The *K*-Means Project

Mohammed Kharma[0000−0001−8280−3285]

Department of Computer Science, Birzeit University, Birzeit, Palestine
mkharmah@birzeit.edu

1 Introduction

Clustering is an unsupervised machine learning approach that groups together comparable data elements based on their feature values [2]. Several clustering methods are available, each with its own set of advantages and disadvantages based on the nature of the data and the use case. The fundamental structure of the data distribution is exploited by clustering algorithms, which also provide criteria for categorizing data that have matching features [2]. K-Means, also known as c-Means, is a popular clustering algorithm and the first version was suggested in 1957 by Stuart Lloyd as a method for pulse-code modulation, his method was kept inside Bell Labs until he published it in 1982 [3]. K-Means is an unsupervised learning algorithm that works on grouping similar data points together based on their feature values. The k-mean process works by constructing different partitions/clusters for the targeted dataset based on the clustering rules and with no previous knowledge of the dataset. Each cluster is made up of comparable data points that are very different from the points in the other clusters [1]. Such a dissimilarity metric depends on the underlying data and the algorithm's goal. Clustering is a fundamental problem in machine learning since it is crucial to many data-driven applications.

The rest of this report is organized as follows, section 2 presents a description of each dataset. Section 3 presents algorithm description, pseudo-code, and mathematical formulation. Section 4 presents the experiments and its results. Finally, section 5 provides some takeaways from this work.

2 Dataset

In this report, we report the results of using the following datasets:

1. **Iris dataset:**

```
# classes: 3
# data points: 150
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column   Non-Null Count   Dtype
---  ------   --------------   -----
```

```
 0   0       150 non-null   float64
 1   1       150 non-null   float64
 2   2       150 non-null   float64
 3   3       150 non-null   float64
 4   4       150 non-null   object

 Dataset description:
                0           1           2           3
count  150.000000  150.000000  150.000000  150.000000
mean     5.843333    3.054000    3.758667    1.198667
std      0.828066    0.433594    1.764420    0.763161
min      4.300000    2.000000    1.000000    0.100000
25%      5.100000    2.800000    1.600000    0.300000
50%      5.800000    3.000000    4.350000    1.300000
75%      6.400000    3.300000    5.100000    1.800000
max      7.900000    4.400000    6.900000    2.500000

Dataset sample:
     0    1    2    3            4
0  5.1  3.5  1.4  0.2  Iris-setosa
1  4.9  3.0  1.4  0.2  Iris-setosa
2  4.7  3.2  1.3  0.2  Iris-setosa
3  4.6  3.1  1.5  0.2  Iris-setosa
4  5.0  3.6  1.4  0.2  Iris-setosa
.
```

2. **Three separable gaussians:**

```
# classes: 3
# data points: 300
Dataset information:
RangeIndex: 300 entries, 0 to 299
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   0       300 non-null   float64
 1   1       300 non-null   float64
 2   2       300 non-null   float64

 Dataset description:
                0           1           2
count  300.000000  300.000000  300.000000
mean     0.433244    2.688586    1.000000
std      1.618409    1.566672    0.817861
min     -2.948656   -0.765892    0.000000
25%     -1.091289    1.162992    0.000000
```

```
50%        0.826162    2.909197    1.000000
75%        1.697028    4.030363    2.000000
max        3.437618    5.474253    2.000000
Dataset sample:
          0          1    2
0  0.428577   4.973997  0.0
1  1.619909   0.067645  1.0
2  1.432893   4.376792  0.0
3 -1.578462   3.034458  2.0
4 -1.658629   2.267460  2.0
.
```

3. **Slightly overlapping three gaussians:**

```
# classes: 3
# data points: 300
Dataset information:
RangeIndex: 300 entries, 0 to 299
Data columns (total 3 columns):
 #   Column   Non-Null Count   Dtype
---  ------   --------------   -----
 0   0        300 non-null     float64
 1   1        300 non-null     float64
 2   2        300 non-null     float64

Dataset description:
                0           1           2
count  300.000000  300.000000  300.000000
mean     0.399093    2.679656    1.000000
std      1.752898    1.721521    0.817861
min     -3.659532   -1.597670    0.000000
25%     -1.006490    1.227617    0.000000
50%      0.629149    2.856643    1.000000
75%      1.759564    4.086663    2.000000
max      4.128793    6.059485    2.000000
Dataset sample:
          0          1    2
0  0.154730   5.309102  0.0
1  1.402230  -0.347364  1.0
2  1.661204   4.413295  0.0
3 -1.604242   3.092746  2.0
4 -1.724491   1.942249  2.0
.
```

4. **Moons**

```
# classes: 2
# data points: 1000
 Dataset information:
 RangeIndex: 1000 entries, 0 to 999
 Data columns (total 3 columns):
  #   Column  Non-Null Count  Dtype
 ---  ------  --------------  -----
  0   0       1000 non-null   float64
  1   1       1000 non-null   float64
  2   2       1000 non-null   float64

 Dataset description:
                  0            1           2
 count  1000.000000  1000.000000  1000.00000
 mean      0.499690     0.248688     0.50000
 std       0.871547     0.496743     0.50025
 min      -1.120606    -0.604452     0.00000
 25%      -0.043376    -0.202149     0.00000
 50%       0.507788     0.242695     0.50000
 75%       1.035700     0.710452     1.00000
 max       2.079038     1.101272     1.00000

 Dataset sample:
           0         1    2
 0 -1.036507  0.392617  0.0
 1  1.014714  0.177547  0.0
 2 -0.661602  0.705367  0.0
 3 -0.286087  0.967387  0.0
 4 -0.790062  0.615586  0.0
 .
```

5. **Circles**

```
# classes: 2
# data points: 1000
 Dataset information:
 RangeIndex: 1000 entries, 0 to 999
 Data columns (total 3 columns):
  #   Column  Non-Null Count  Dtype
 ---  ------  --------------  -----
  0   0       1000 non-null   float64
  1   1       1000 non-null   float64
  2   2       1000 non-null   float64

 Dataset description:
                  0            1           2
```

```
count    1000.000000    1000.000000    1000.00000
mean       -0.000930      -0.001957       0.50000
std         0.560831       0.561995       0.50025
min        -1.140872      -1.111948       0.00000
25%        -0.439114      -0.444116       0.00000
50%        -0.008458       0.000451       0.50000
75%         0.434248       0.436093       1.00000
max         1.105457       1.110823       1.00000

Dataset sample:
          0          1    2
0  1.047437  -0.245648  0.0
1  0.420065   0.168314  1.0
2  0.223487  -0.337189  1.0
3 -0.254356   0.497842  1.0
4  0.055423  -1.014569  0.0
```

3 Design and Implementation

3.1 Algorithm illustration

The following illustrates the basic flow of the K-means algorithm:

1. Initialize the center of each cluster K by selecting point P (one instance from the dataset) from the dataset randomly as a center for each cluster C
2. Loop over the whole dataset instances (points) and assign each data point to the closest cluster center based on calculating the Euclidean distance between each data point and each cluster center. Then assign each instance to the closest cluster center.
3. Update the cluster centers K by calculating the mean of all instances assigned in each cluster and update the cluster center of each cluster to be the mean of its data points.
4. Repeat steps 2 and 3 until the cluster centers K has reached the maximum number of iterations or algorithm convergence.
5. Finally, return the cluster centers and their assigned data points

