K-Means Algorithm
(Cmpe481 Fall 2023 Assignment1)

Alp Tuna - 2019400288

1) How to run the project?

1) Go to the codes folder.
2) Install the required packages. They are written in the `requirements.txt` file which is in misc folder.
3) Select a mode and a file name.
   2.1) If you select mode = 1, code generates data points which are suitable for clustering and draws them using matplotlib. Also saves the data points as a txt file with a given filename in the same folder.
   2.2) If you select mode = 2, code generates 3 circles inside each other and draws them using matplotlib. These data points are not suitable for k-means clustering algorithms. Also saves the data points as a txt file with a given filename in the same folder.
   2.3) If you select mode = 3, code applies custom k_means clustering that I have implemented to the dataset. Dataset is read from the .txt file with a given filename. You can use the data generated in the previous steps. It visualizes the first 3 iterations, last iteration and the change of the objective function throughout the algorithm.
   2.4) If you select mode = 4, code applies k_means clustering algorithm using scikit library and displays the final result using matplotlib.
   2.5) If you select mode = 5, code plots the cost function of k values from k =1 to k = 10. You can change the range in the `find_optimal_k()` function. By using elbow point, we can determine the optimal option for k.

   You can specify the mode and filename here.

```python

if __name__ == "__main__":
    filename = "cluster_data.txt"
    k = 2
    mode = 3

    if mode == 1:
        generate_data_suitable_for_clustering(filename)
    elif mode == 2:
        generate_data_not_suitable_for_clustering(filename)
    elif mode == 3:
        data = read_data(filename)
        k_means(data, k)
    elif mode == 4:
        data = read_data(filename)
        k_means_scikit(data, k)
    elif mode == 5:
        data = read_data(filename)
        find_optimal_k(data)
    else:
        raise Exception("Enter a valid mode")

```

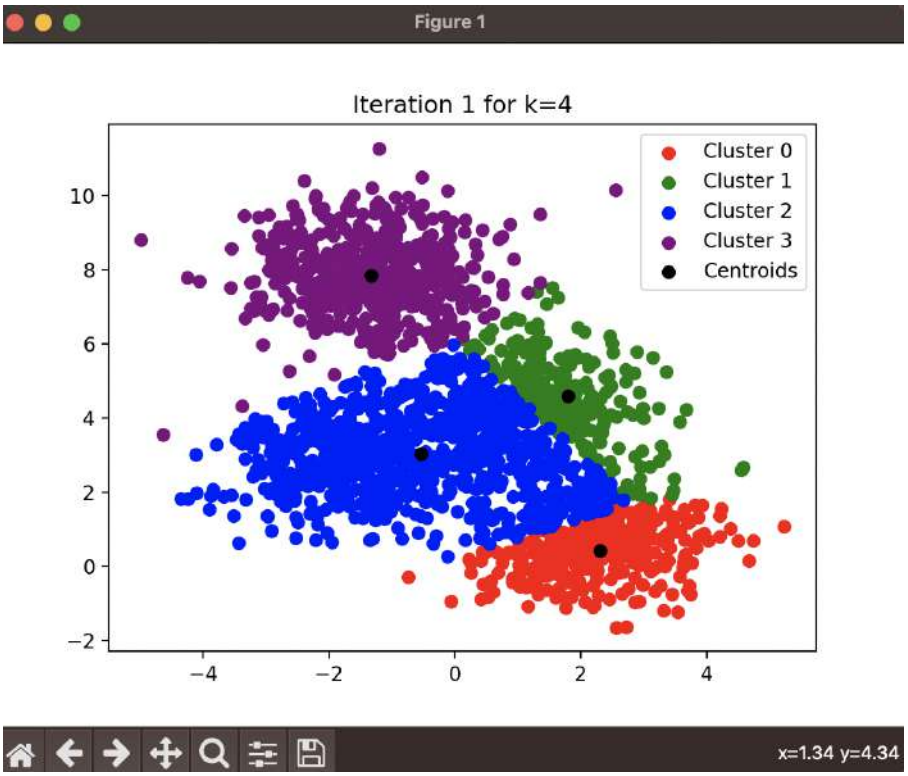
(variable) filename: Literal['cluster_data.txt']

4) Run `python assignment1.py` or `python3 assignment1.py` depending on your python version.

2) Custom K-Means Algorithm Results

Note: Up to point 5, we refer to nice_dataset.txt in misc folder

For k = 4:



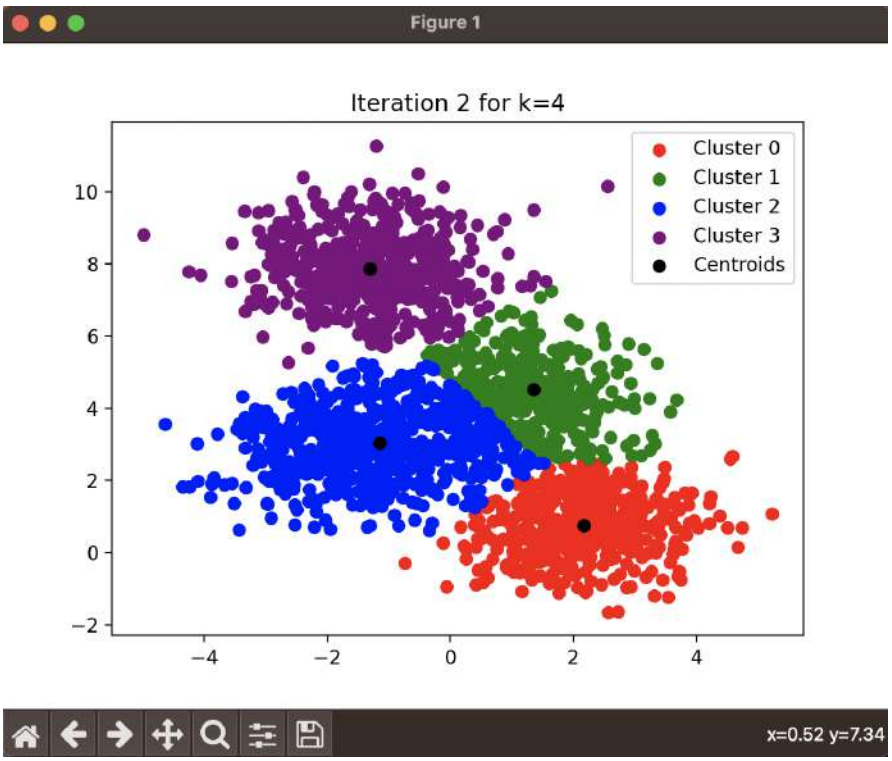


Figure 1
Iteration 2 for k=4
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Centroids
10
8
6
4
2
0
−2
−4
−2
0
2
4
x=0.52 y=7.34

