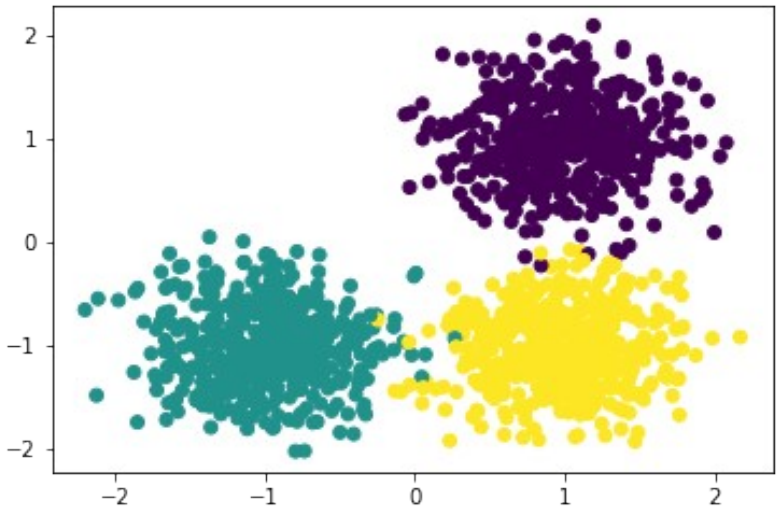
Data Clustering – Moses Mbabaali - 20th April 2021

This assignment was based on working with data clustering algorithms using synthetic data and hand written digits. The algorithms in picture that were used are k-means, fuzzy c-means, possibilistic c-means and graded possiblilistic c-means. Several tasks were involved, the first was about generating synthetic data, acquisition of the hand written digits, developing the algorithms and using the algorithms.

Task 1

For the data generation task I used sklearn python library. This was a pretty smooth task because almost all the heavy lifting is done by sklearn. To generate the data just call from "***from sklearn.datasets import make_blobs***" and then define the centers, "***centers = [[1, 1], [-1, -1], [1, -1]]"*** and finally call place the data in X and y ***X,y=make_blobs(n_samples=1500,centers=centers,cluster_std=0.4,random_state=0).*** So in this case we are making 1500 samples with defined centers and the stand deviation of 0.4. The output of the operation can be visualized below.



Task 2

Task 2 was pretty straight forward without any hiccups, basically the goal was data importation for the handwritten digits. I did that successfully.

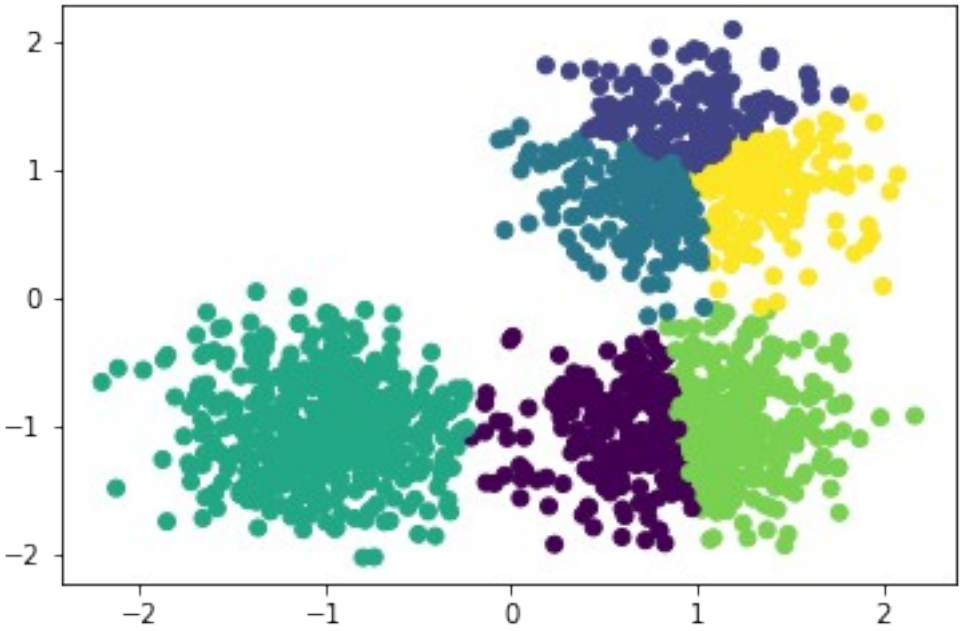
Task 3

The task at hand in this section was development of clustering algorithms. Several approaches were used in this section. For K-means I used the sklearn python library as one of the sources to facilitate this task, then for fuzzy c-means I used another library called fcmeans, then for possiblilistic c-means skfuzzy was used which is part

of the python scientific libraries. After developing of the algorithms I tested them on the available data that was generated earlier and below are the results.