In the following, the algorithm pseudo-code:

```
# Initialize the cluster centers randomly
clusterCenters <- select c instances randomly from the dataset
maxIteration <- some number like 100 iteration
errorLimit <- some small value to control the algorithm convergence
currentLoopIteration <- 0

While maxIteration < currentLoopIteration || error rate > errorLimit:
    # Assign each data instance to the closest cluster center.
    for each data point:

        Calculate the similarity measure between the data point and each
        cluster center using for example Euclidean distance function.

        Assign the data instances to the nearest cluster center's cluster.

    # Update the cluster center

    for all points per each cluster:
        Calculate the mean of all data points assigned to the cluster.

        Update the cluster center of the cluster to the calculated mean.

    # Calculate the error rate/algorithm convergence based on the change
    in distance between each cluster point and its current center
    for
    all points per each cluster:
        Calculate the Euclidean distance of all data points
        assigned to the cluster.
    Get the sum of all Euclidean distances for all points (currentErrorRate)
    and divide it by the number of instances in the dataset.

    if currentErrorRate<=errorLimit:
        break
    else
        currentLoopIteration++

Return each cluster center and its assigned data points.
```

3.2 Mathematical equations

To calculate the Euclidian distance between the two instances, assume points, P1(X1, Y1) and P2 (X2, Y2). The Euclidian distance equation is shown in equation 1:

$$D(P1, P2) = \sqrt{(P2-P1)^2} \tag{1}$$

To calculate the new center in each iteration after the assignment step, calculate the mean of each feature using equation 2 where C is the cluster center of a particular cluster J and X is the assigned data points into cluster J.

$$C_j = \frac{1}{C_j}\sum_{1}^{C_j} X_i \tag{2}$$

To calculate the cost function on each iteration, assume we have cluster centers K and datapoints X. So we need to calculate the distance using equation 1 between each data point in X between i and n with the related cluster center j. See equation 3

$$1/n * \sum_{1<j<k} \sum_{X_i \in C_j} D(x_i, c_j) \tag{3}$$

4 Experiments and results

4.1 Experiments using Iris dataset

The experiment setup and execution results are as follows:

- Dataset: Iris dataset.
- Maximum number of iterations: 200 (0-199) iterations.
- Initialize features scaling in order to normalize the data input to values between zero and one.
- Applying dimensionality reduction to 2 classes before running k-means in order to help in reducing the computational cost and improve the performance of the algorithm, as well as to identify patterns or relationships in the data that may not be easily visible in higher dimensions.
- Using the number of clusters equal two, the classification results with the demonstration of clusters center shown in figure 1. The algorithm convergence is shown in figure 2.
- Using the number of clusters equal to three, the classification results with the demonstration of clusters center shown in figure 3. The algorithm convergence is shown in figure 4. The X-axis refers to the number of iterations we use to update and enhance the selection of the cluster centers. The Y-axis refers to the calculated cost function as per to 3.
- Considering that we don't know the distribution of the data and we don't have prior knowledge about the actual number of classes in the dataset. Hence, we are dealing with an unlabeled dataset, then we run the clustering over a different number of clusters starting from two clusters to 15 clusters to test the best data points distribution over each cluster center. We stop the iteration when we reach the point where the difference between the error rate in the previous iteration is less than or equal to 0.02. Figure 5 shows the dataset over the best number of found clusters. Figure 6 shows the algorithm iteration convergence over a number of iterations to find the best number of clusters (The baseline of numbers starts from 2, and the X-axis refers to the number of clusters, the Y-axis refers to the cost function).

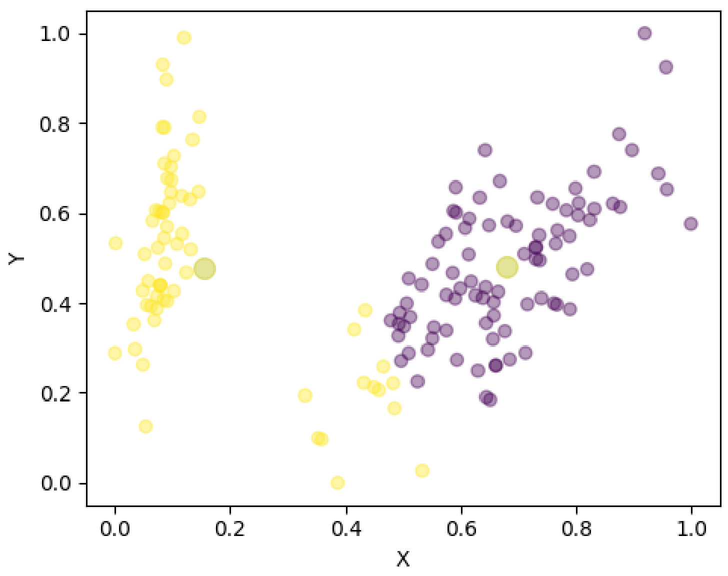

Fig. 1. Iris dataset over 2 clusters .

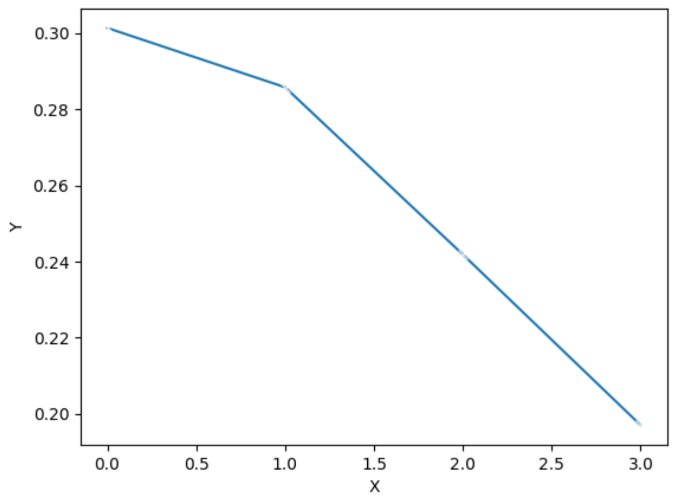

Fig. 2. Iris dataset - Algorithm iteration convergence - 2 clusters.

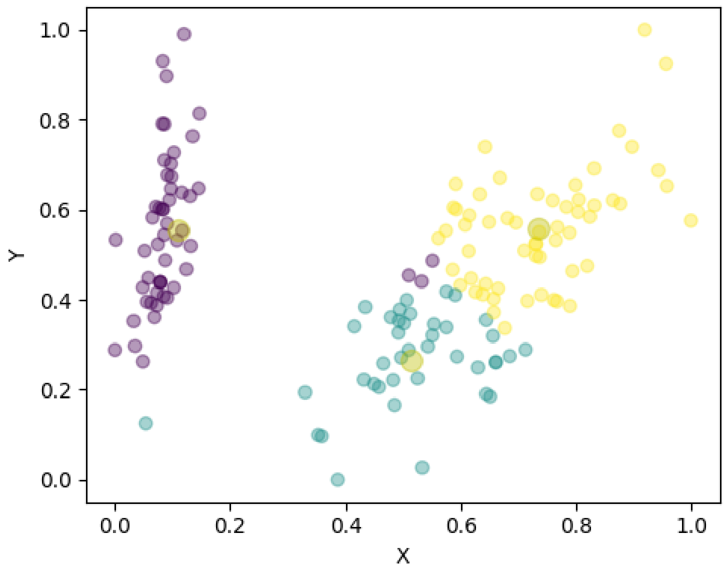

Fig. 3. Iris dataset over 3 clusters .

