Pattern Recognition Homework (5)

K-Means, GMM, SVMs

Instructed by Dr. Zohreh Azimifar

Abbas Mehrbanian

abbas.mrbn@gmail.com, Stu ID: 40130935

Reza Tahmasebi

saeed77t@gmail.com, Stu ID: 40160957

Table of Contents

1 Introduction

In This project we are going to detect covid-19 infection for a blood test dataset of patients that are suspected to have covid infection, using a rule based fuzzy system with help of type 1 and IT2 fuzzy sets.

2 K-Means

K-means is a popular machine learning and data mining algorithm that discovers potential clusters within a dataset. Finding these clusters in a dataset can often reveal interesting and meaningful structures underlying the distribution of data. K-means clustering has been applied to many problems in science and still remains popular today for its simplicity and effectiveness.

2.1 Implementation

The main objective of the K-Means algorithm is to minimize the sum of distances between the points and their respective cluster centroid. It consists of the following steps:

- Step 1: Choose the number of clusters K
- Step 2: Select k random points from the data as centroids
- Step 3: Assign all the points to the closest cluster centroid
- Step 4: Recompute the centroids of newly formed clusters
- Step 5: Repeat steps 3 and 4 until convergence or reaching the maximum number of iterations.

The above procedure can be formulated as follows.

Step 1: Choose the number of clusters (K)

Step 2: give a random initial value for the cluster center of gravity $\mu_j = \{j = 1, 2, \dots, K\}$

Step 3: Calculate the formula below and assign each data to a cluster.

$$r_{ij} = \begin{cases} 1 & if\ j = argmin_{\ k} \left\| x^{(i)} - \mu_k \right\|^2 \\ 0 & otherwise \end{cases}$$

Step 4: To update the cluster center of gravity, calculate the above formula below:

$$\mu_j = \frac{\sum_i r_{ij} x^{(i)}}{\sum_i r_{ij}}$$

Step 5: Repeat steps 3 & 4 to continue updating the cluster. Iteration continues until no more clusters are updated or until the maximum number of iterations pre-defined is reached.

We implemented this algorithm as class named KMeans and the procedure we discussed above is implemented in *fit* function of this class. The KMeans class accepts three parameters: 1. Number of clusters. 2. Maximum number of iterations for algorithm. And 3. Initialization method of algorithm parameters.

We implemented two ways of initializing first means: random & kfirst. In random approach we select K random points from the dataset and in kfirst method we simply select the first K points from the given dataset.

To initialize first center of gravities (means) we implemented a function named *initialise_centroids* which animalizes and returns K means based of pre-defined initialization method.

2.1.1 Elbow curve

In cluster analysis, the elbow method is a heuristic used in determining the number of clusters in a data set. The method consists of plotting the explained variation as a function of the number of clusters, and picking the elbow of the curve as the number of clusters to use. In another words, the optimal number of clusters is determined by calculating the loss function of the k-means method while varying the number of clusters and illustrating the results.

$$L = \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij} \left\| x^{(i)} - \mu_j \right\|^2$$

2.2 Results

2.2.1 TSNV dataset

2.2.1.1 Clustering results and silhouette analysis

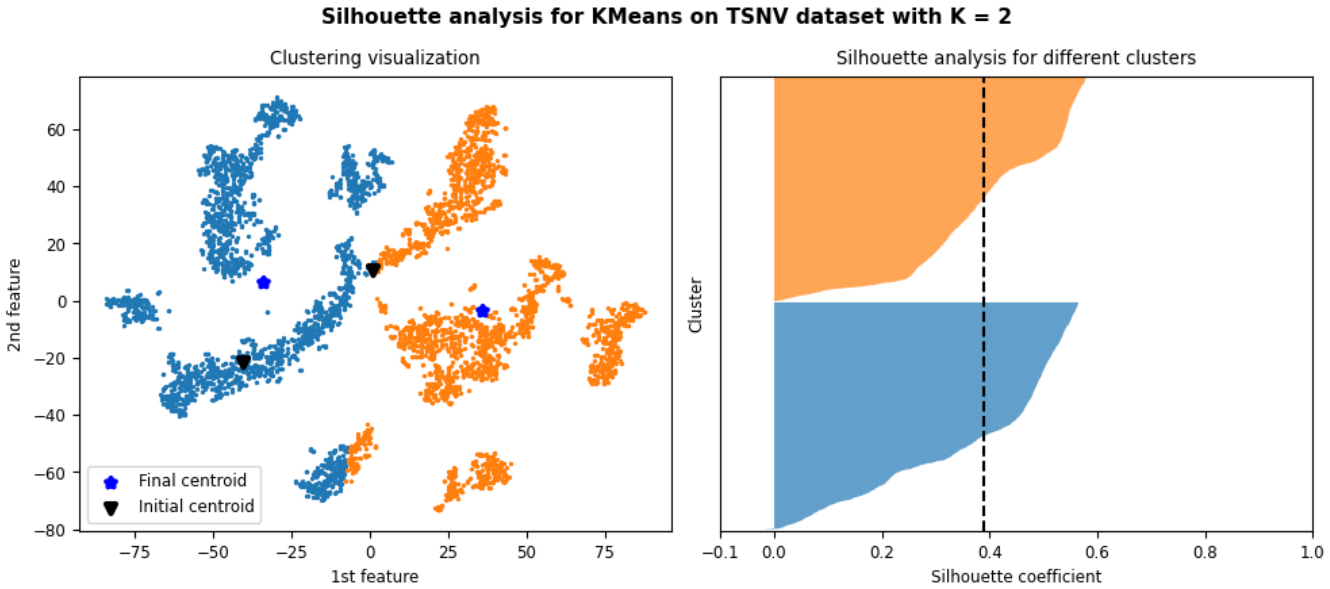


Figure 1: Results and silhouette analysis for K-Means on TSNV dataset with K = 2

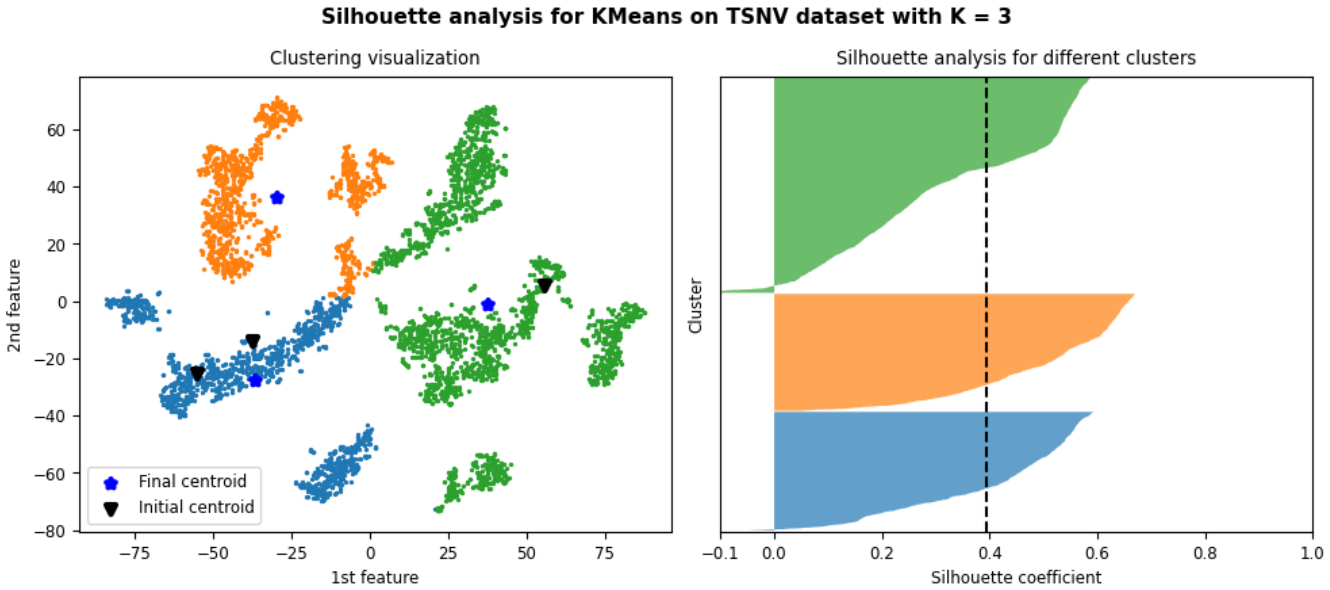


Figure 2: Results and silhouette analysis for K-Means on TSNV dataset with K = 3

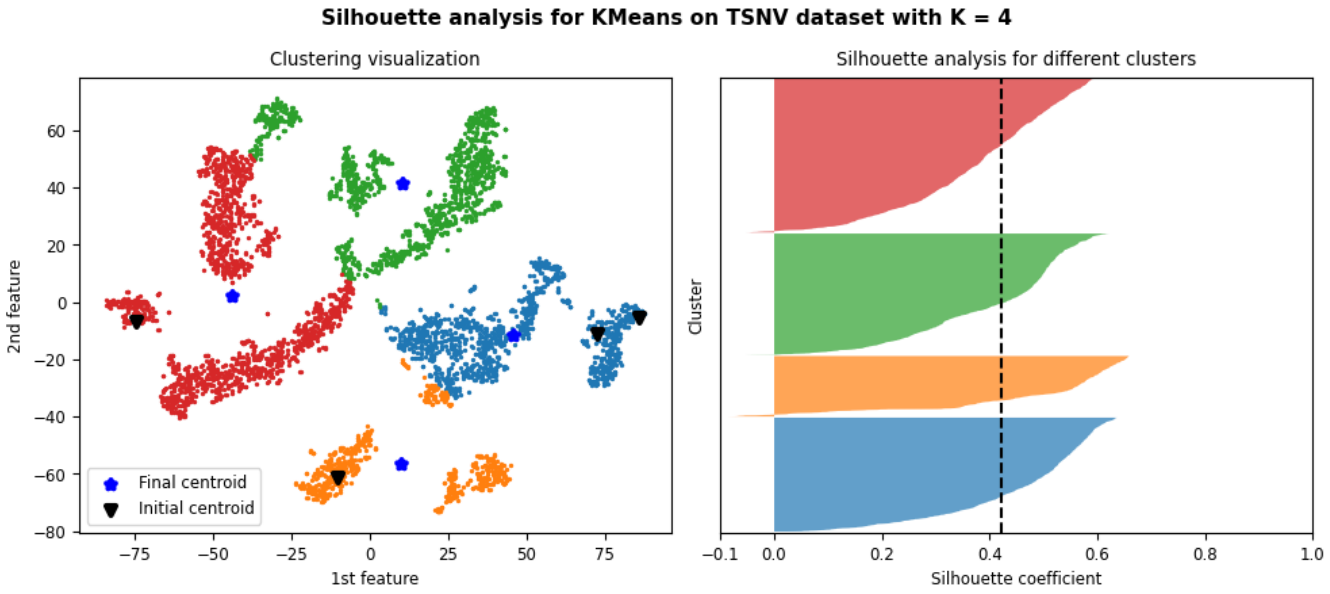


Figure 3: Results and silhouette analysis for K-Means on TSNV dataset with K = 4

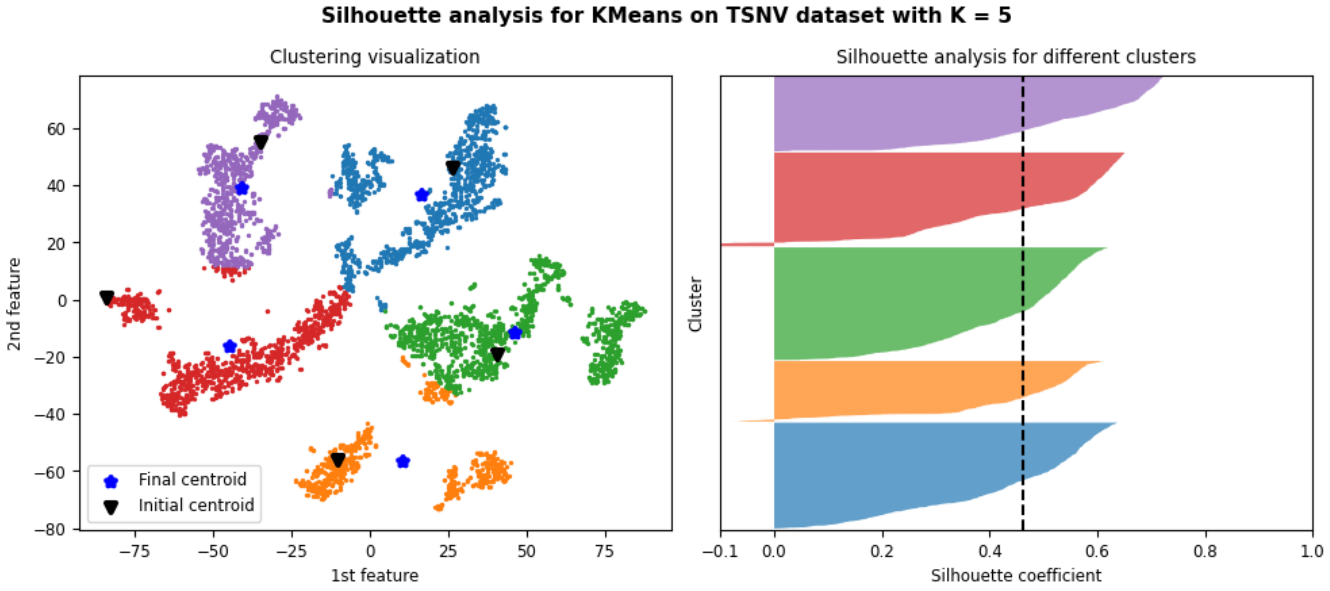


Figure 4: Results and silhouette analysis for K-Means on TSNV dataset with K = 5

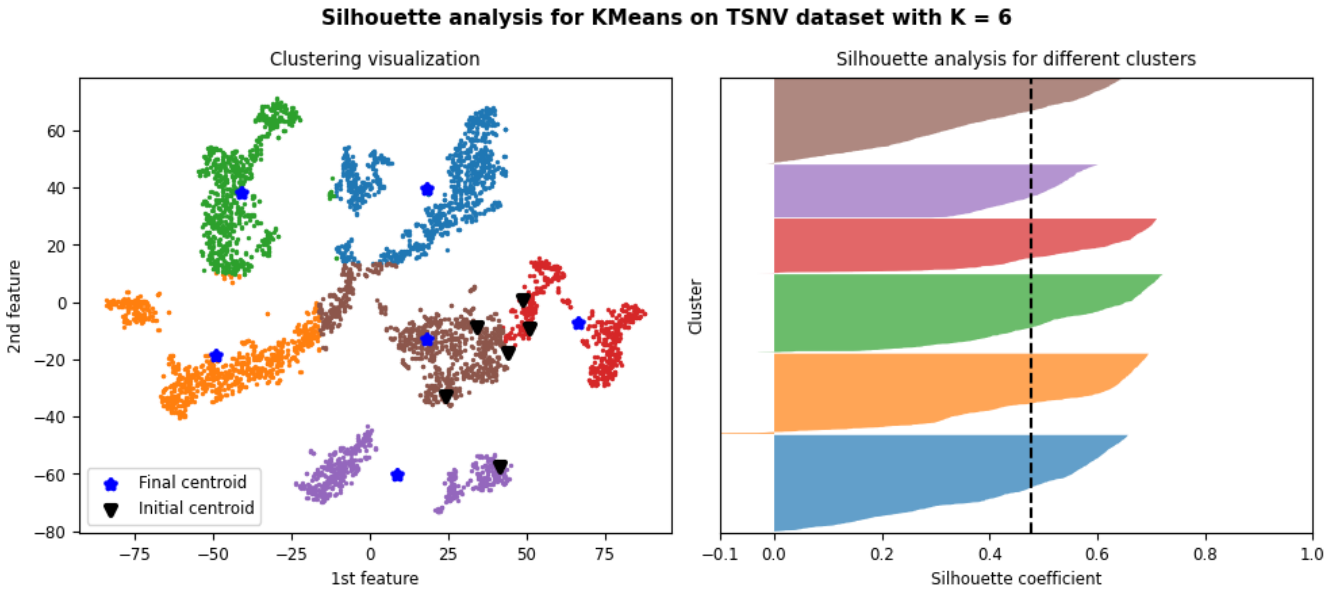


Figure 5: Results and silhouette analysis for K-Means on TSNV dataset with K = 6

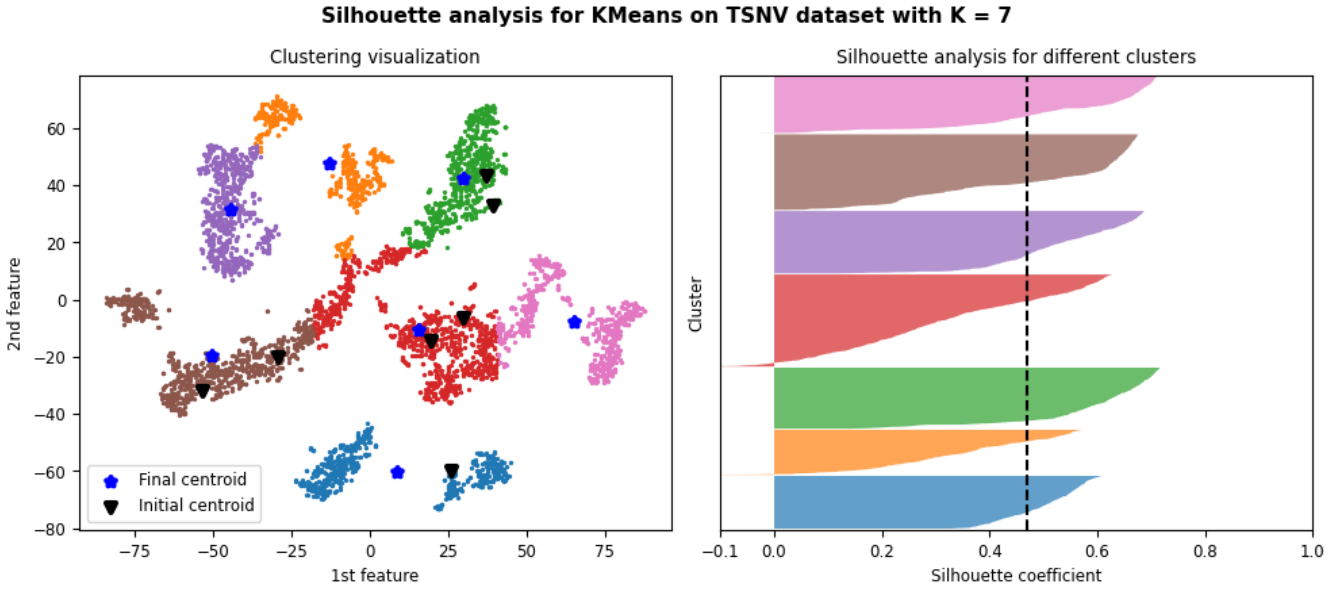


Figure 6: Results and silhouette analysis for K-Means on TSNV dataset with K = 7

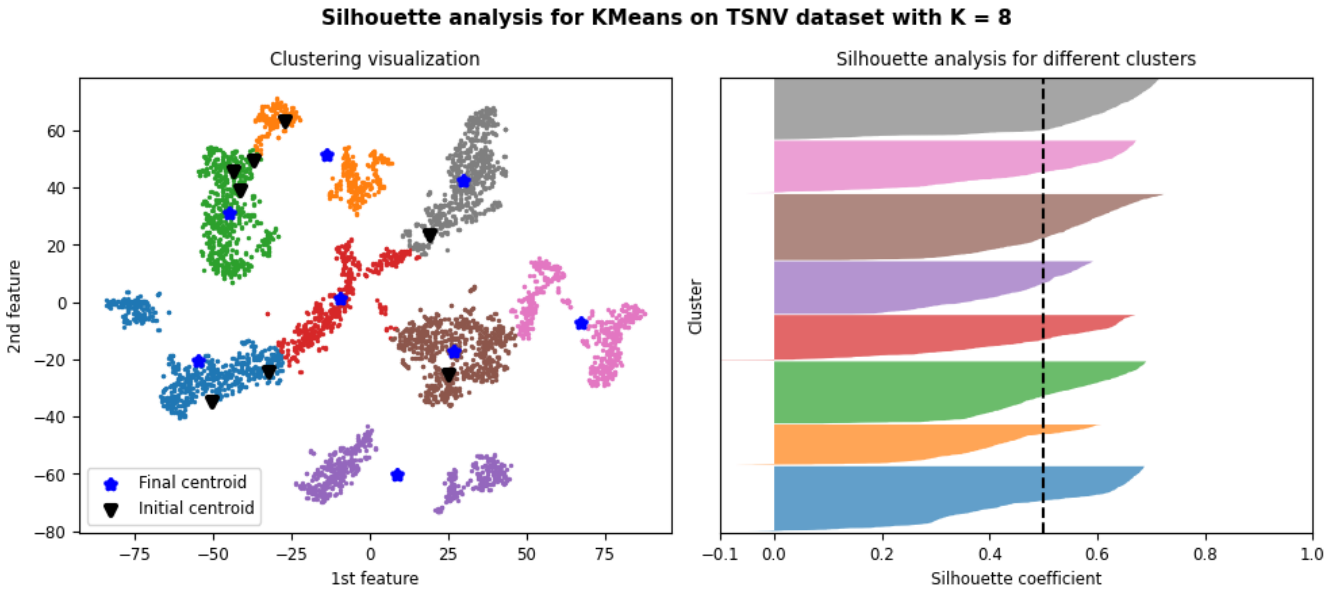


Figure 7: Results and silhouette analysis for K-Means on TSNV dataset with K = 8

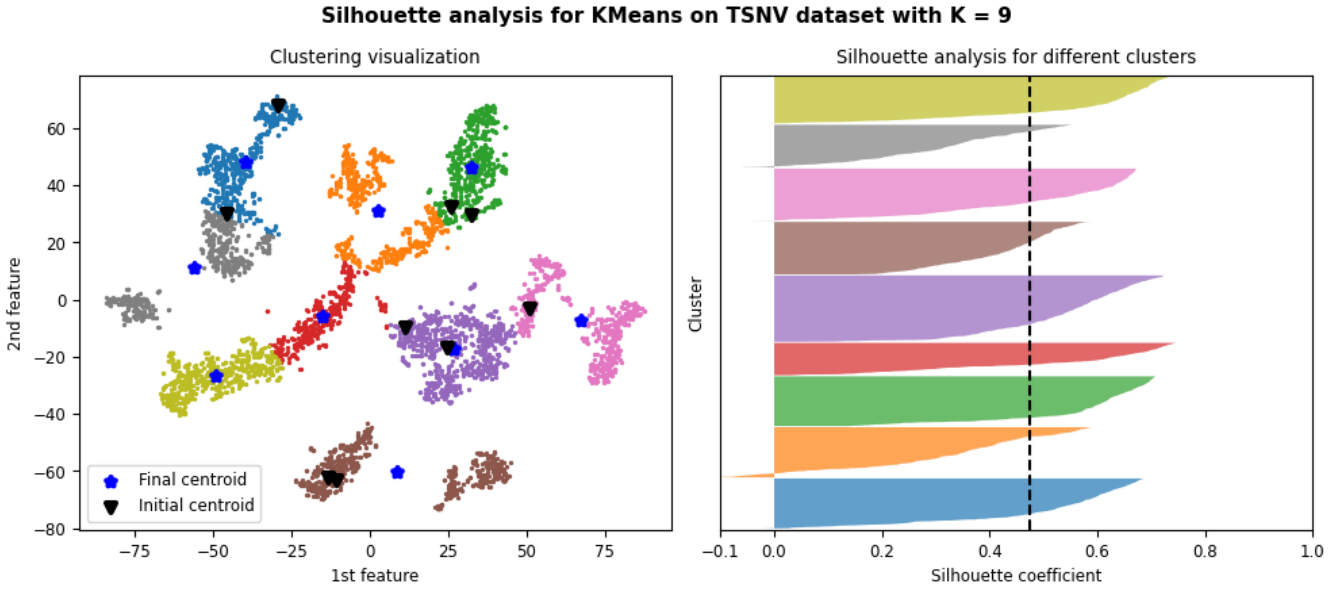


Figure 8: Results and silhouette analysis for K-Means on TSNV dataset with K = 9

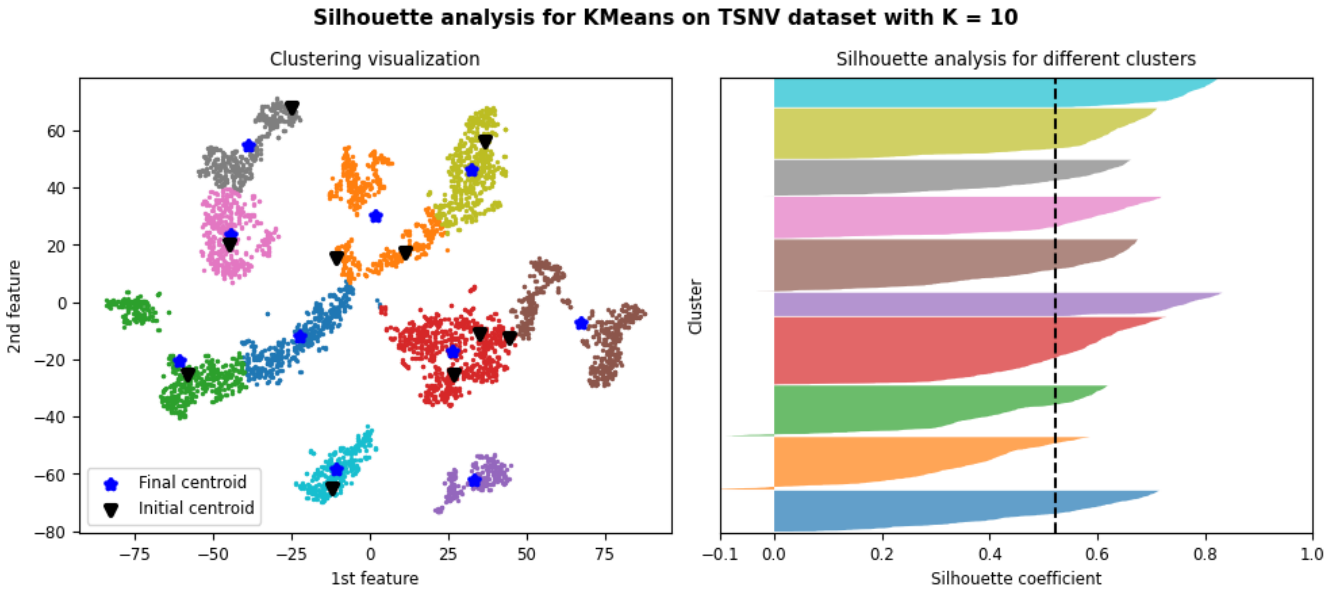


Figure 9: Results and silhouette analysis for K-Means on TSNV dataset with K = 10

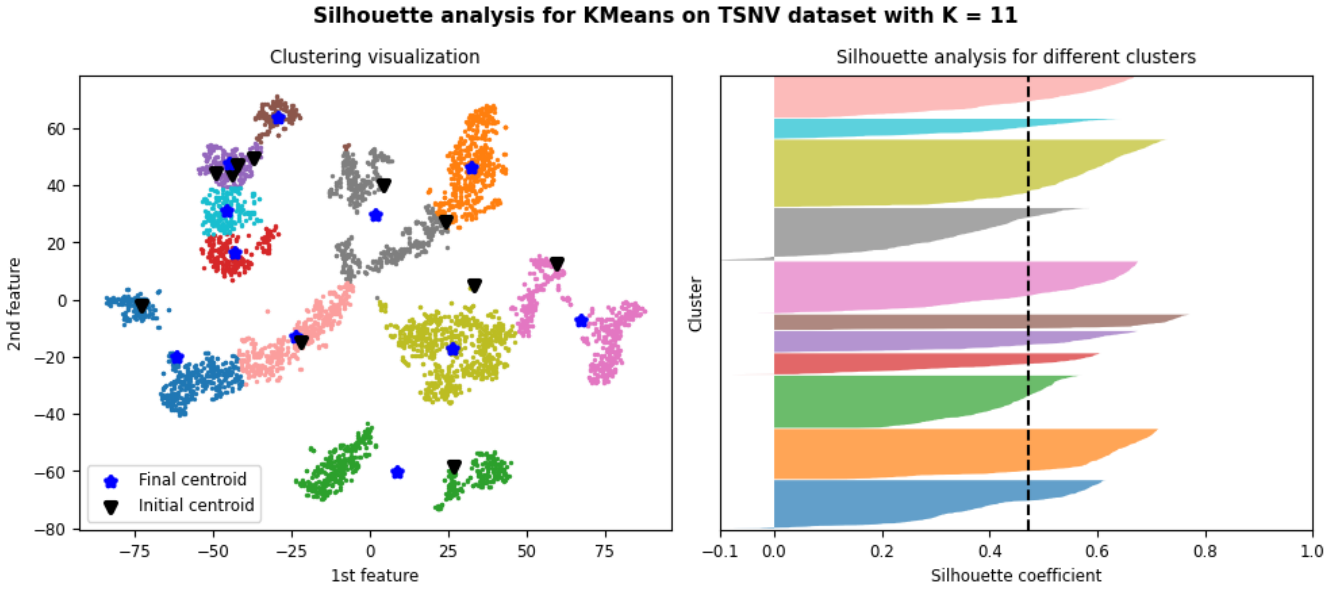


Figure 10: Results and silhouette analysis for K-Means on TSNV dataset with K = 11

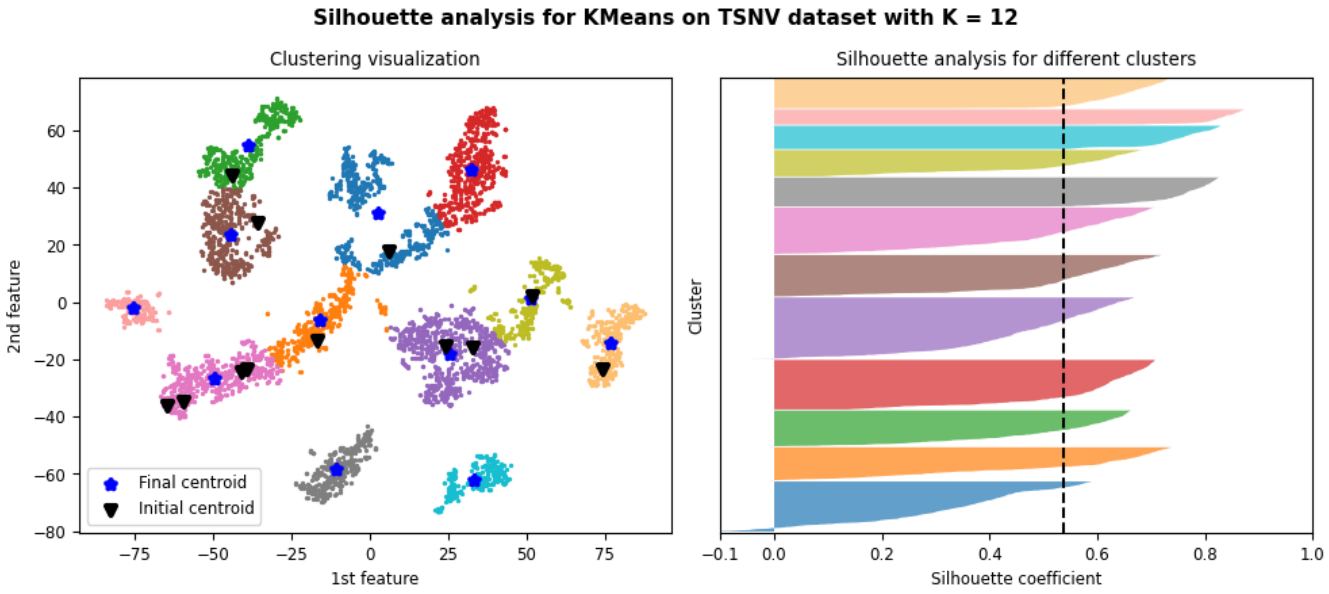


Figure 11: Results and silhouette analysis for K-Means on TSNV dataset with K = 12