Kmeans (The best iteration was chosen)

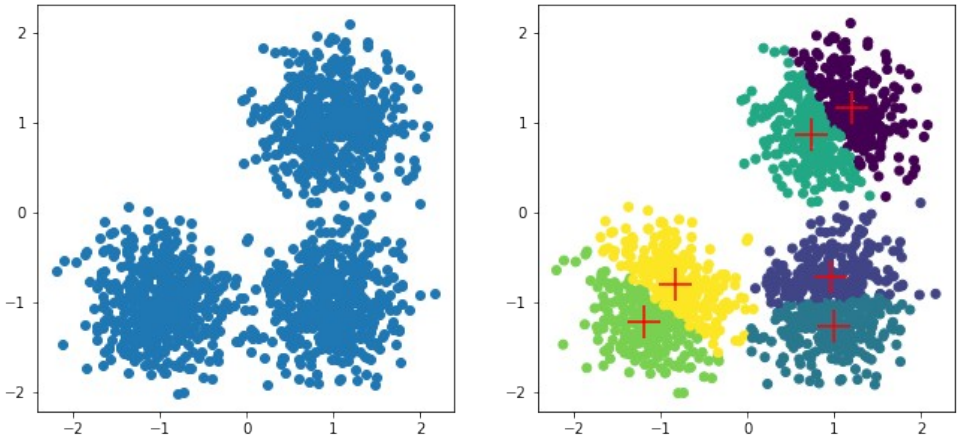
The function Kmeans_c(clusters, iterations, X) is used as the implemented version of the algorithm. It takes in the number of clusters, iterations needed, and the X which is the data to cluster. The best iteration is chosen among all, that is one where the least distortion or Inertia happens.



The graphic above shows the clustering of the data into 6 clusters.

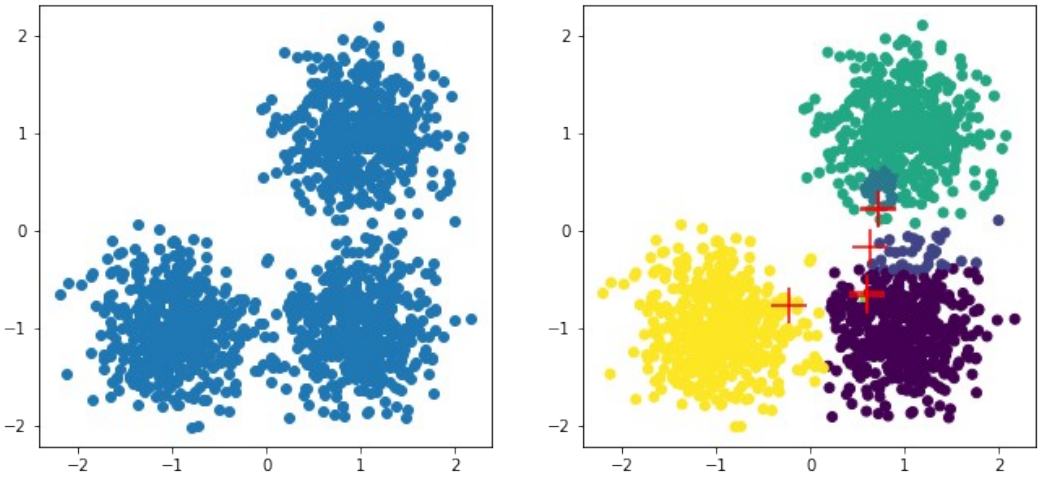
Fuzzy c-means

In the case of fuzzy c-means the function fcmeans(X,clusters) was implemented, it takes in the data X and the number of clusters that one wants to get out the dataset. After implementation the following results were produced. With further introspection we can easily notice that the data was clustered a little differently than how K-Means did its clustering.