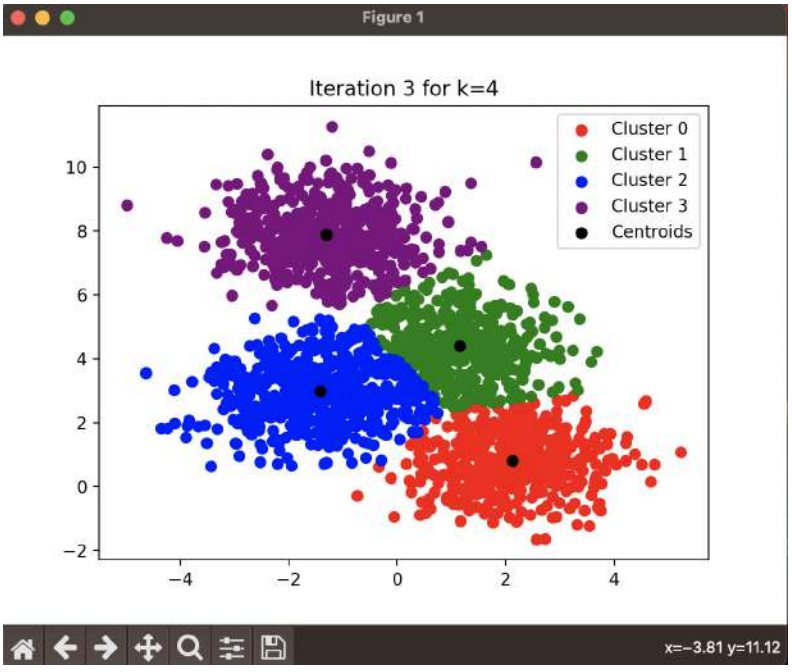
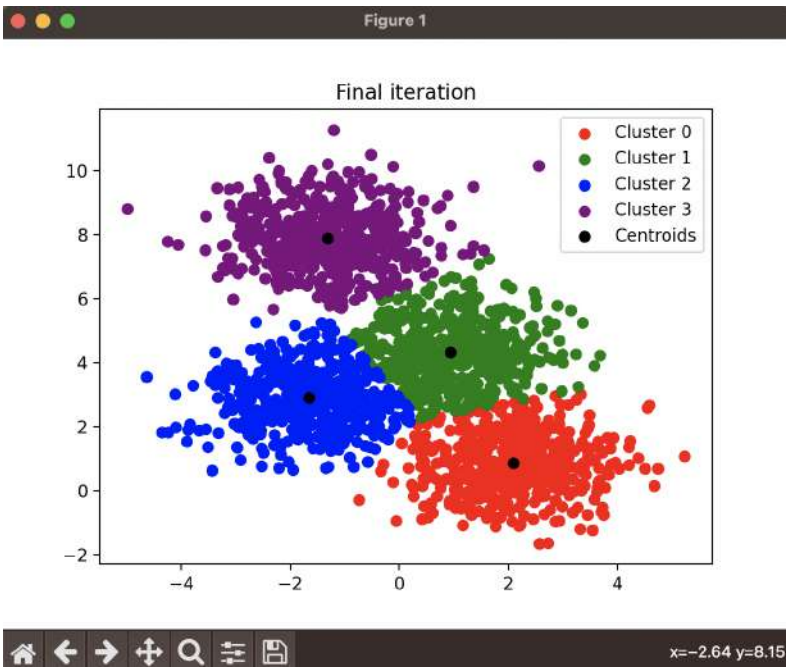
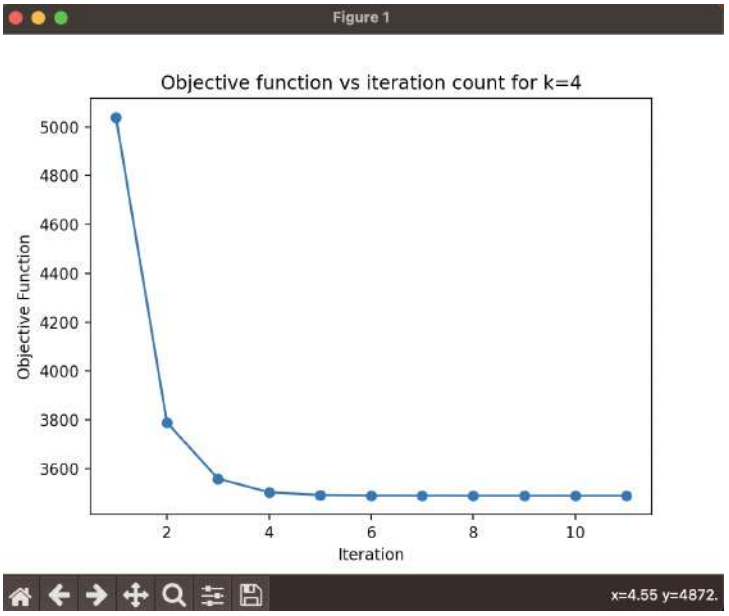


Figure 1
Iteration 3 for k=4
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Centroids
10
8
6
4
2
0
−2
−4
−2
0
2
4
x=−3.81 y=11.12





As you can see, objective function decreases sharply in the first 3 iterations and cluster centers settle down after third iteration although they still change in a very small amount. We can verify that also by looking at cluster graphs. However, the process may be a bit different in each run since we choose the initial cluster centers randomly.

For k = 2:

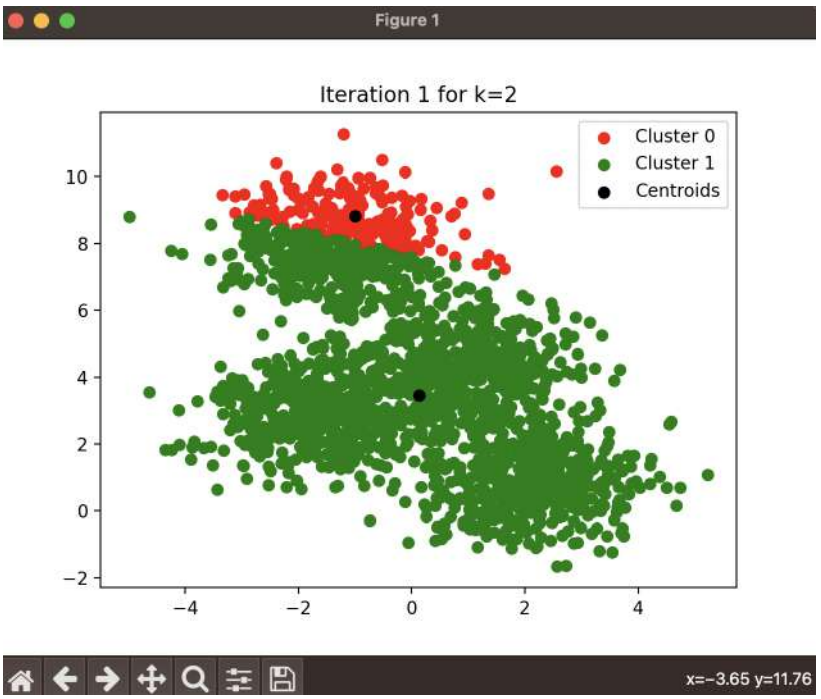


Figure 1
Iteration 1 for k=2
Cluster 0
Cluster 1
Centroids
10
8
6
4
2
0
−2
−4
−2
0
2
4
x=−3.65 y=11.76