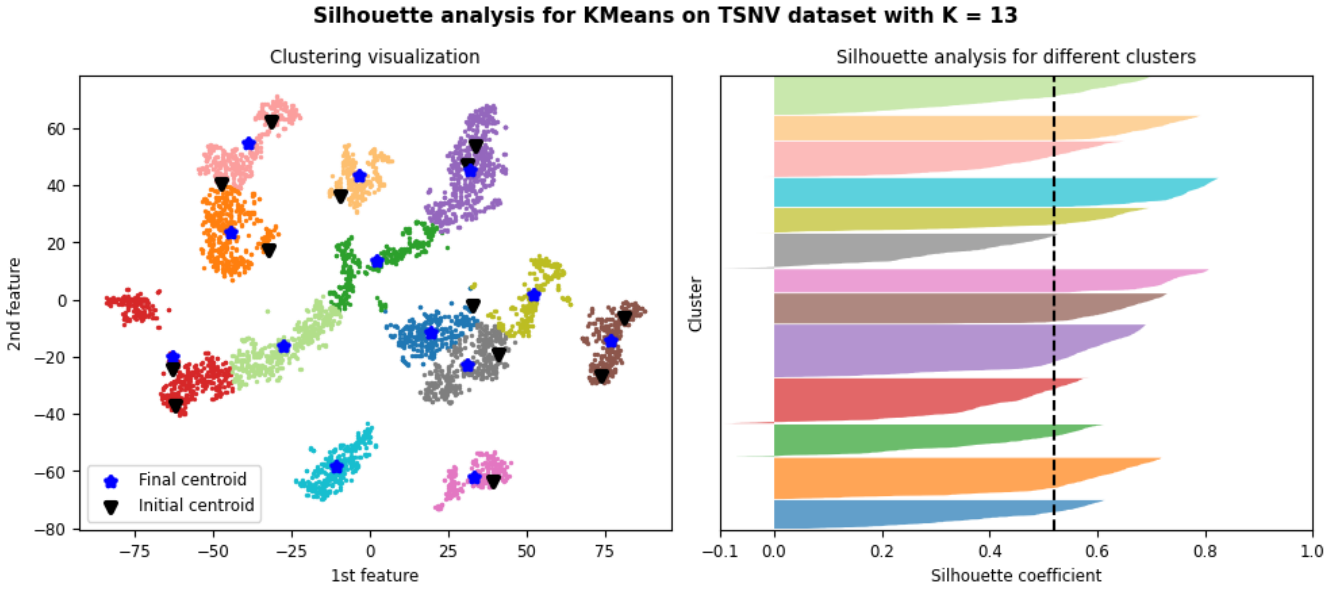


Figure 12: Results and silhouette analysis for K-Means on TSNV dataset with K = 13

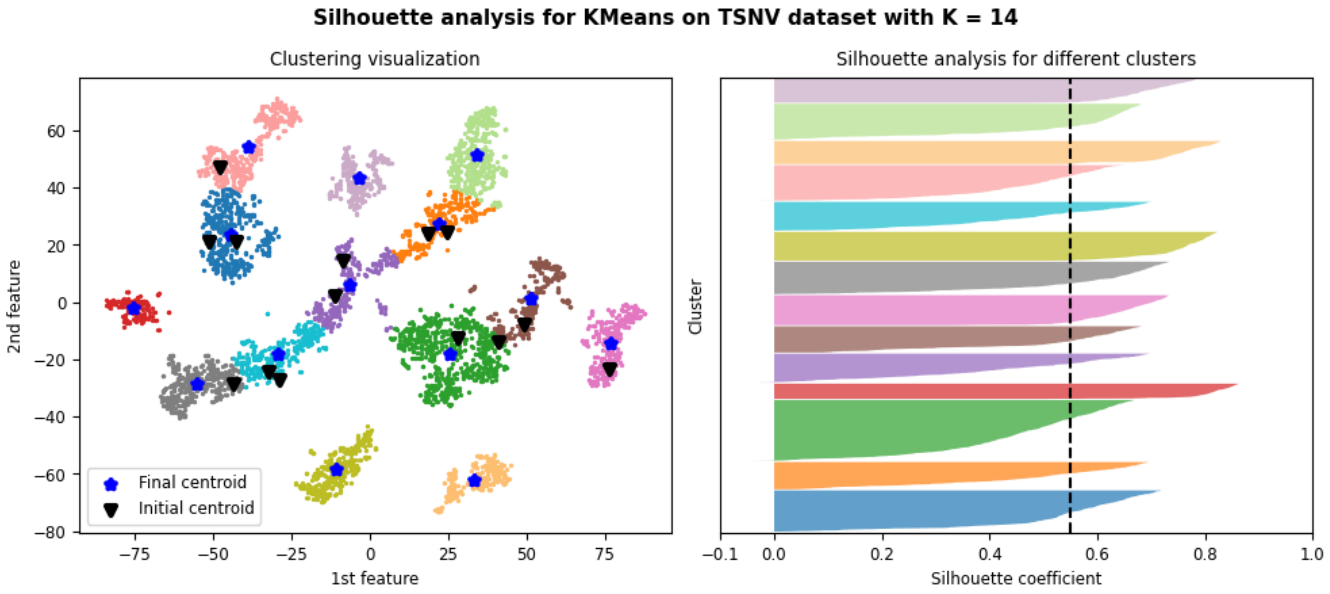


Figure 13: Results and silhouette analysis for K-Means on TSNV dataset with K = 14

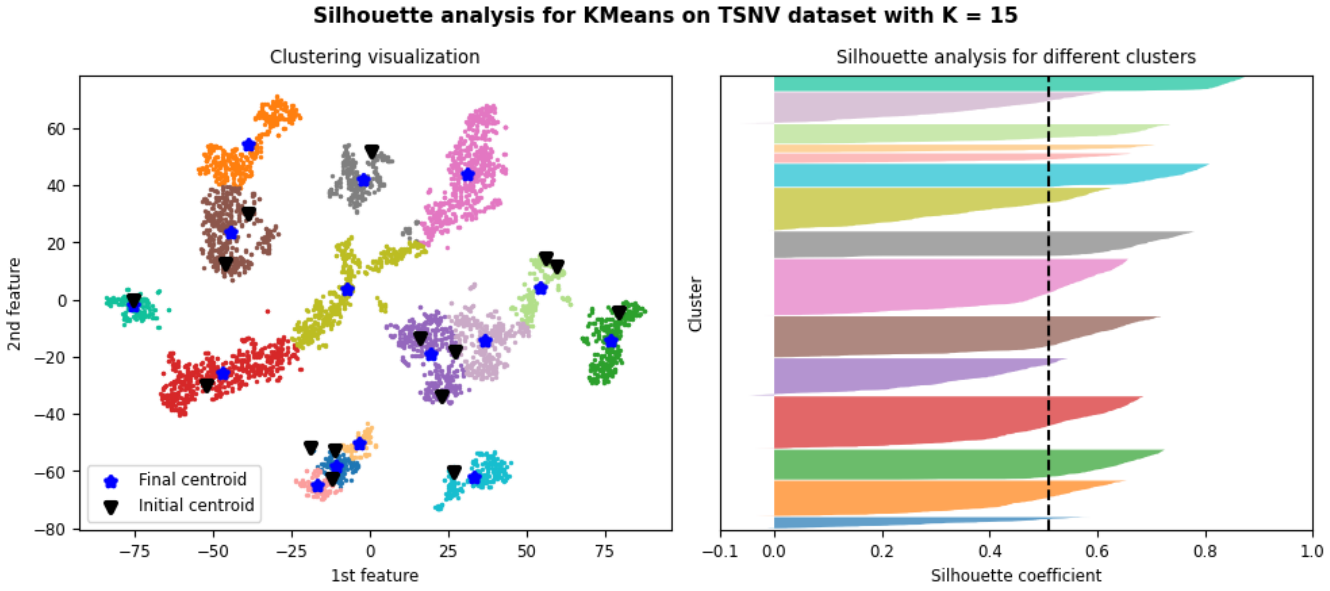
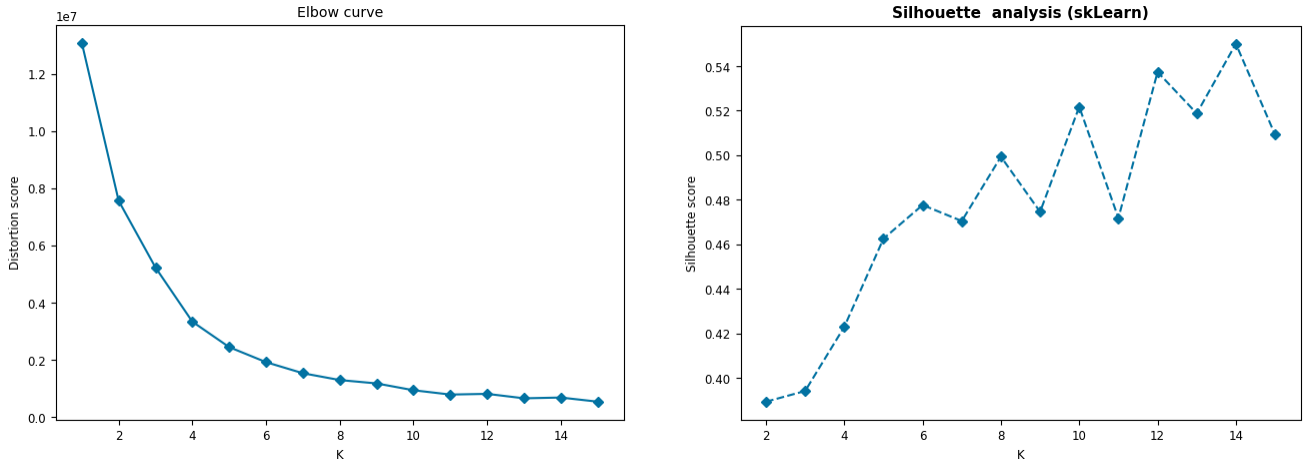


Figure 14: Results and silhouette analysis for K-Means on TSNV dataset with K = 15

2.2.1.2 Choosing the best K



(a) *(b)*

Figure 15: metrics result (a) elbow method. (b) Silhouette analysis

The location of the bend in elbow method seems to be a bit ambiguous. And It's hard to determine an optimal number of K using elbow method for this dataset.

Using silhouette method, we can choose some sub-optimal number of clusters.

According to Figure 7, the choice of K = 8 can be one of optimal choices since it stands well against all the three measuring criteria for silhouette analysis where all clusters' plot is beyond average Silhouette score, with mostly uniform thickness and do not have wide fluctuations in the size. For same reasons the choice of K = 2 & 5 to 10 can be optimal as well using this method.

Choosing number of clusters for this dataset depends on the need and we saw using both method the procedure of choosing best K was ambiguous.

We concluded that clustering on this dataset specially using k-means algorithm is hard and nearly impossible.

2.2.2 Blobs dataset

2.2.2.1 Clustering results and silhouette analysis

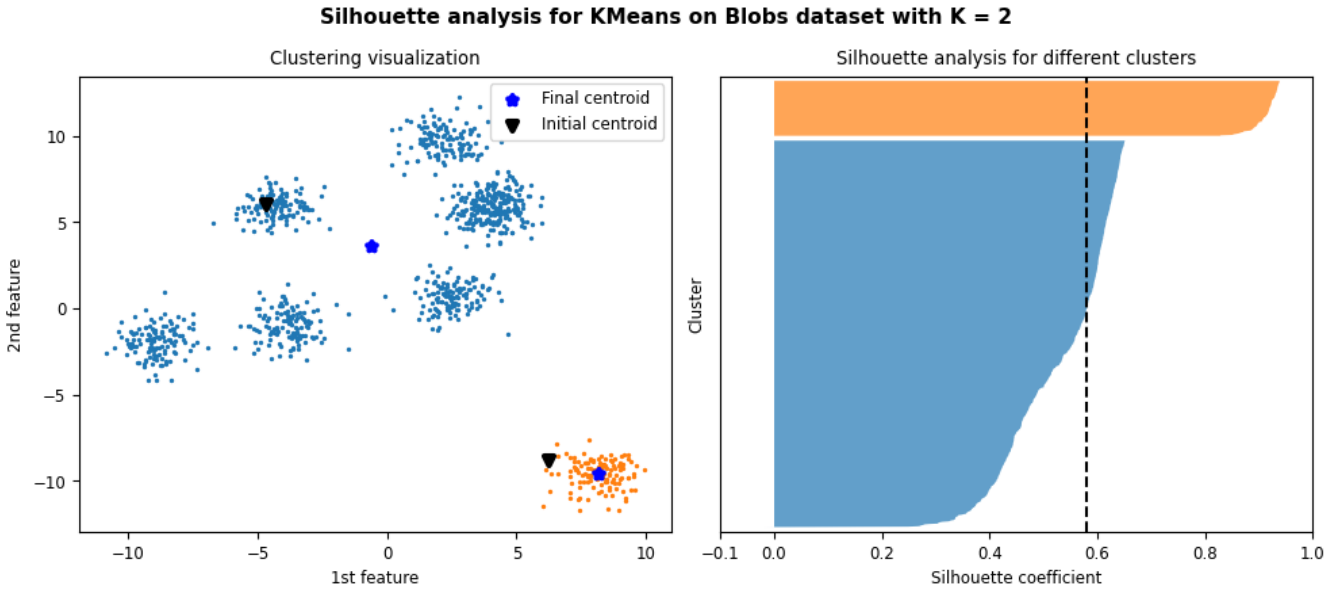


Figure 16: Results and silhouette analysis for K-Means on Blobs dataset with K = 2

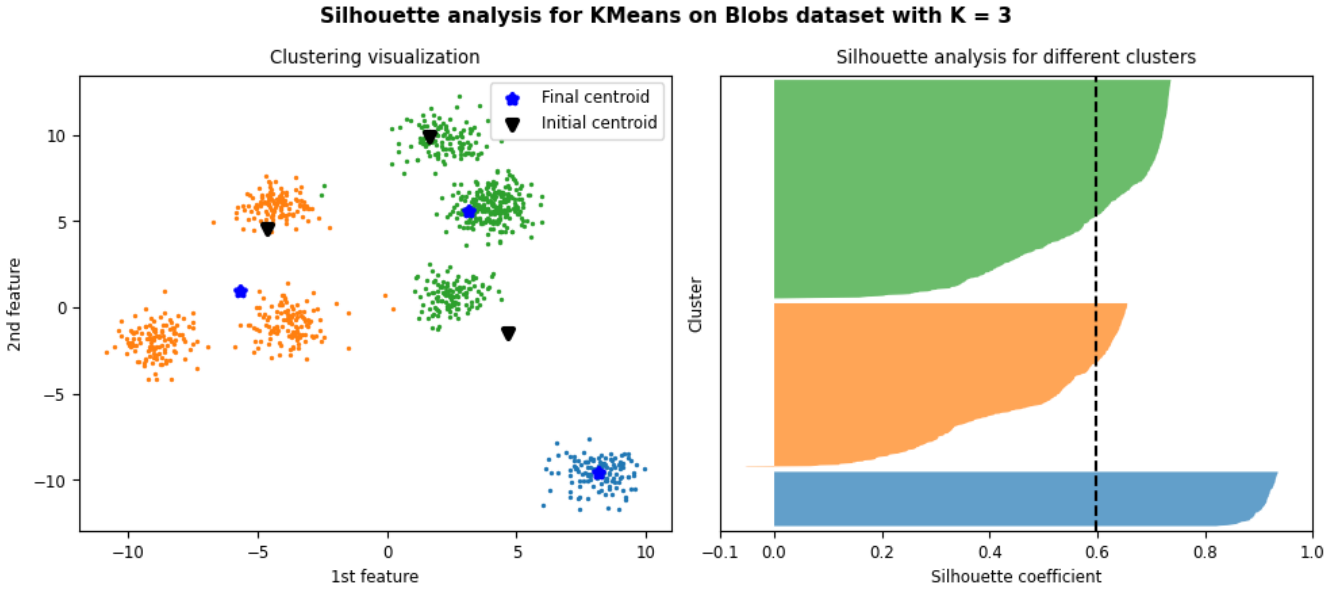


Figure 17: Results and silhouette analysis for K-Means on Blobs dataset with K = 3

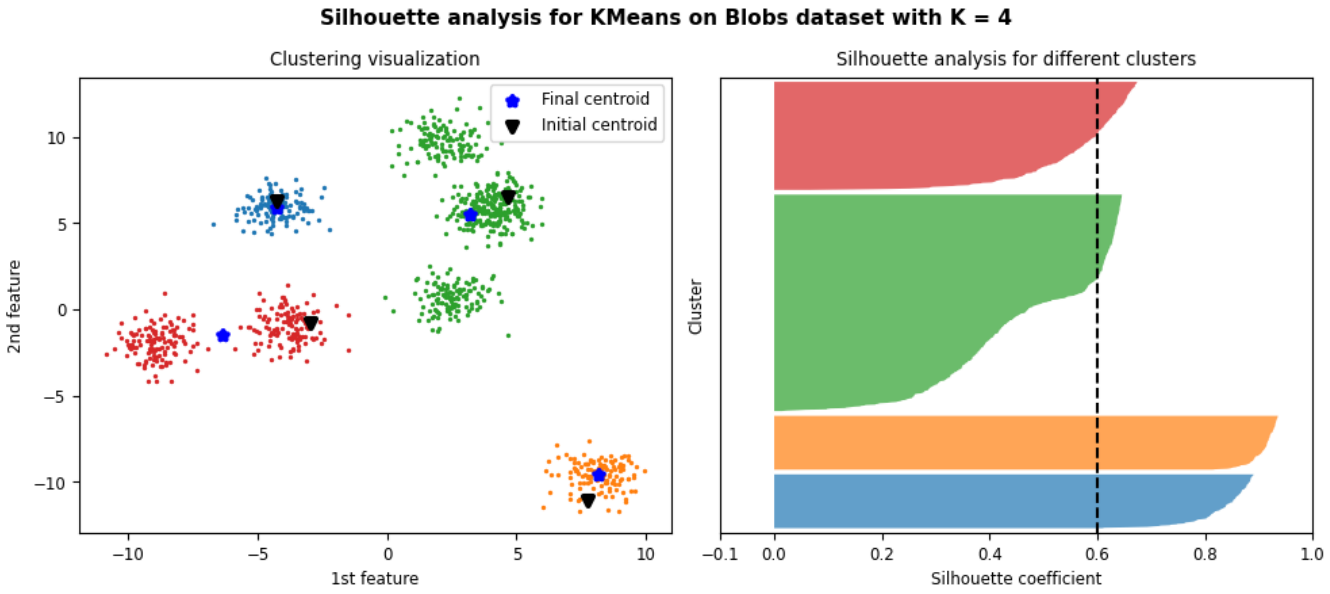


Figure 18: Results and silhouette analysis for K-Means on Blobs dataset with K = 4

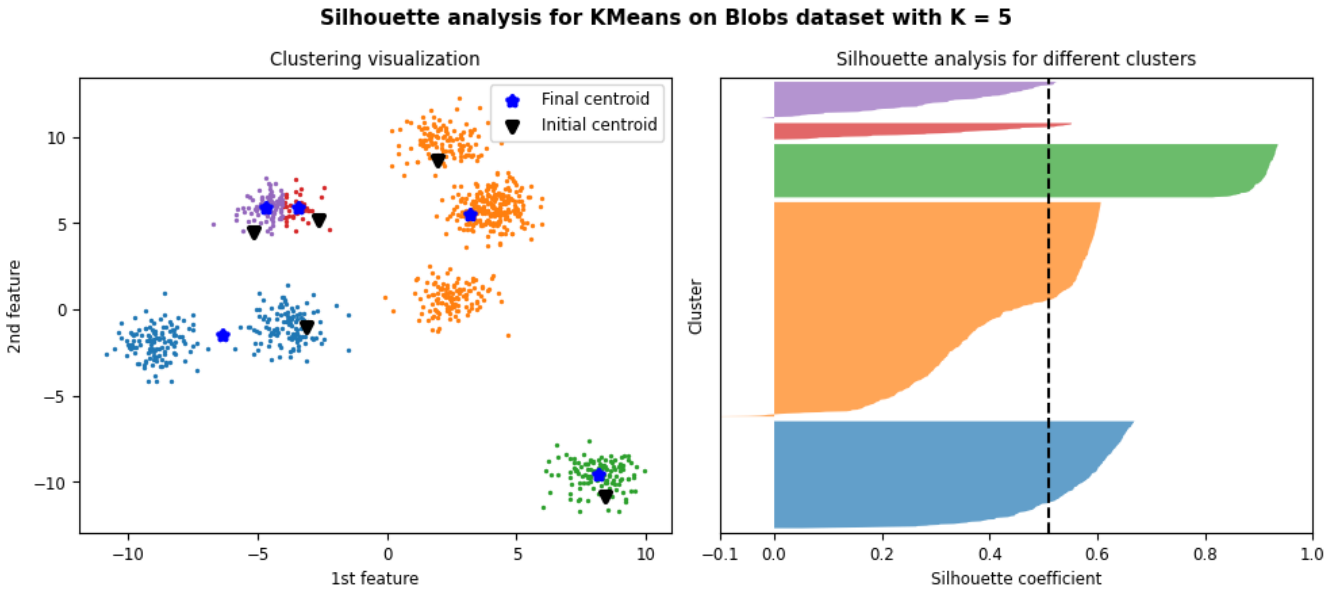


Figure 19: Results and silhouette analysis for K-Means on Blobs dataset with K = 5

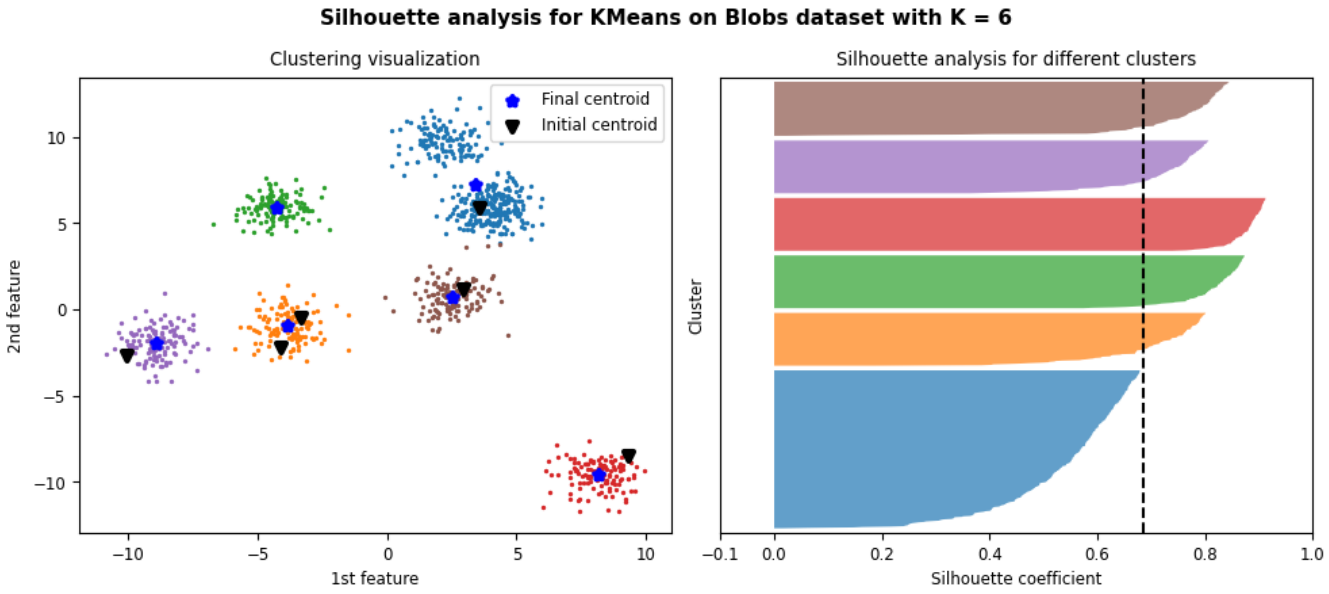


Figure 20: Results and silhouette analysis for K-Means on Blobs dataset with K = 6

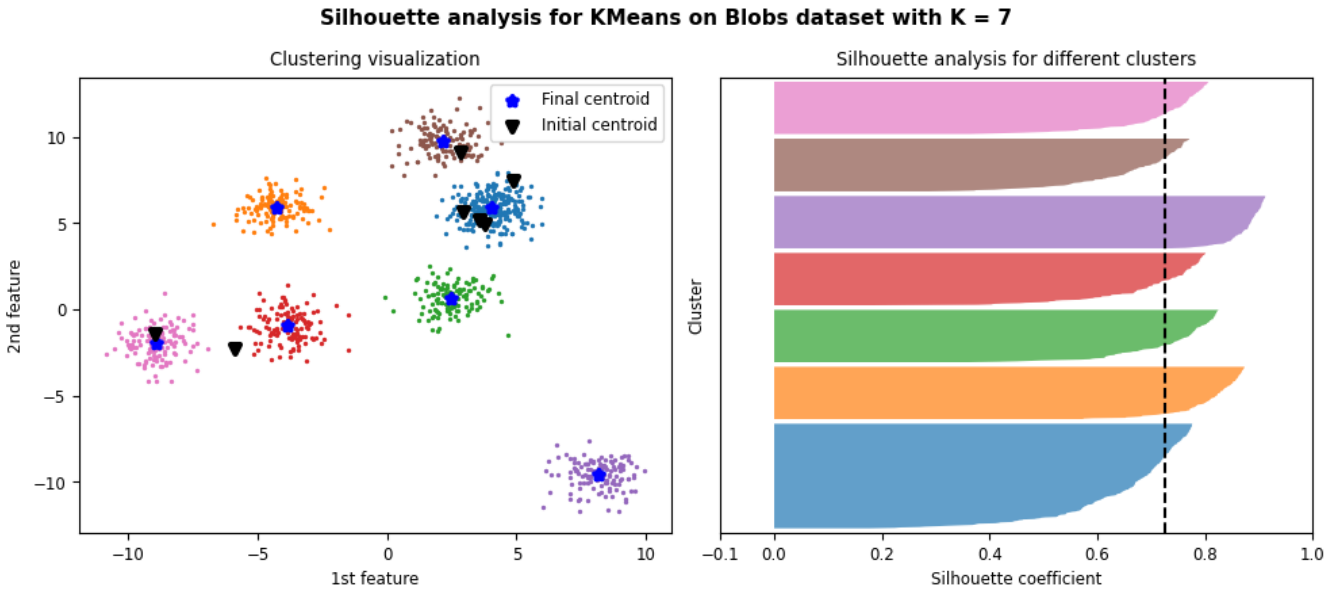


Figure 21: Results and silhouette analysis for K-Means on Blobs dataset with K = 7

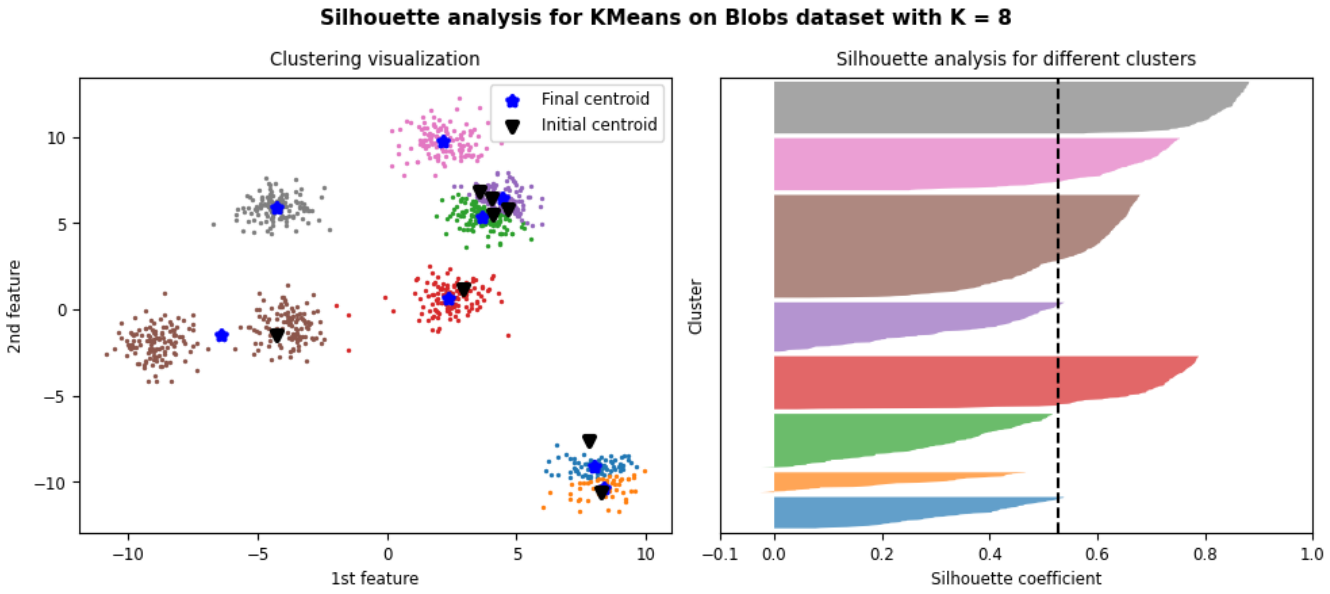


Figure 22: Results and silhouette analysis for K-Means on Blobs dataset with K = 8