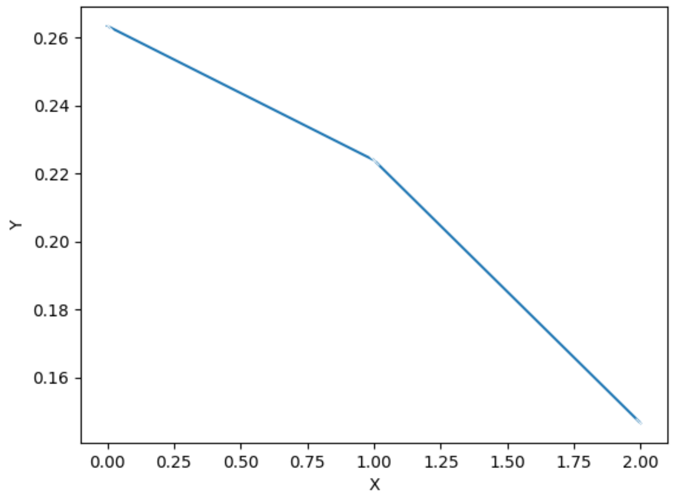

Fig. 4. Iris dataset - Algorithm iteration convergence - 3 clusters.

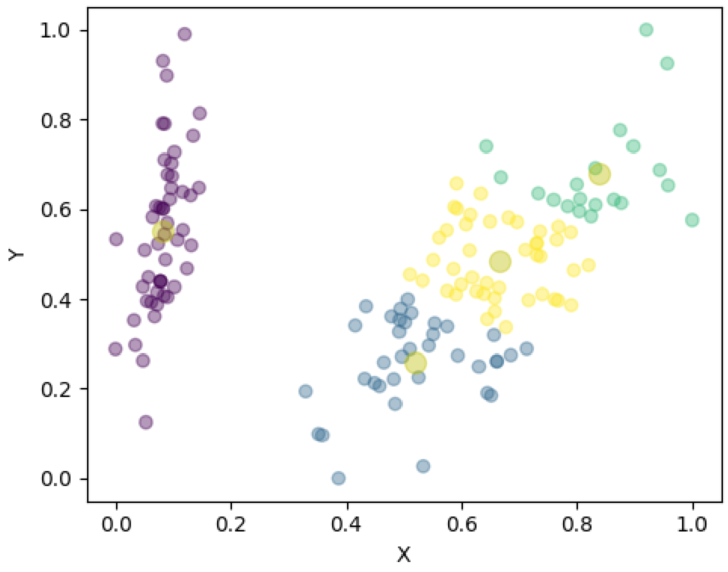

Fig. 5. Iris dataset over the best number of found clusters.

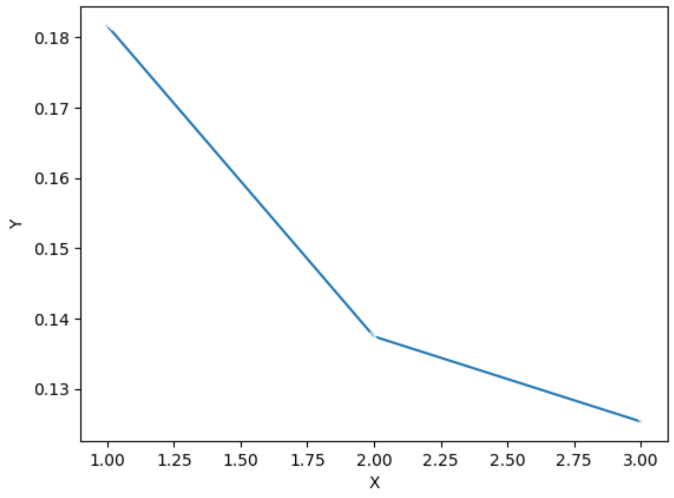

Fig. 6. Iris dataset - Algorithm iteration convergence over a number of iterations to find the best number of clusters

4.2 Experiments using three separable gaussians dataset

The experiment setup and execution results are as follows:

– Dataset: Three separable gaussians dataset.
– Maximum number of iterations: 200 (0-199) iterations.
– Initialize features scaling in order to normalize the data input to values between zero and one.
– Using the number of clusters equal two, the classification results with the demonstration of clusters center shown in figure 7. The algorithm convergence is shown in figure 8.
– Using the number of clusters equal to three, the classification results with the demonstration of clusters center shown in figure 9. The algorithm convergence is shown in figure 10. The X-axis refers to the number of iterations we use to update and enhance the selection of the cluster centers. The Y-axis refers to the calculated cost function as per to 3.
– Considering that we don't know the distribution of the data and we don't have prior knowledge about the actual number of classes in the dataset. Hence, we are dealing with an unlabeled dataset, then we run the clustering over a different number of clusters starting from two clusters to 15 clusters to test the best data points distribution over each cluster center. We stop the iteration when we reach the point where the difference between the error rate in the previous iteration is less than or equal to 0.02. Figure 11 shows the dataset over the best number of found clusters. Figure 12 shows the algorithm iteration convergence over a number of iterations to find the best number of clusters (The baseline of numbers starts from 2, and the X-axis refers to the number of clusters, the Y-axis refers to the cost function)

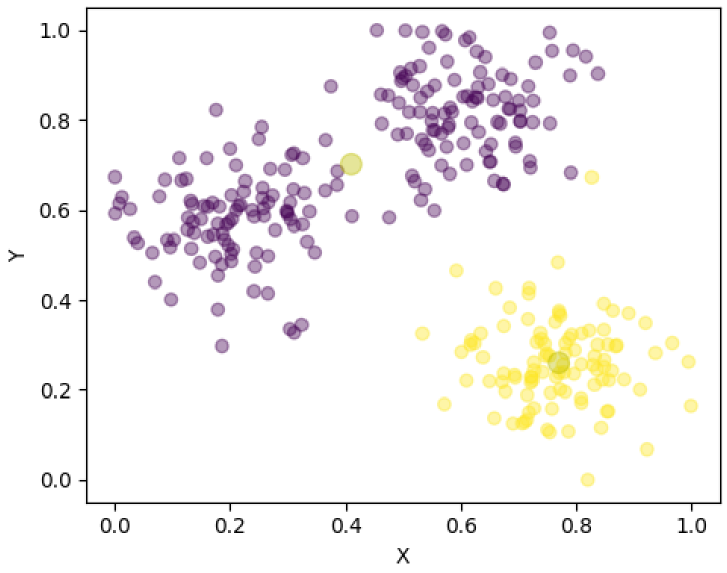

Fig. 7. Three separable gaussians dataset over 2 clusters .