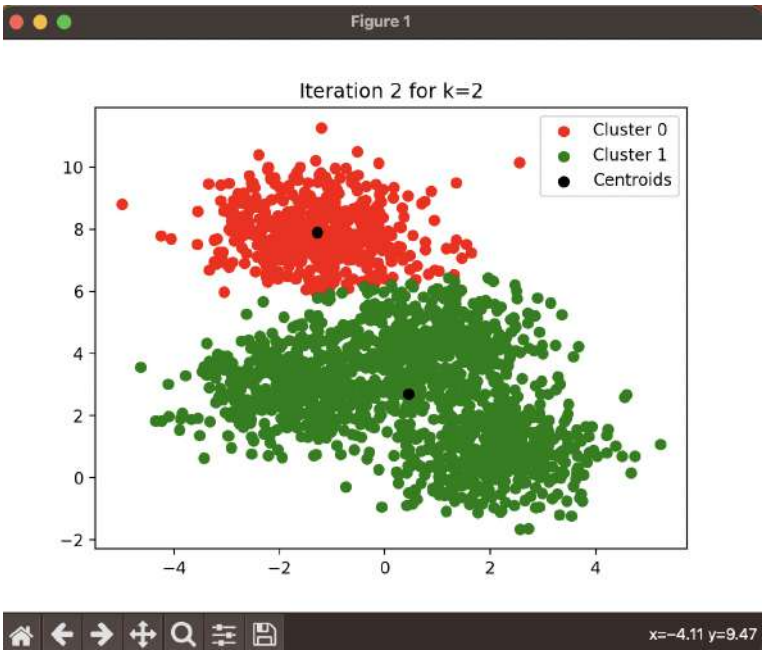


Figure 1
Iteration 2 for k=2
Cluster 0
Cluster 1
Centroids
10
8
6
4
2
0
−2
−4
−2
0
2
4
x=−4.11 y=9.47

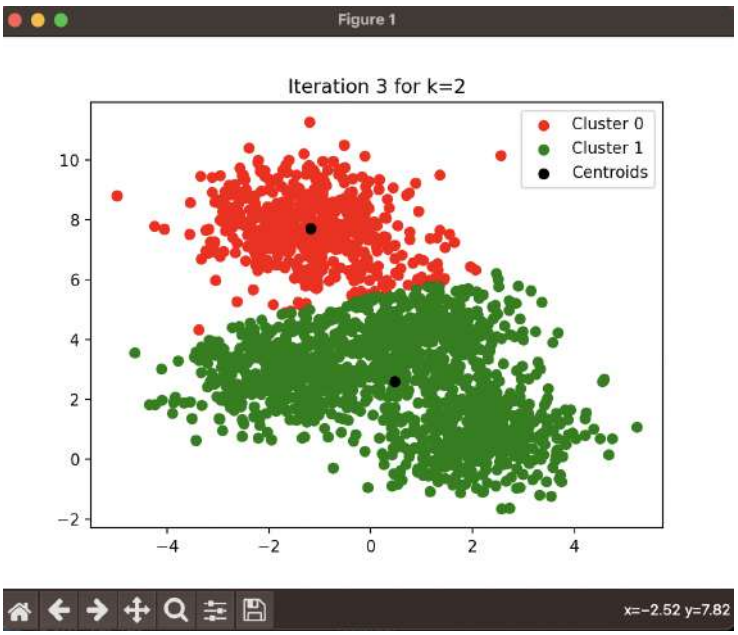


Figure 1
Iteration 3 for k=2
Cluster 0
Cluster 1
Centroids
10
8
6
4
2
0
−2
−4
−2
0
2
4
x=−2.52 y=7.82

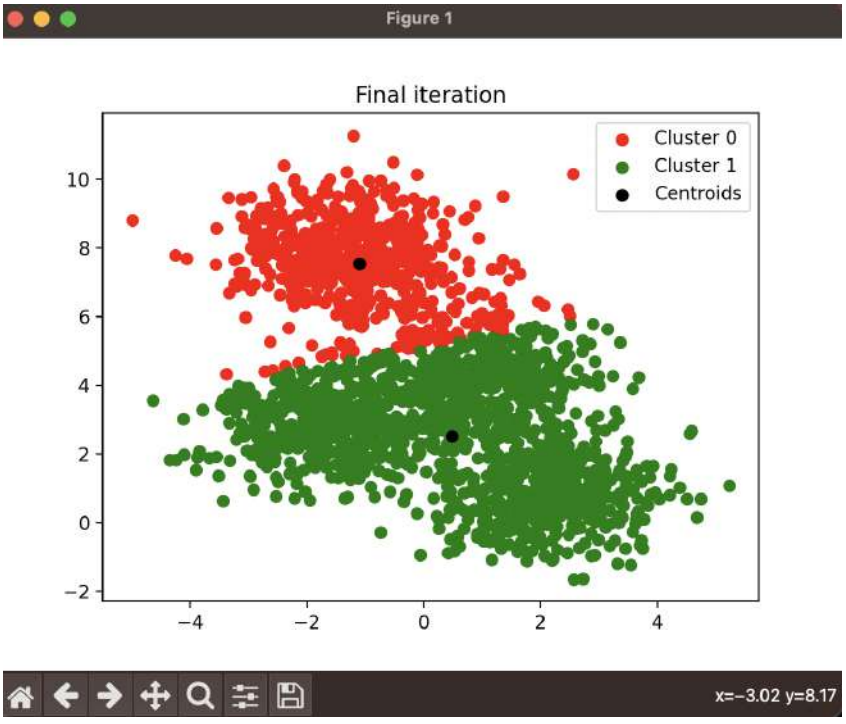
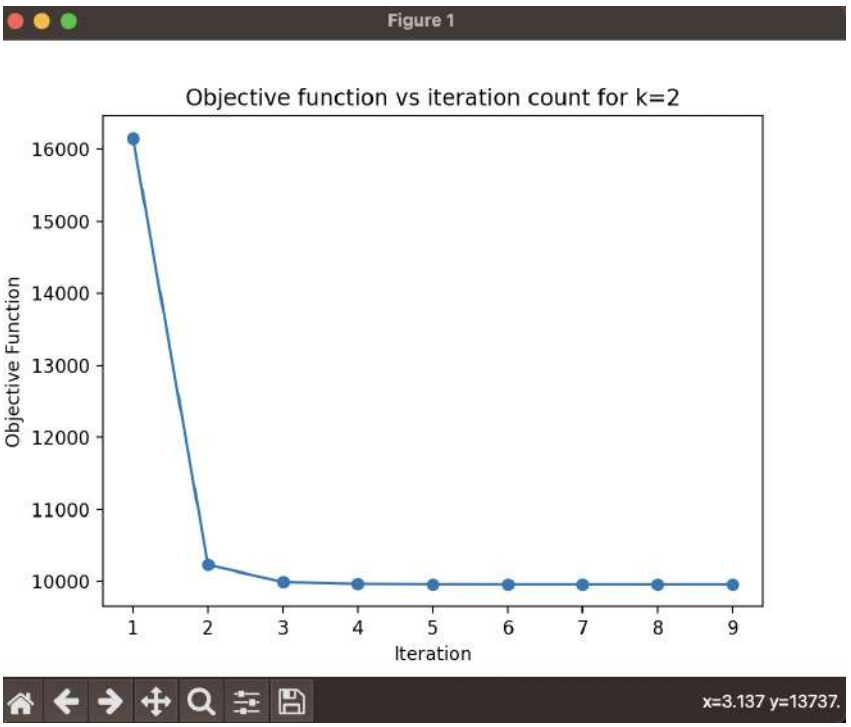