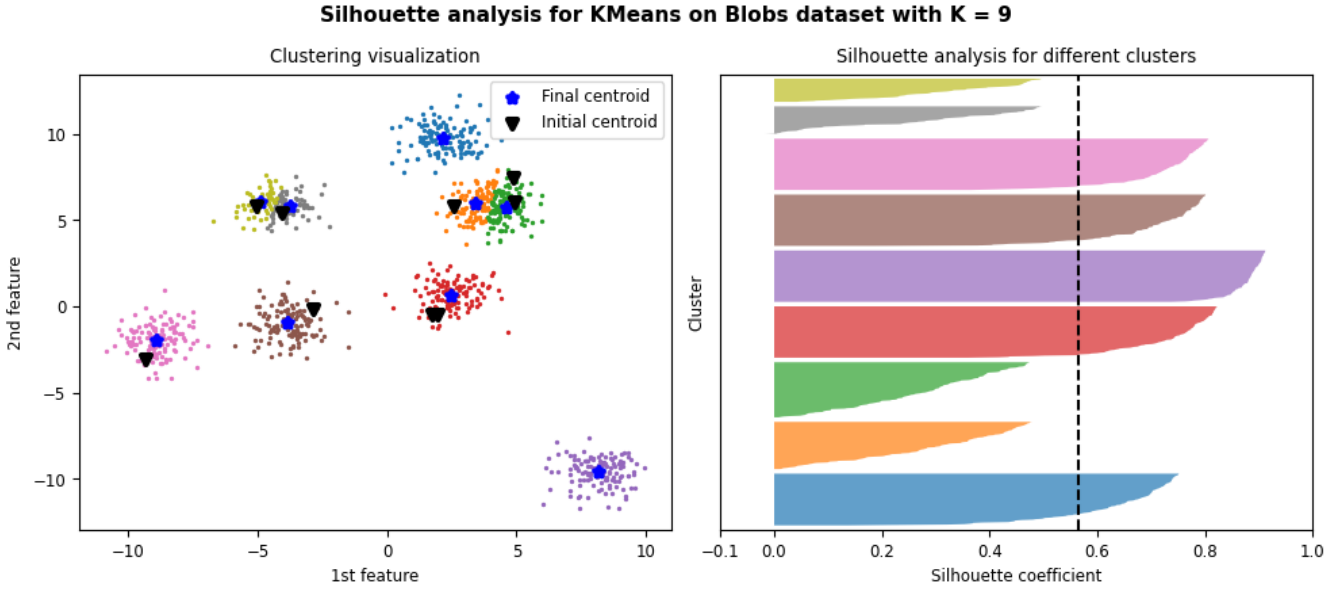


Figure 23: Results and silhouette analysis for K-Means on Blobs dataset with K = 9

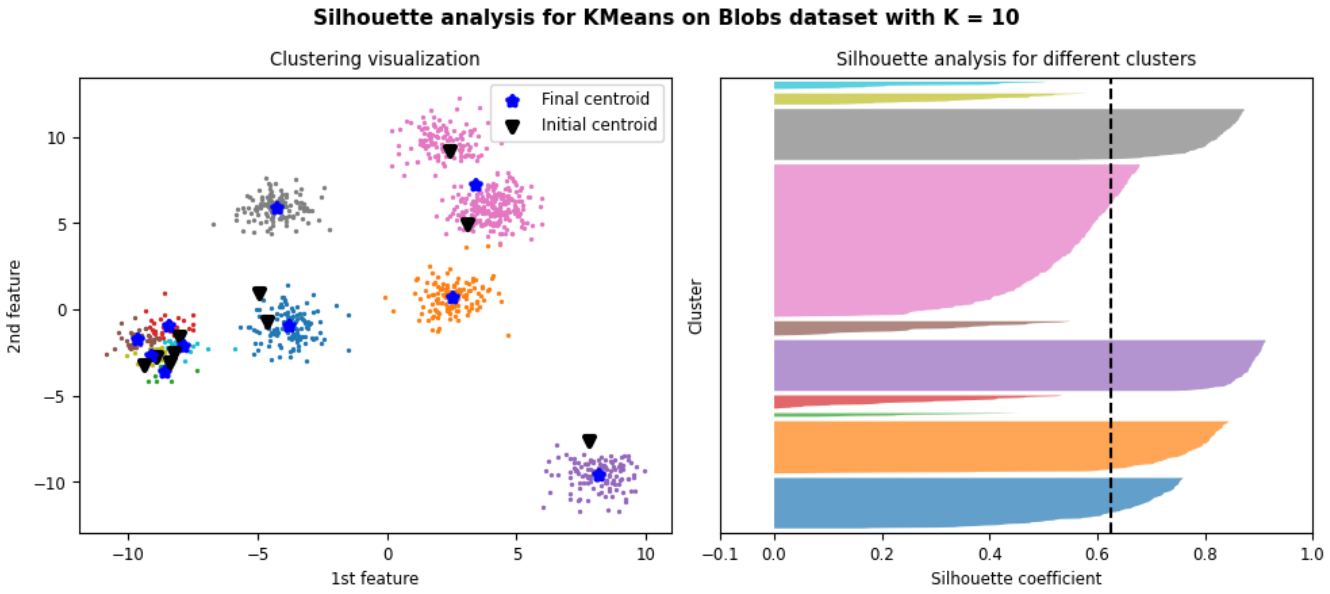


Figure 24: Results and silhouette analysis for K-Means on Blobs dataset with K = 10

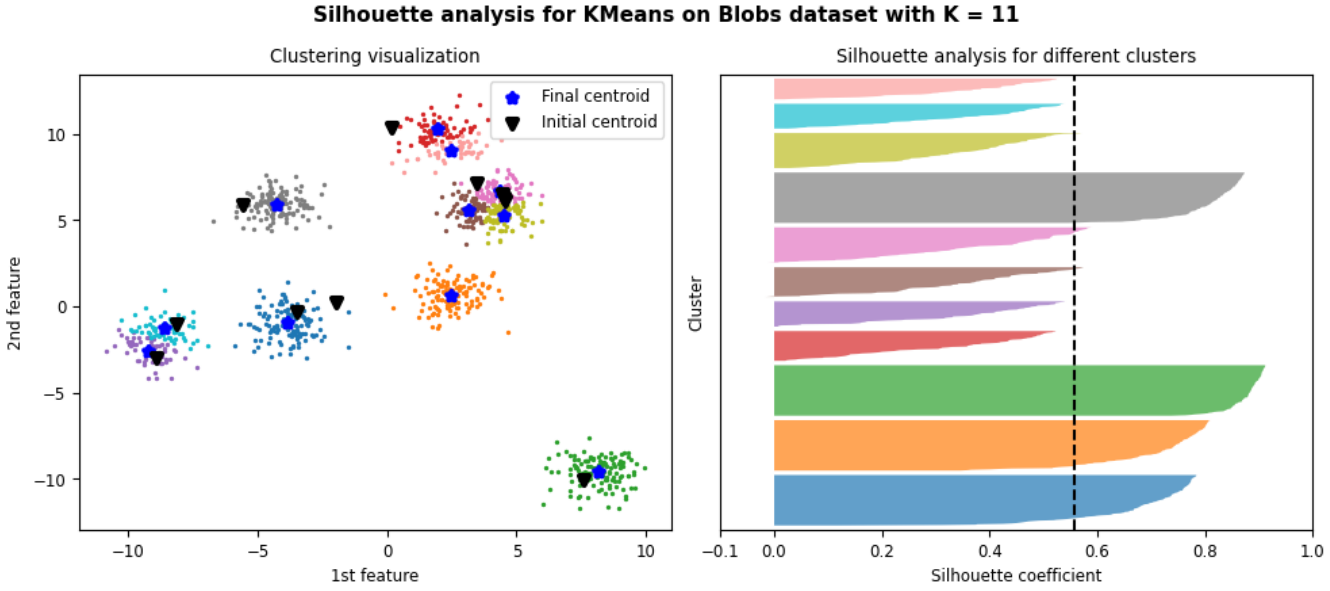


Figure 25: Results and silhouette analysis for K-Means on Blobs dataset with K = 11

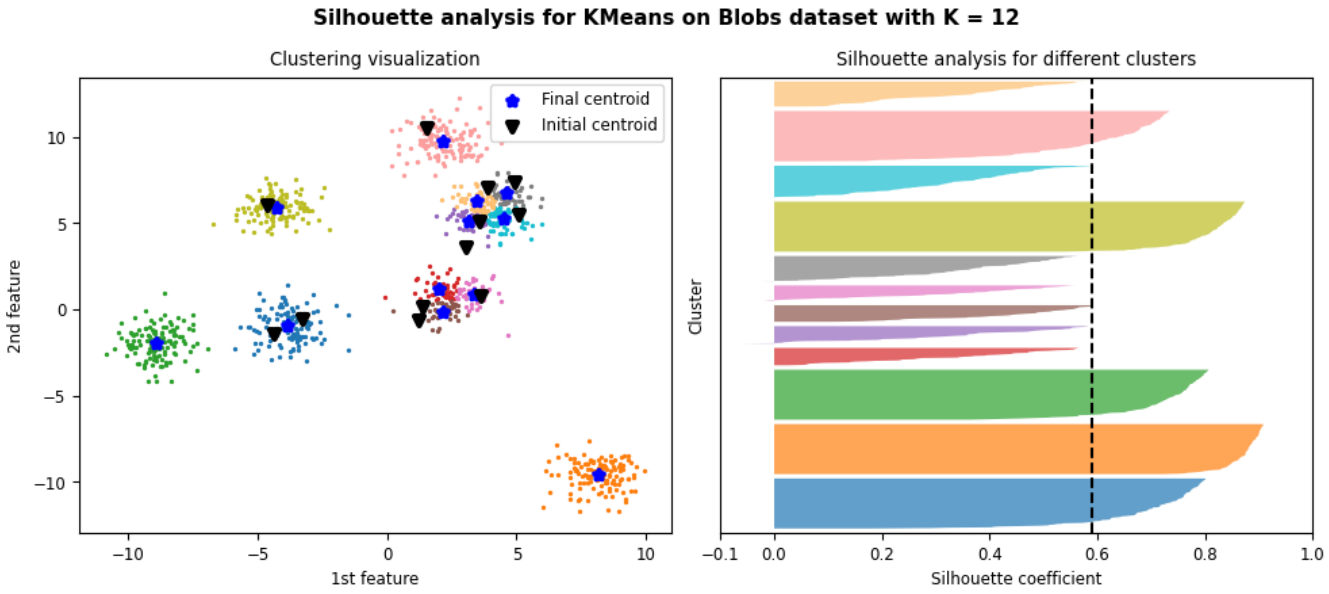


Figure 26: Results and silhouette analysis for K-Means on Blobs dataset with K = 12

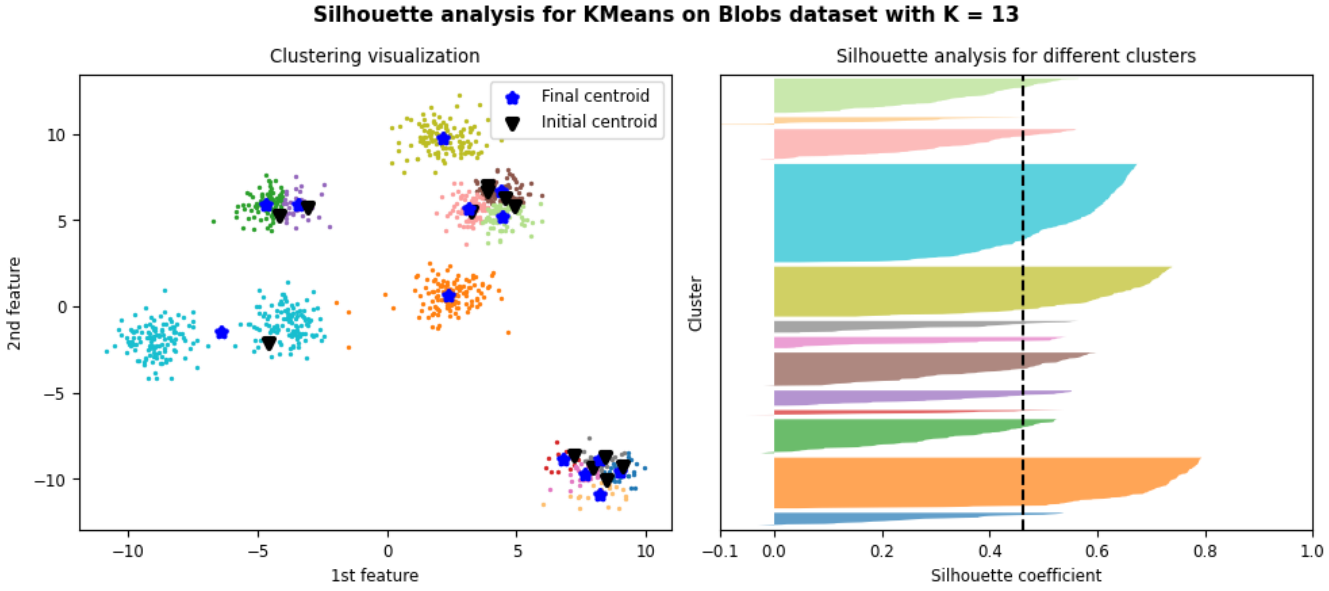


Figure 27: Results and silhouette analysis for K-Means on Blobs dataset with K = 13

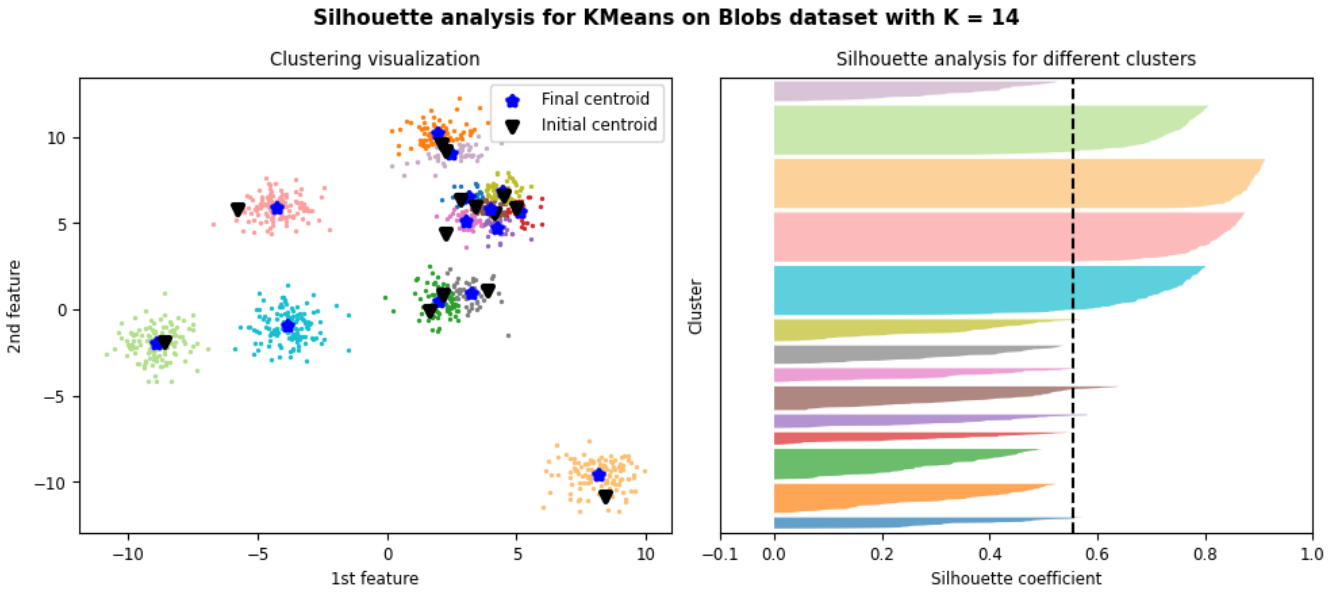


Figure 28: Results and silhouette analysis for K-Means on Blobs dataset with K = 14

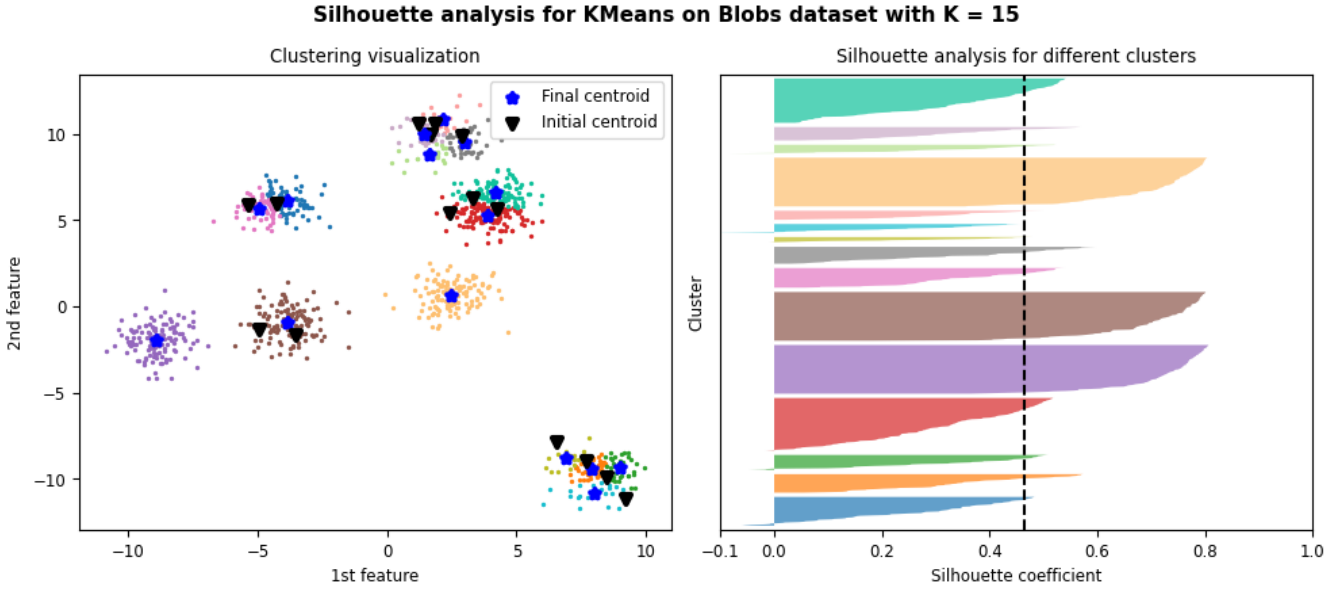
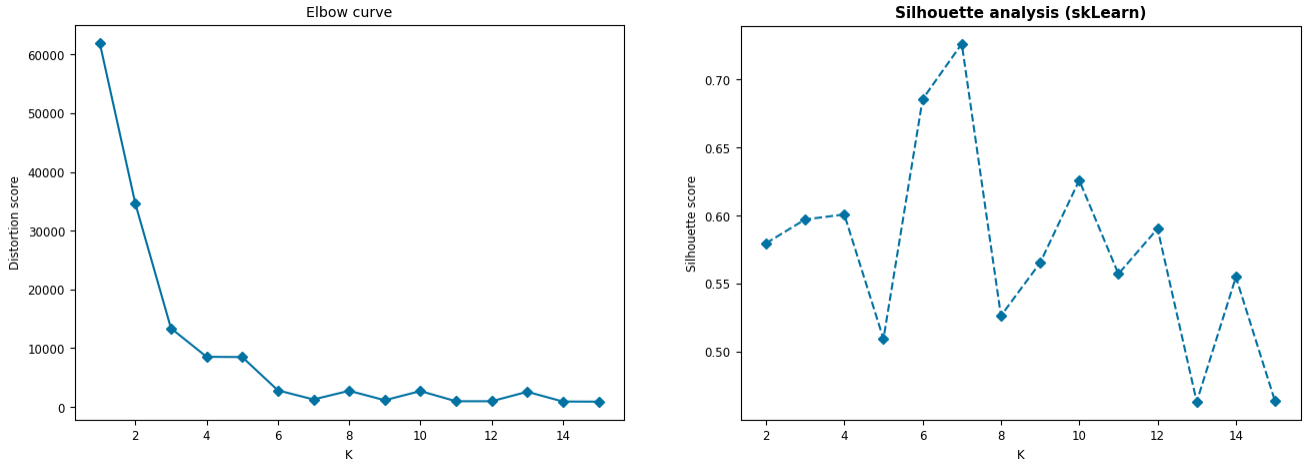


Figure 29: Results and silhouette analysis for K-Means on Blobs dataset with K = 15

2.2.2.2 Choosing the best K



(a) *(b)*

Figure 30: metrics result (a) elbow method. (b) Silhouette analysis

Using elbow method, we can see the bend in broken in K = 4, so it can a sub optimal number of clusters for this dataset but by looking at plots it seems it's not the best choice.

According to Figure 21, the choice of K = 7 can be one of best choices since it stands well against all the three measuring criteria for silhouette analysis where all clusters' plot is beyond average Silhouette score, with mostly uniform thickness and do not have wide fluctuations in the size. It also has the highest average Silhouette score.

Choosing the number of clusters was much better using Silhouette score. Clustering using k-means algorithm work pretty well for blobs dataset.

2.2.3 Elliptical dataset

2.2.3.1 Clustering results and silhouette analysis

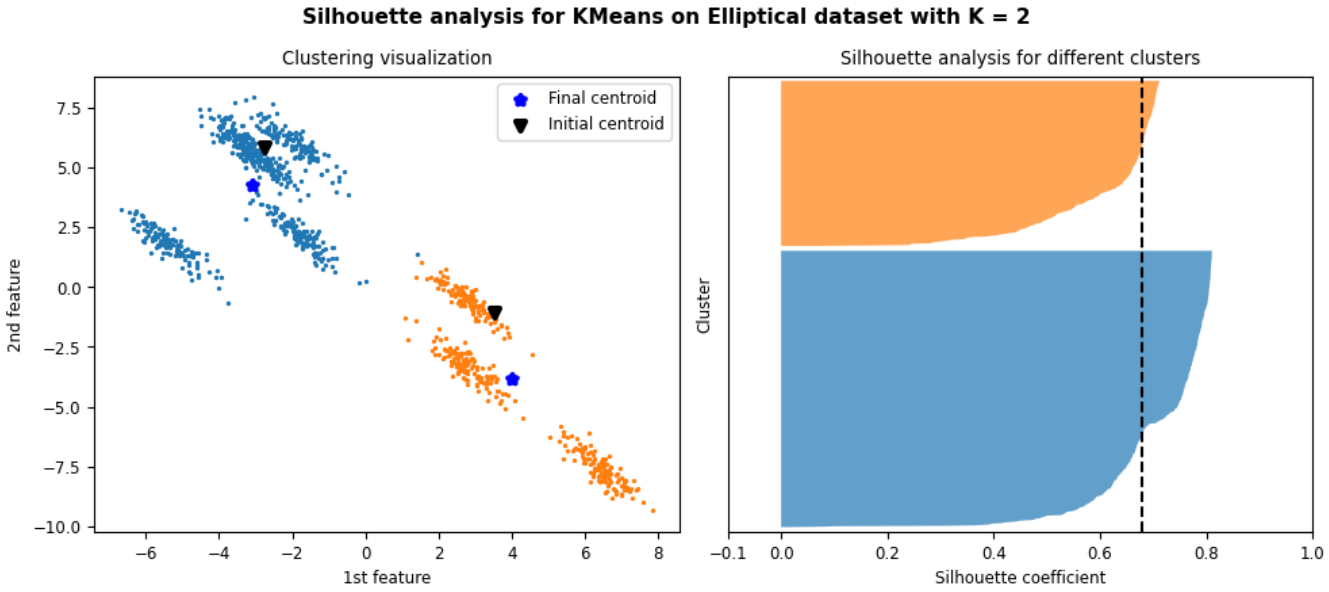


Figure 31: Results and silhouette analysis for K-Means on Elliptical dataset with K = 2

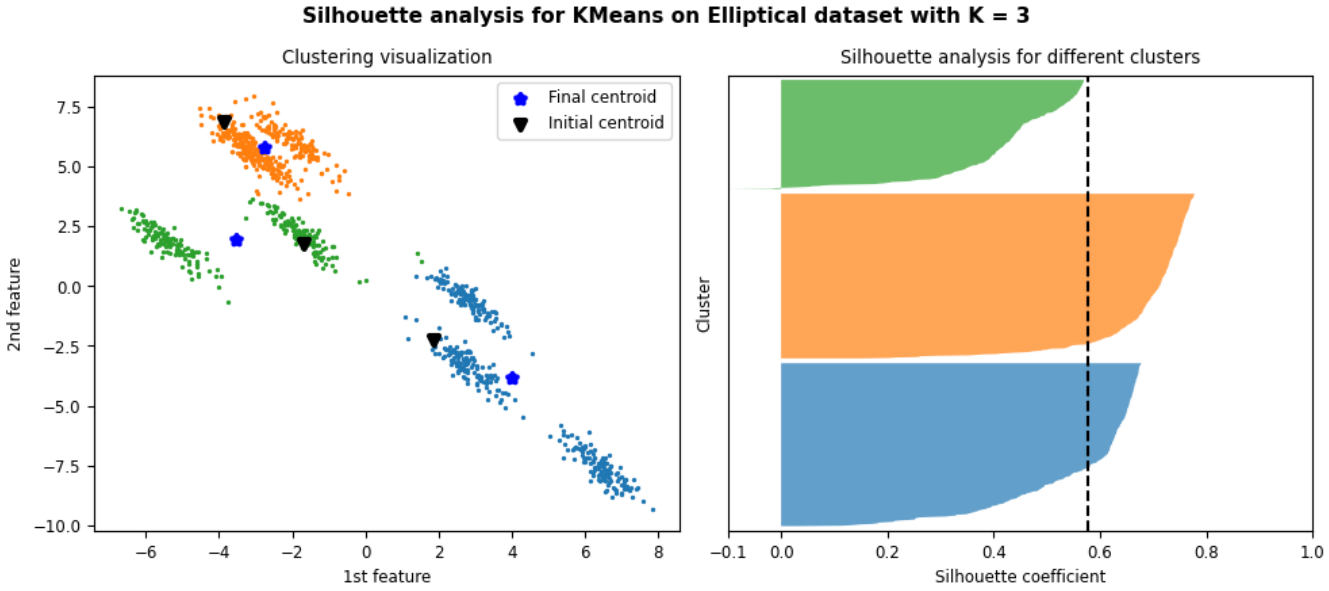


Figure 32: Results and silhouette analysis for K-Means on Elliptical dataset with K = 3

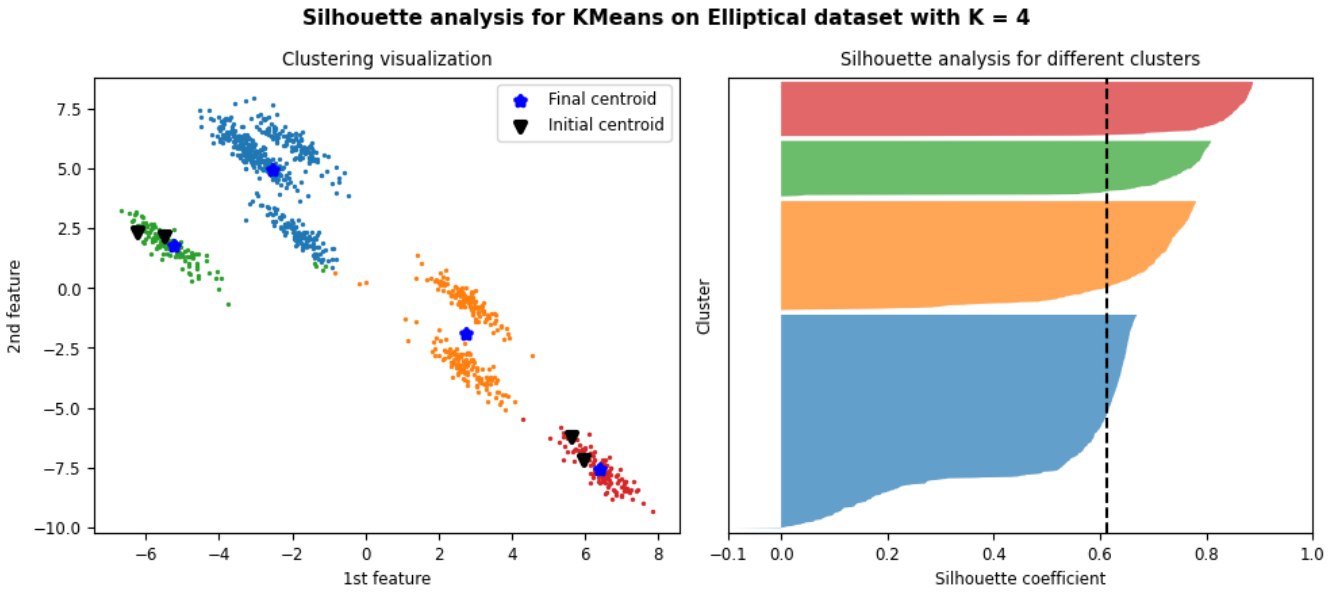


Figure 33: Results and silhouette analysis for K-Means on Elliptical dataset with K = 4

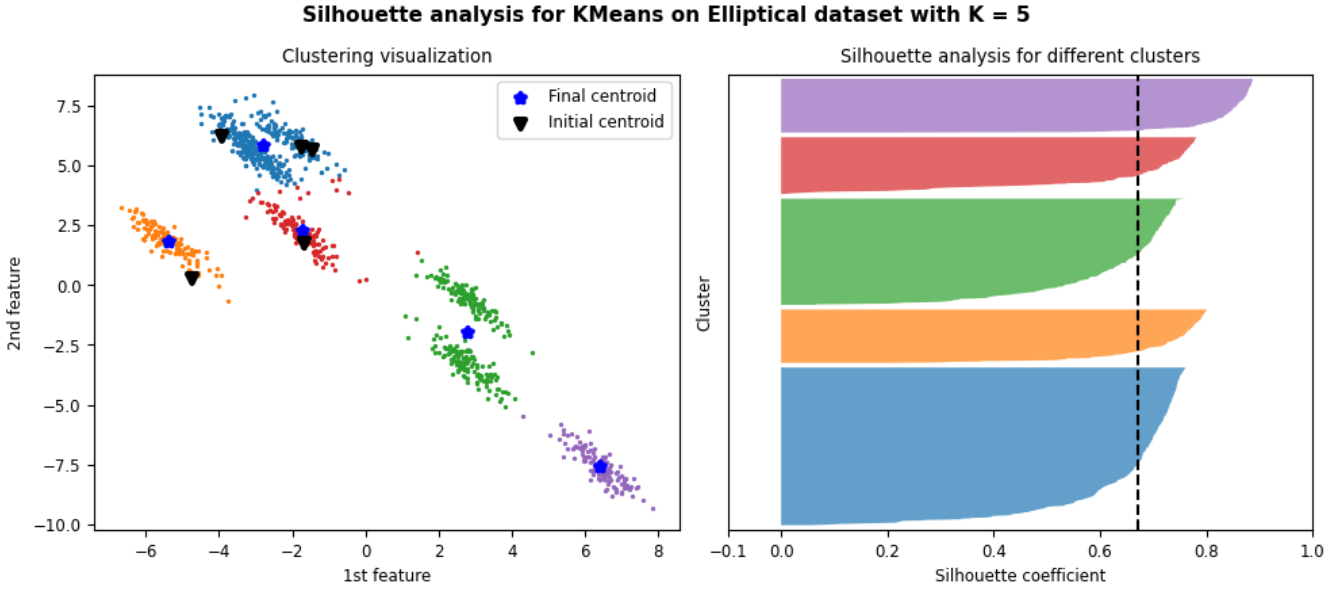


Figure 34: Results and silhouette analysis for K-Means on Elliptical dataset with K = 5