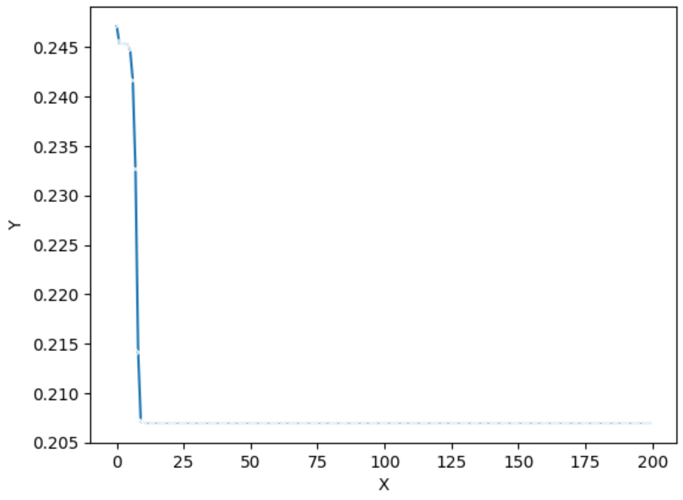

Fig. 8. Three separable gaussians - Algorithm iteration convergence - 2 clusters.

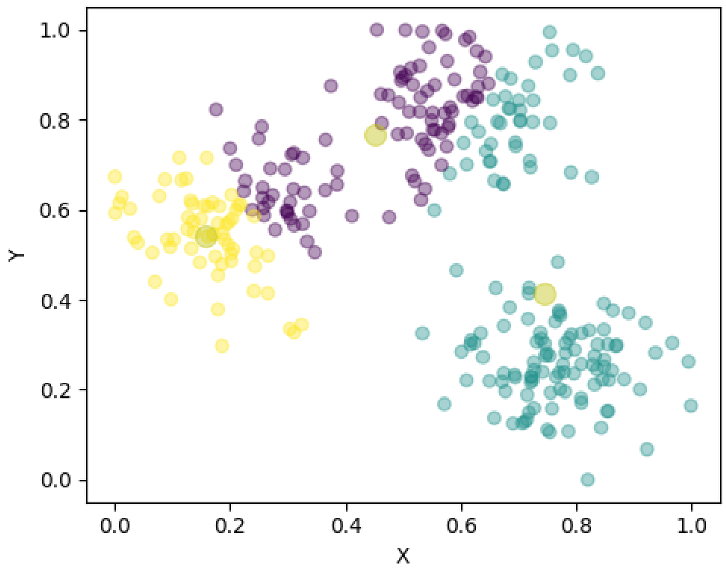

Fig. 9. Three separable gaussians dataset over 3 clusters.

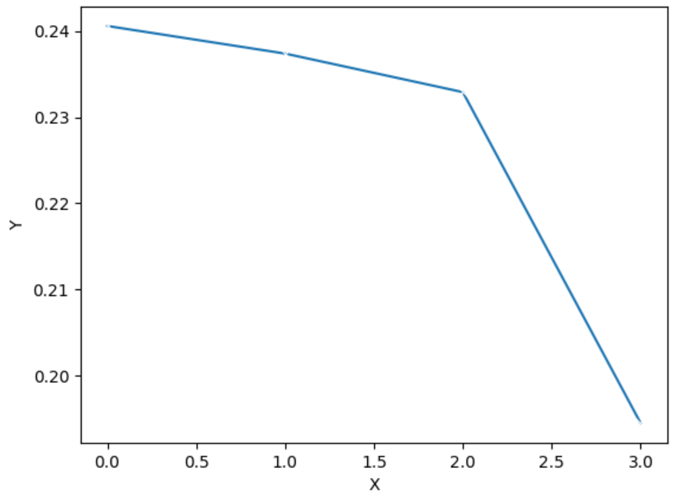

Fig. 10. Three separable gaussians - Algorithm iteration convergence - 3 clusters.

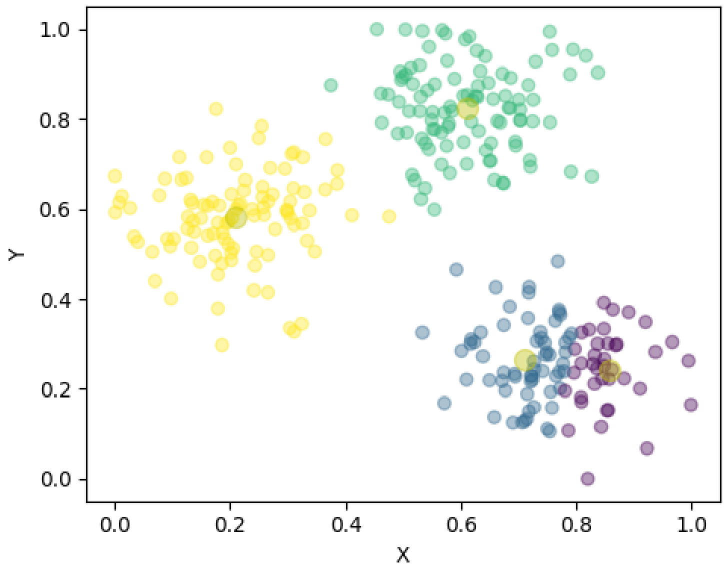

Fig. 11. Three separable gaussians dataset over the best number of found clusters.

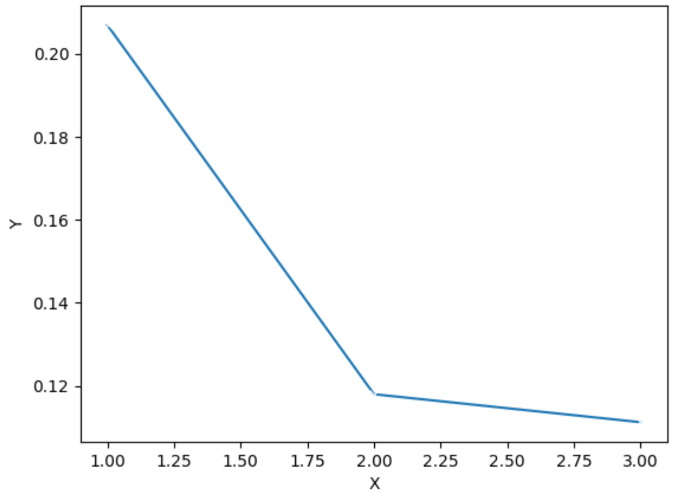

Fig. 12. Three separable gaussians dataset - Algorithm iteration convergence over a number of iterations to find the best number of clusters

4.3 Experiments using slightly overlapping three separable gaussians dataset

The experiment setup and execution results are as follows:

- Dataset: Slightly overlapping three separable gaussians dataset.
- Maximum number of iterations: 200 (0-199) iterations.
- Initialize features scaling in order to normalize the data input to values between zero and one.
- Using the number of clusters equal two, the classification results with the demonstration of clusters center shown in figure 13. The algorithm convergence is shown in figure 14
- Using the number of clusters equal to three, the classification results with the demonstration of clusters center shown in figure 15. The algorithm convergence is shown in figure 16. The X-axis refers to the number of iterations we use to update and enhance the selection of the cluster centers. The Y-axis refers to the calculated cost function as per to 3.
- Considering that we don't know the distribution of the data and we don't have prior knowledge about the actual number of classes in the dataset. Hence, we are dealing with an unlabeled dataset, then we run the clustering over a different number of clusters starting from two clusters to 15 clusters to test the best data points distribution over each cluster center. We stop the iteration when we reach the point where the difference between the error rate in the previous iteration is less than or equal to 0.02. Figure 17 shows the dataset over the best number of found clusters. Figure 18 shows the algorithm iteration convergence over a number of iterations to find the best number of clusters (The baseline of numbers starts from 2, and the X-axis refers to the number of clusters, the Y-axis refers to the cost function)

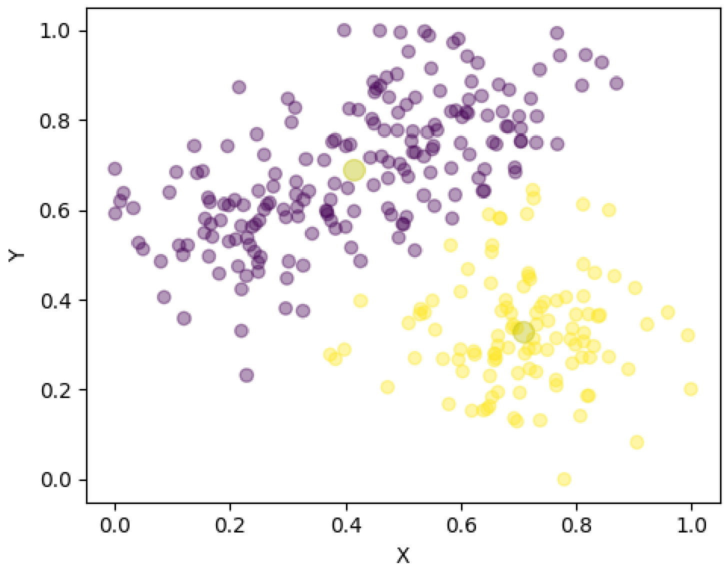

Fig. 13. Slightly overlapping three separable gaussians dataset over 2 clusters.

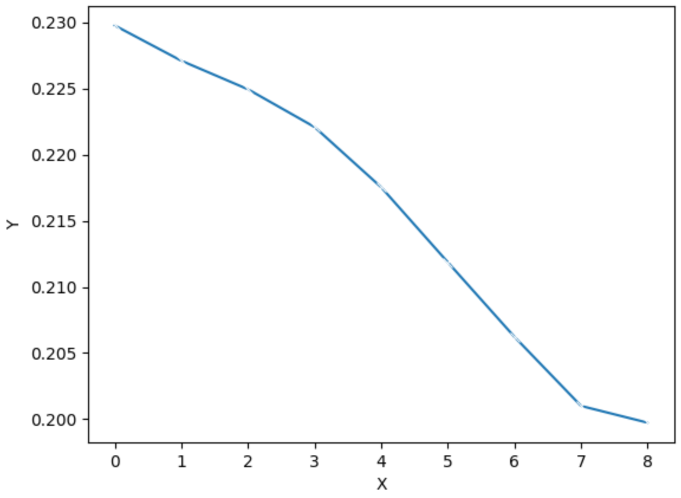

Fig. 14. Slightly overlapping three separable gaussians - Algorithm iteration convergence - 2 clusters.

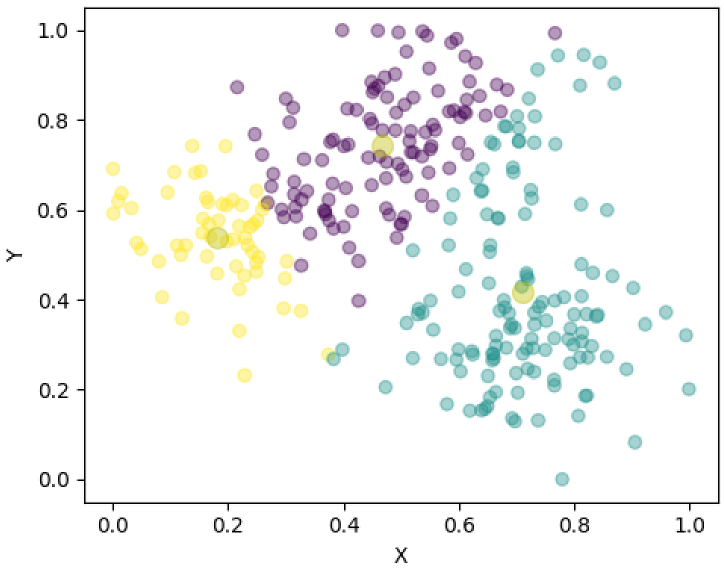

Fig. 15. Slightly overlapping three separable gaussians dataset over 3 clusters.

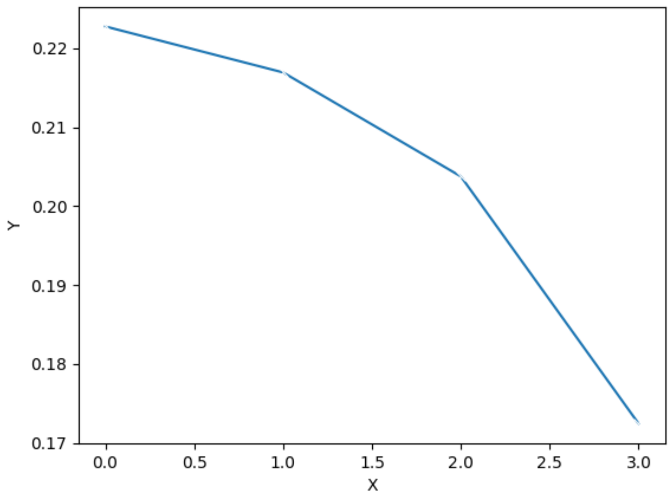

Fig. 16. Slightly overlapping three separable gaussians - Algorithm iteration convergence - 3 clusters.

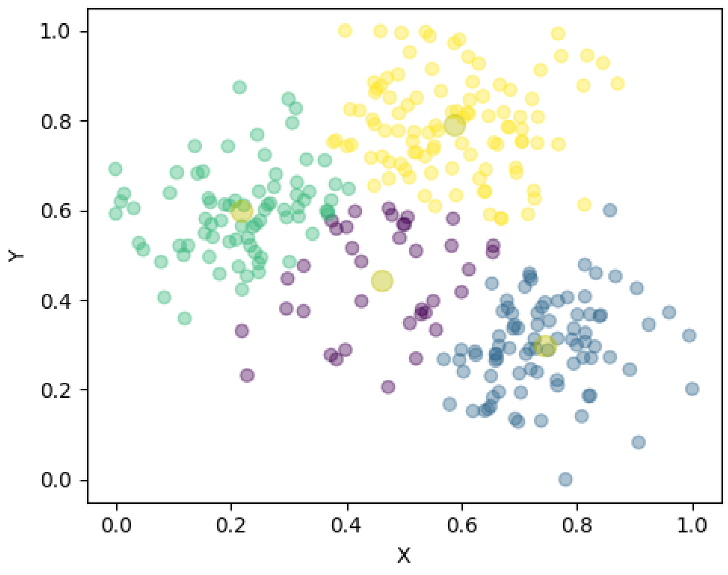

Fig. 17. Slightly overlapping three separable gaussians dataset over the best number of found clusters.