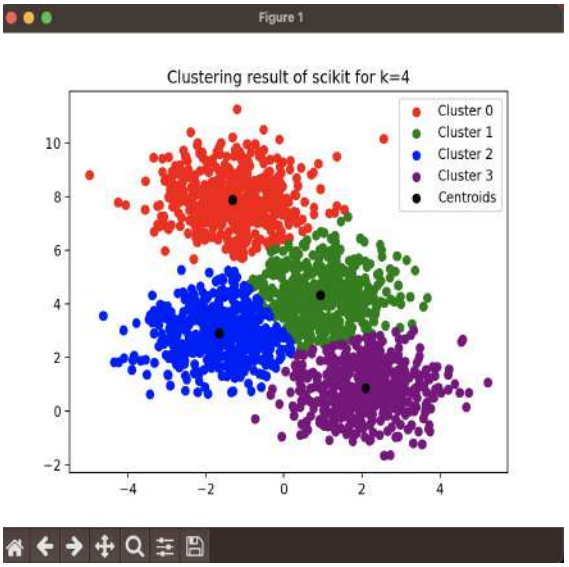
Figure 1
Final iteration
Cluster 0
Cluster 1
Centroids
10
8
6
4
2
0
−2
−4
−2
0
2
4
x=−3.02 y=8.17

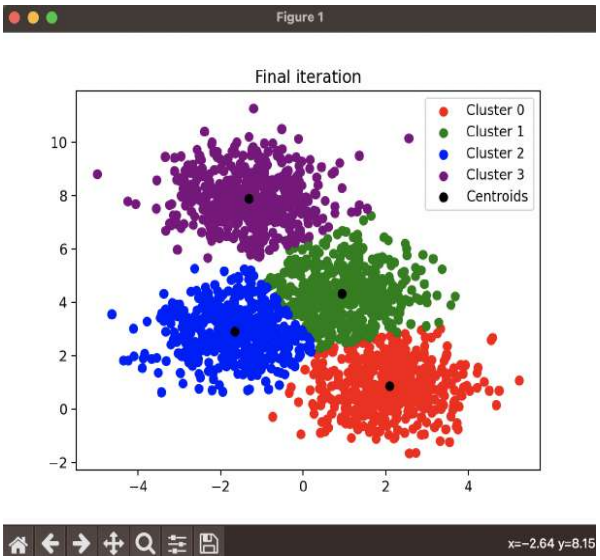


This time, for k = 2, objective function decreases sharply and settles down quite fast. We can check it also from the graph and data points and their cluster centers after the second iteration and the final iteration are quite similar. However, it may take also a lot of iteration(i.e more than 5) since the initial cluster centers are arbitrary.

3) Scikit K-Means Algorithm Results

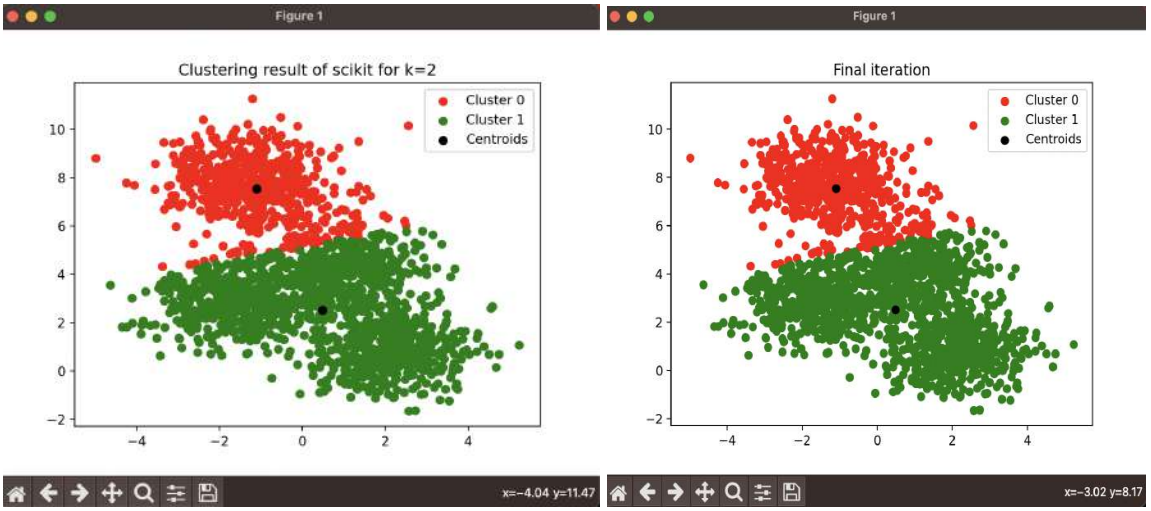
For k = 4





Scikit K-Means **Custom K-Means**

For k = 2



Scikit K-Means **Custom K-Means**

As you can see, scikit also gives similar result.(Although colors are different 🙂)
This is a good sign since it suggests that my custom k-means algorithm works well.