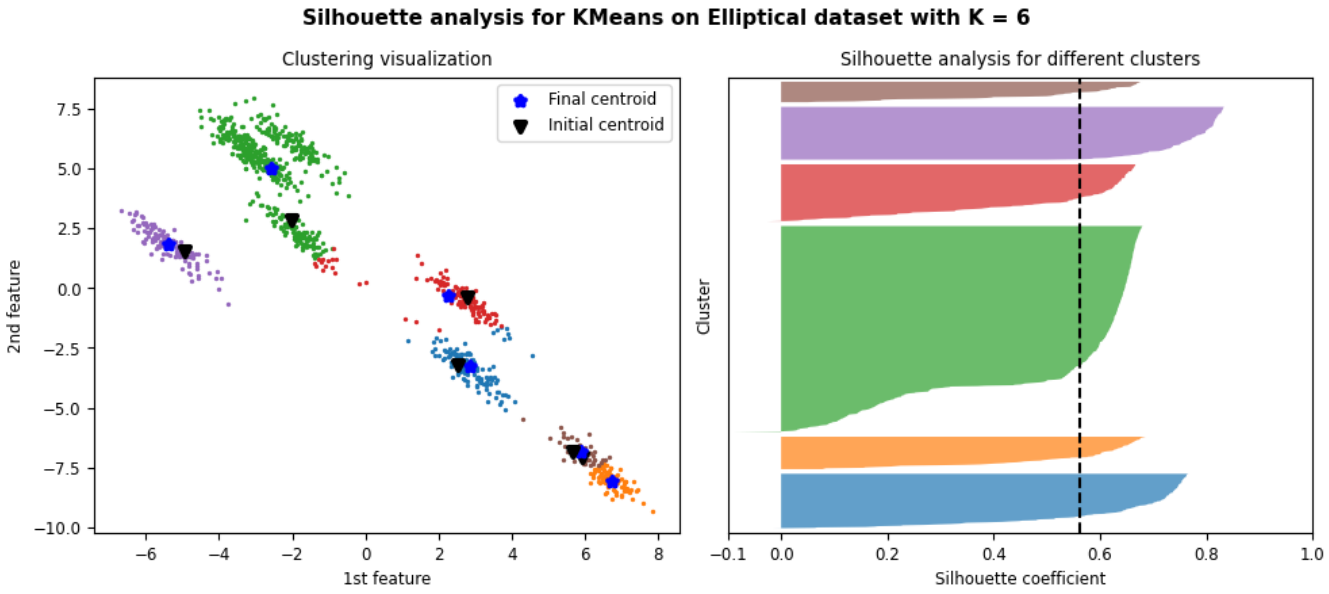


Figure 35: Results and silhouette analysis for K-Means on Elliptical dataset with K = 6

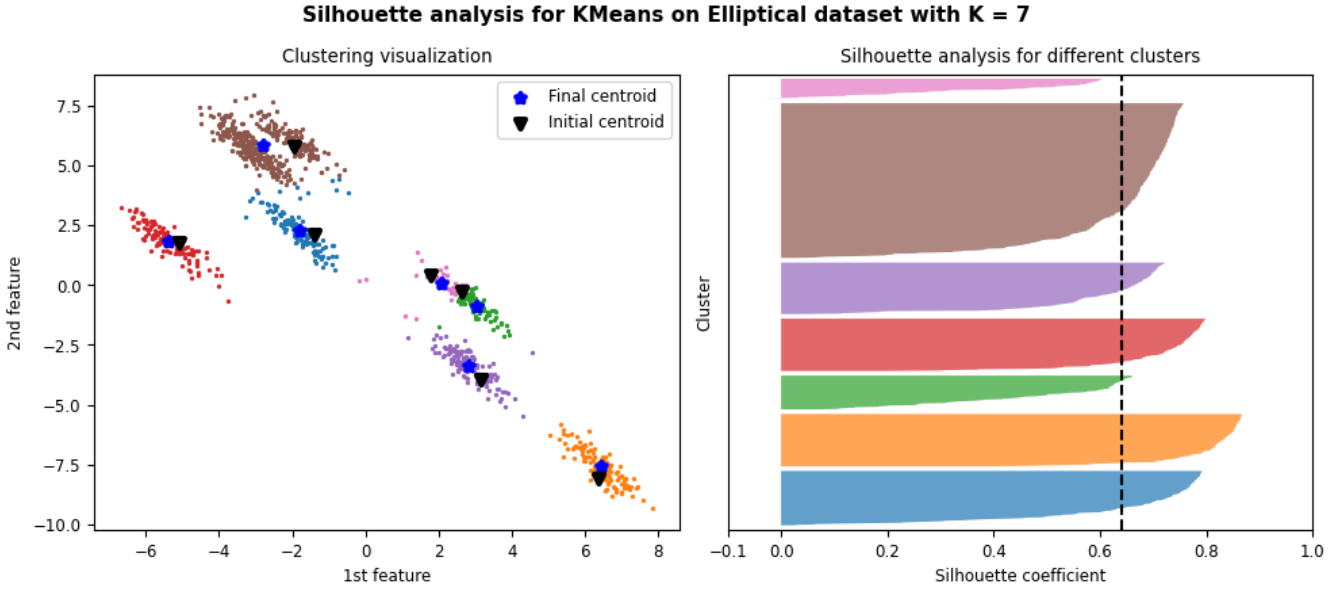


Figure 36: Results and silhouette analysis for K-Means on Elliptical dataset with K = 7

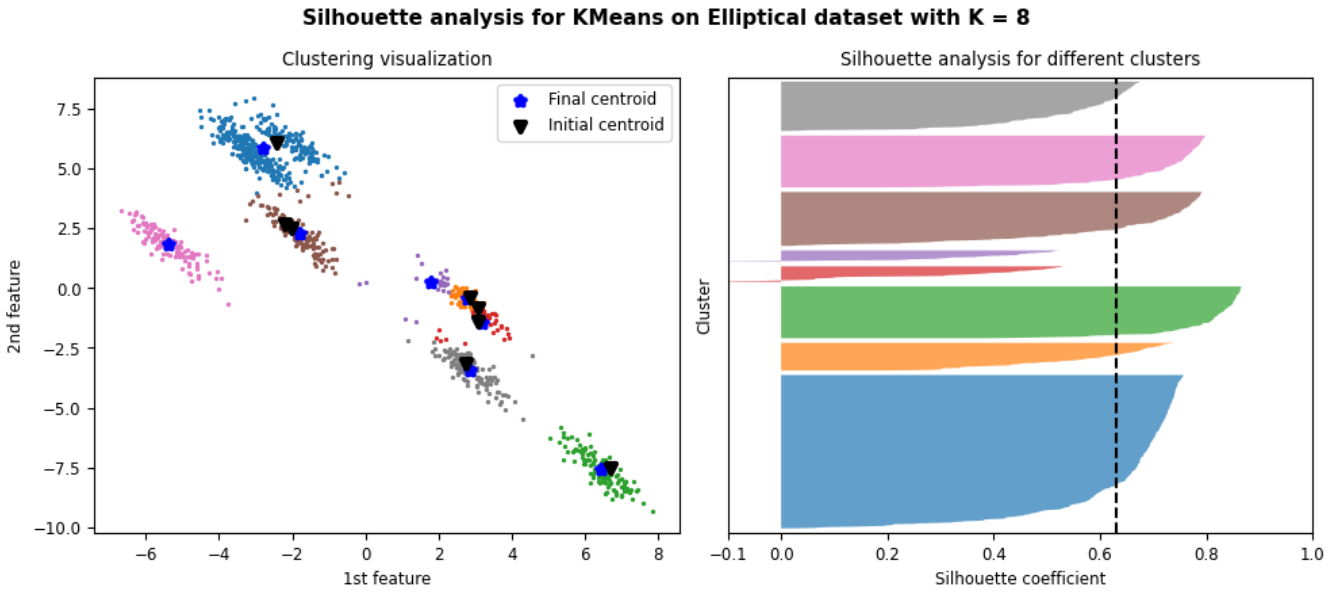


Figure 37: Results and silhouette analysis for K-Means on Elliptical dataset with K = 8

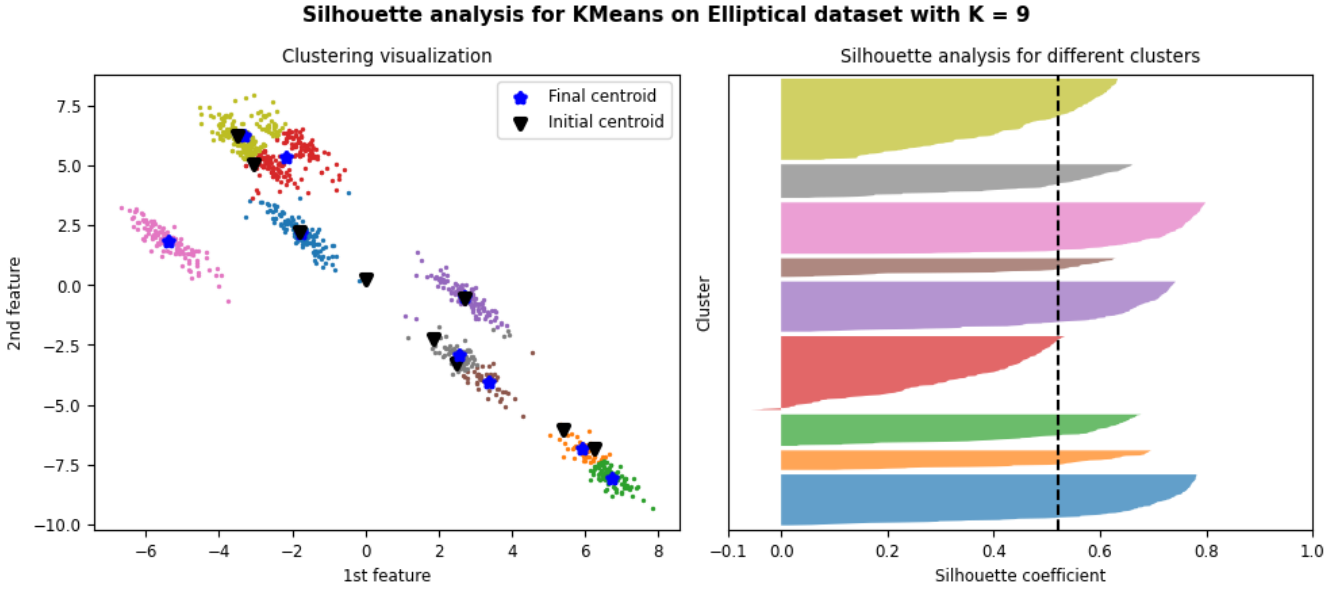


Figure 38: Results and silhouette analysis for K-Means on Elliptical dataset with K = 9

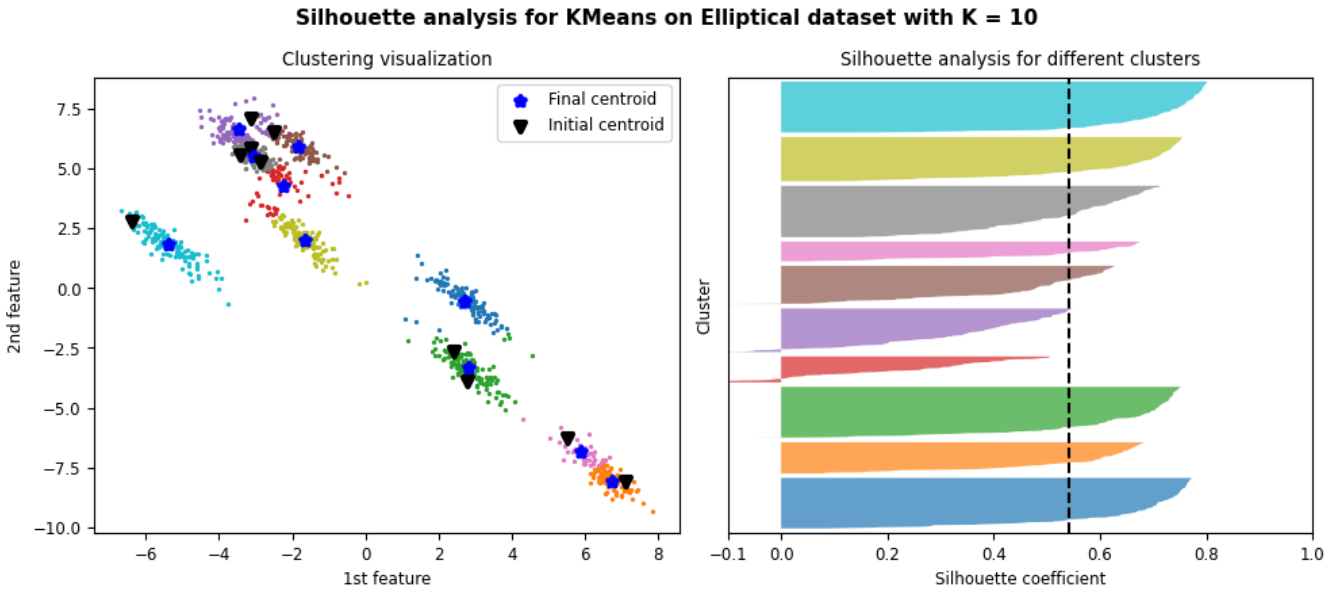


Figure 39: Results and silhouette analysis for K-Means on Elliptical dataset with K = 10

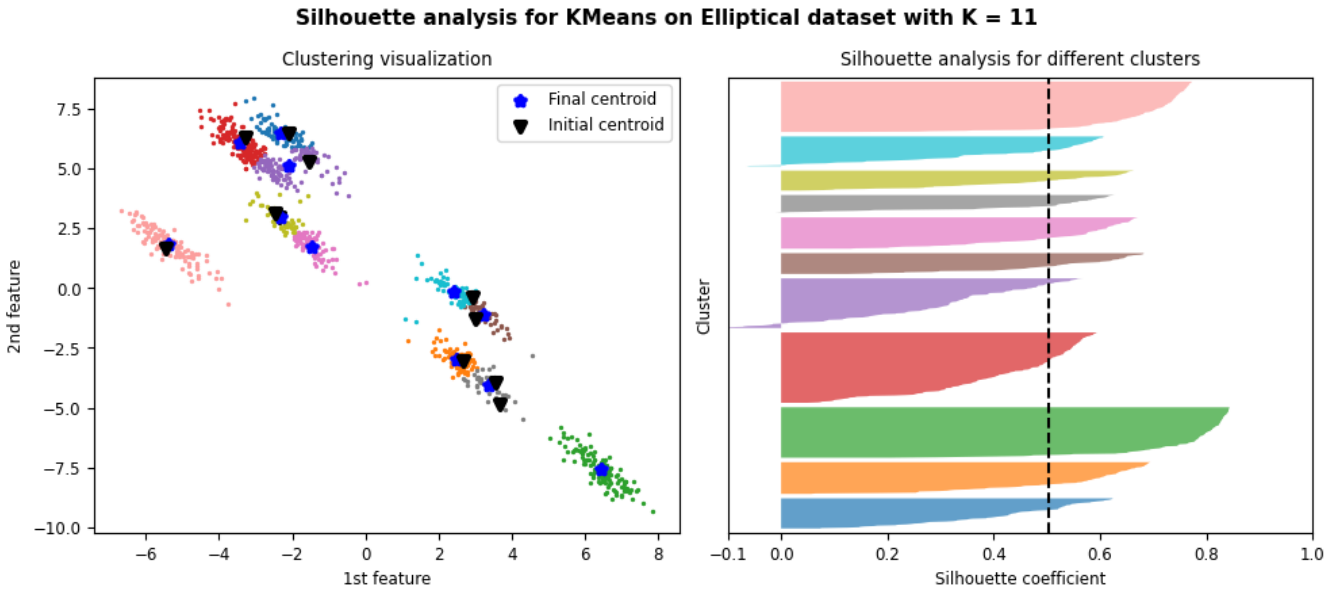


Figure 40: Results and silhouette analysis for K-Means on Elliptical dataset with K = 11

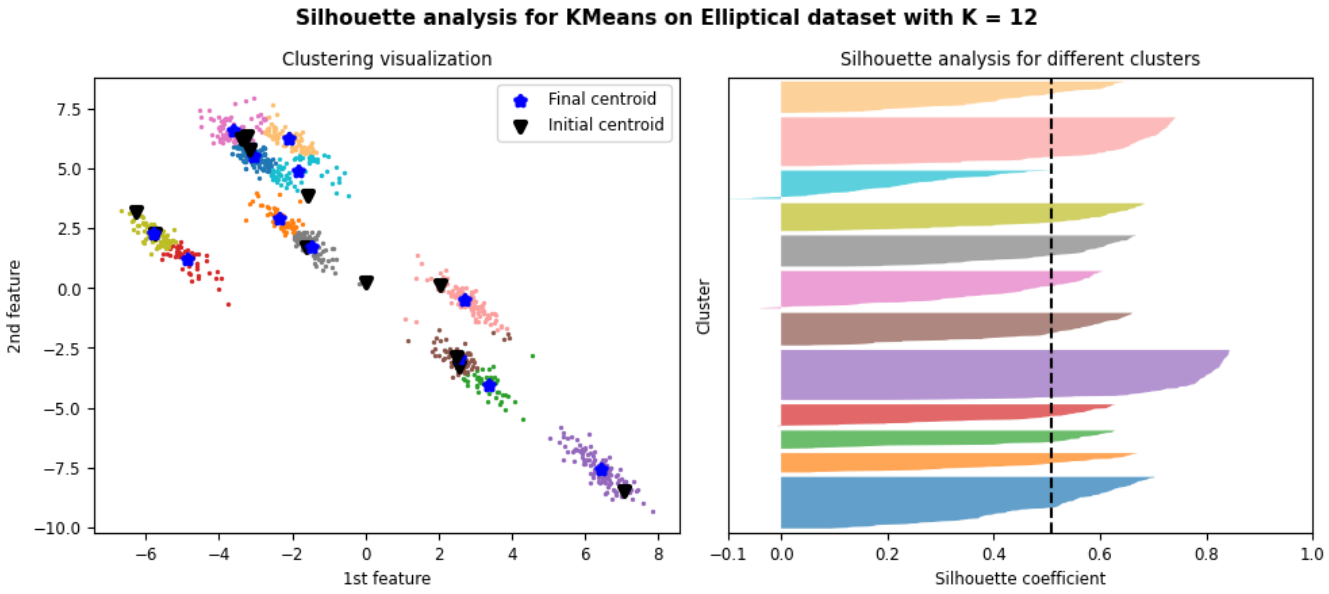


Figure 41: Results and silhouette analysis for K-Means on Elliptical dataset with K = 12

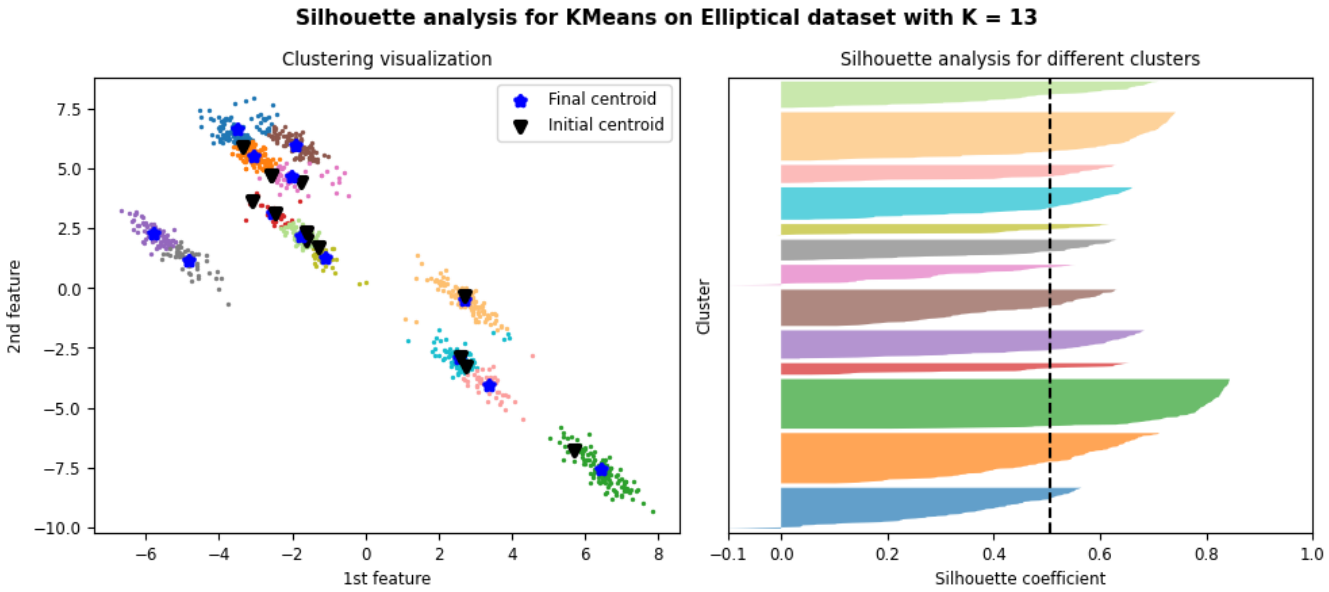


Figure 42: Results and silhouette analysis for K-Means on Elliptical dataset with K = 13

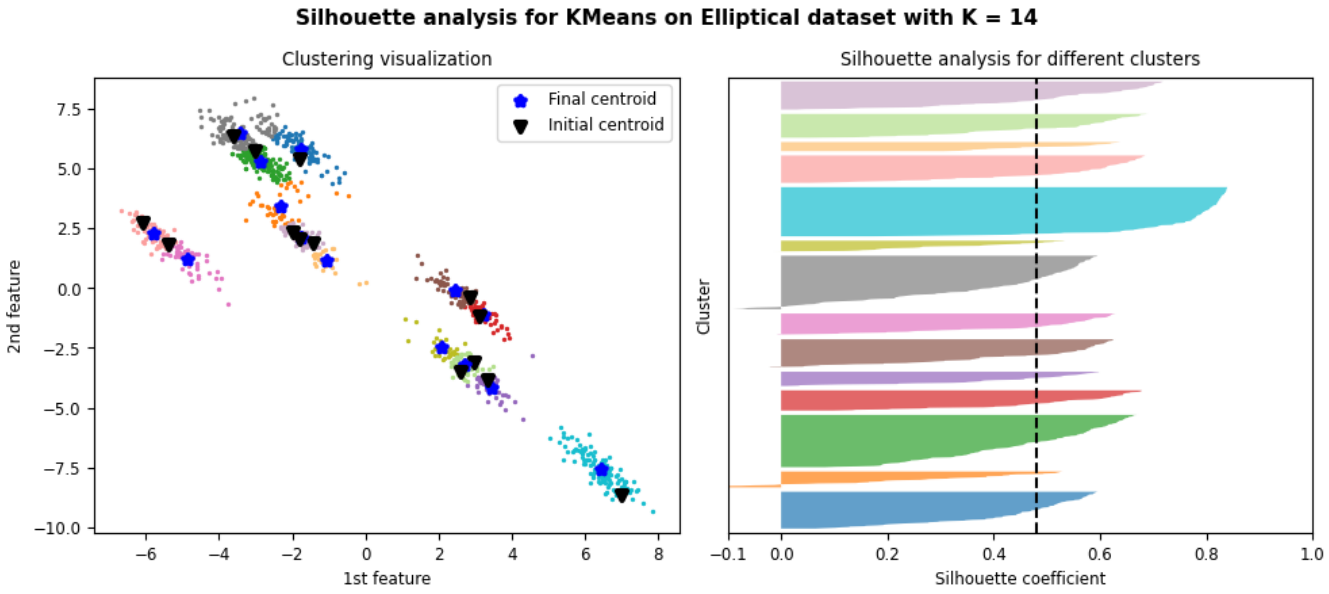


Figure 43: Results and silhouette analysis for K-Means on Elliptical dataset with K = 14

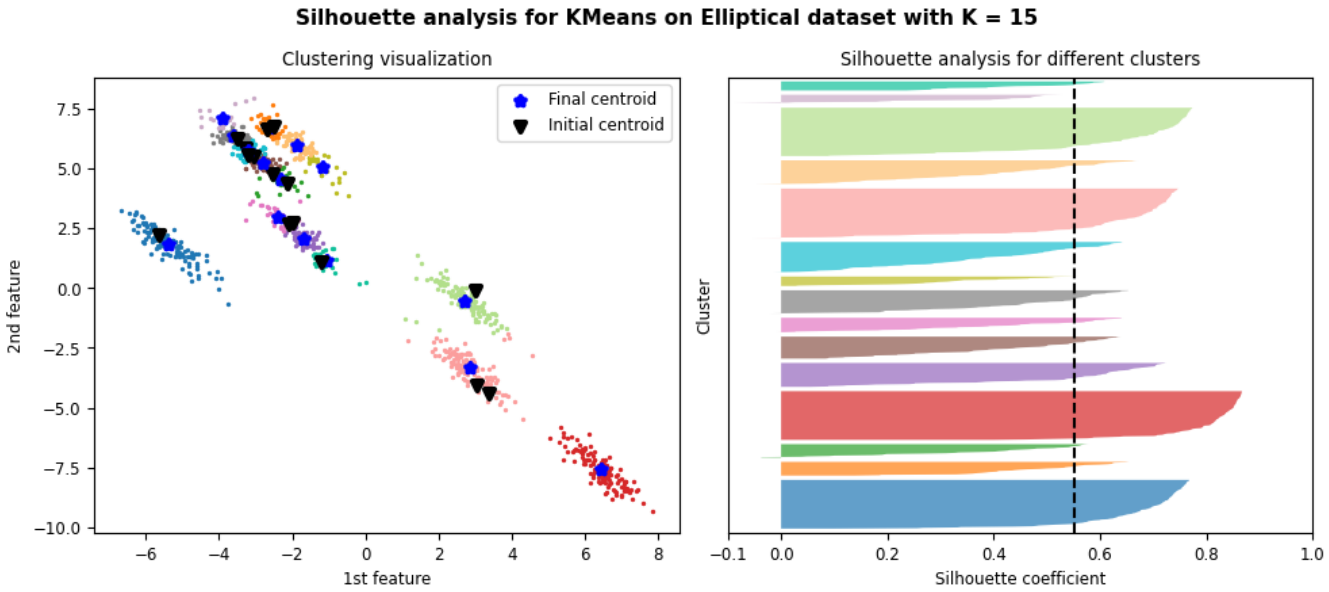
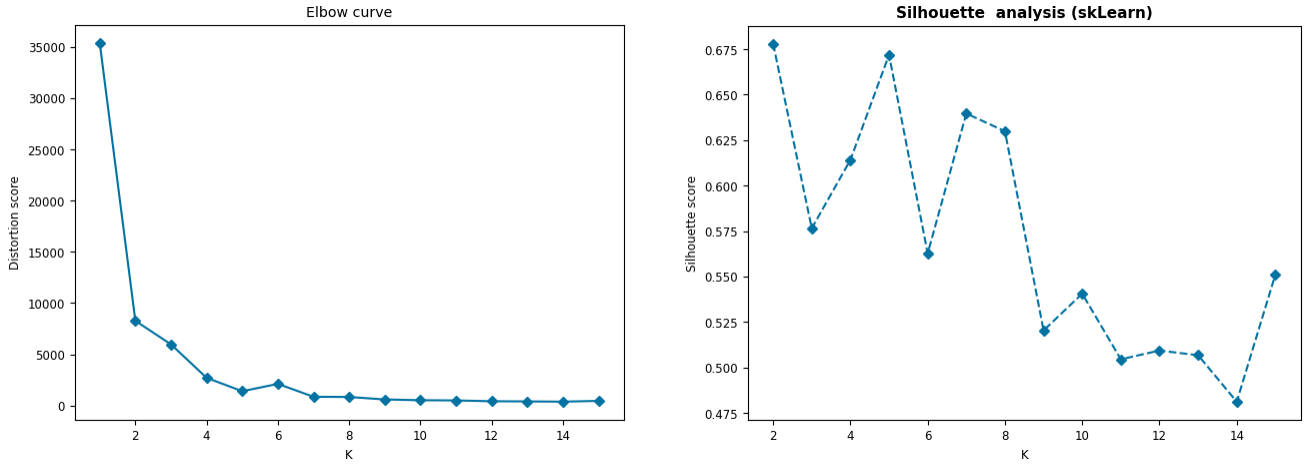


Figure 44: Results and silhouette analysis for K-Means on Elliptical dataset with K = 15

2.2.3.2 Choosing the best K



(a) *(b)*

Figure 45: metrics result (a) elbow method. (b) Silhouette analysis

Using elbow method, we can see the bend is broken in K = 5, so it can be chosen as the best number of clusters using this method.

Also using Silhouette score method and according to Figure 5 we can see K = 5 can be a sub-optimal choice since it does not satisfy only one criteria which is non-uniform thickness. We can see K = 5 has the highest average Silhouette score in figure above.

Choosing the number of clusters using both methods had pretty much same results for this dataset. Clustering using k-means algorithm can still work for this dataset, but by looking at the plot of dataset we concluded that since some of clusters are so close to each other, it can make it harder for k-means algorithm for perfect clustering but still not impossible.