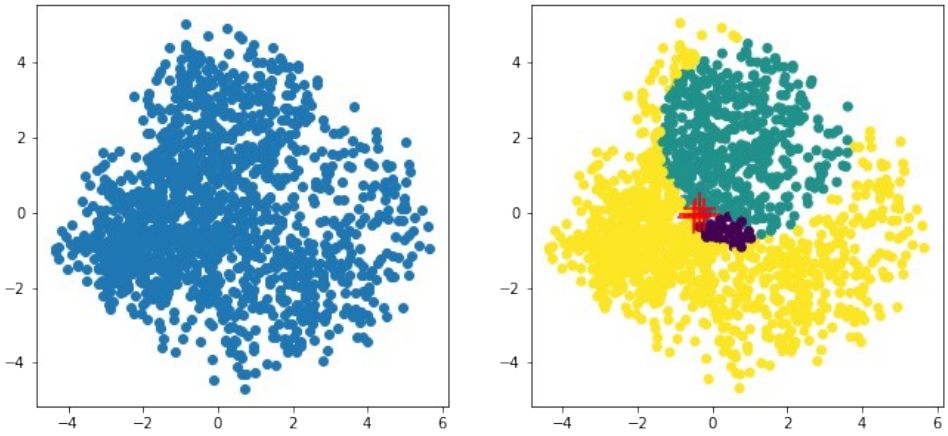
Possiblilistic c-means

The possiblistic c-means was implemented with the following function **PossiblisticC(data, nclusters,centroids, iterations)** basically the function takes in the data, then the required number of clusters, the centroids and number of iterations. The centroids are the initial points where the algorithm will start. It will go on adjusting until it finds a sweet spot for all the data. After the implementation the following clusters were realized.



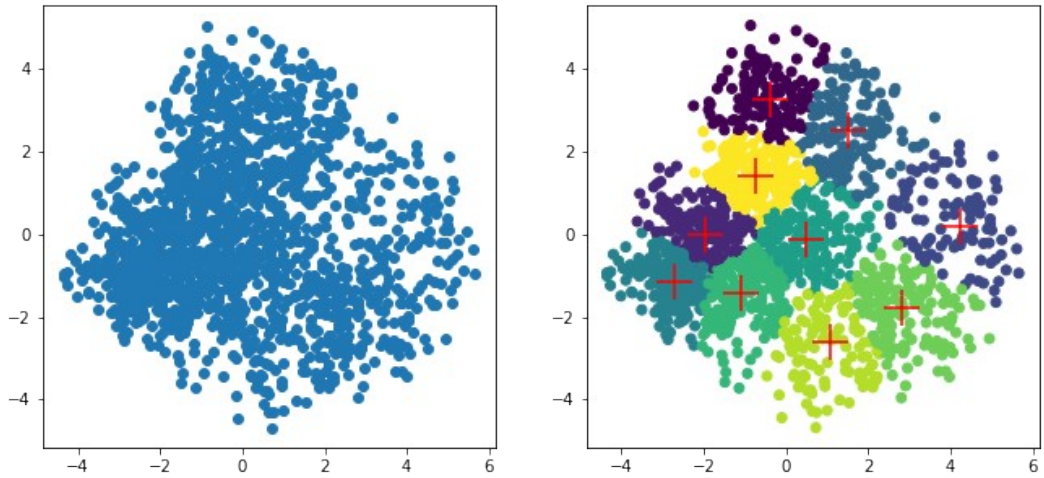
Unlike the previous algorithms this one does not do perform as well even if the conditions are pretty much the same.

Task 4 (Unsupervised Analysis of the hand written data)



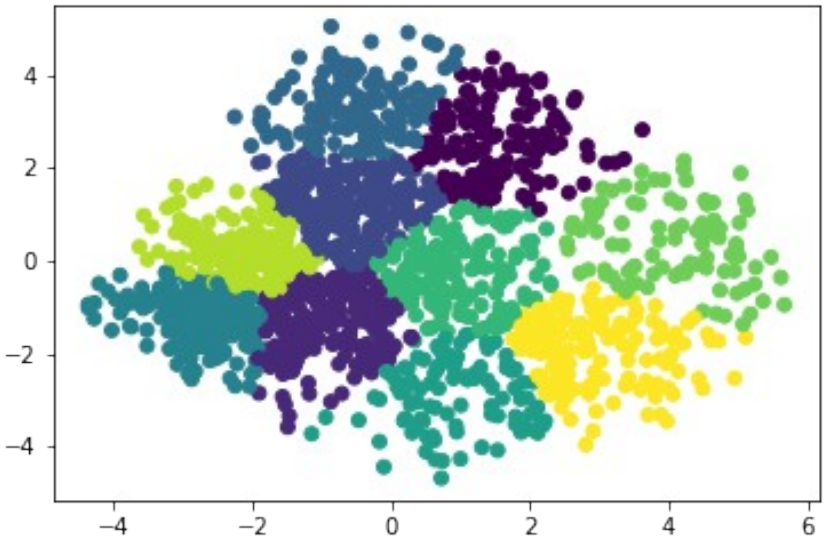
For this task I though it would be necessary to first have a sneak preview of the data. This meant first doing some quick plots. But because the dataset has many columns I performed some pca on the data to get the key principal components. This also involved trying to cluster the data using he algorithms developed above. The first plot is for possiblistic cmeans.

