**Report for Scientific Data Management Assignment 1**

**Group 1**

[**Benjamin Neckam**](https://www.facebook.com/benjamin.neckam?fref=pb&hc_location=profile_browser)**,** [**Martin Sipiczki**](https://www.facebook.com/martin.sipiczki?fref=pb&hc_location=profile_browser)**,** [**Petko Kaltchev**](https://www.facebook.com/petko.kaltchev?fref=pb&hc_location=profile_browser)**, Yuntao Hu**

1. **Problem statement**

A Gaussian cluster data generator was built, which can take number of clusters, number of points and dimensionality as input. More over, the discreteness of Gaussian cluster can be customized with specific standard derivation of the Gaussian distribution and distance between clusters.

The initial Gaussian cluster and clustering results are visualized in 2D, in which clusters are differed in shapes and colors.

Algorithms of four different variants of K-means are implemented. Using Gaussian cluster with different characteristic, their convergence behavior are compared.

1. **Implementation**

**Class: Point**

A “Point” class was defined for description of the points in the clusters. Each “Point” object contains an array list variable stored the coordinates of this point and an integer variable stored the index of cluster it belong to. The “Point” class also contains several methods to access and change the variable of the object.

**Class: GaussianClusterGenerator**

A “GaussianClusterGenetor” class was defined for generating of the Gaussian clusters. Each “GaussianClusterGenetor” object contains three integer variables k, n and d to store the information of number of clusters, number of points and dimensionality. In addition, four float variables, deviation, minPointDistance, rangeMin, and rangeMax to store the information of the Gaussian clusters. The cluster is stored in an array list of Point objects.

The Gaussian cluster generator first generates Gaussian coordinates based on specific derivation and number range and distance between different points are then checked. The main point for each cluster is created according to number of clusters and dimensionality. At last the points are generated around the main points in each cluster based on the Guassian distribution coordinates calculated before.

**Class: Plot2D**

A “Plot2D” class was defined for visualization of the clusters. JFreeChart package is used the implantation of the class. Different cluster is distinguished with colors and shapes of the points.

**Class: Kmeans**

A “Kmeans” class was defined for clustering the data sets. The initialization method return an array list of centroids in each cluster, then centroids of every cluster are updated.

**Implementation of a) initialization strategy:**

The a) initialization is implemented in function “getCentroidOfCluster”. All points in the dataset are randomly assigned to a cluster. For each cluster, the mean of all the points in this cluster are calculated and stored as initial centroids.

**Implementation of b) initialization strategy:**

Centroids of all clusters are randomly selected from the dataset.

**Distance calculating:**

The distance between two points is calculated with the function “dist”. The distance is calculated as the Euclidean distance, which is the square root of sum of the square of the differences between corresponding values.

**Implementation of a) Lloyd update strategy:**

In each iteration, the distances of each point to all centroids are calculated. The point is reclassified to cluster of the centroid with the shortest distance. After all points are updated, the centroids of all clusters are updated as well.

Then the new centroids list is compared with that of the iteration before. If they don’t meet the minimum difference, iteration continues, otherwise, iteration stops.

**Implementation of b) MacQueen update strategy:**

In first iteration, the distances of each point to all centroids are calculated. The point is reclassified to cluster of the centroid with the shortest distance.

If a point is updated, the centroids of all clusters should be updated.

Then the new centroids list is compared with that of the iteration before. If they don’t meet the minimum difference, iteration continues, otherwise, iteration stops.

1. **Results**

To compare the results, value scale of the points are set from 0 to 50, number of clusters is set to 8. Three different data point numbers (N), 100, 1000 and 10000 used for testing. Derivations are set to 5 and 15 for dataset with different discreteness. All the four variants are implemented for 100 times to get an average of iteration numbers.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Lloyd | | MacQueen | |
| N | Derivation | Initialize (a) | Initialize (b) | Initialize (a) | Initialize (b) |
| 100 | 5 | 4 | 69 | 1249 | 1234 |
| 15 | 6 | 81 | 1498 | 1272 |
| 1000 | 5 | 10 | 246 | 12847 | 15071 |
| 15 | 21 | 373 | 14079 | 14220 |
| 10000 | 5 | 102 | 534 | 136491 | 142365 |
| 15 | 238 | 619 | 143389 | 147238 |

Table 1. Iteration number of different methods for different datasets

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Lloyd | | MacQueen | |
| N | Derivation | Initialize (a) | Initialize (b) | Initialize (a) | Initialize (b) |
| 100 | 5 | 2 | 23 | 46 | 82 |
| 15 | 3 | 32 | 85 | 43 |
| 1000 | 5 | 10 | 167 | 741 | 827 |
| 15 | 15 | 367 | 740 | 744 |
| 10000 | 5 | 96 | 1684 | 81162 | 88726 |
| 15 | 229 | 1677 | 88166 | 95364 |