2.2.4 Moon dataset

2.2.4.1 Clustering results and silhouette analysis

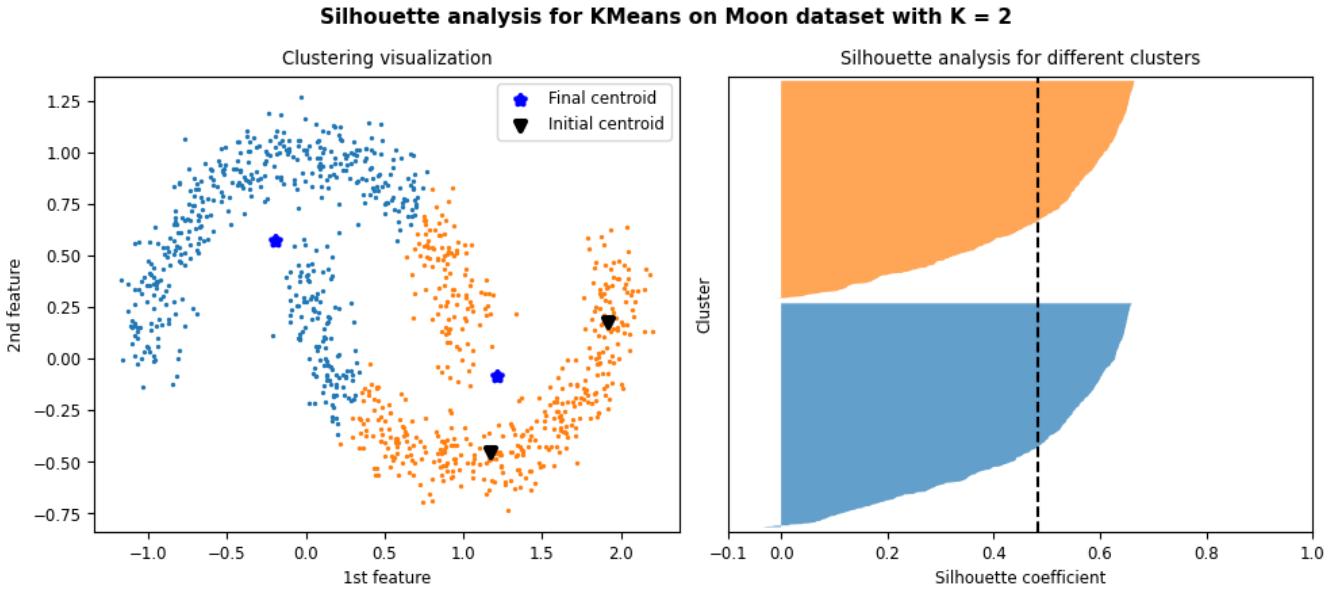


Figure 46: Results and silhouette analysis for K-Means on Moon dataset with K = 2

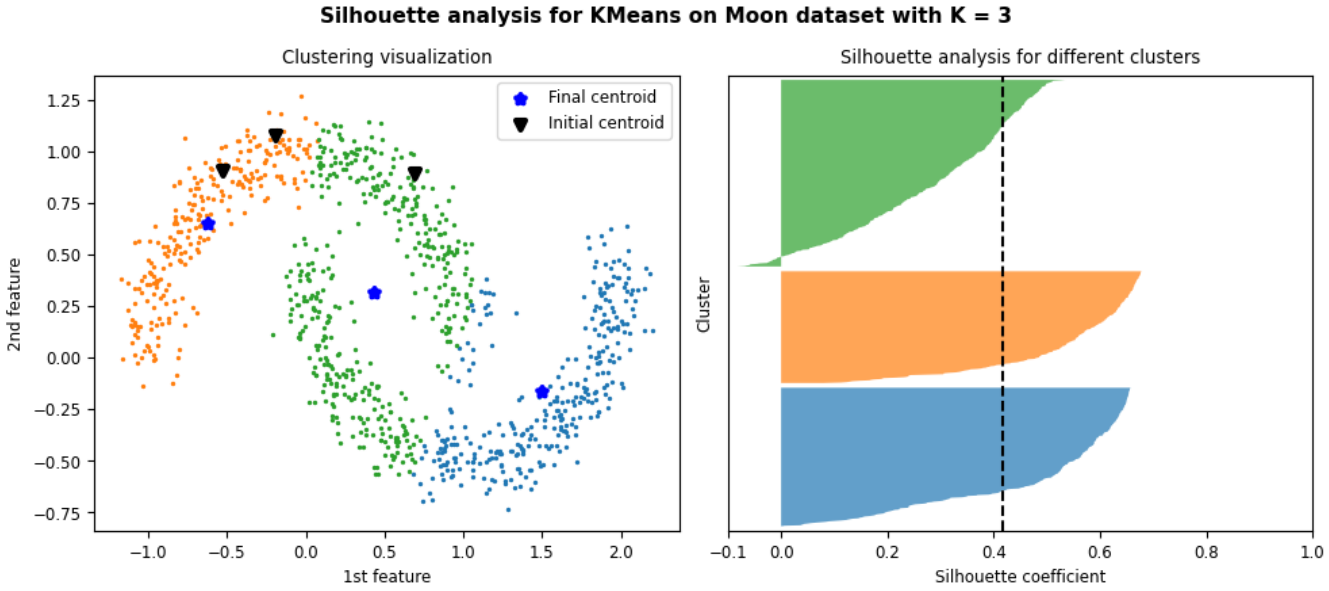


Figure 47: Results and silhouette analysis for K-Means on Moon dataset with K = 3

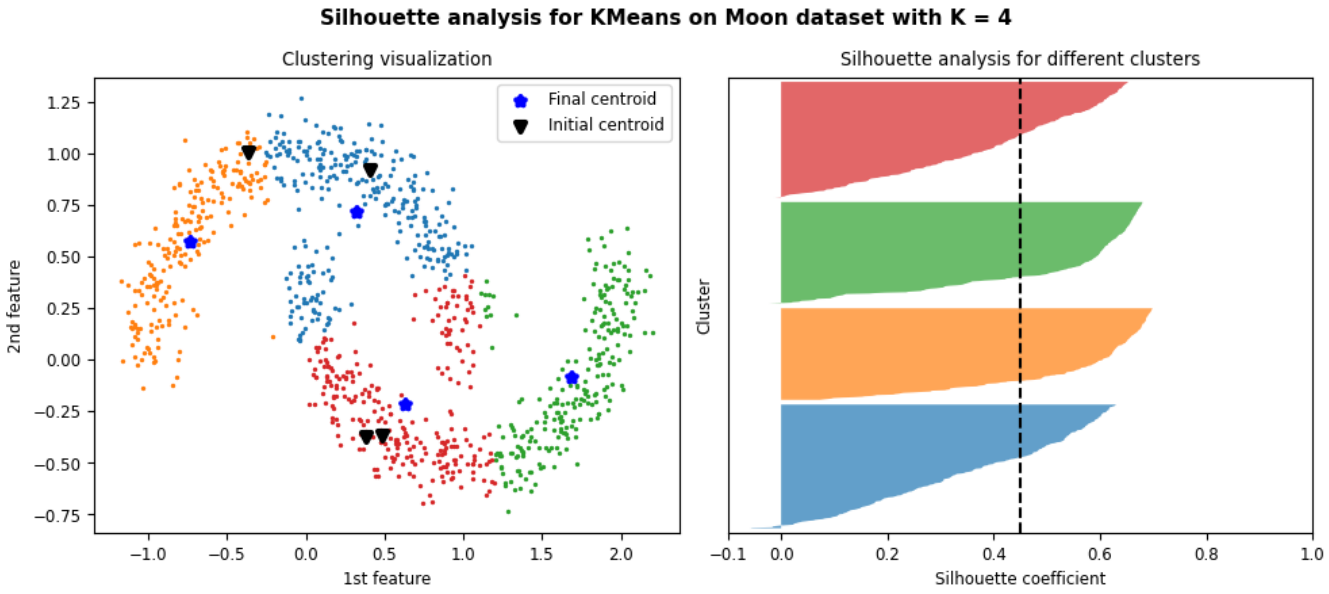


Figure 48: Results and silhouette analysis for K-Means on Moon dataset with K = 4

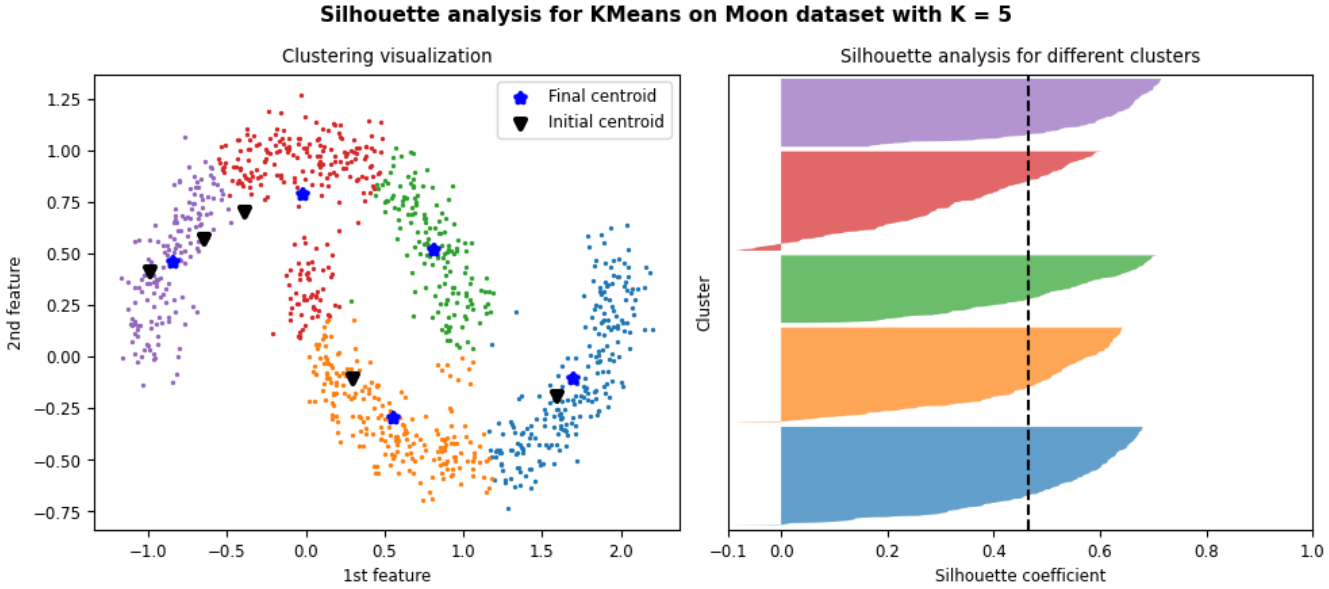


Figure 49: Results and silhouette analysis for K-Means on Moon dataset with K = 5

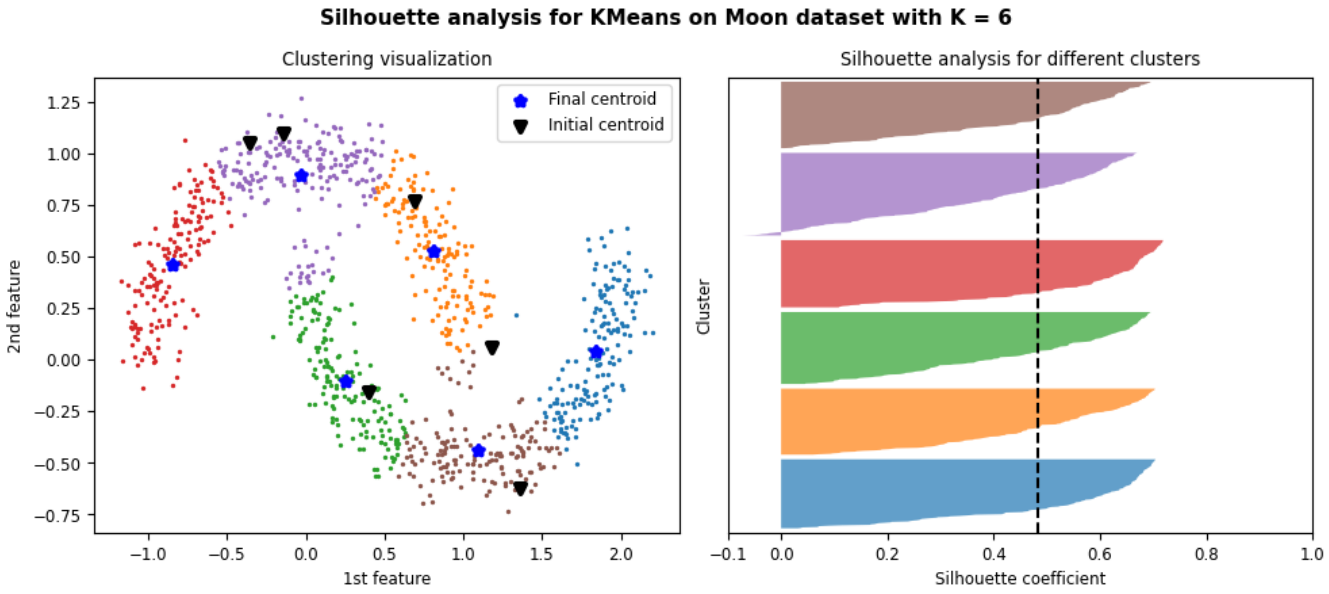


Figure 50: Results and silhouette analysis for K-Means on Moon dataset with K = 6

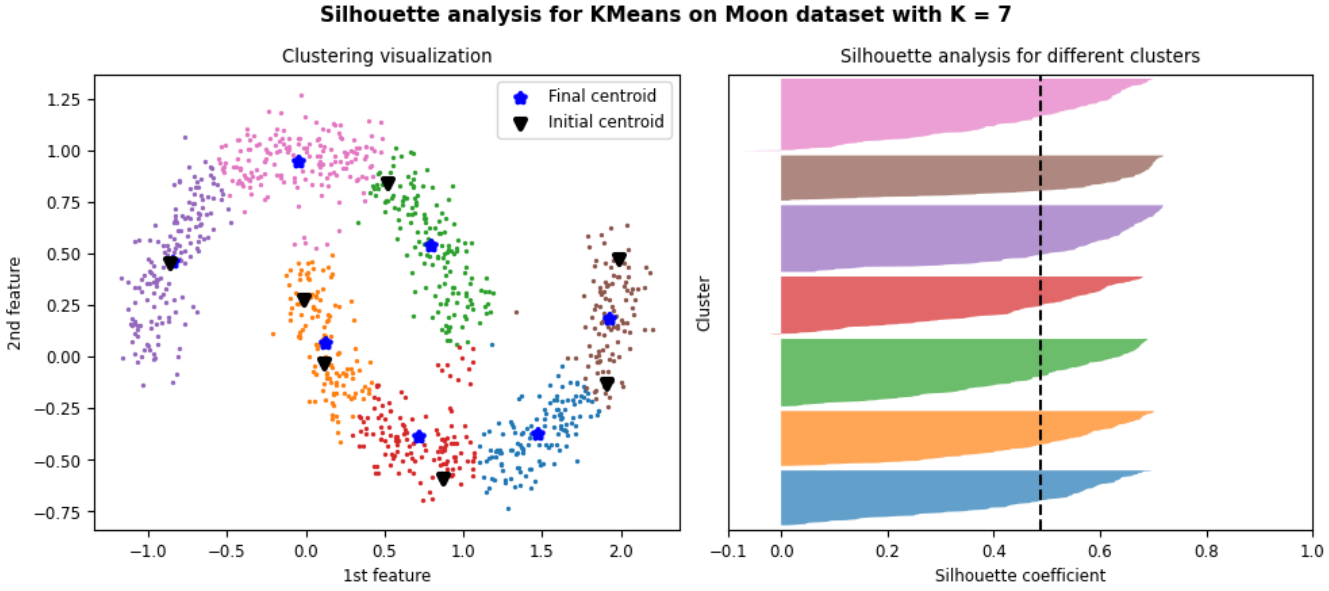


Figure 51: Results and silhouette analysis for K-Means on Moon dataset with K = 7

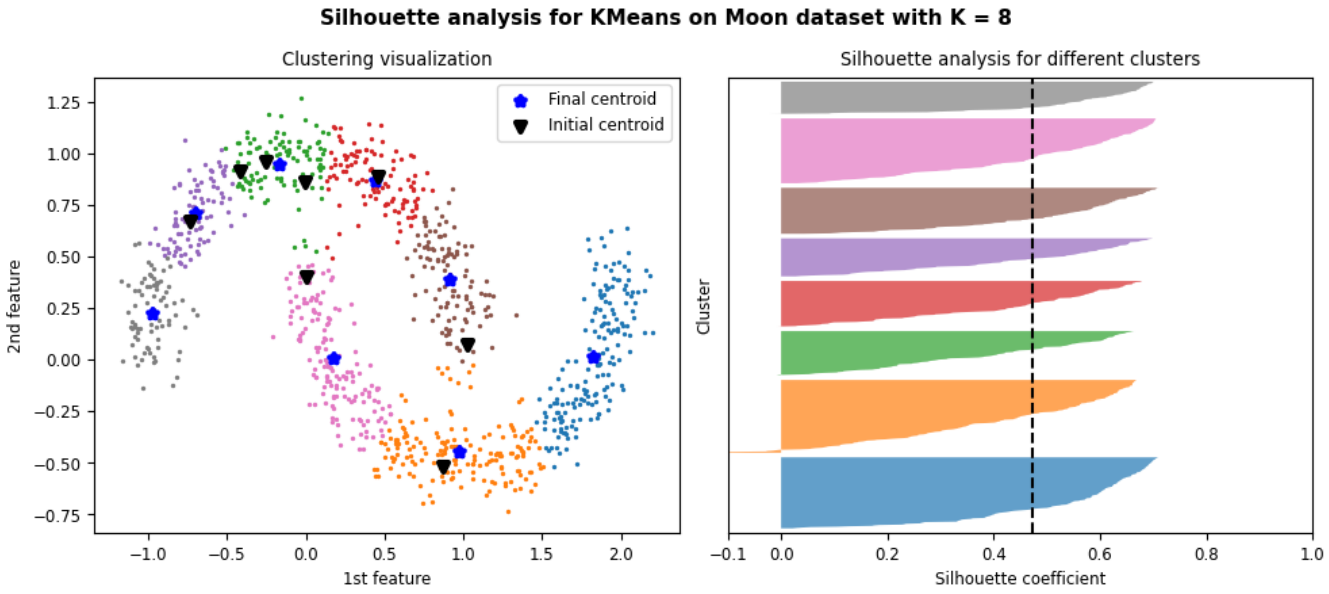


Figure 52: Results and silhouette analysis for K-Means on Moon dataset with K = 8

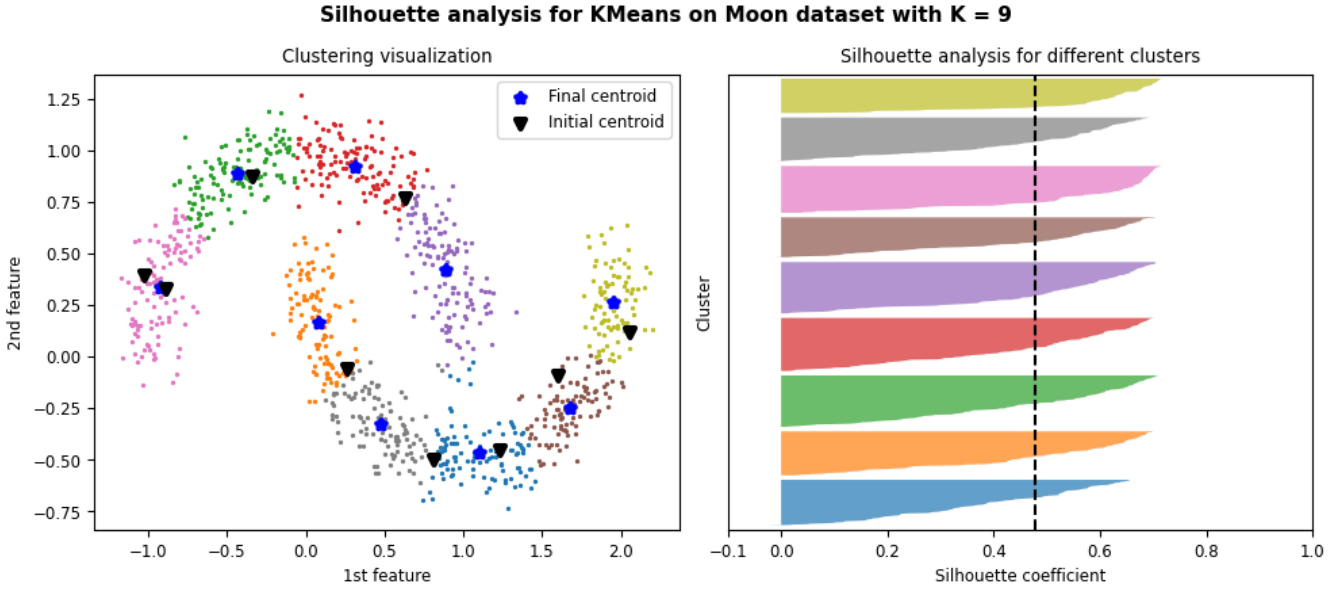


Figure 53: Results and silhouette analysis for K-Means on Moon dataset with K = 9

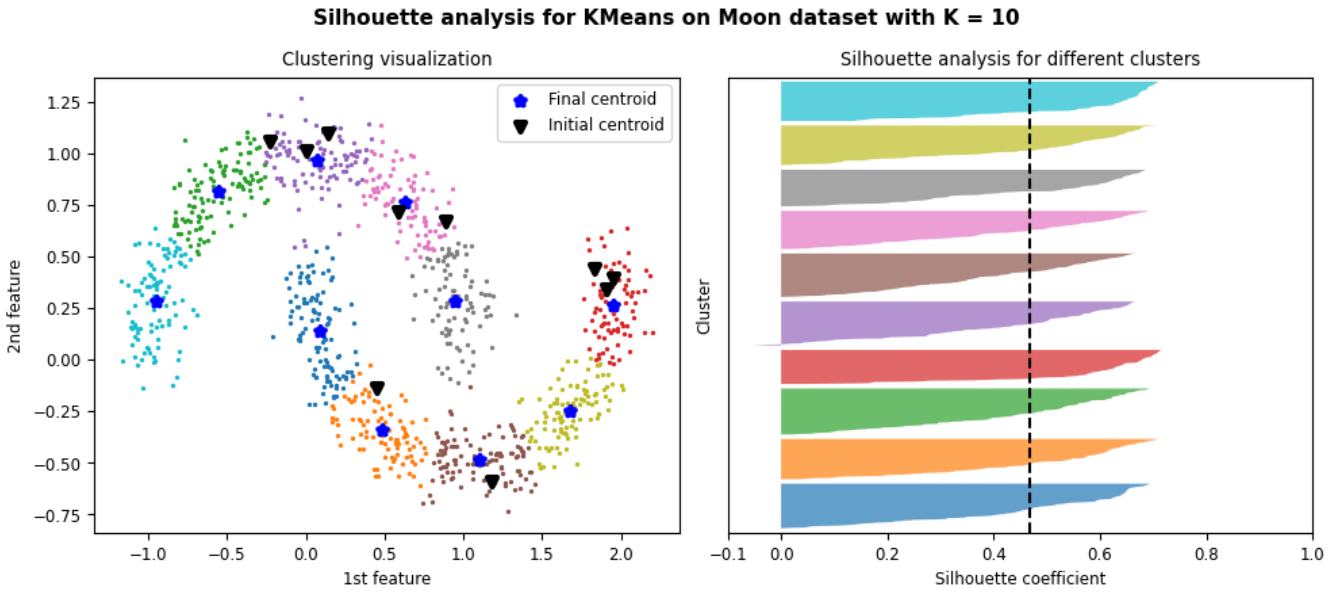


Figure 54: Results and silhouette analysis for K-Means on Moon dataset with K = 10

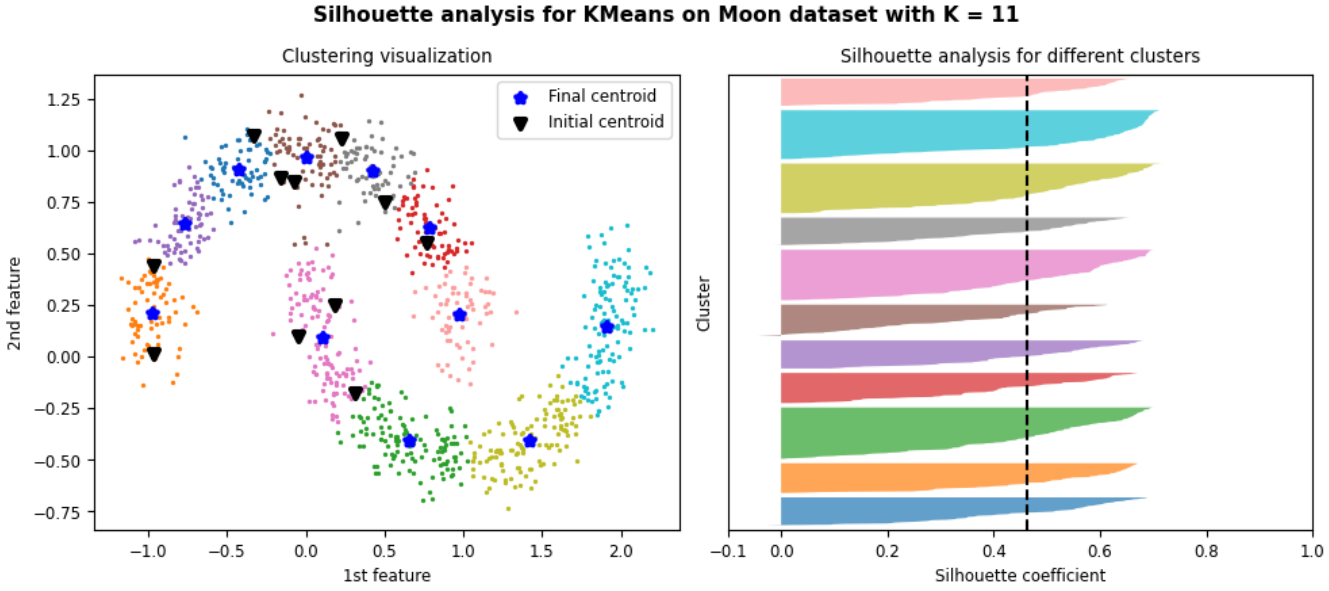


Figure 55: Results and silhouette analysis for K-Means on Moon dataset with K = 11

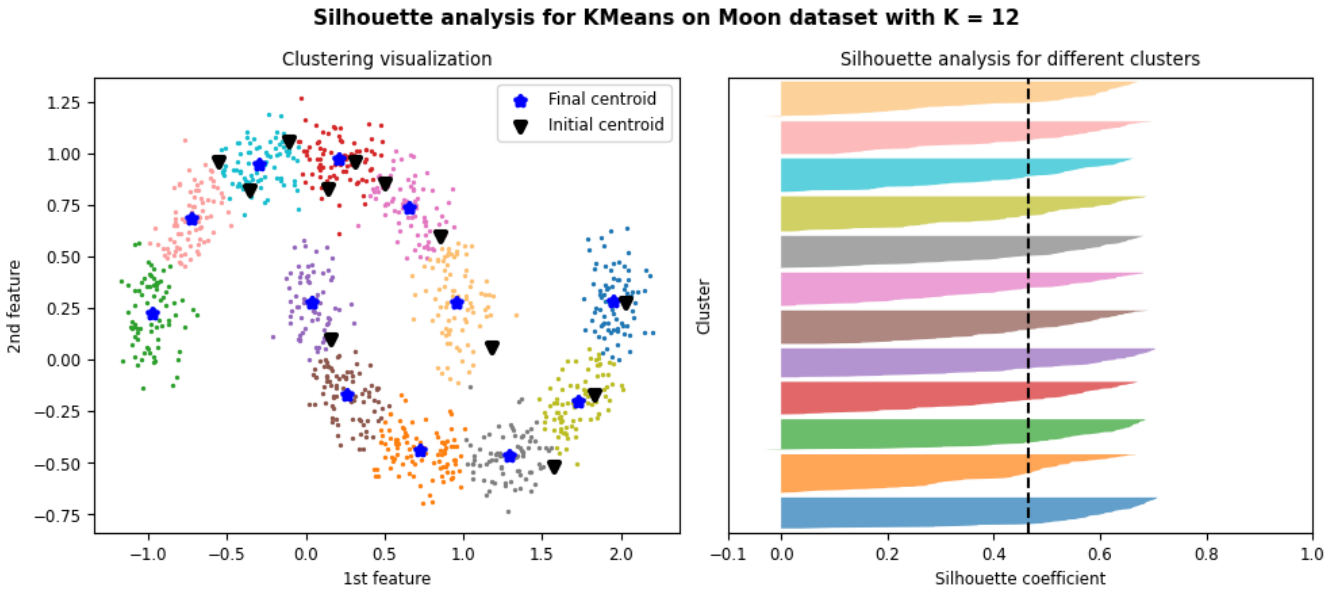


Figure 56: Results and silhouette analysis for K-Means on Moon dataset with K = 12

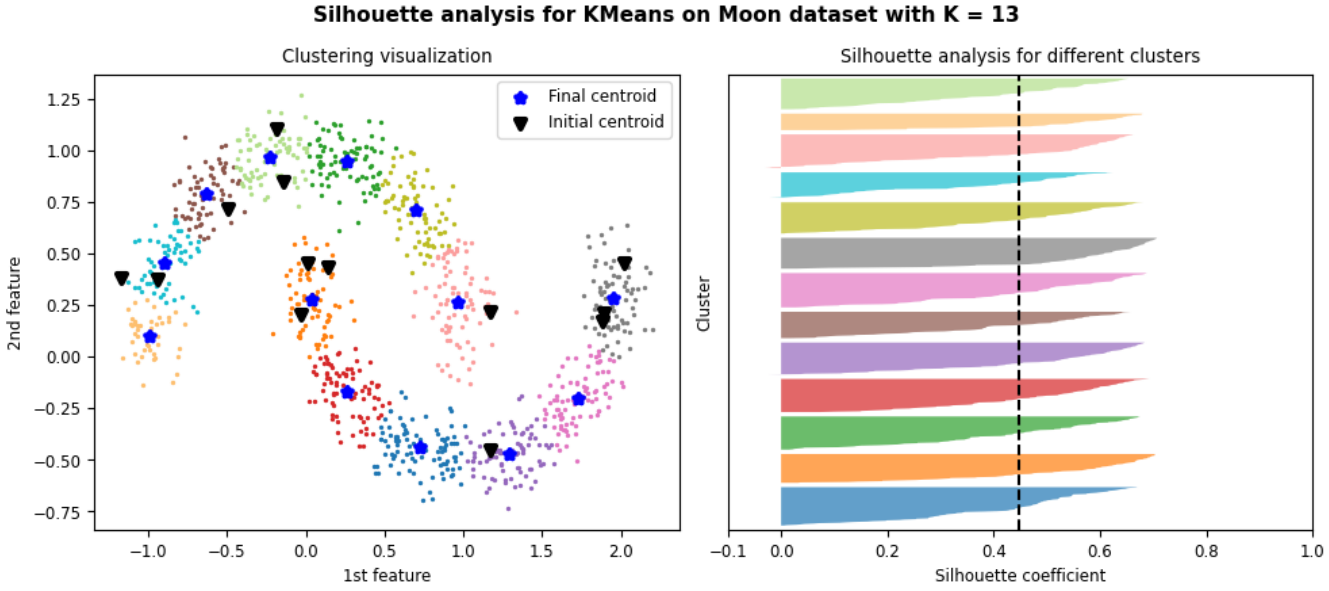


Figure 57: Results and silhouette analysis for K-Means on Moon dataset with K = 13

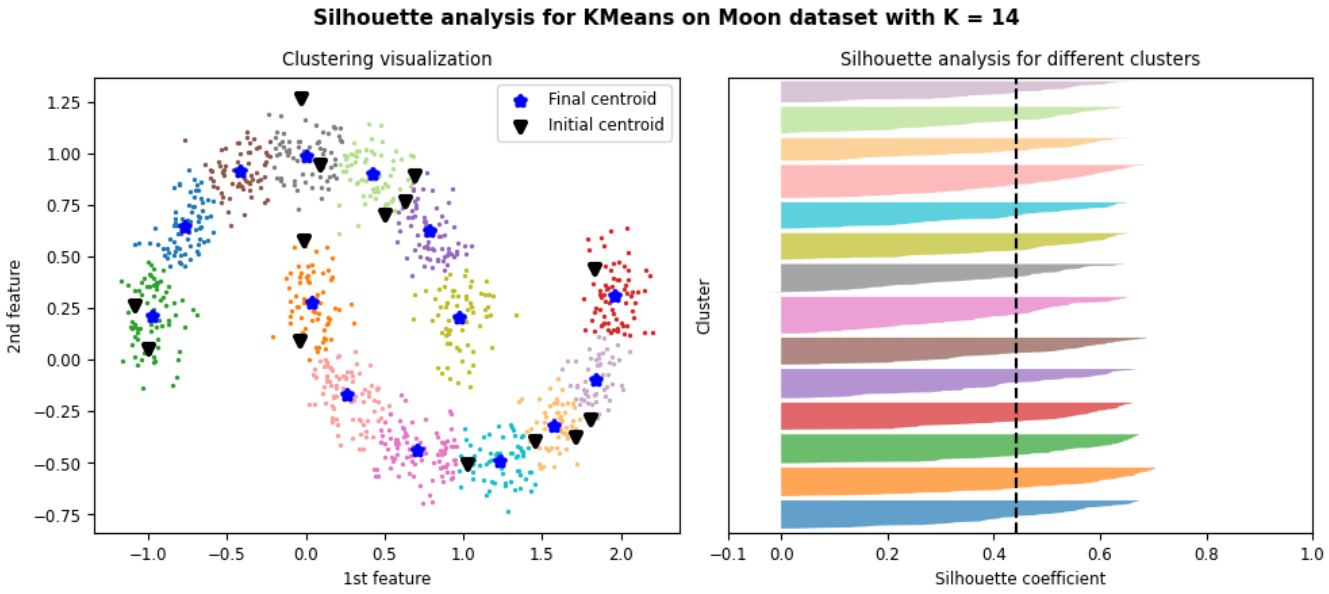


Figure 58: Results and silhouette analysis for K-Means on Moon dataset with K = 14