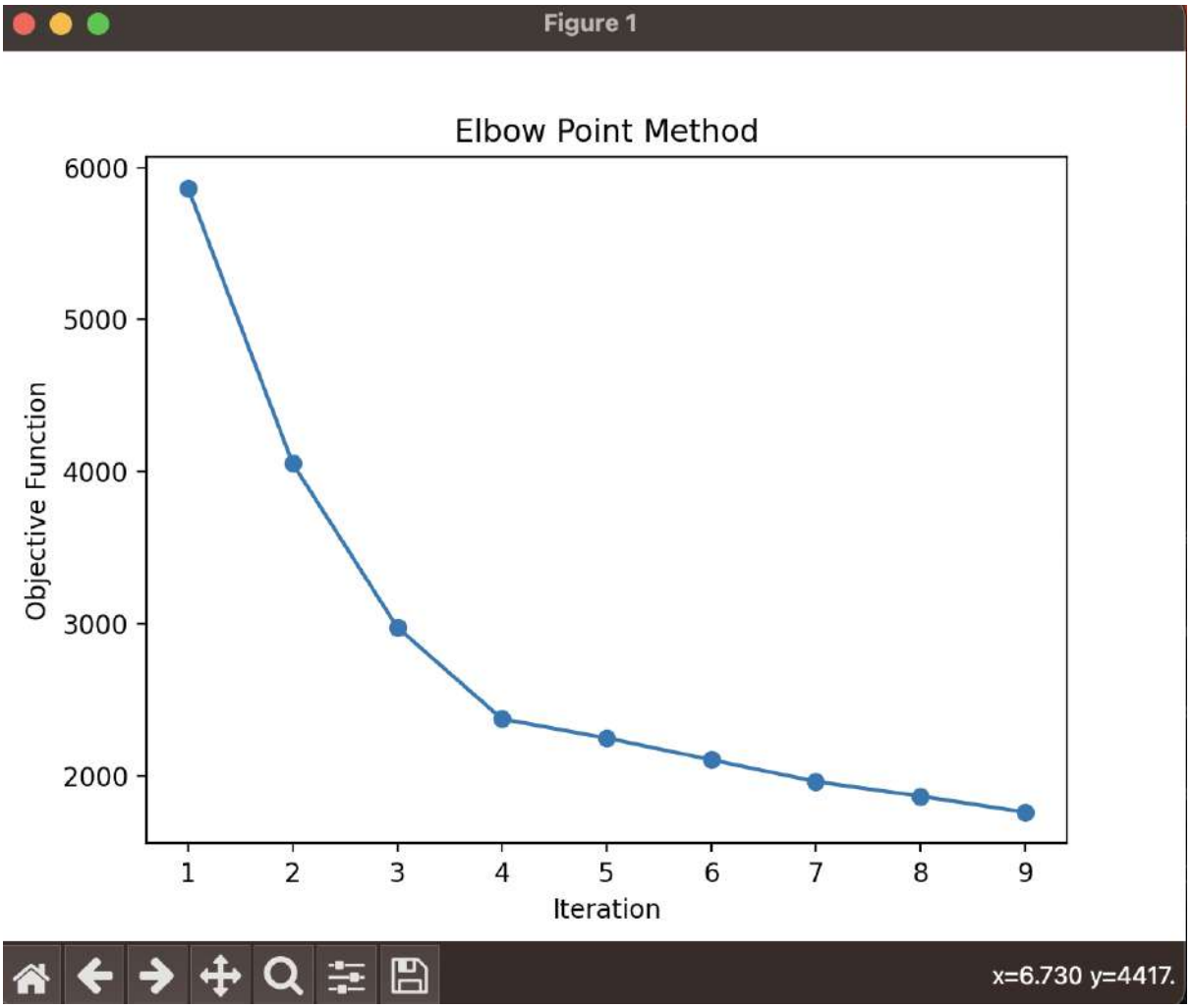
4) Finding Optimal k for K-Means Algorithm

I have made a research about how to find k in the best way and I have chosen “elbow method” to implement due to its simplicity and efficiency. By plotting the within-cluster sum of squares (WCSS) against different k values, it provides a clear "elbow" point where the rate of change in WCSS begins to slow down, indicating the optimal number of clusters. This method offers a quick and effective way to make informed decisions about the appropriate number of clusters without the need for complex calculations, making it widely applicable in various domains. One downside of this algorithm could be the situations where elbow point is not clear and there is not a sharp change in the graph’s shape and it decreases steadily.

I have utilized the following video and articles in my research of elbow point method.
StatQuest: K-means clustering
https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans
https://medium.com/analytics-vidhya/elbow-method-of-k-means-clustering-algorithm-a0c916adc540

The result of elbow point algorithm in my data set is as follows:

.



It can be clearly seen that after the fourth iteration, there is a sharp change in the plot's shape. We can mathematically calculate it using slopes but it is quite intuitive and easier to see it in a graph.
The best number for k is 4 in this dataset.

5) Ugly Dataset (ugly_dataset.txt in misc folder)

Let's apply the whole procedures this time for ugly dataset.

5.1) Custom K-Means Algorithm Results

For k = 3

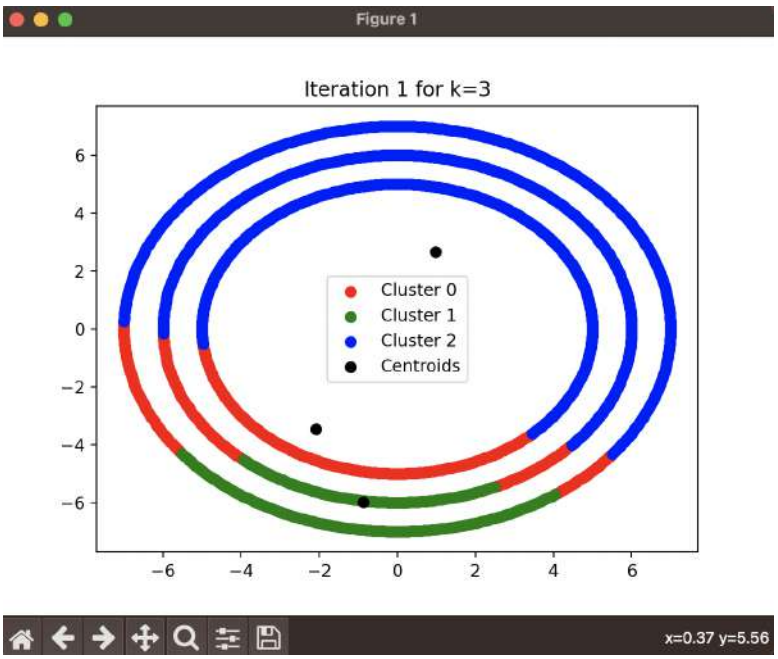


Figure 1
Iteration 1 for k=3
Cluster 0
Cluster 1
Cluster 2
Centroids
x=0.37 y=5.56