Table 2. Implementation time (ms) of different methods for different datasets

Visualization of cluster with N=1000, derivation=5

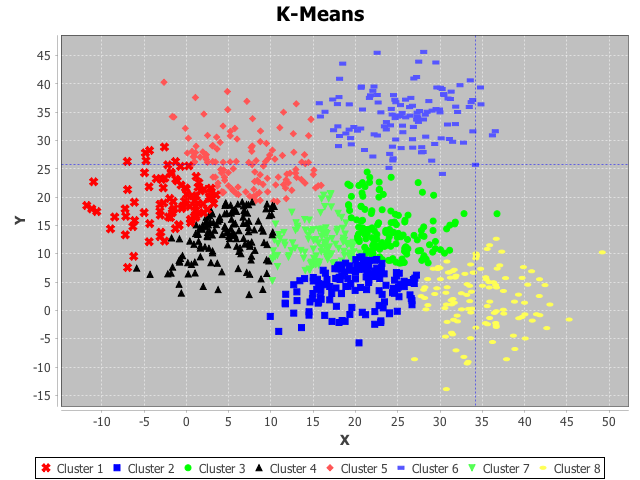
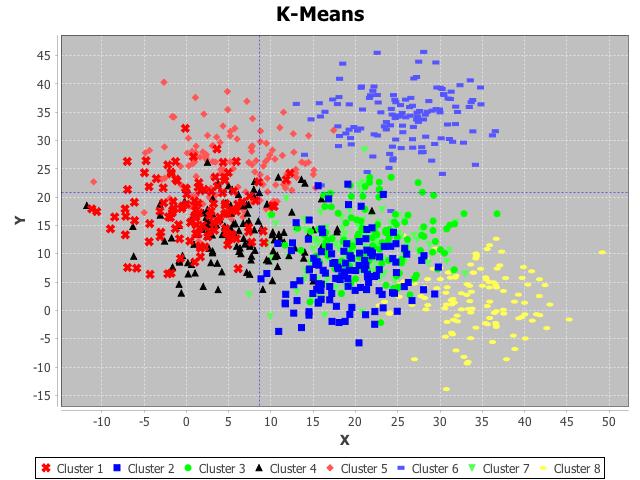


Figure 1: Lloyd with initialization method a (before and after clustering)

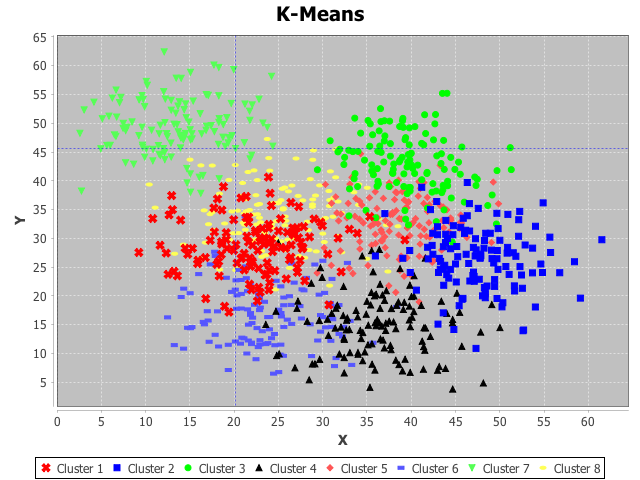
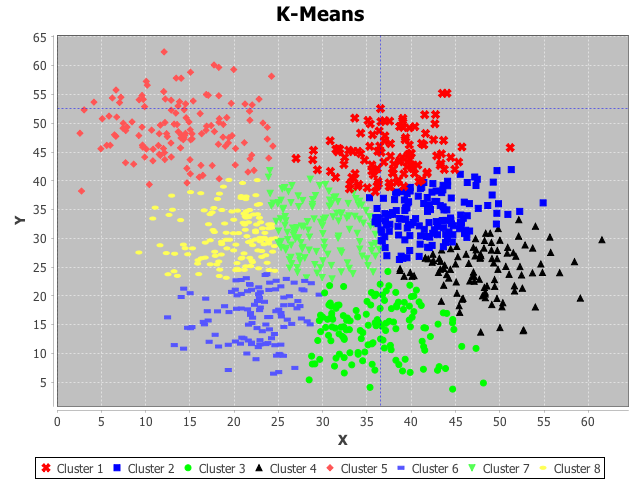
 