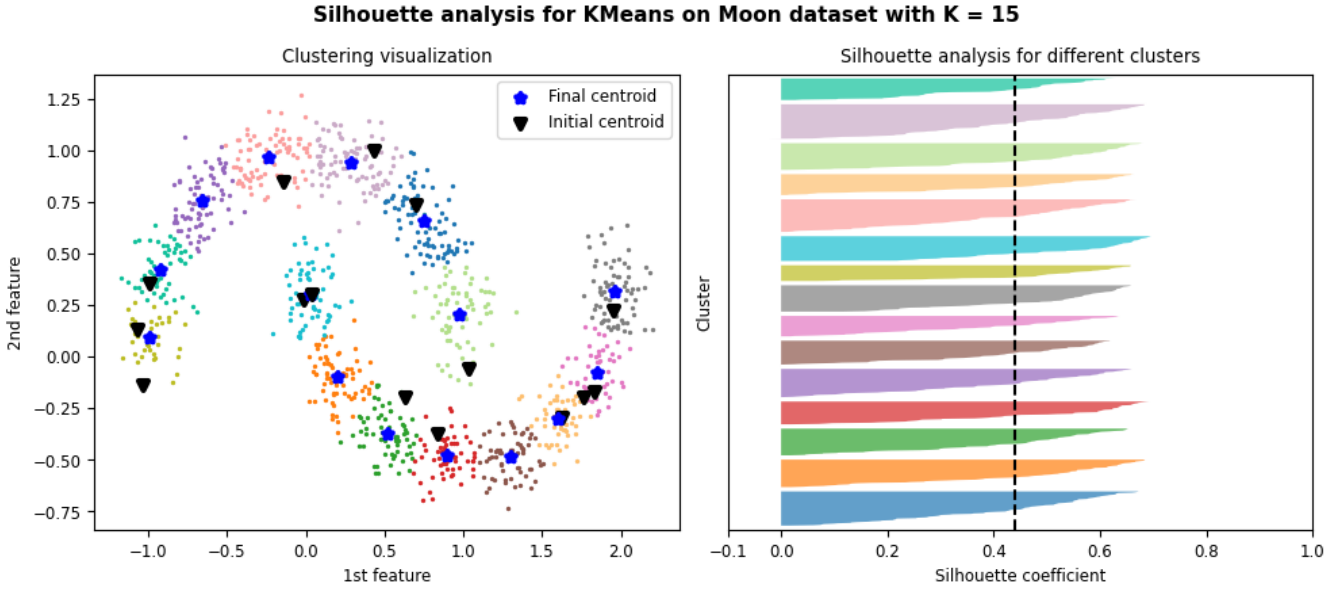
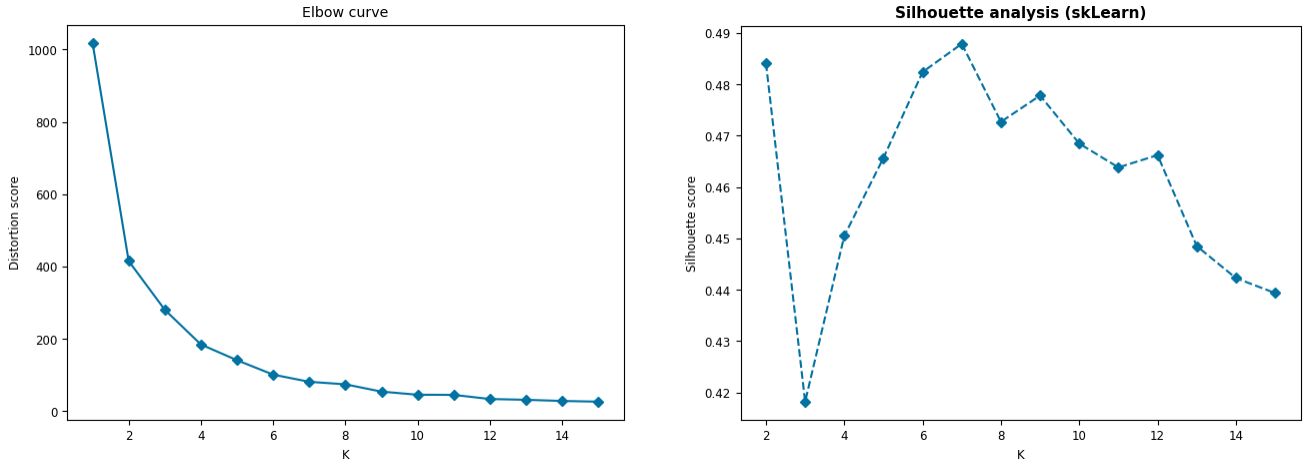


Figure 59: Results and silhouette analysis for K-Means on Moon dataset with K = 15

2.2.4.2 Choosing the best K



(a) *(b)*

Figure 60: metrics result (a) elbow method. (b) Silhouette analysis

We can't use elbow method since it's a bit ambiguous as you can see in the figure above.

Using Silhouette score is also ambiguous for the results we got, since pretty much every choice of K all three conditions, to choose the best one, are met. One might choose according to highest average Silhouette score but it cannot be valid since in some of the tries the result might change.

We concluded that k-means isn't a suitable choice for clustering this dataset.

2.2.5 Circle dataset

2.2.5.1 Clustering results and silhouette analysis

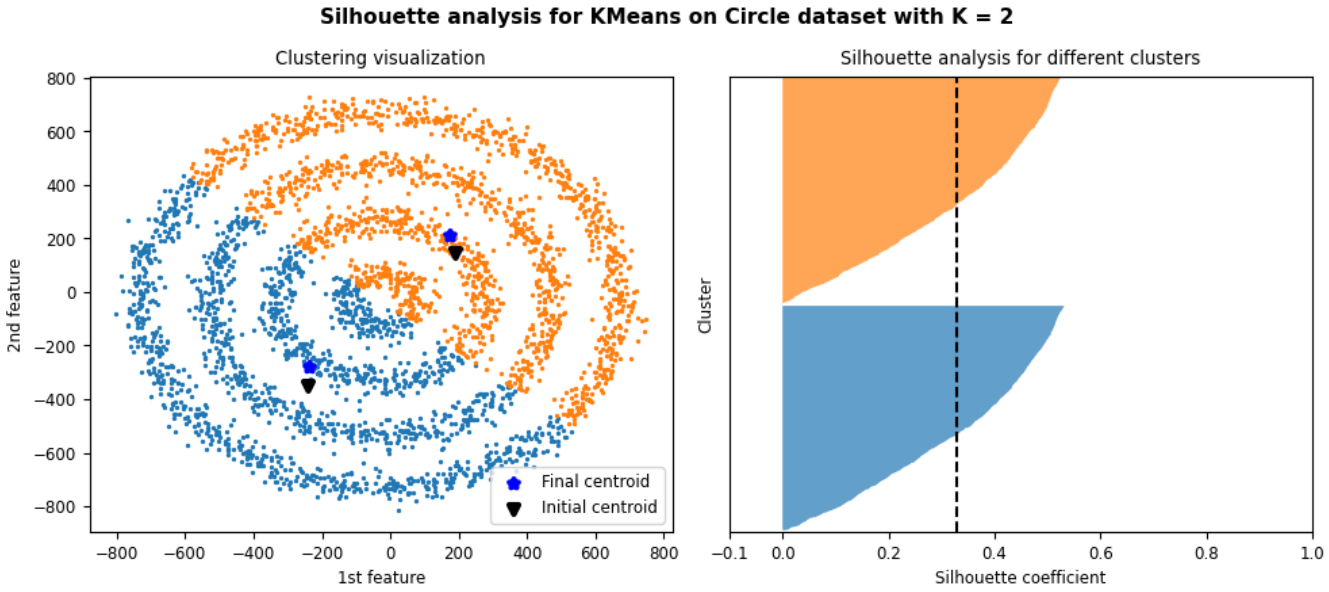


Figure 61: Results and silhouette analysis for K-Means on Circle dataset with K = 2

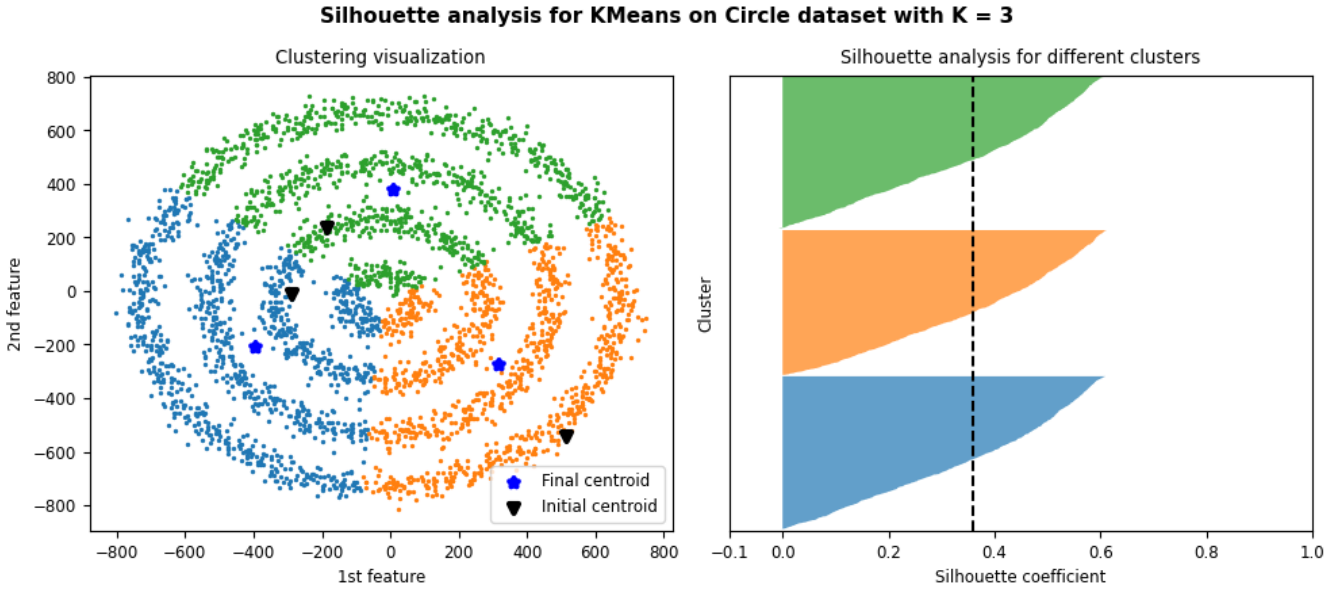


Figure 62: Results and silhouette analysis for K-Means on Circle dataset with K = 3

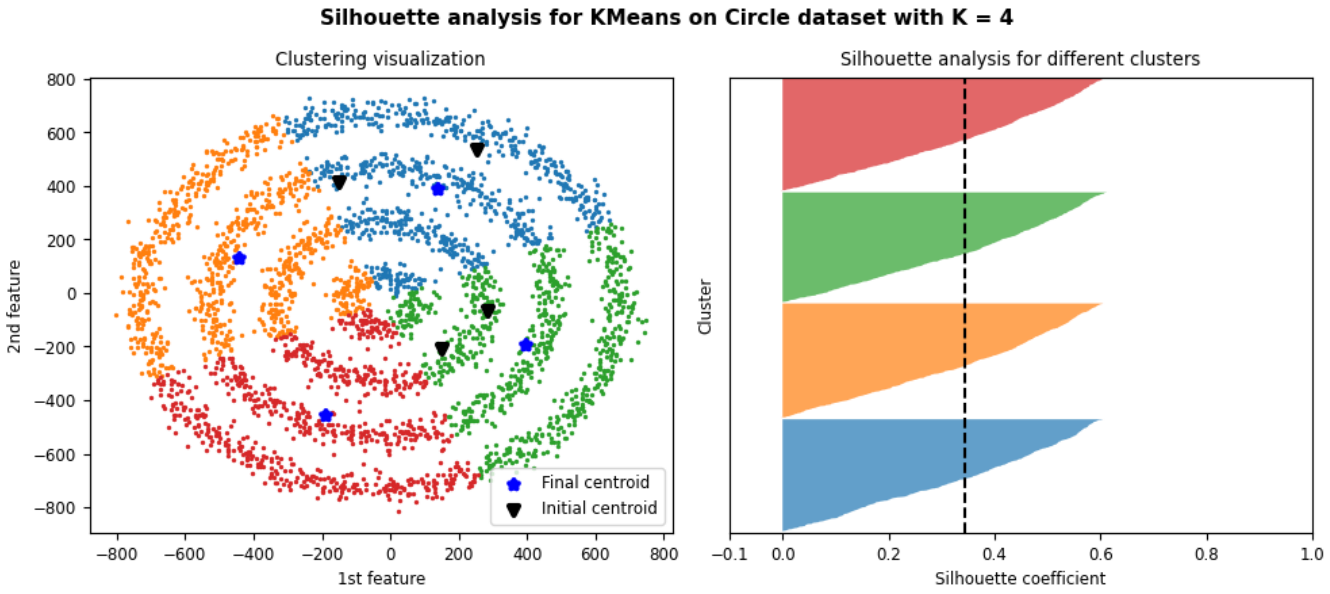


Figure 63: Results and silhouette analysis for K-Means on Circle dataset with K = 4

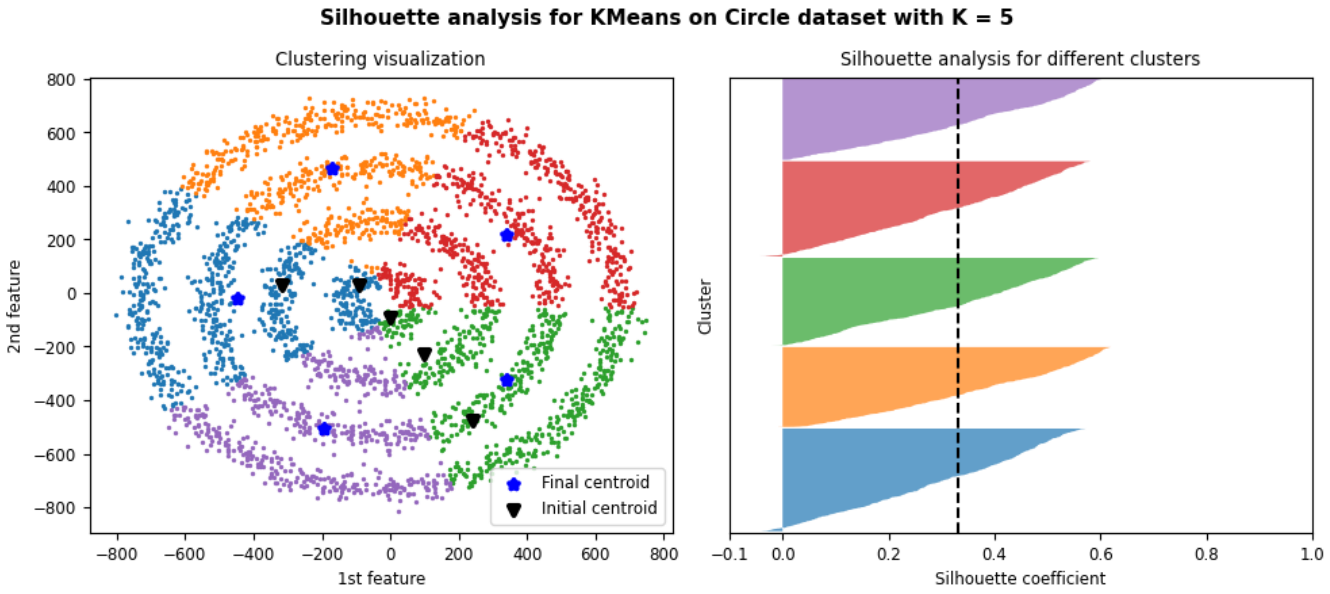


Figure 64: Results and silhouette analysis for K-Means on Circle dataset with K = 5

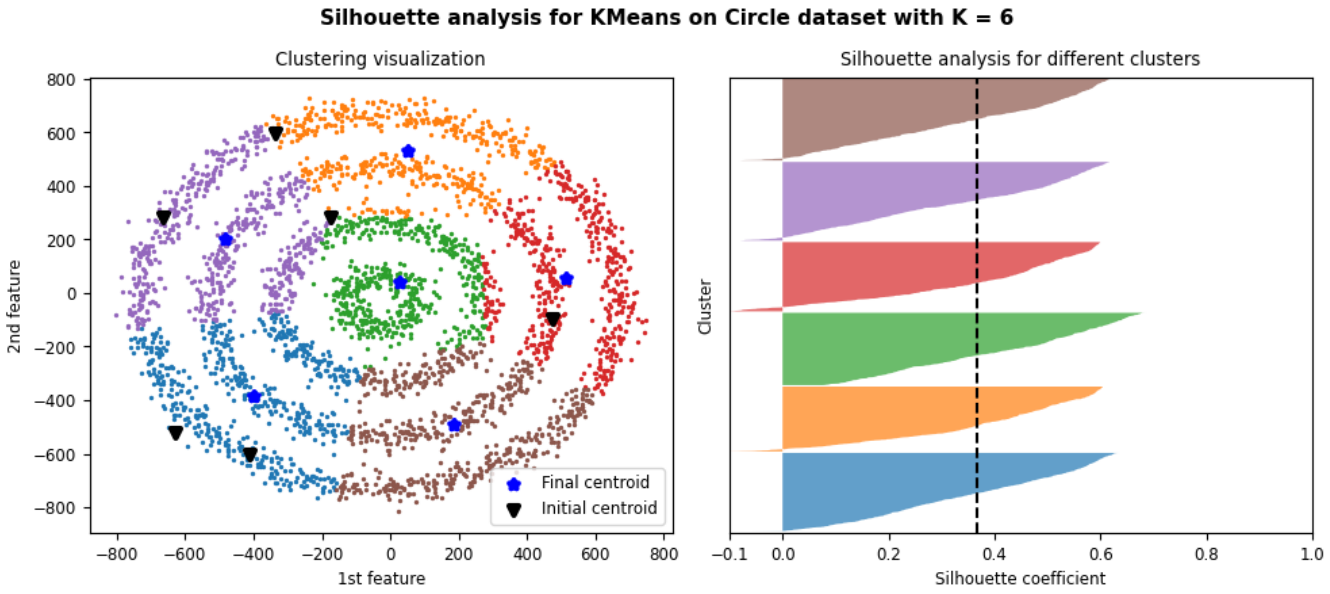


Figure 65: Results and silhouette analysis for K-Means on Circle dataset with K = 6

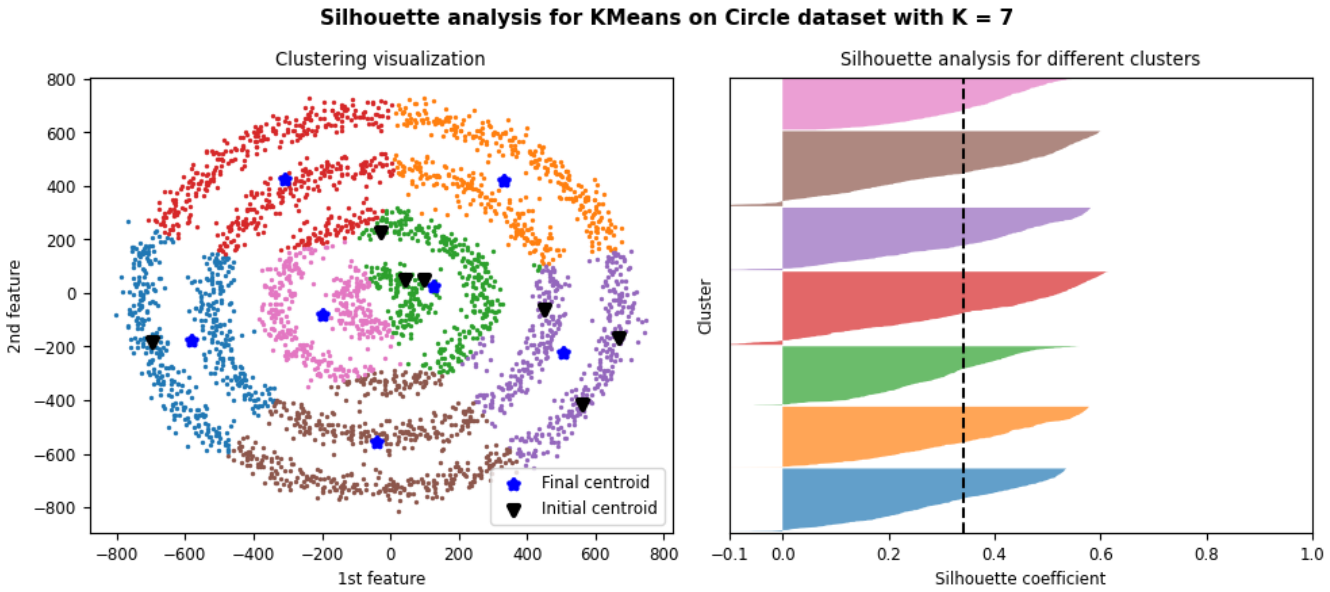


Figure 66: Results and silhouette analysis for K-Means on Circle dataset with K = 7

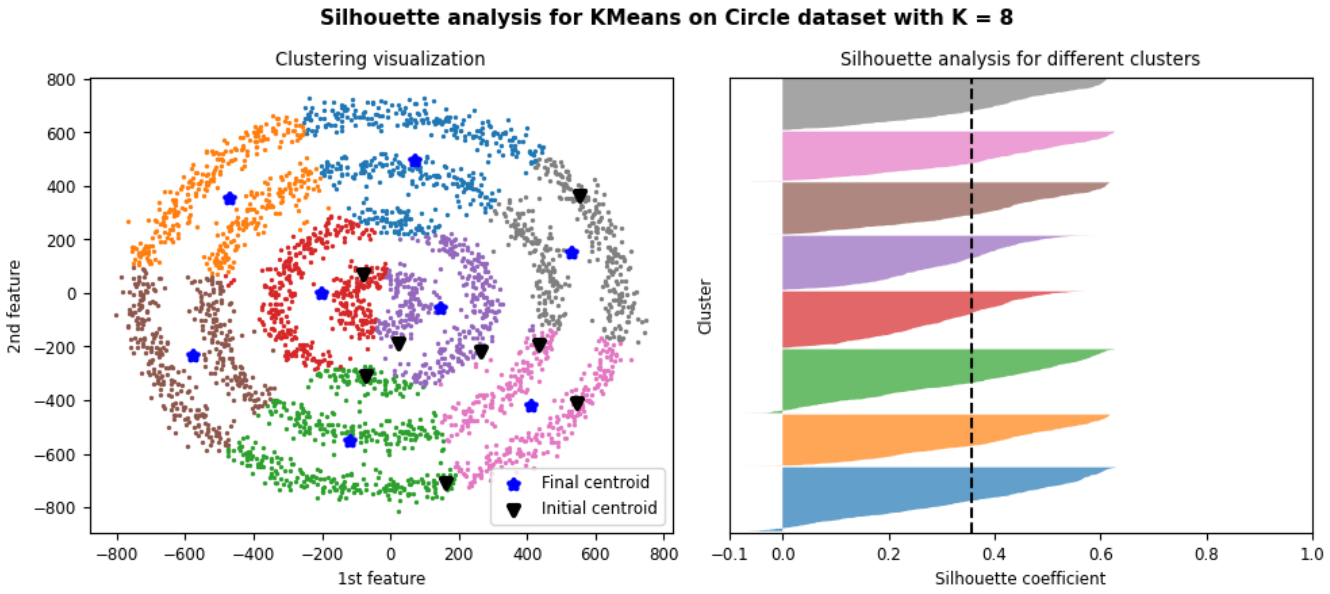


Figure 67: Results and silhouette analysis for K-Means on Circle dataset with K = 8

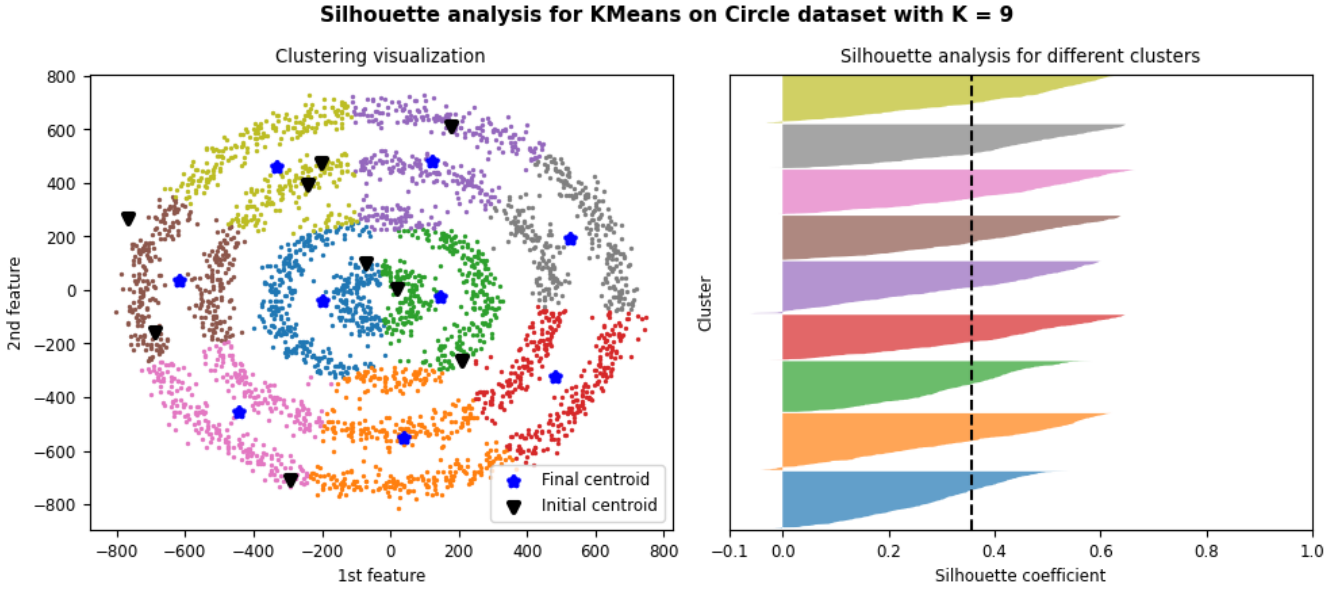


Figure 68: Results and silhouette analysis for K-Means on Circle dataset with K = 9

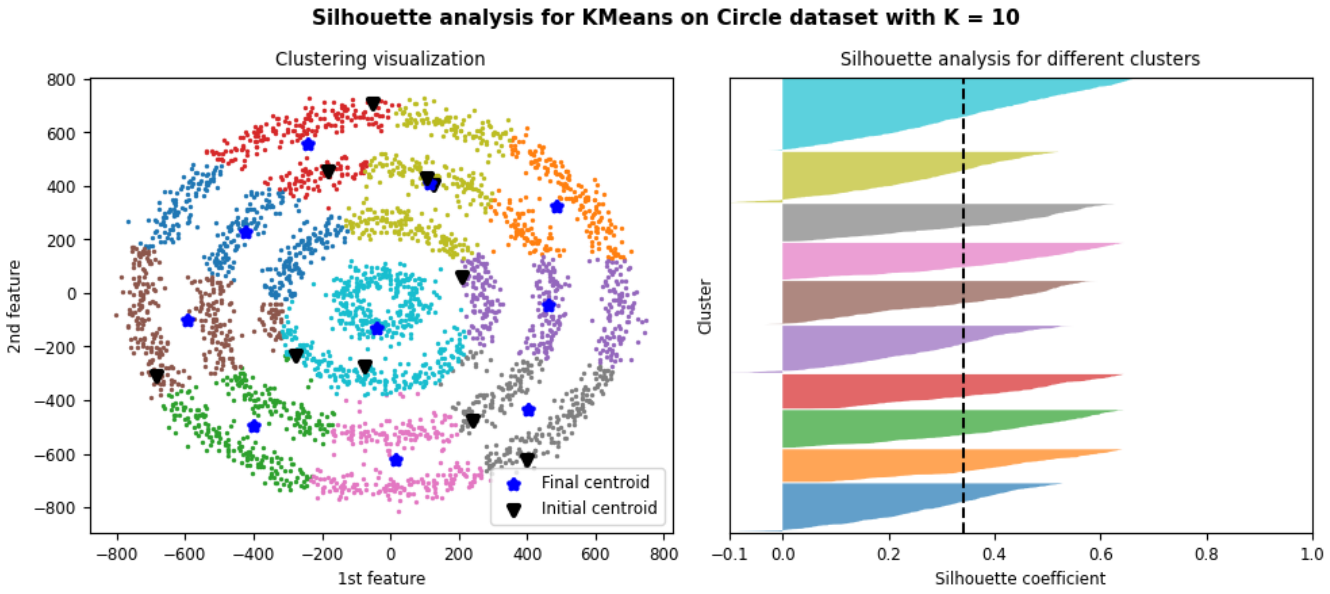


Figure 69: Results and silhouette analysis for K-Means on Circle dataset with K = 10