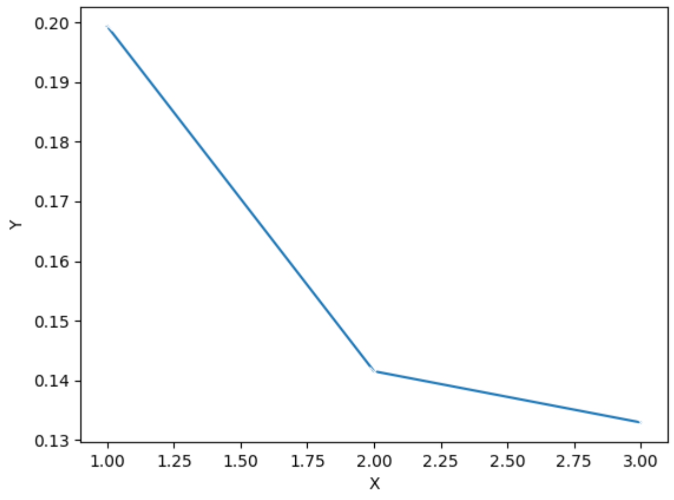

Fig. 18. Slightly overlapping three separable gaussians dataset - Algorithm iteration convergence over a number of iterations to find the best number of clusters

4.4 Experiments using Moons dataset

The experiment setup and execution results are as follows:

- Dataset: Moons dataset.
- Maximum number of iterations: 200 (0-199) iterations.
- Initialize features scaling in order to normalize the data input to values between zero and one.
- Using the number of clusters equal two, the classification results with the demonstration of clusters center shown in figure 19. The algorithm convergence is shown in figure 20.
- Using the number of clusters equal to three, the classification results with the demonstration of clusters center shown in figure 21. The algorithm convergence is shown in figure 22. The X-axis refers to the number of iterations we use to update and enhance the selection of the cluster centers. The Y-axis refers to the calculated cost function as per to 3.
- Considering that we don't know the distribution of the data and we don't have prior knowledge about the actual number of classes in the dataset. Hence, we are dealing with an unlabeled dataset, then we run the clustering over a different number of clusters starting from two clusters to 15 clusters to test the best data points distribution over each cluster center. We stop the iteration when we reach the point where the difference between the error rate in the previous iteration is less than or equal to 0.02. Figure 23 shows the dataset over the best number of found clusters. Figure 24 shows the algorithm iteration convergence over a number of iterations to find the best number of clusters (The baseline of numbers starts from 2, and the X-axis refers to the number of clusters, the Y-axis refers to the cost function).

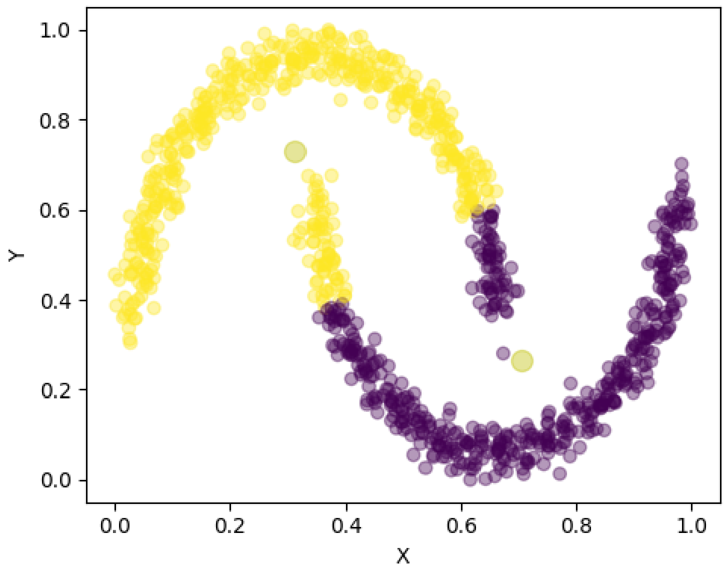

Fig. 19. Moons dataset over 2 clusters .

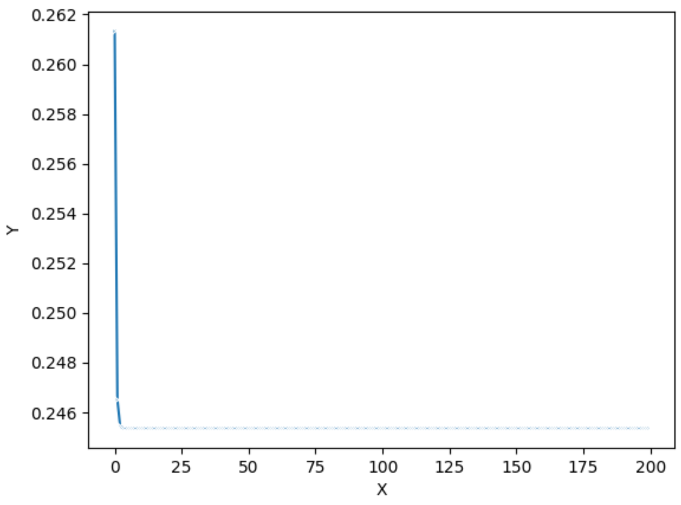

Fig. 20. Moons dataset - Algorithm iteration convergence - 2 clusters.

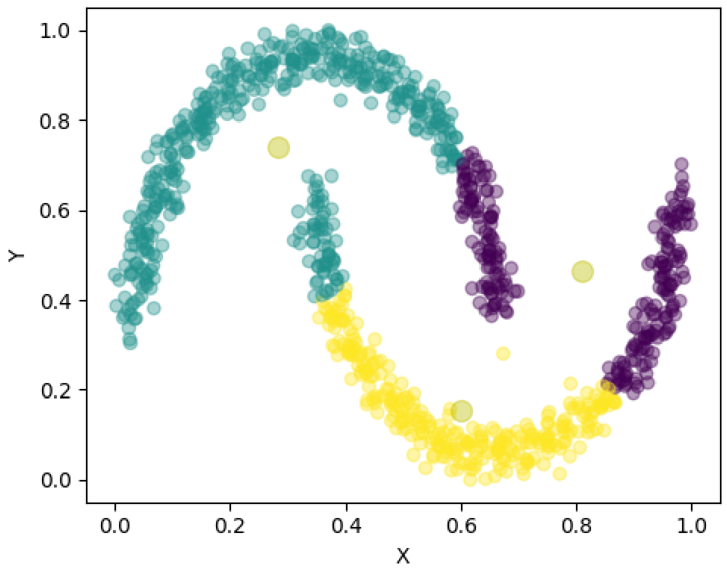

Fig. 21. Moons dataset over 3 clusters .