Figure 2: Lloyd with initialization method b (before and after clustering)

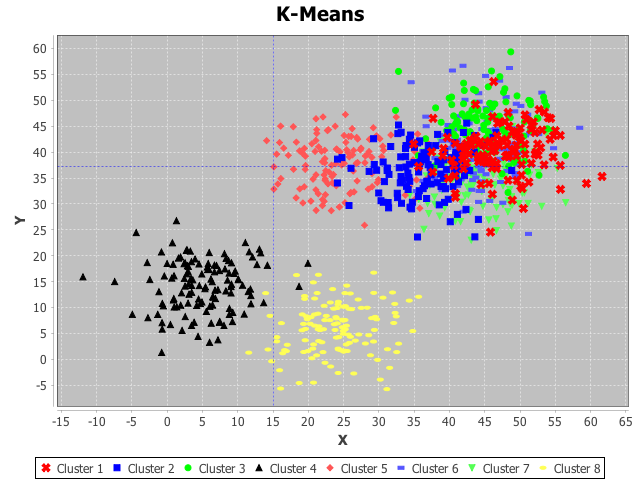
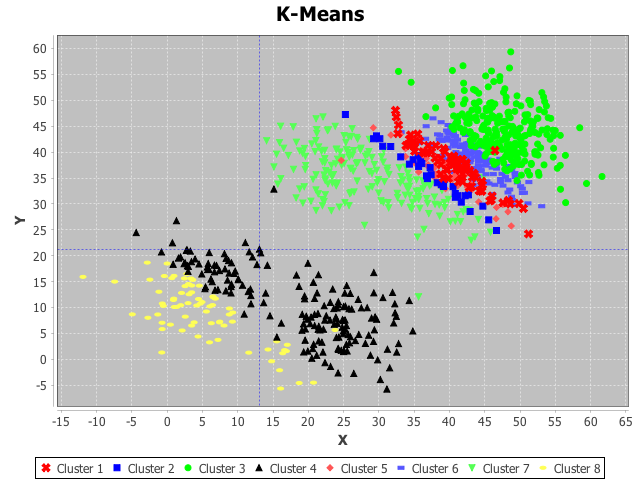
 

Figure 3: MacQueen with initialization method a (before and after clustering)

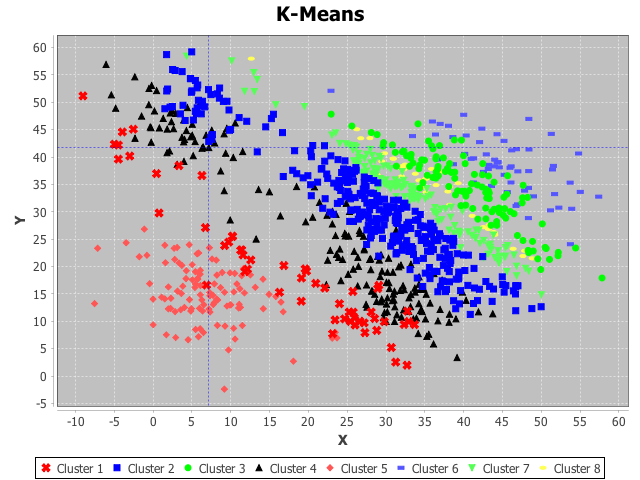
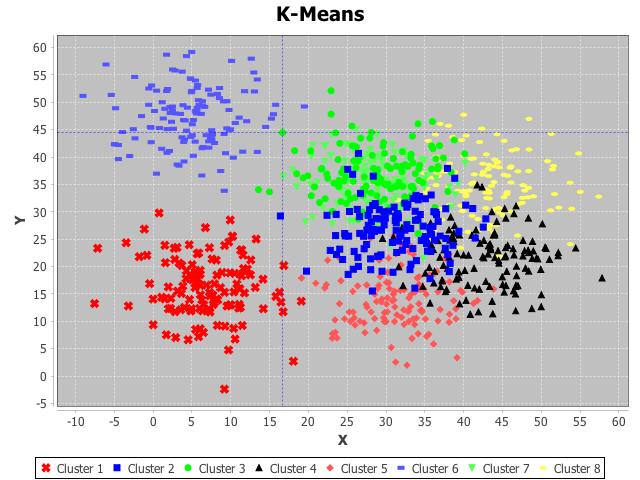


Figure 4: MacQueen with initialization method b (before and after clustering)