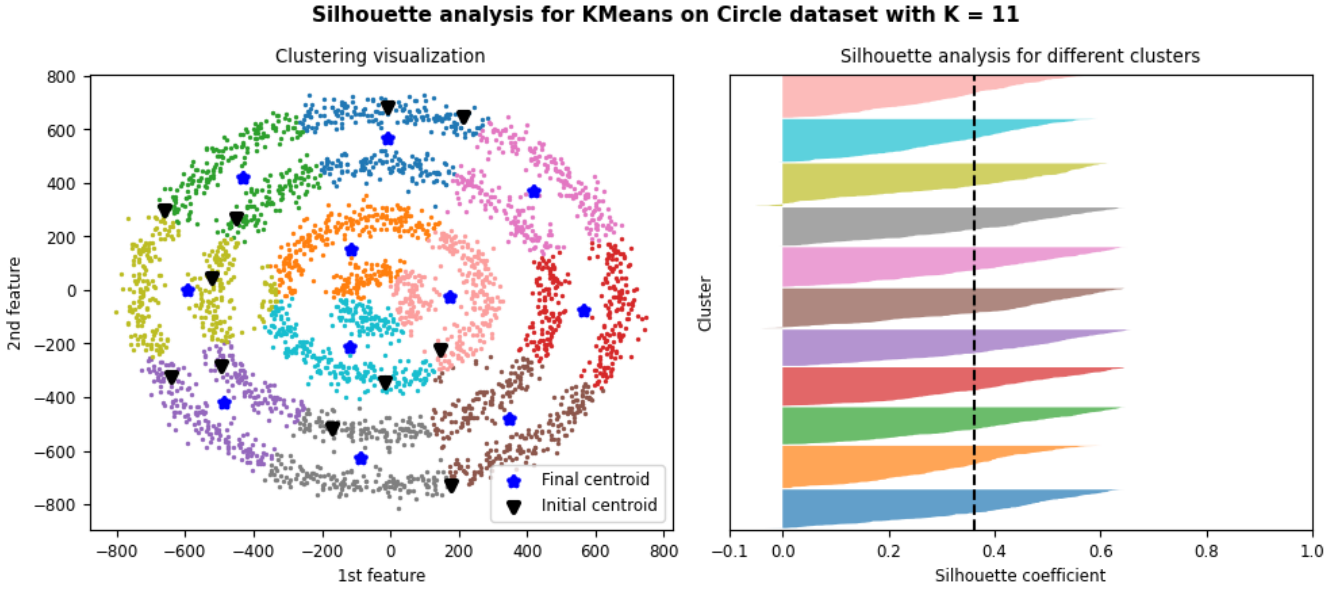


Figure 70: Results and silhouette analysis for K-Means on Circle dataset with K = 11

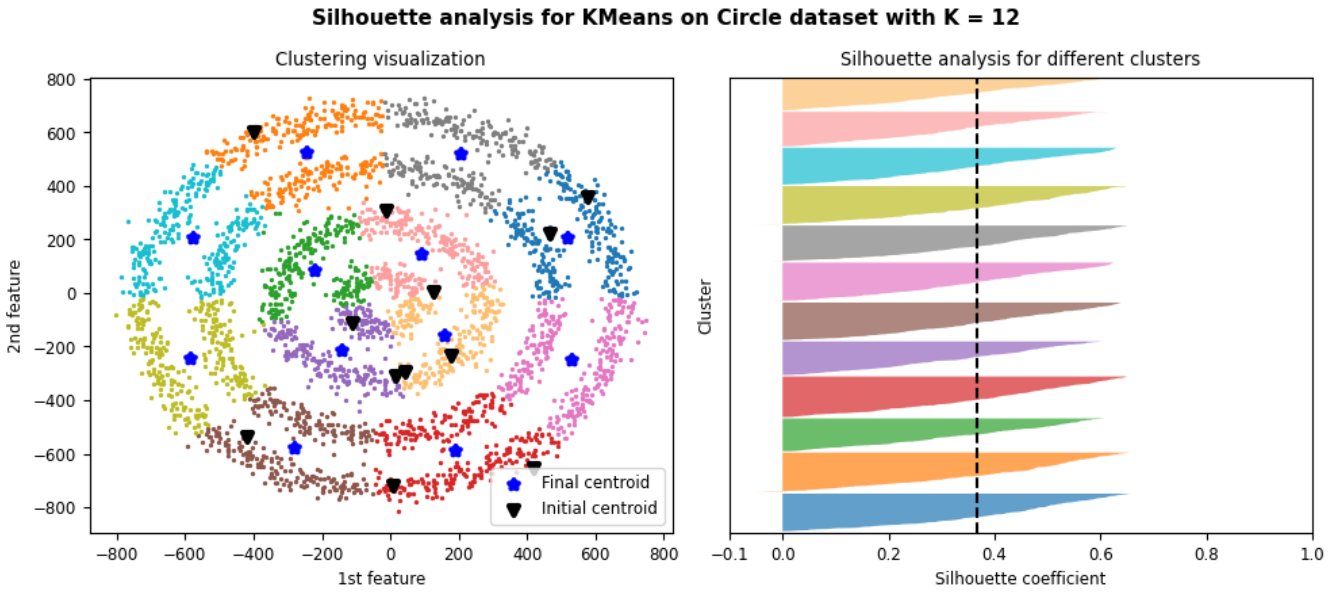


Figure 71: Results and silhouette analysis for K-Means on Circle dataset with K = 12

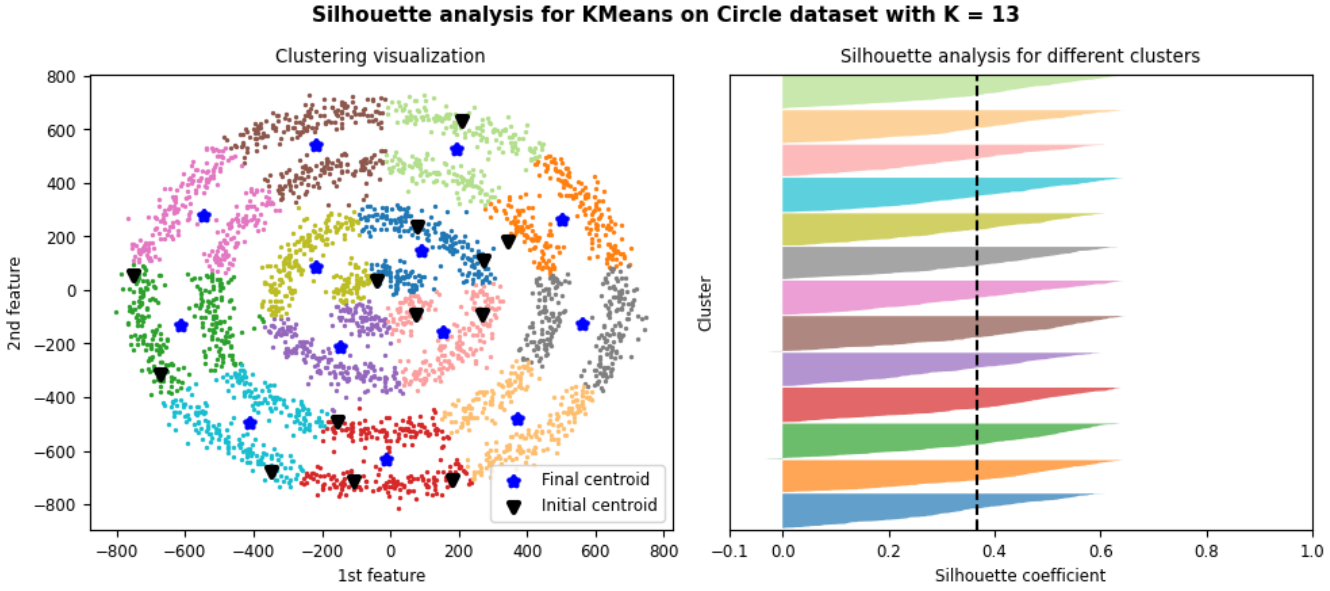


Figure 72: Results and silhouette analysis for K-Means on Circle dataset with K = 13

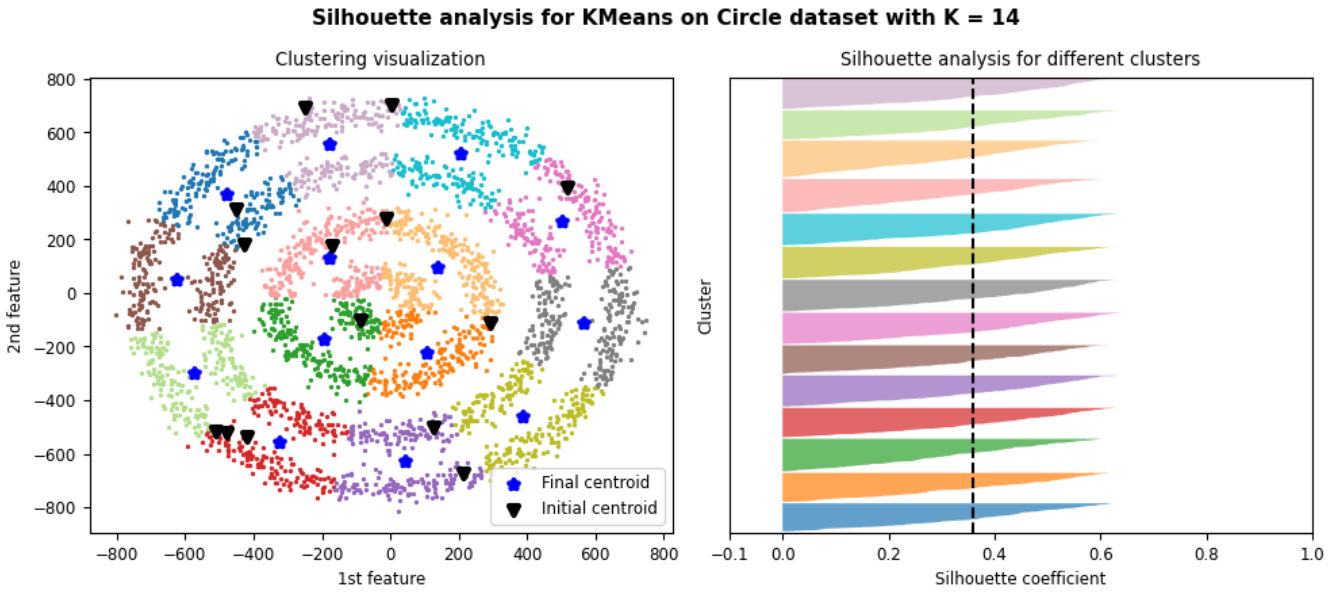


Figure 73: Results and silhouette analysis for K-Means on Circle dataset with K = 14

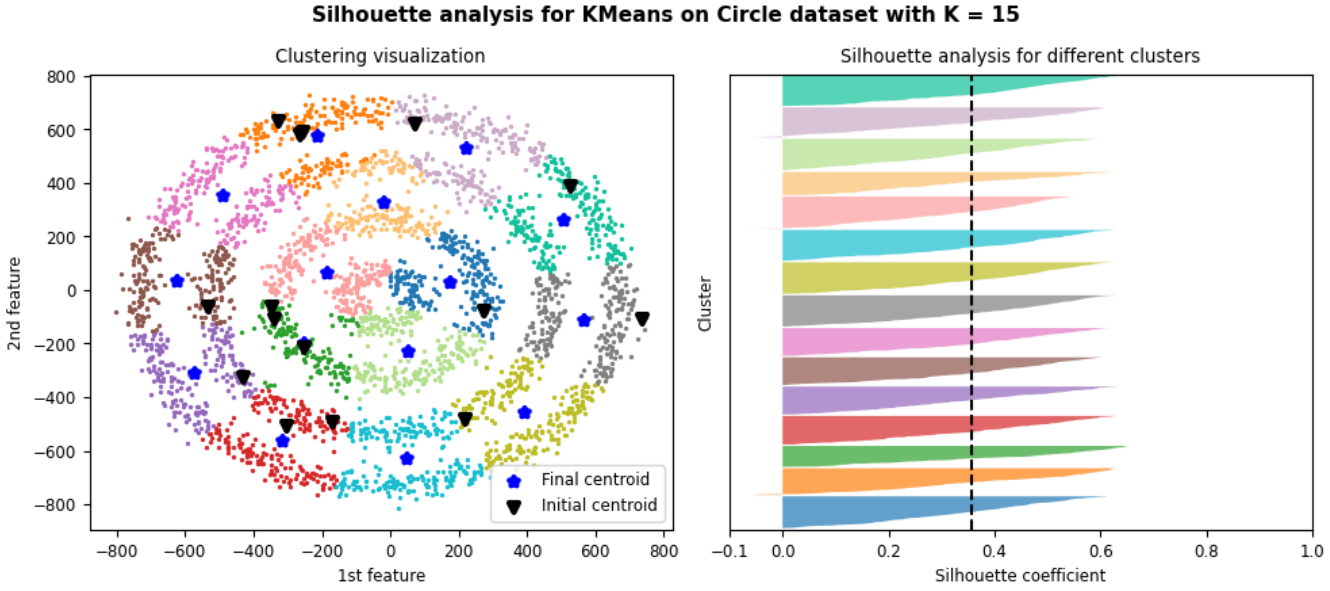
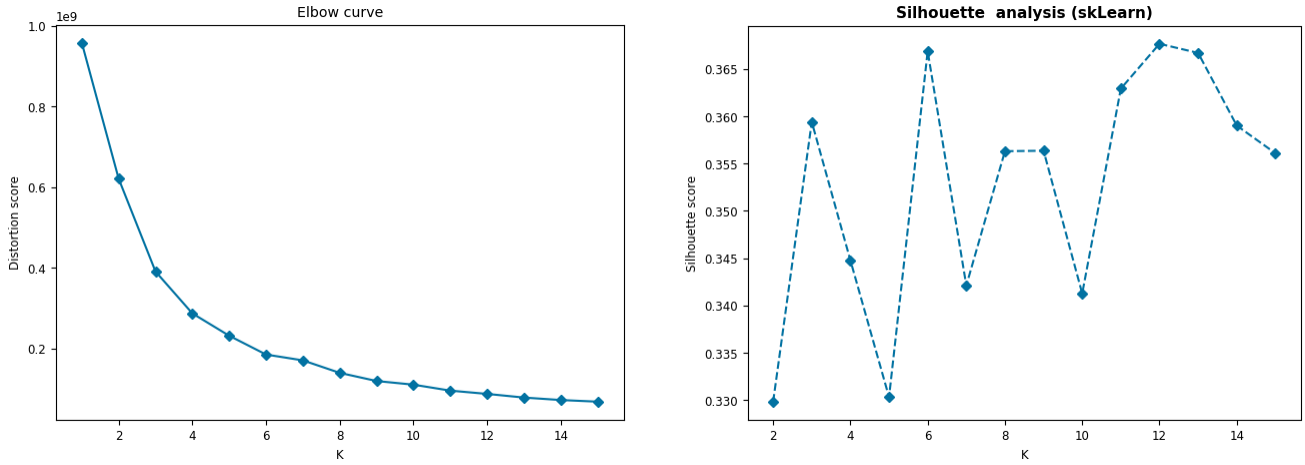


Figure 74: Results and silhouette analysis for K-Means on Circle dataset with K = 15

2.2.5.2 Choosing the best K



(a) *(b)*

Figure 75: metrics result (a) elbow method. (b) Silhouette analysis

For the same reasons we discussed in 2.2.4.2 for Moon dataset, both elbow method and Silhouette score doesn't work to choose the best number of clusters for this dataset and k-means algorithm is not a suitable choice for clustering on this dataset.

2.2.6 Best clustering using K-means

According to what we discussed in previous sections about each dataset Blobs dataset had the best clustering results using K-means algorithm. Clustering elliptical dataset using K-means was not so bad but still not perfect because of some dense and close clusters. For the other datasets K-means performed poorly especially in case of circle and moon datasets.

3 Gaussian mixture model (GMM)

The Gaussian mixture model (GMM) is well-known as an unsupervised learning algorithm for clustering. Here, "Gaussian" means the Gaussian distribution, described by mean and variance; mixture means the mixture of more than one Gaussian distribution. GMM uses Expectation Maximization (EM) to train a GMM model. A GMM model can be employed to estimate the PDF of some samples (like a parametric density estimator).

3.1 Implementation

We used EM-algorithm in this implementation. It is divided in two steps: 1. E-step & M-step. We implemented this algorithm in form of a class name GMM.

This class accepts number of clusters, maximum number of iterations and parameter initialization method.

3.1.1 Initializing parameters

Before getting into e-step and m-step, let's talk about parameters initialization. We have three parameters to initialize at the beginning of this algorithm which are: means, covariance matrixes and π.

We implemented two approaches for parameter initialization: 1. Using K-Means algorithm 2. Random. It's common to use k-means algorithm to select initial cluster centroids using sampling based on an empirical probability distribution of the points' contribution to the overall inertia. This technique speeds up convergence.

We implemented initialization process as function named initialize in GMM class.

3.1.2 E-step

During this step which is also known as the Estimation step, we calculate the r matrix. It is calculated using the formula below.

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j)}$$

r matrix is also known as 'responsibilities' matrix, it can be interpreted in the following way. Rows are the samples from the dataset, while columns represent every cluster, the elements of this matrix are interpreted as follows r_{nk} is the probability of sample n to be

part of cluster k. When the algorithm will converge, we will use this matrix to predict the clusters for each point. For normal distribution we used *multivariate_normal* function from *spicy* package.

This step is implemented as function named *e_step* in GMM class which accepts dataset then calculates and returns the r matrix.

3.1.3 M-step

During M-step which is also known as the Maximization-step we will update the values of parameters using updated responsibility matrix from E-step. To do suck task we will use the following formulas. Where N indicates the size.

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (x_n - \mu_k)(x_n - \mu_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

This step is implemented as a function named *m_step* in GMM class which accepts dataset and updated r matrix, then updates and returns the parameters above.

3.1.4 Repeat till convergence

We used the functions we discussed in previous sections inside of fit function in GMM class which accepts dataset and updates the initialized parameters corresponding to that dataset until convergence or reaching the maximum iteration pre-defined.

3.2 Evaluation methods

Since this algorithm is considered as an unsupervised learning algorithm. We can use clustering evaluation method to evaluate the quality of a clustering assignment.

3.2.1 Silhouette Coefficient

We used this method to evaluate clustering assignment for K-means algorithm as well. In average scores for this method a higher Silhouette Coefficient score relates to a model

with better defined clusters. The Silhouette Coefficient is defined for each sample and is composed of two scores:

- **a**: The mean distance between a sample and all other points in the same class.
- **b**: The mean distance between a sample and all other points in the next nearest cluster.

The Silhouette Coefficient s for a single sample is then given as:

$$s^{(i)} = \frac{b^{(i)} - a^{(i)}}{\max(a^{(i)}, b^{(i)})}$$

The silhouette coefficient, by its definition, falls in the [−1, 1] interval. When the silhouette coefficient is calculated and averaged over all data, the closer to 1, the better the clustering performance.

3.2.2 Calinski-Harabasz Index

If the ground truth labels are not known, the Calinski-Harabasz index - also known as the Variance Ratio Criterion - can be used to evaluate the model, where a higher Calinski-Harabasz score relates to a model with better defined clusters. This score is higher when clusters are dense and well separated, which relates to a standard concept of a cluster.

3.2.3 Davies-Bouldin Index

In this method a lower Davies-Bouldin index relates to a model with better separation between the clusters.

This index signifies the average 'similarity' between clusters, where the similarity is a measure that compares the distance between clusters with the size of the clusters themselves.

Zero is the lowest possible score. Values closer to zero indicate a better partition.

We implemented a method named *report_scores* which accepts data and an array containing different values of K = number of gaussian components to use with GMM algorithm and then runs GMM 50 times for each K and saves the average of 10 best scores. In the end plots the results for given Ks for three metrics we discussed above. To calculate each score in order we used the following functions from *sklearn* library: *silhouette_score*, *calinski_harabasz_score* and *davies_bouldin_score*.

3.3 Results

We tried GMM algorithm on datasets with K = 2, 3, 5, 7, 8, 10 Gaussian components. We didn't use 1 because silhouette coefficient method implemented by skLearn, needs at least two unique labels.

3.3.1 TSNV dataset

3.3.1.1 Clustering results and silhouette analysis

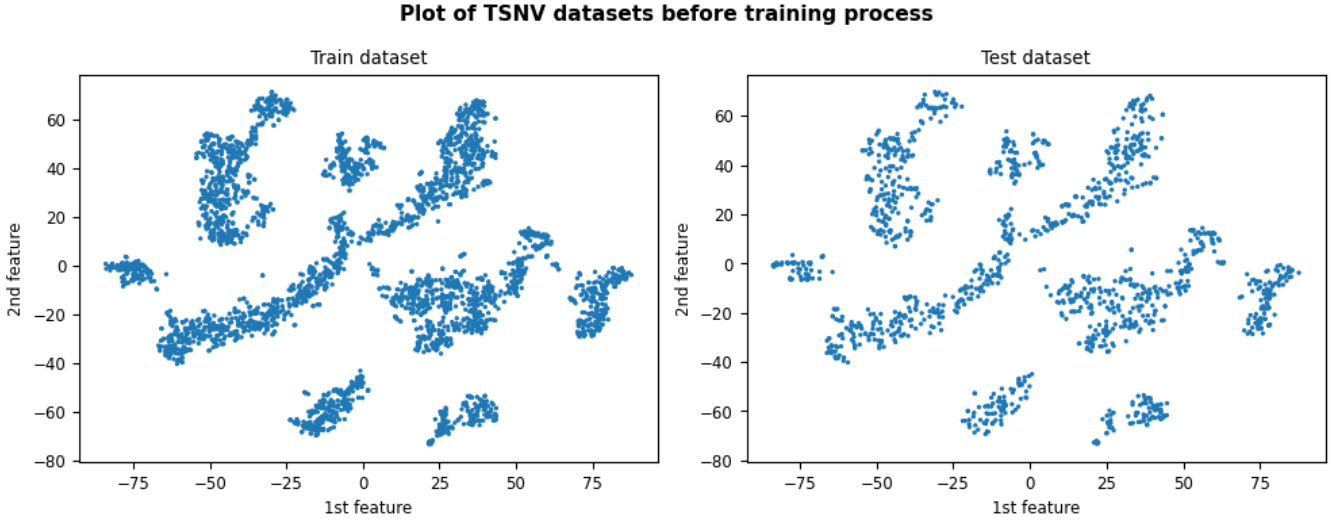


Figure 76: Plot of train & test datasets for TSNV dataset

Since this dataset didn't have any labels, we used a single label "1" for all samples in the dataset.

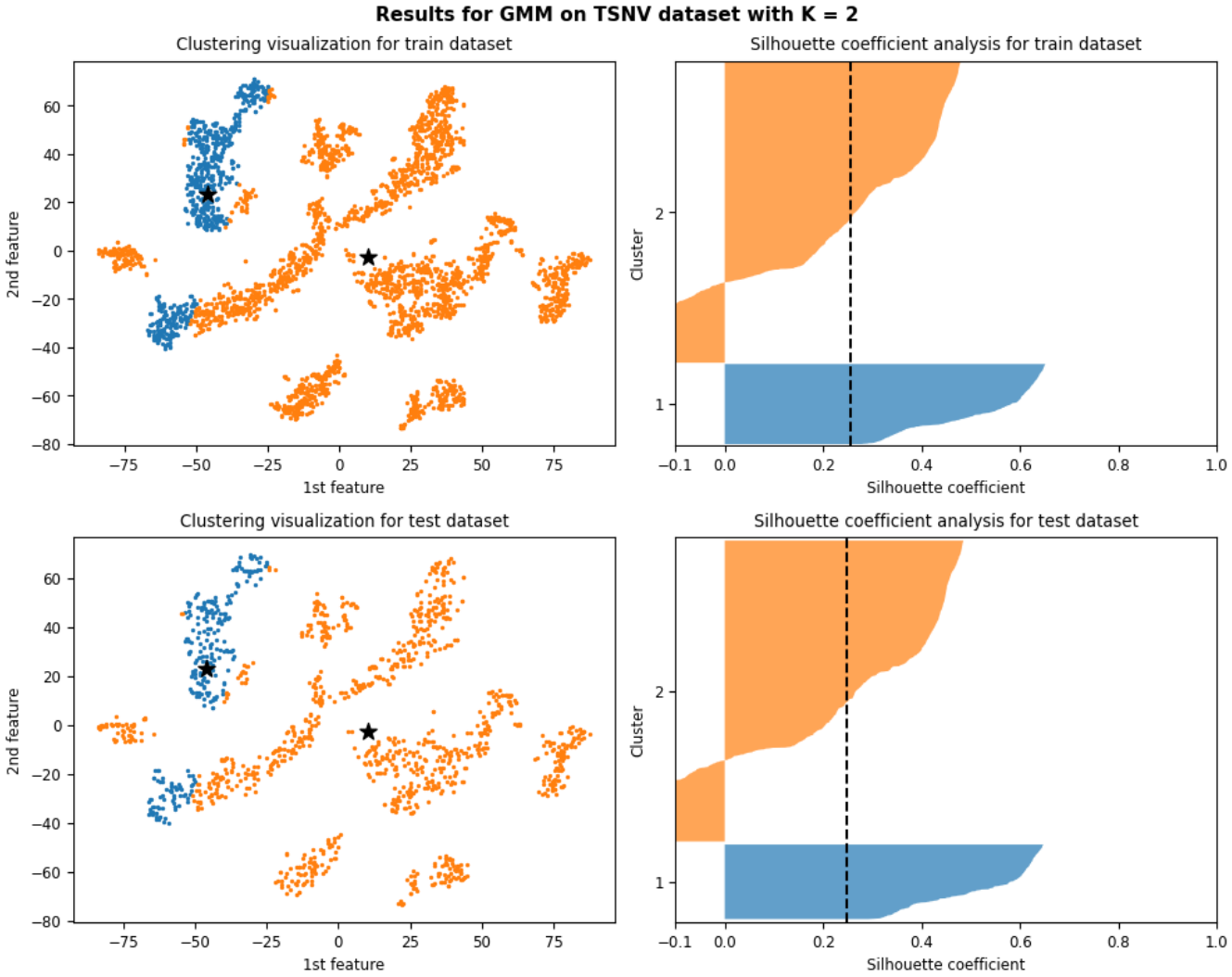


Figure 77: Results and silhouette analysis for running GMM on TSNV dataset with K = 2

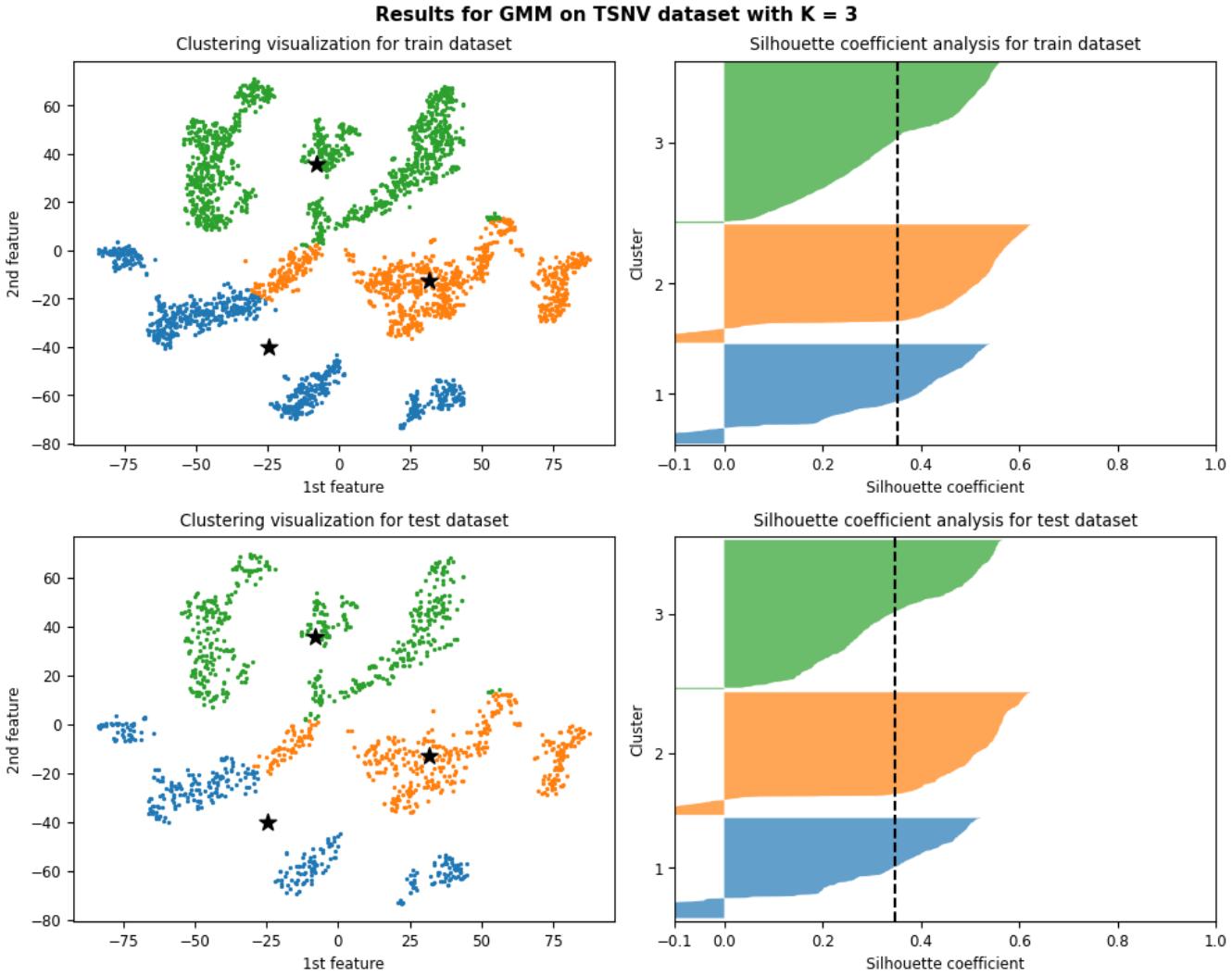


Figure 78: Results and silhouette analysis for running GMM on TSNV dataset with K = 3