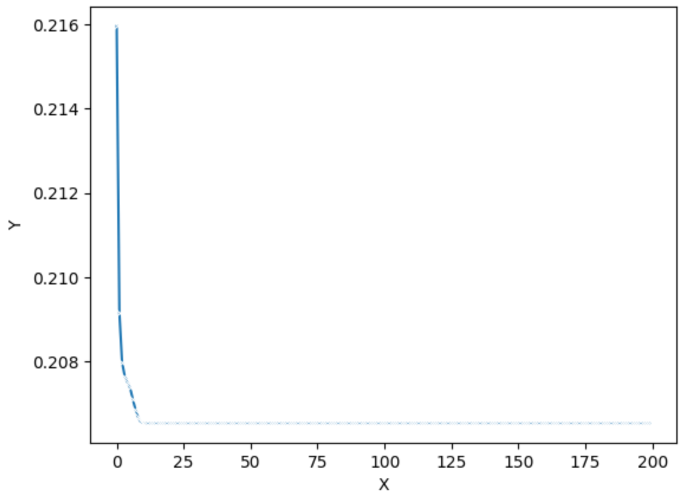

Fig. 22. Moons dataset - Algorithm iteration convergence - 3 clusters.

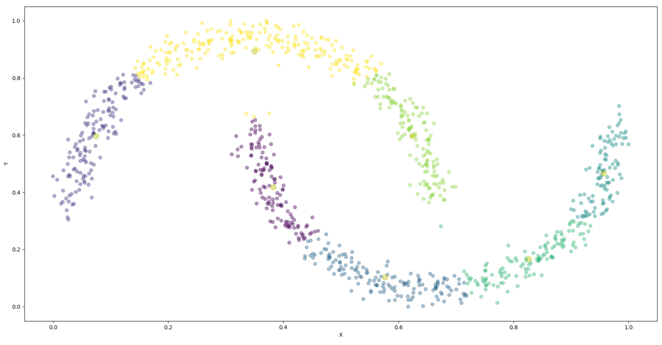

Fig. 23. Moons dataset over the best number of found clusters.

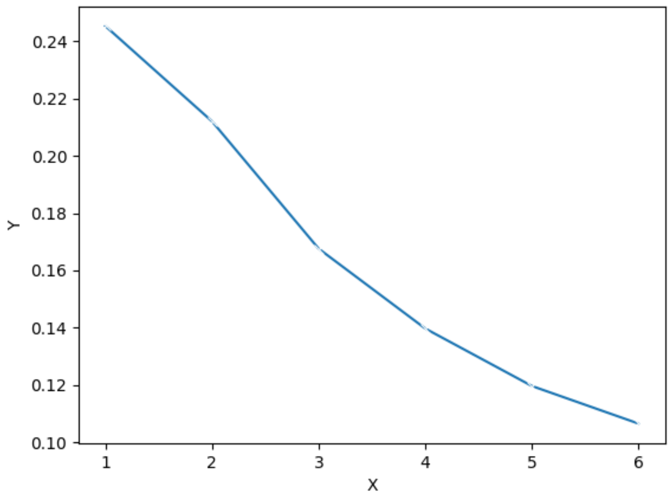

Fig. 24. Moons dataset - Algorithm iteration convergence over a number of iterations to find the best number of clusters

4.5 Experiments using Circles dataset

The experiment setup and execution results are as follows:

- Dataset: Circles dataset.
- Maximum number of iterations: 200 (0-199) iterations.
- Initialize features scaling in order to normalize the data input to values between zero and one.
- Using the number of clusters equal two, the classification results with the demonstration of clusters center shown in figure 25. The algorithm convergence is shown in figure 26
- Using the number of clusters equal to three, the classification results with the demonstration of clusters center shown in figure 27. The algorithm convergence is shown in figure 28. The X-axis refers to the number of iterations we use to update and enhance the selection of the cluster centers. The Y-axis refers to the calculated cost function as per to 3.
- Considering that we don't know the distribution of the data and we don't have prior knowledge about the actual number of classes in the dataset. Hence, we are dealing with an unlabeled dataset, then we run the clustering over a different number of clusters starting from two clusters to 15 clusters to test the best data points distribution over each cluster center. We stop the iteration when we reach the point where the difference between the error rate in the previous iteration is less than or equal to 0.02. Figure 29 shows the dataset over the best number of found clusters. Figure 30 shows the algorithm iteration convergence over a number of iterations to find the best number of clusters (The baseline of numbers starts from 2, and the X-axis refers to the number of clusters, the Y-axis refers to the cost function).

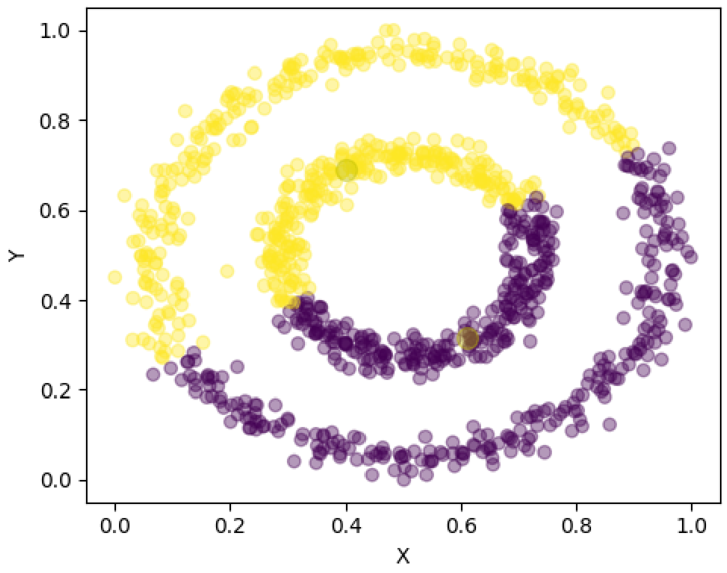

Fig. 25. Circles dataset over 2 clusters .

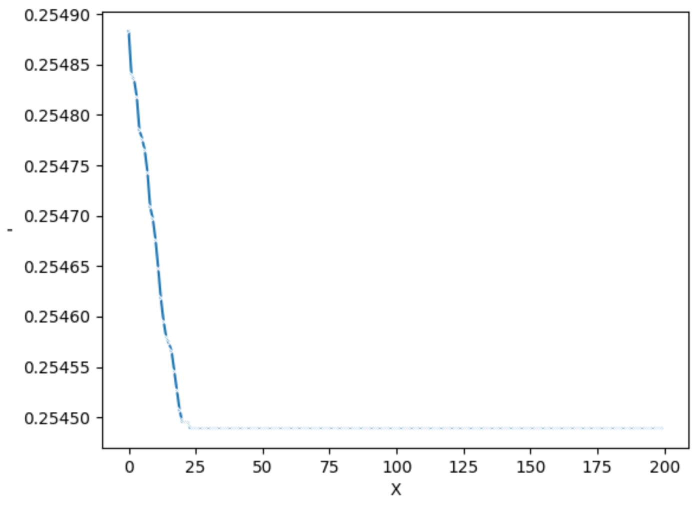

Fig. 26. Circles dataset - Algorithm iteration convergence - 2 clusters.