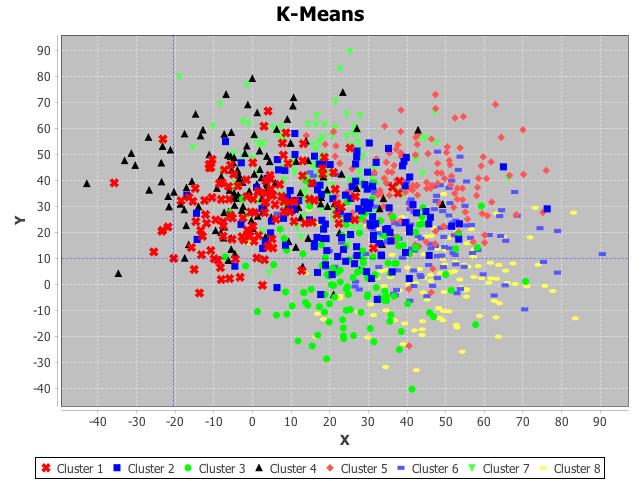
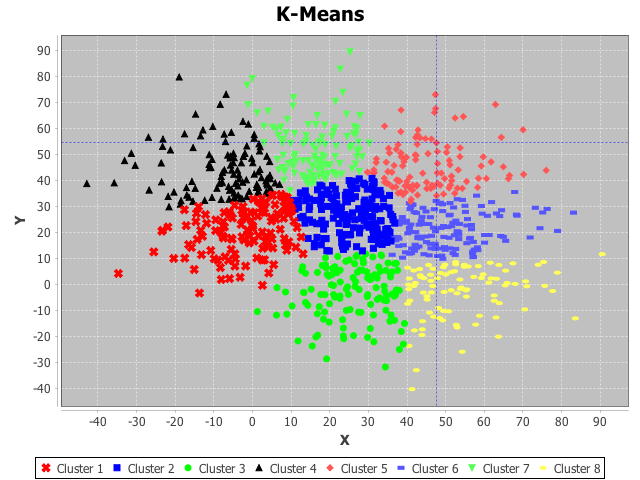
Visualization of cluster with N=1000, derivation=15  

Figure 5: Lloyd with initialization method a (before and after clustering)

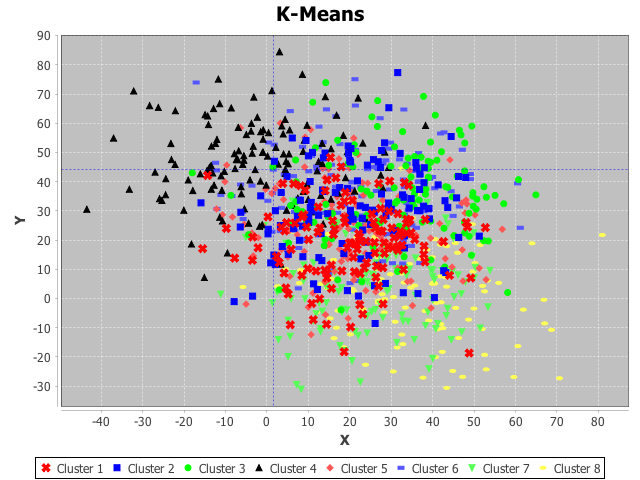
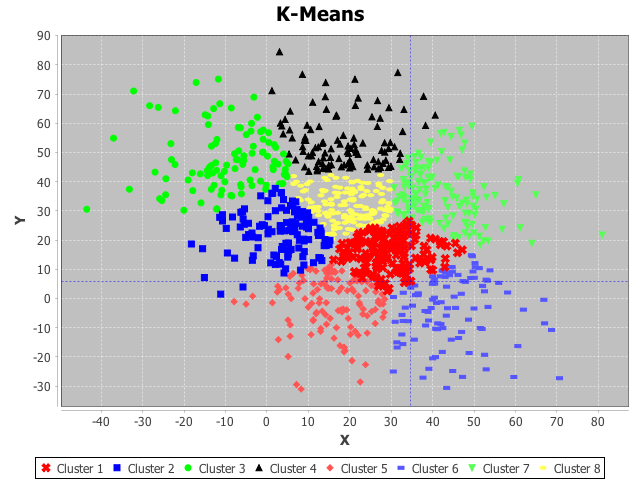
 