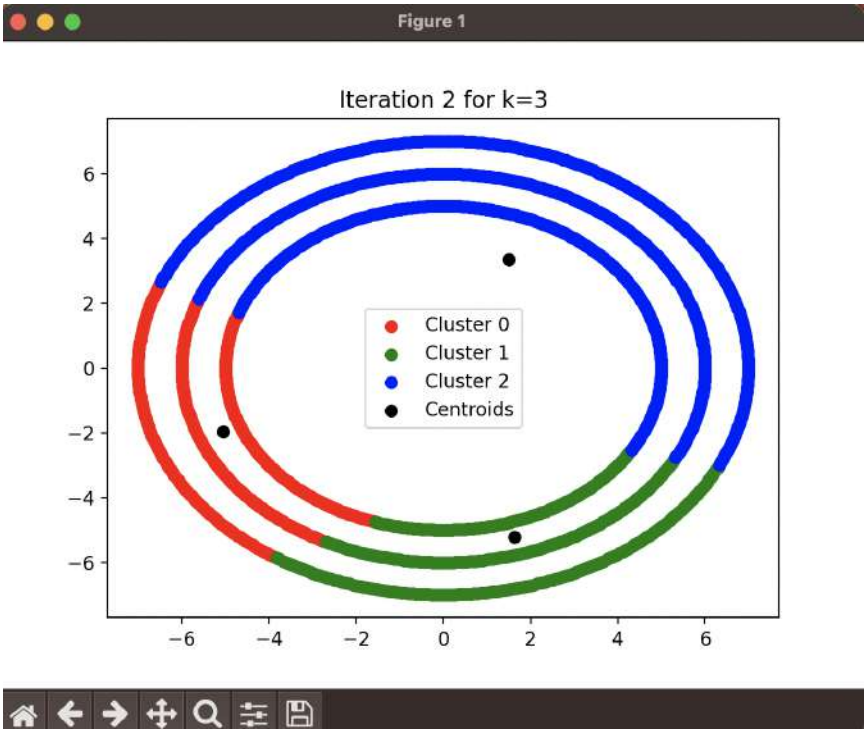


Figure 1
Iteration 2 for k=3
Cluster 0
Cluster 1
Cluster 2
Centroids

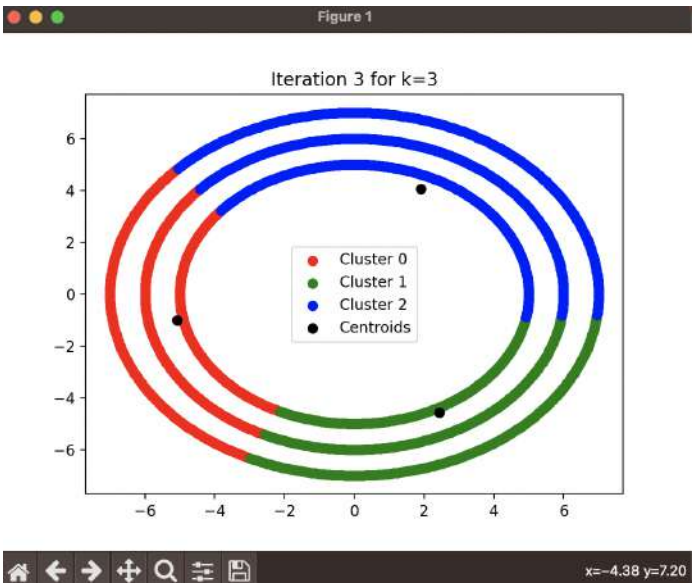


Figure 1
Iteration 3 for k=3
Cluster 0
Cluster 1
Cluster 2
Centroids
x=−4.38 y=7.20

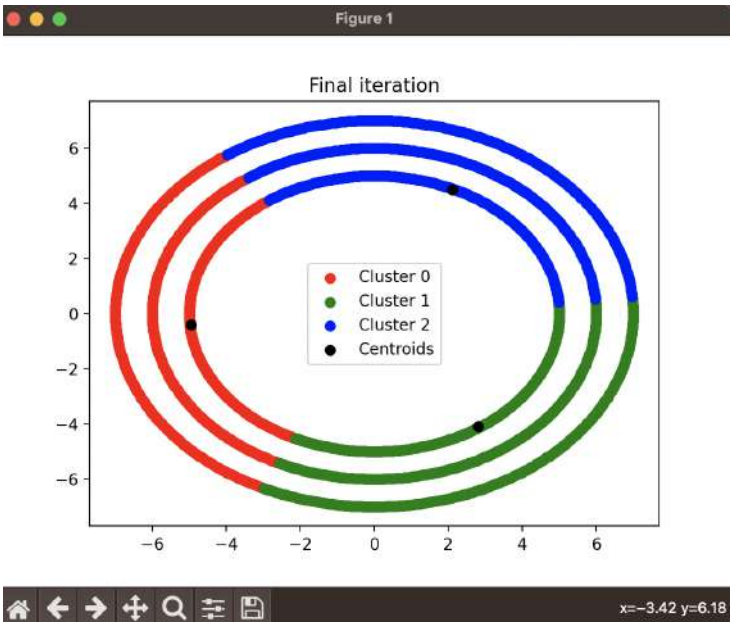


Figure 1
Final iteration
Cluster 0
Cluster 1
Cluster 2
Centroids
x=−3.42 y=6.18

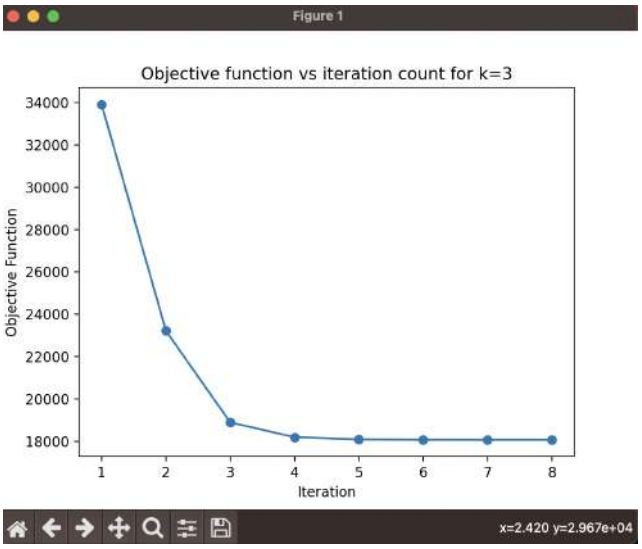
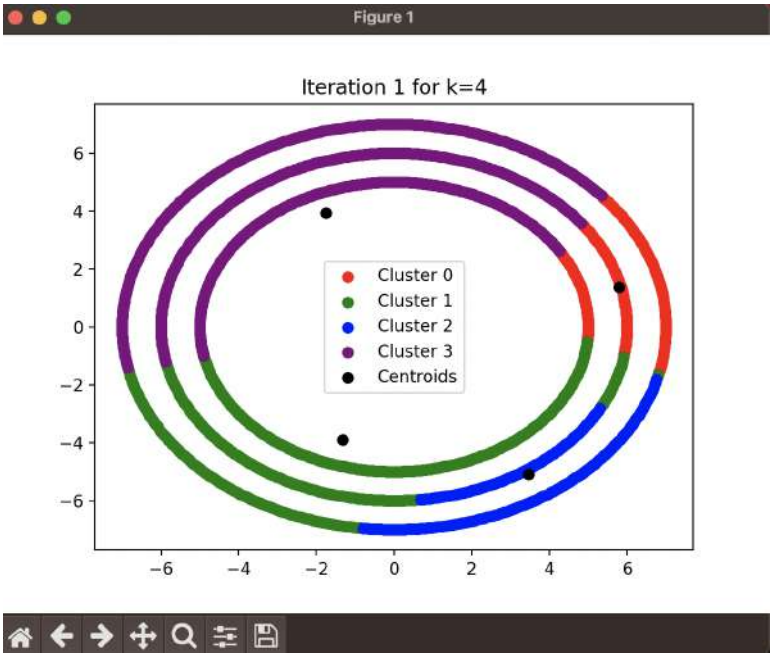
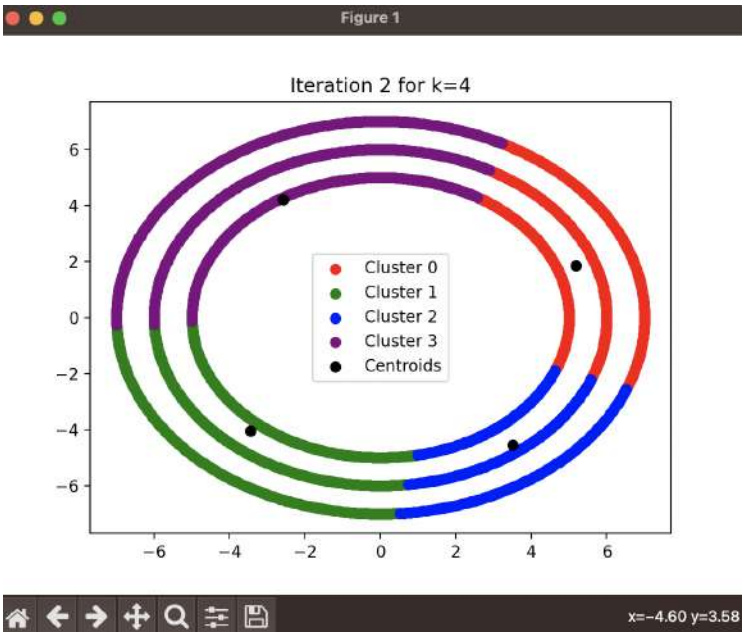