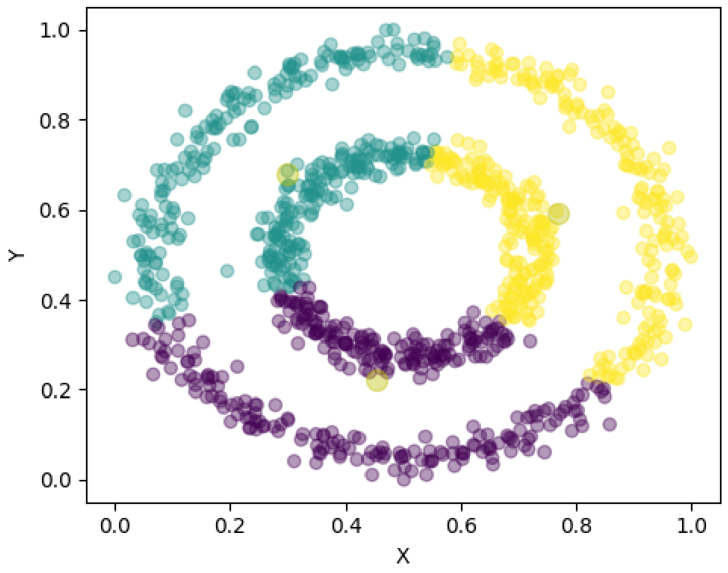

Fig. 27. Circles dataset over 3 clusters .

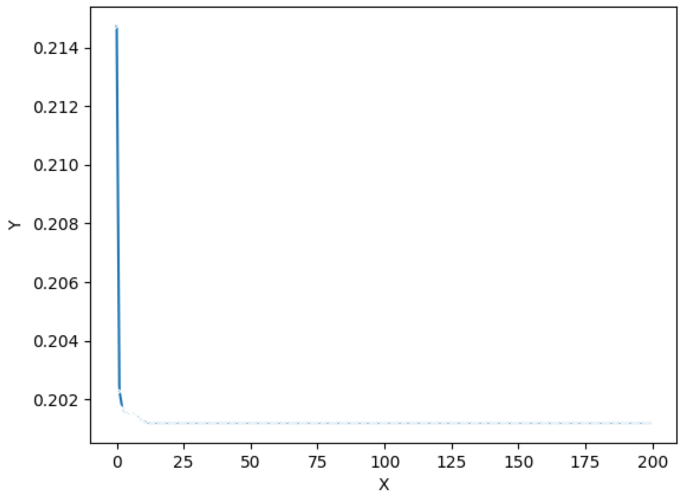

Fig. 28. Circles dataset - Algorithm iteration convergence - 3 clusters.

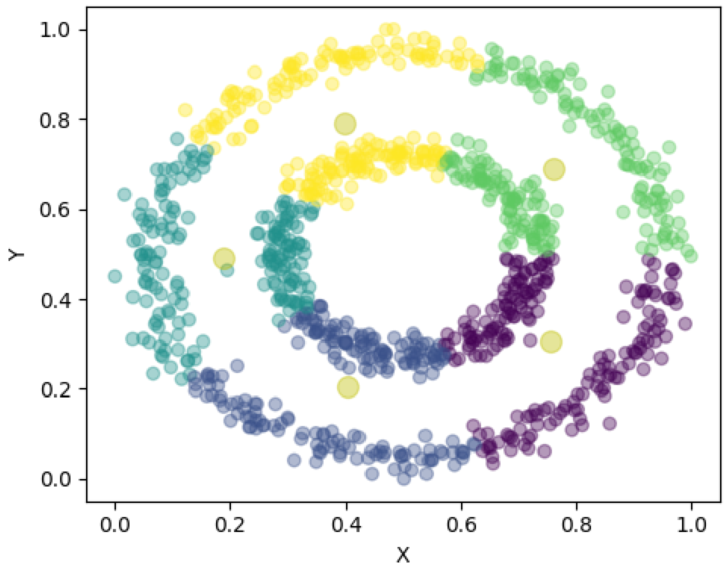

Fig. 29. Circles dataset over the best number of found clusters.

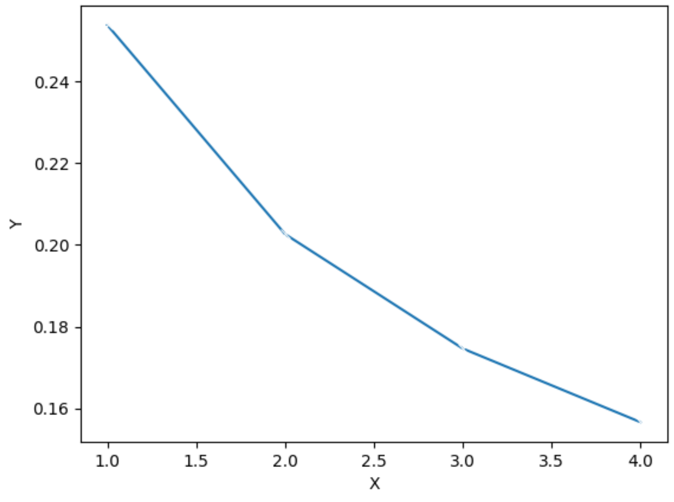

Fig. 30. Circles dataset - Algorithm iteration convergence over a number of iterations to find the best number of clusters

5 Discussion

Based on the conducted experiments, the following takeaway about the K-means:

– Clusters centers based algorithm: Where each cluster center represents the mean of all data points belonging to that cluster.
– Sensitivity for outlier data points: Where the data point distribution is controlling the cluster center calculation over each round of iterations.
– The impact of the selected starting cluster centers: The selected Centers of the initial clusters have a substantial influence on the final clusters generated by the K-Means method, it is critical to carefully select at the initialization time the initial centers of the cluster that really reflect the nature of data.
– Produce same results under the same conduction: K-means implementation is deterministic where when giving the same number of clusters, the same dataset, the same number of maximum iterations, and the same Algorithm iteration convergence limit (error rate). The produced clusters will be the same.
– The implementation considers that any particular data point is hard clustered and it is assigned to one and only one single cluster and is exclusively associated with that cluster.
– The number of clusters must be determined before running the classification iterations: One of the K-Means algorithm's disadvantages is that the number of clusters must be set in advance, which can be challenging in many real-world situations where the nature of data is not well understood.

References

1. Ahmed, M., Seraj, R., Islam, S.M.S.: The k-means algorithm: A comprehensive survey and performance evaluation. Electronics **9**(8), 1295 (2020)
2. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Comput. Surv. **31**(3), 264–323 (1999). https://doi.org/10.1145/331499.331504, https://doi.org/10.1145/331499.331504
3. Lloyd, S.P.: Least squares quantization in PCM. IEEE Trans. Inf. Theory **28**(2), 129–136 (1982). https://doi.org/10.1109/TIT.1982.1056489, https://doi.org/10.1109/TIT.1982.1056489