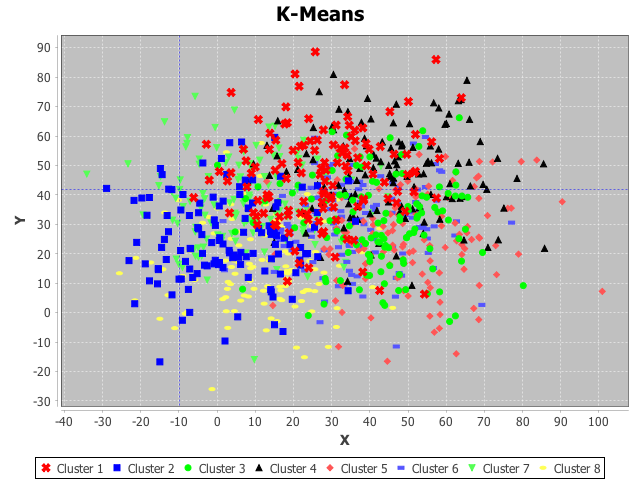
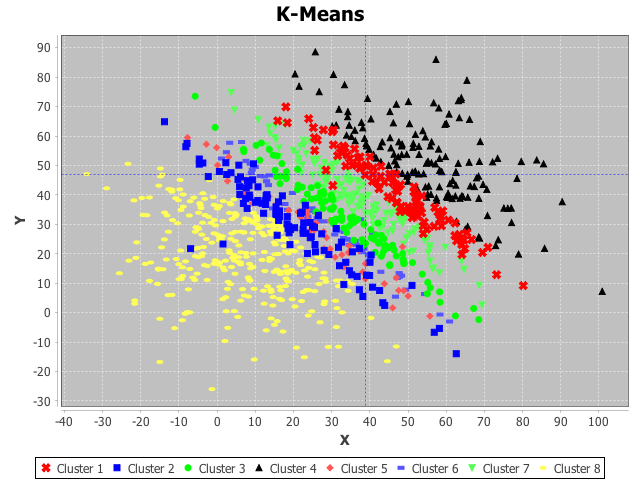
Figure 6: Lloyd with initialization method b (before and after clustering) 

Figure 7: MacQueen with initialization method a (before and after clustering)

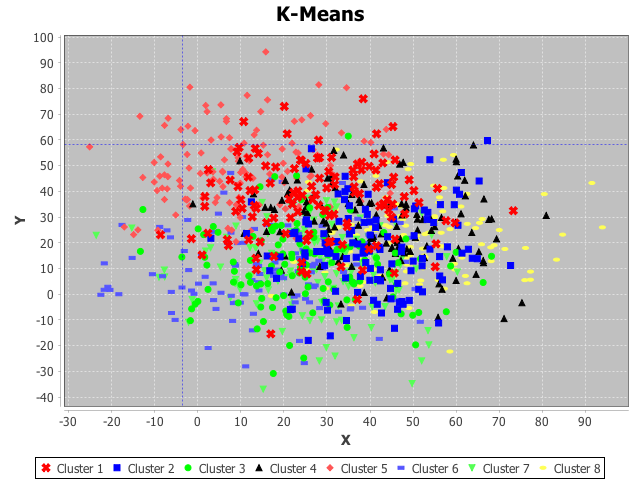
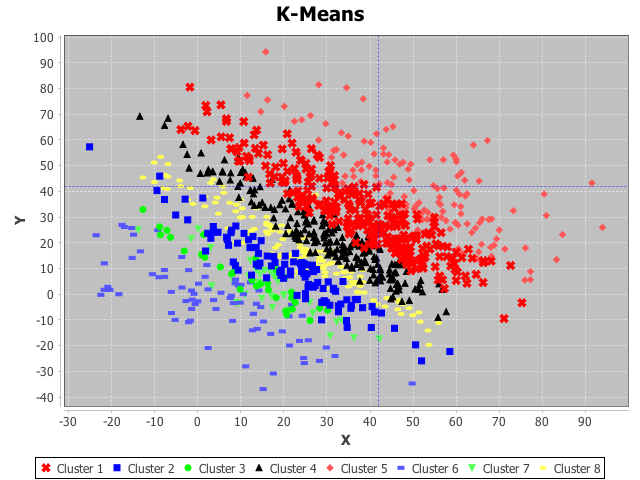
 

Figure 8: MacQueen with initialization method b (before and after clustering)

1. **Discussion**

As the calculation of iteration number MacQueen and Lloyd is different, we use the implementation time of the algorithm to indicate the convergence rate of different algorithms.

It is obvious that with the increase of N the iteration number and implementation time increase dramatically. While in all case, the discreteness of the original dataset shows significant on the implementation of all algorithms. More specifically, large derivation (less discreteness) of the data needs more iteration to cluster the data.

The initialization methods affect the updated methods in different ways. In both case, the initialization method (a) is better than (b). There are not much difference when update the cluster with MacQueen, but a large increase in iteration number was observed when the cluster is updated with Lloyd.

1. **Remark**