C-means Handwritten digits visualization with PCA.



Similarly the clustering for the dataset is pretty different than how possiblistic cmeans did its clustering.

Kmeans Clustering Handwritten Digits



Again the clustering in this case is also different from the ones above.

The cardinality of the data in this case for each of the 10 clusters is shown below.

Task4 Cardinality

C-means.

```
Class : 6 Count : (array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]), array([161, 162,
159, 159, 161, 159, 161, 158, 155, 158]))
```

For possiblistic c-means the data was split in only one class. This is class 6. From the visualization we can tell that it kind of does things differently in comparison to the rest of the algorithms.

K-means Cardinality.

For k-means the data was as well clustered differently and placed in different classes.

```
Cluster : 0 Class count : (array([0, 2, 4, 5, 6, 9]), array([ 4,  1, 75,  3,
49,  1]))
Cluster : 1 Class count : (array([1, 3, 5, 6, 8, 9]), array([  4, 120,  11,
1,  11,  11]))
Cluster : 2 Class count : (array([0, 1, 2, 4, 5, 6, 7]), array([  5,   3,
4,  51,   4, 102,   2]))
Cluster : 3 Class count : (array([0, 1, 2, 3, 4, 5, 8]), array([  1,  10,
107,   1,   2,   3,   5]))
Cluster : 4 Class count : (array([1, 4, 5, 6, 7, 8, 9]), array([ 53,  16,
5,   4, 131,   3,   7]))
Cluster : 5 Class count : (array([0, 2, 3, 5, 7, 8, 9]), array([  4,  34,
8,  12,   1, 104,  19]))
Cluster : 6 Class count : (array([2, 3, 4, 5, 6, 7, 8, 9]), array([ 1,  3,
5, 60,  1,  7, 12, 74]))
Cluster : 7 Class count : (array([0, 6]), array([143,   2]))
Cluster : 8 Class count : (array([0, 3, 5, 6, 8, 9]), array([ 4, 19, 61,  2,
19, 43]))
Cluster : 9 Class count : (array([1, 2, 3, 4, 7, 8, 9]), array([92, 12,  8,
12, 17,  1,  3]))
```

Since there are 10 clusters, above is the count for each of the individual values in the cluster. For example in in cluster 0, there are 4 zeros, 1 two, 75 fours ,3 fives, 49 sixes, and one 9.

C-means Cardinality.

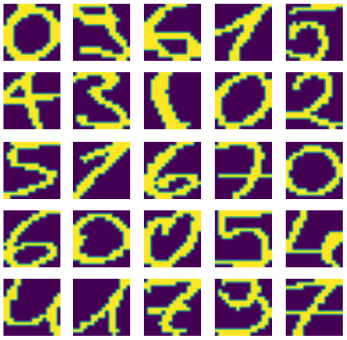
```
Cluster : 0 Class count : (array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
array([101,   7,  39,  85,  52, 114,  83,   4,  60,  78]))
Cluster : 1 Class count : (array([1, 2, 3, 4, 5, 6, 7, 8, 9]), array([148,
109,  65,  36,  24,  15, 153,  77,  59]))
Cluster : 2 Class count : (array([0, 2, 3, 4, 5, 6, 8]), array([ 8,  2,  3,
16,  2,  2,  2]))
Cluster : 3 Class count : (array([0, 2, 4, 5, 6, 8, 9]), array([5, 2, 2, 5,
2, 4, 6]))
Cluster : 4 Class count : (array([0, 1, 3, 4, 5, 6, 7, 8, 9]), array([1, 6,
2, 3, 2, 3, 1, 5, 4]))
Cluster : 5 Class count : (array([1, 4, 5, 6, 8, 9]), array([1, 9, 3, 1, 1,
3]))
```

```
Cluster : 6 Class count : (array([2, 4, 6, 8]), array([1, 2, 2, 3]))
Cluster : 7 Class count : (array([0, 2, 4, 5, 6, 8, 9]), array([46,  2, 36,
8, 53,  3,  6]))
Cluster : 8 Class count : (array([2, 4]), array([1, 5]))
Cluster : 9 Class count : (array([2, 3, 5, 9]), array([3, 4, 1, 2]))
```

The function d_fcmeans(tesB,tcat) is used to the cardinality count. From the visuals above we can easily tell that the clustering in each of the cases is done differently.

Task 4 identifying ambiguous labels.