Figure 1
Objective function vs iteration count for k=3
Objective Function
Iteration
x=2.420 y=2.967e+04

As it can be seen from objective function vs iteration count graph, after the third iteration cluster centers almost settled down and after the eighth iteration our algorithm terminated. However, the result it probably different than what we expected initially. We probably want to have 3 clusters and each cluster should represent one circle in the graph. However, since k-means clusters data points by spatial distance, it couldn't make the desired clustering. This is one of the situations where k-means algorithm fail to cluster correctly.
These arguments work for k = 4 too as shown below.

For k = 4





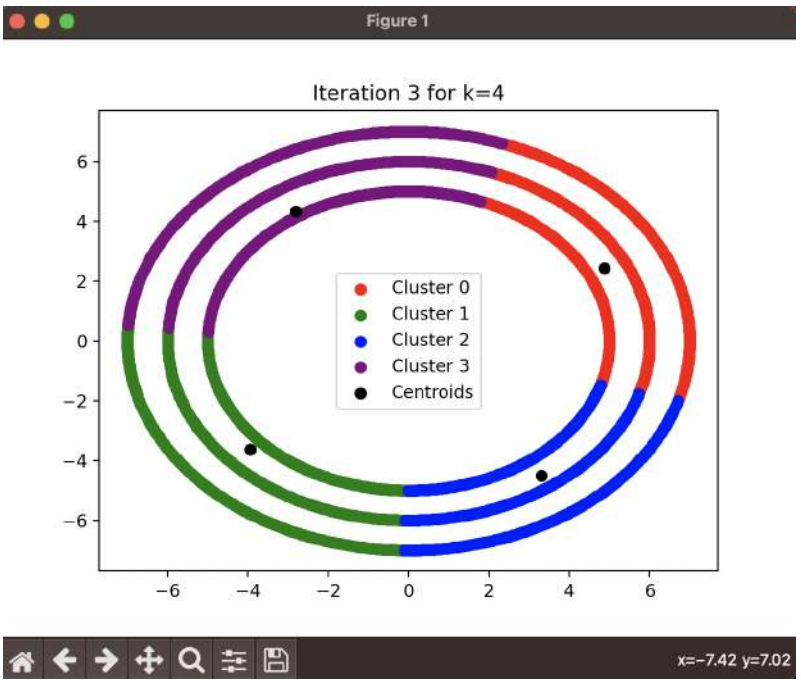


Figure 1
Iteration 3 for k=4
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Centroids
x=−7.42 y=7.02

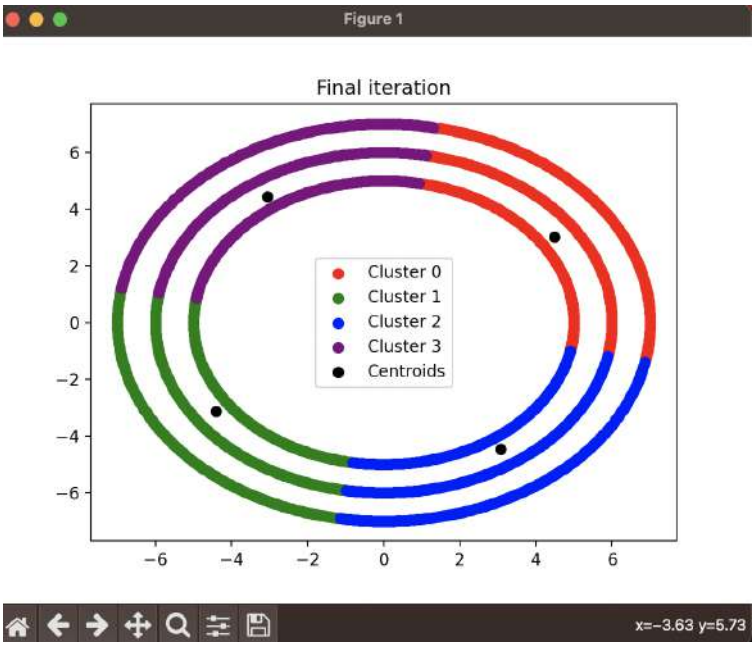
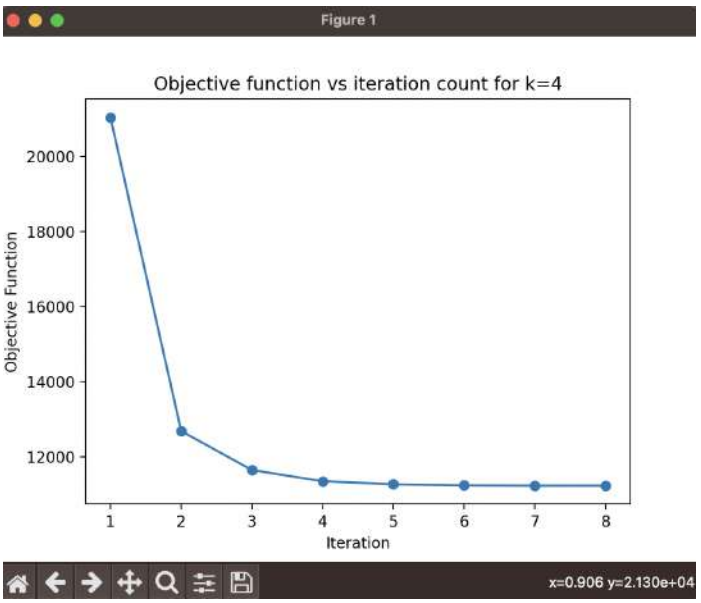
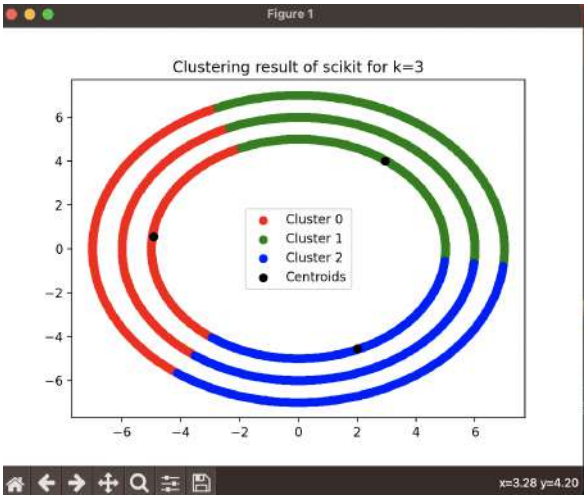


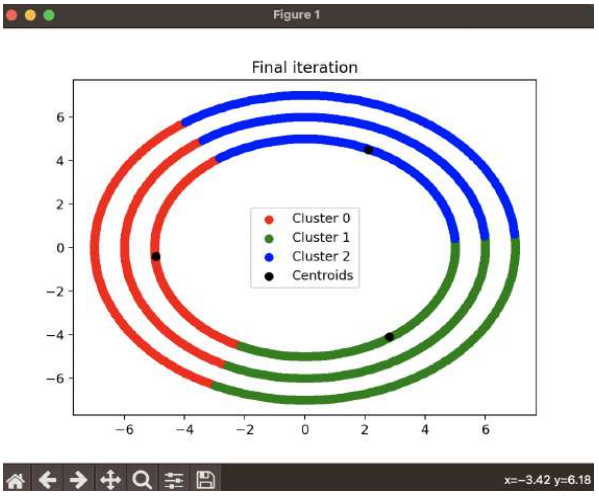
Figure 1
Final iteration
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Centroids
x=−3.63 y=5.73



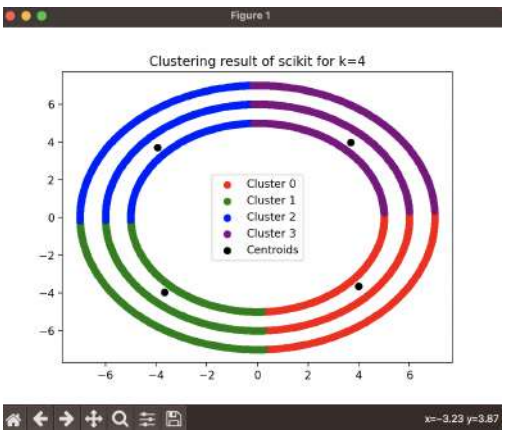
5.2) Scikit K-Means Algorithm Results

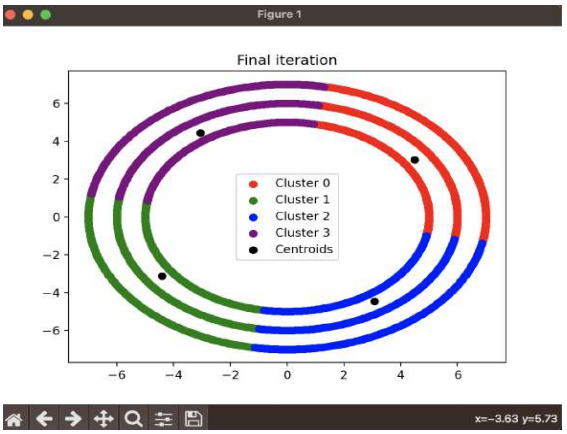
For k = 3





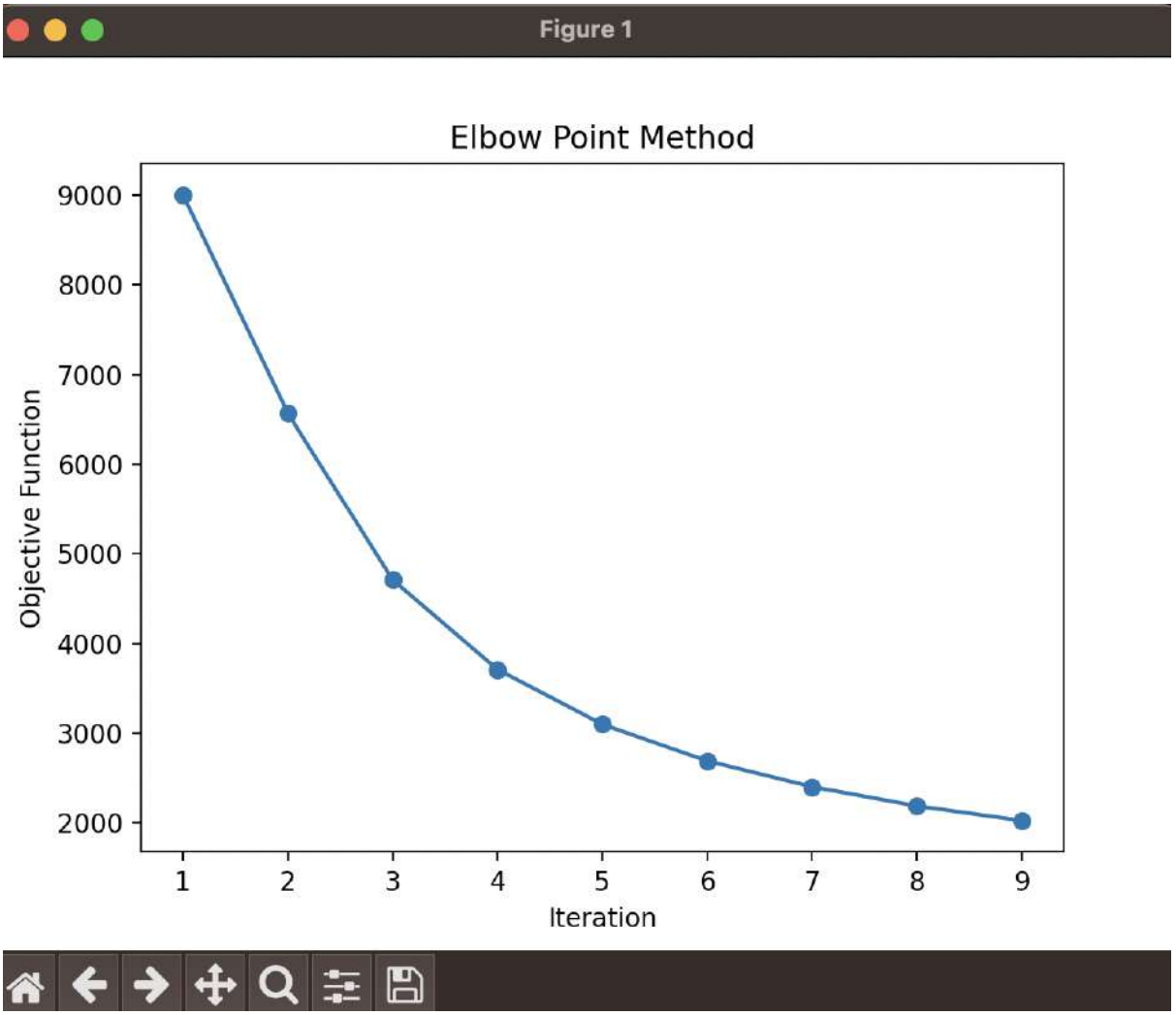
For k = 3





Scikit algorithm yielded the same result which is good. My custom algorithm seems to be successful.

5.3) Finding Optimal k for K-Means Algorithm



As you can see, there is not a sharp decline in the graph. k = 3 seems the be the best alternative but the smooth and steady decline indicates that there is not much difference between different k values if we use k-means algorithm.