In this task I ended up using deep learning to identify those labels that are ambiguous. It was necessary to first view the digits.



Above is the random view of the data. But there are some numbers that are mostly misclassified. To get this numbers I had to do some deep learning with keras using a sequential model to identify these digits. After running the algorithm it was discovered that the most misclassified and ambiguous digits were

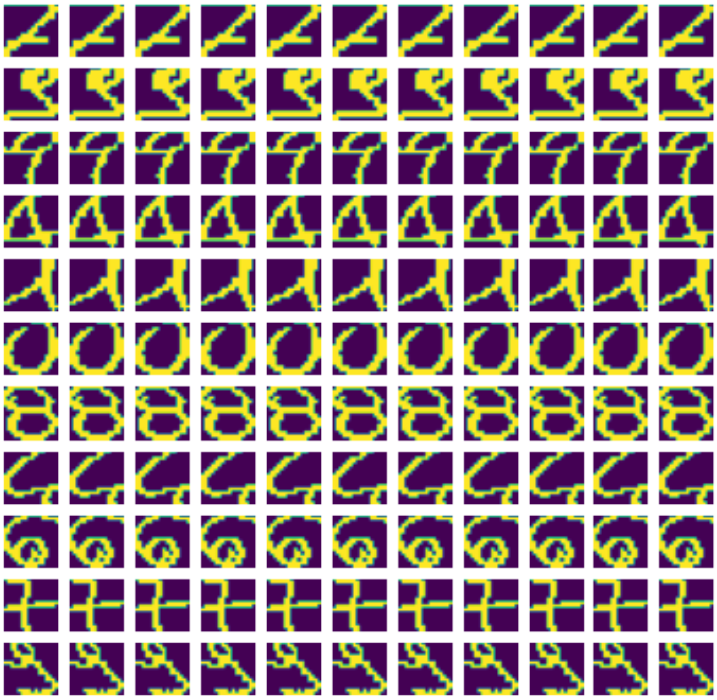
```
array([3, 3, 9, 2, 4, 6, 9, 2, 7, 3, 3, 2, 8, 9, 1, 3, 3, 2, 9, 0, 1, 5,
       1, 2, 4, 9, 4, 3, 1, 3, 3, 3, 2, 5, 5, 1, 4, 1, 9, 8, 1, 4, 9, 9,
       8, 7, 9, 3, 0])
```

The count of those values produced

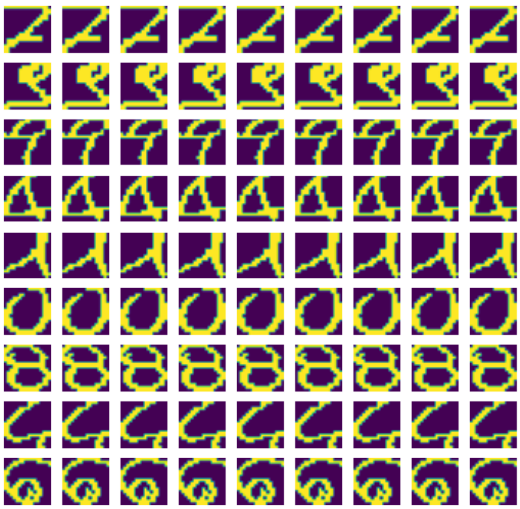
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]), array([ 2,  7,  6, 11,  5,  3,  1,
2,  3,  9]))
```

From the data we can easily ascertain that 9s fall prey to missclassifications, 3s are also culprit to that, 1s and 2s.

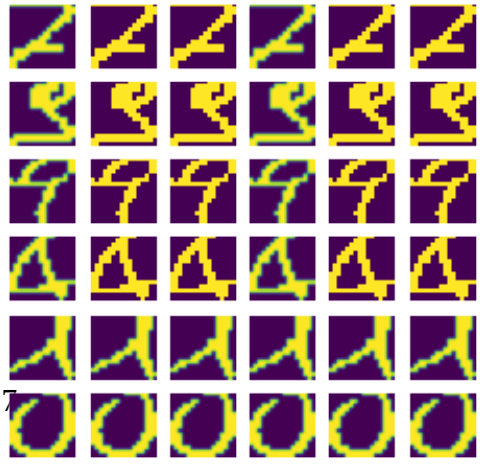
Digits that were wrongly classified as threes.



Digits that were wrongly classified as nines.



Digits that were wrongly classified as twos.



From above we can easily see the digits that were wrongly classified. This can either be alluded to the fact that some numbers were poorly written and so the classifier and clustering algorithm would but them in wrong categorizations. In all it was a challenging exercise but also rewarding.