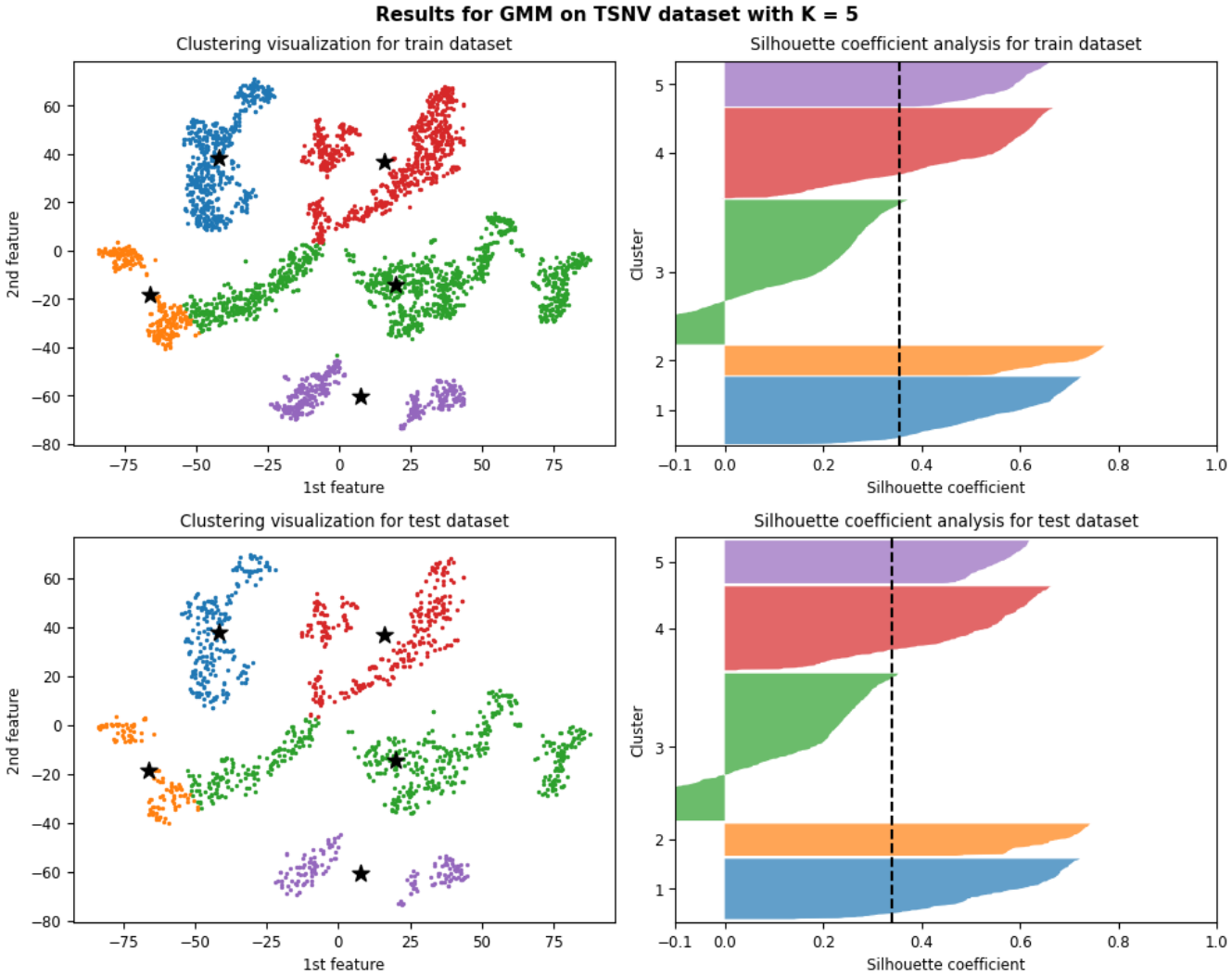


Figure 79: Results and silhouette analysis for running GMM on TSNV dataset with K = 5

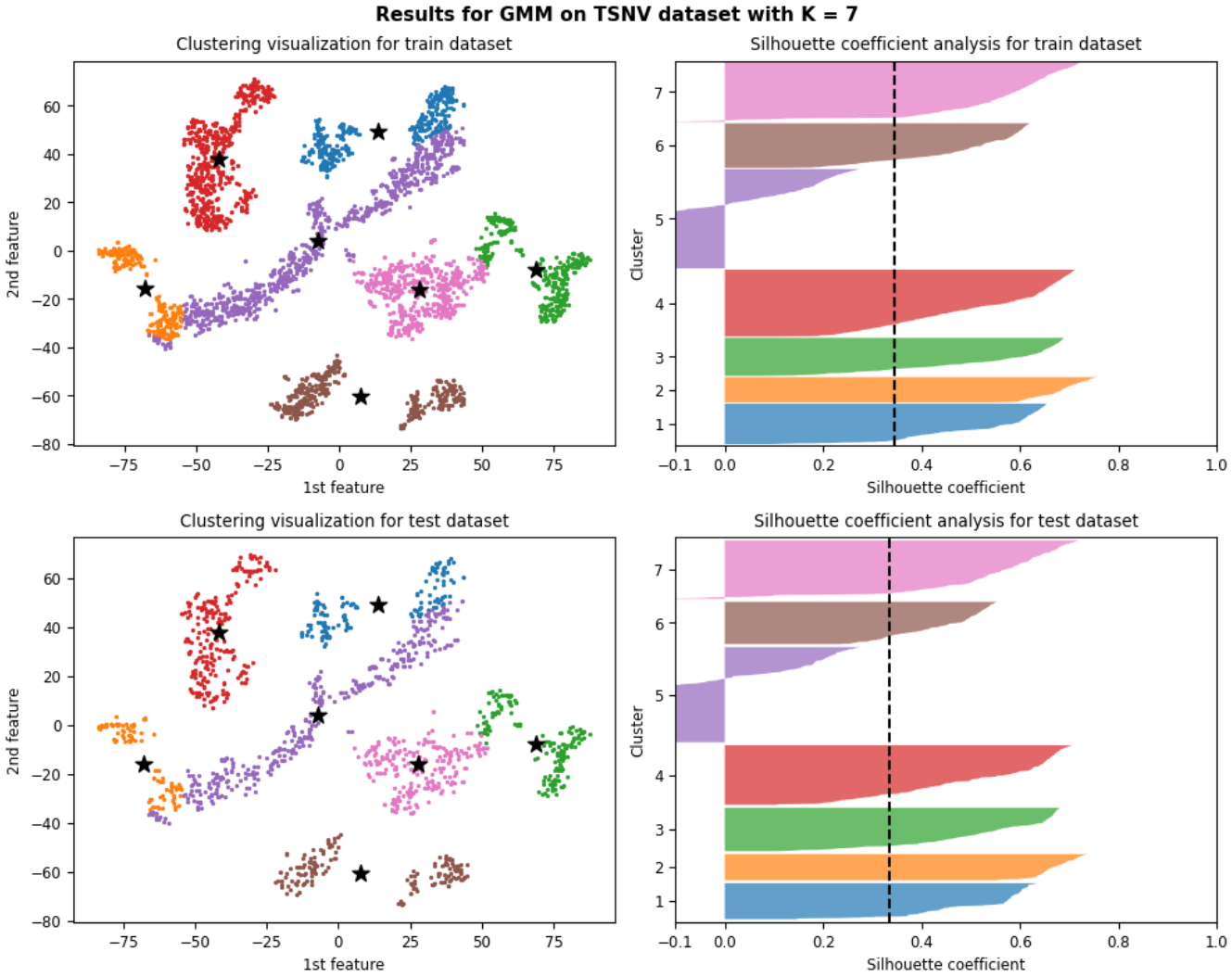


Figure 80: Results and silhouette analysis for running GMM on TSNV dataset with K = 7

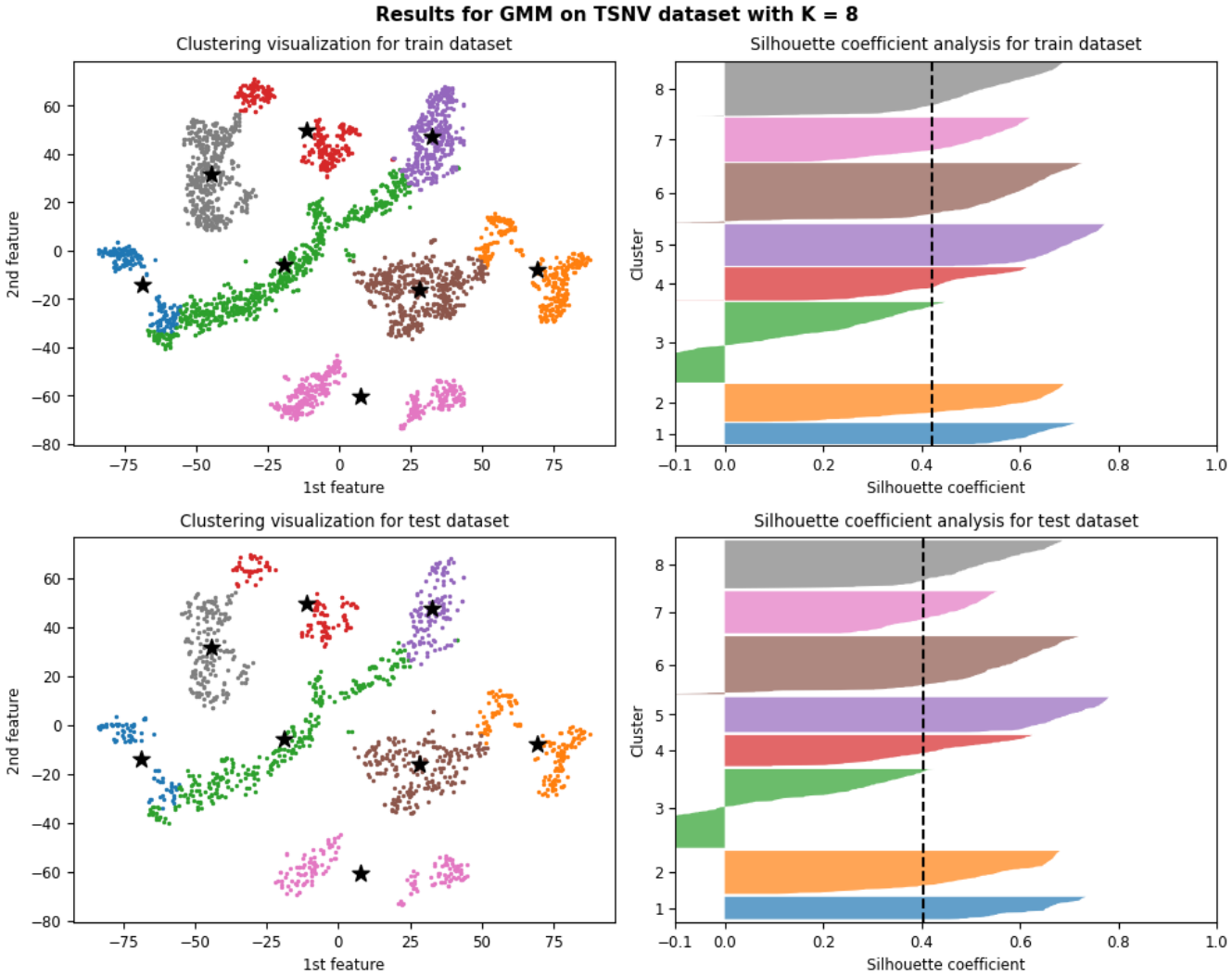


Figure 81: Results and silhouette analysis for running GMM on TSNV dataset with K = 8

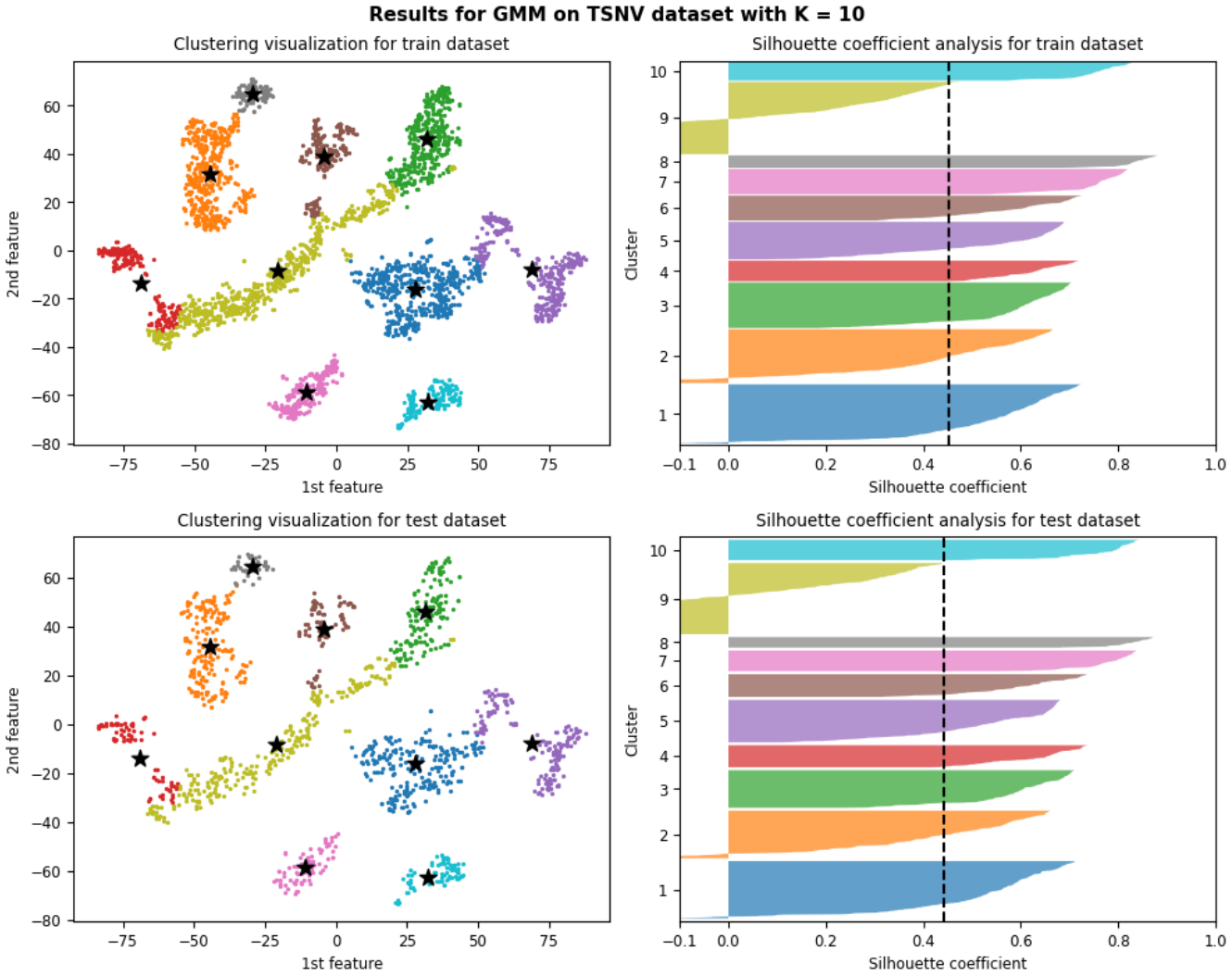


Figure 82: Results and silhouette analysis for running GMM on TSNV dataset with K = 10

3.3.1.2 Choosing the best K

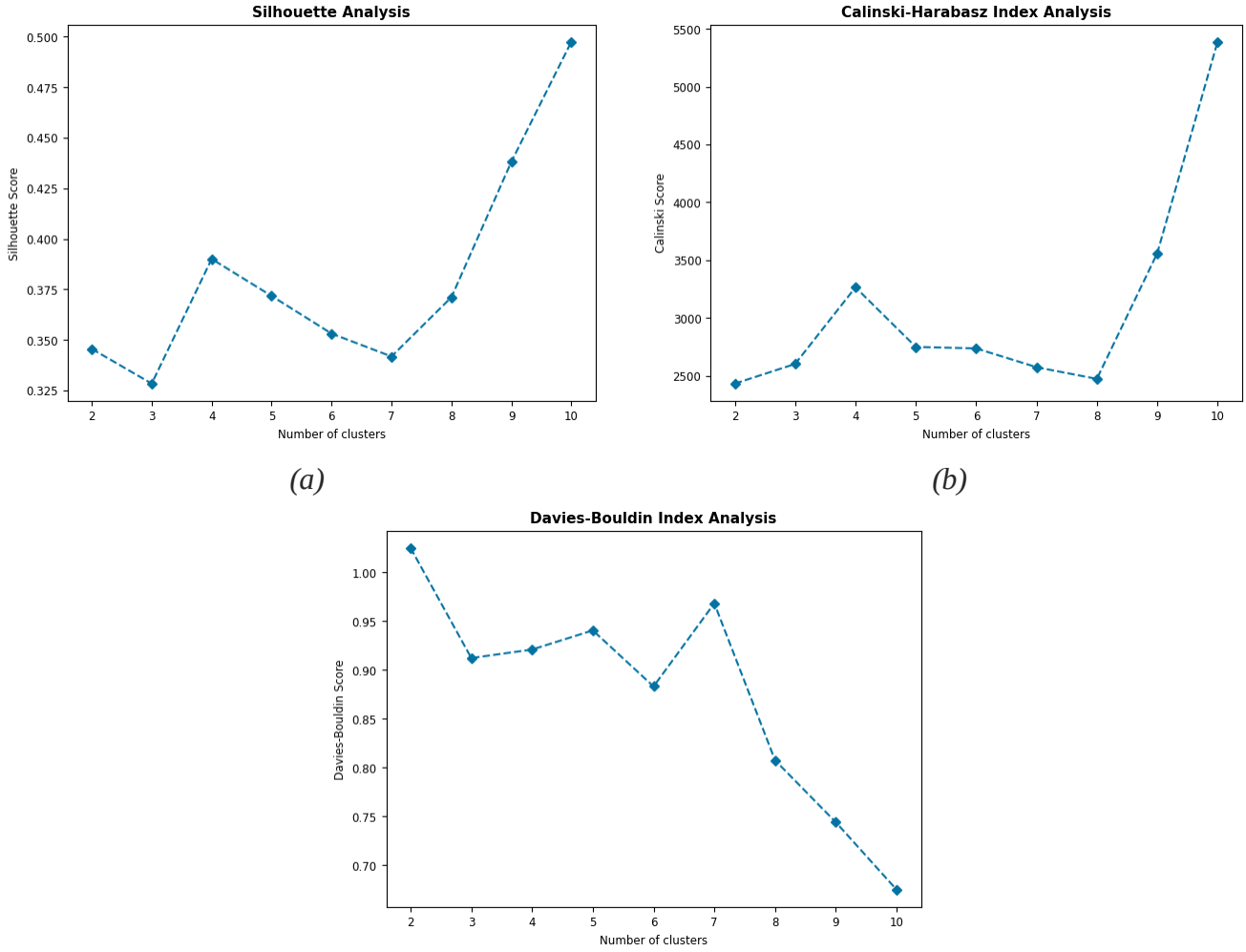


Figure 83: metrics result (a) Average Silhouette score analysis. (b) Calinski-Harabasz Index Analysis. (c) Davies-Bouldin Index Analysis

According to metrics above K = 10 Gaussian components is the best choice since according to definition it has the highest score in both average silhouette and Calinski-Harabasz Index analysis. And lowest value in Davies-Bouldin Index Analysis.

We can also see for K = 10 in Silhouette analysis the cluster number 9 has a huge negative value which shows some of its samples may belong to adjacent clusters.

We can choose other optimal number of gaussian components like K = 8 according to 3 Silhouette coefficient conditions and its plot in Figure 81.

3.3.2 Blobs dataset

3.3.2.1 Clustering results and silhouette analysis

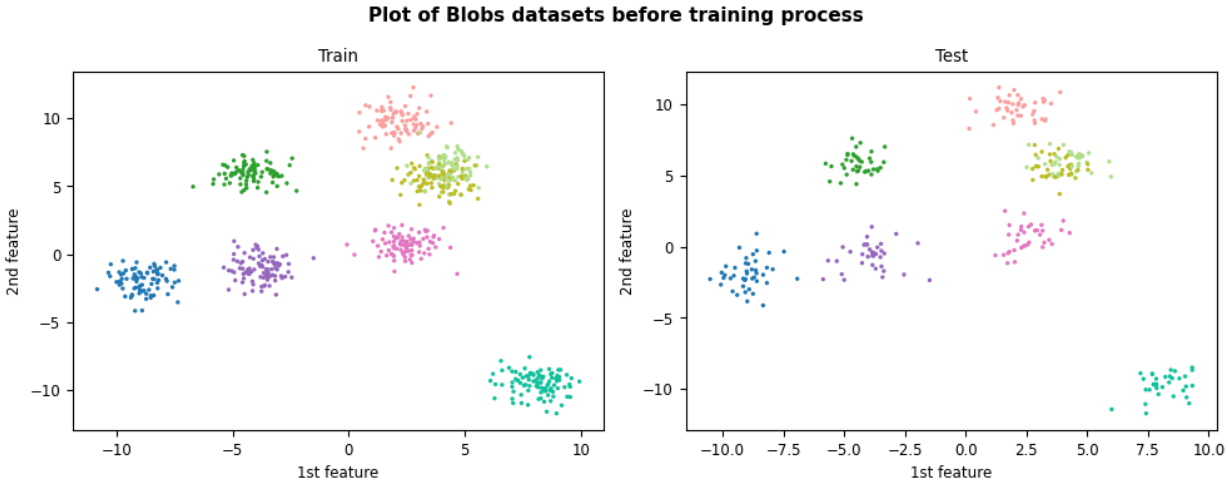


Figure 84: Plot of train & test datasets for Blobs dataset

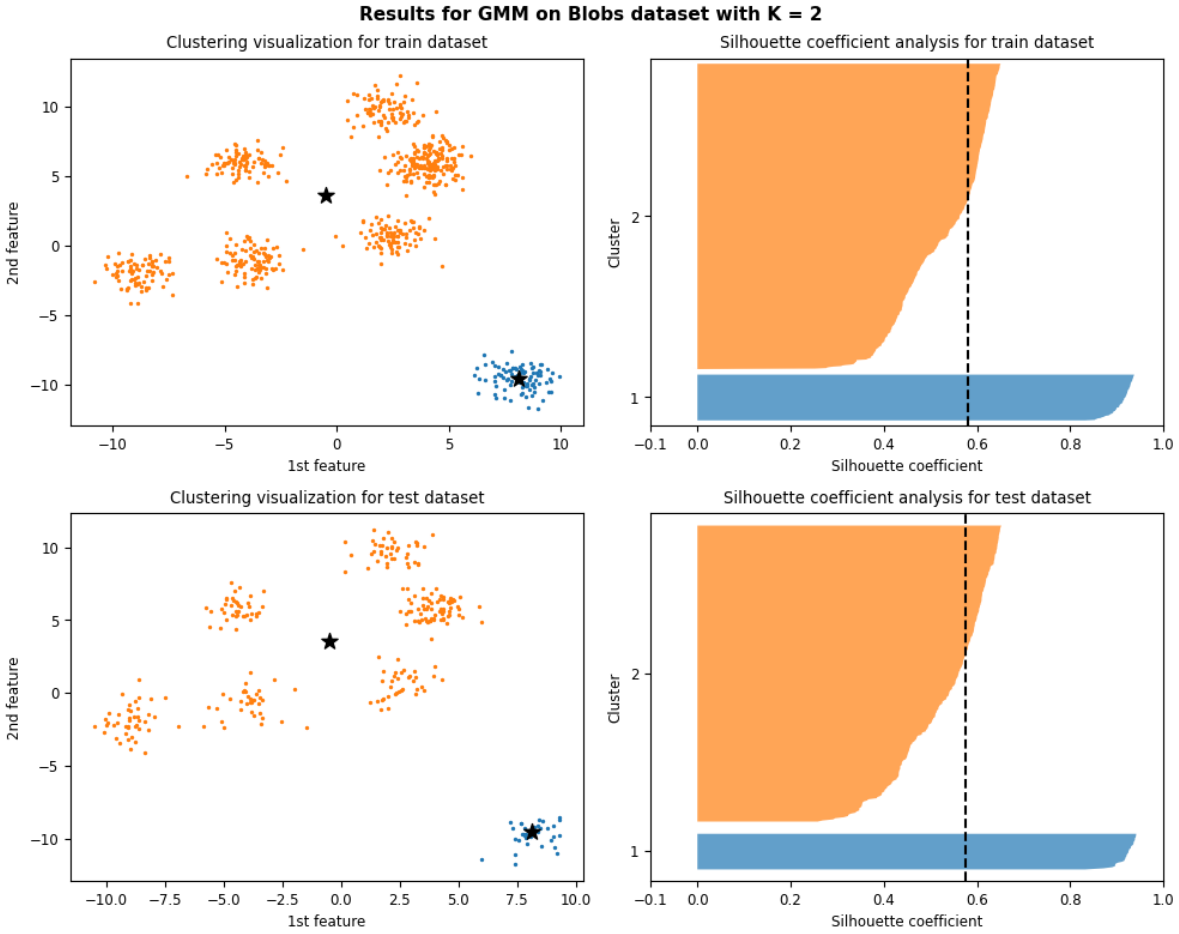


Figure 85: Results and silhouette analysis for running GMM on Blobs dataset with K = 2

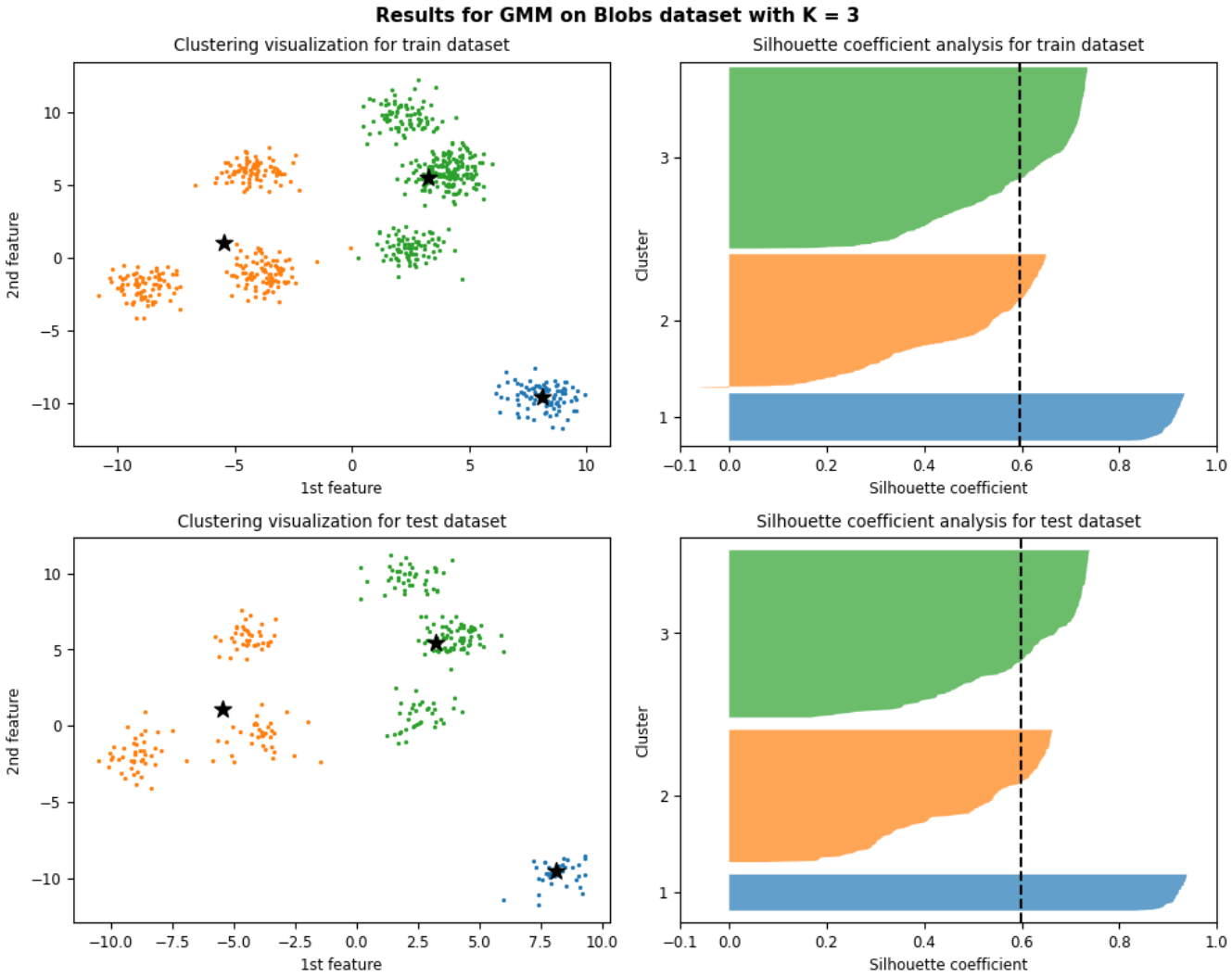


Figure 86: Results and silhouette analysis for running GMM on Blobs dataset with K = 3

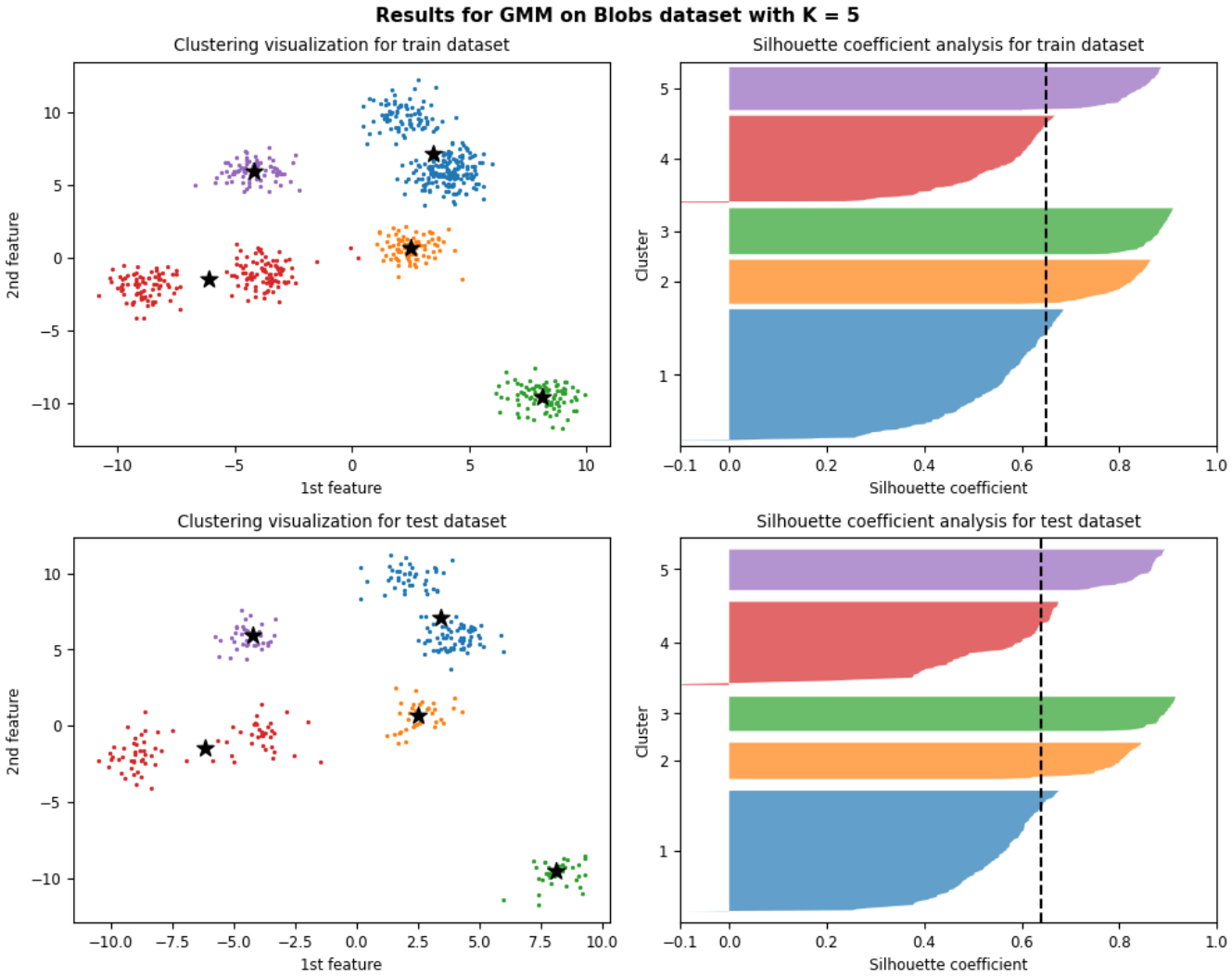


Figure 87: Results and silhouette analysis for running GMM on Blobs dataset with K = 5

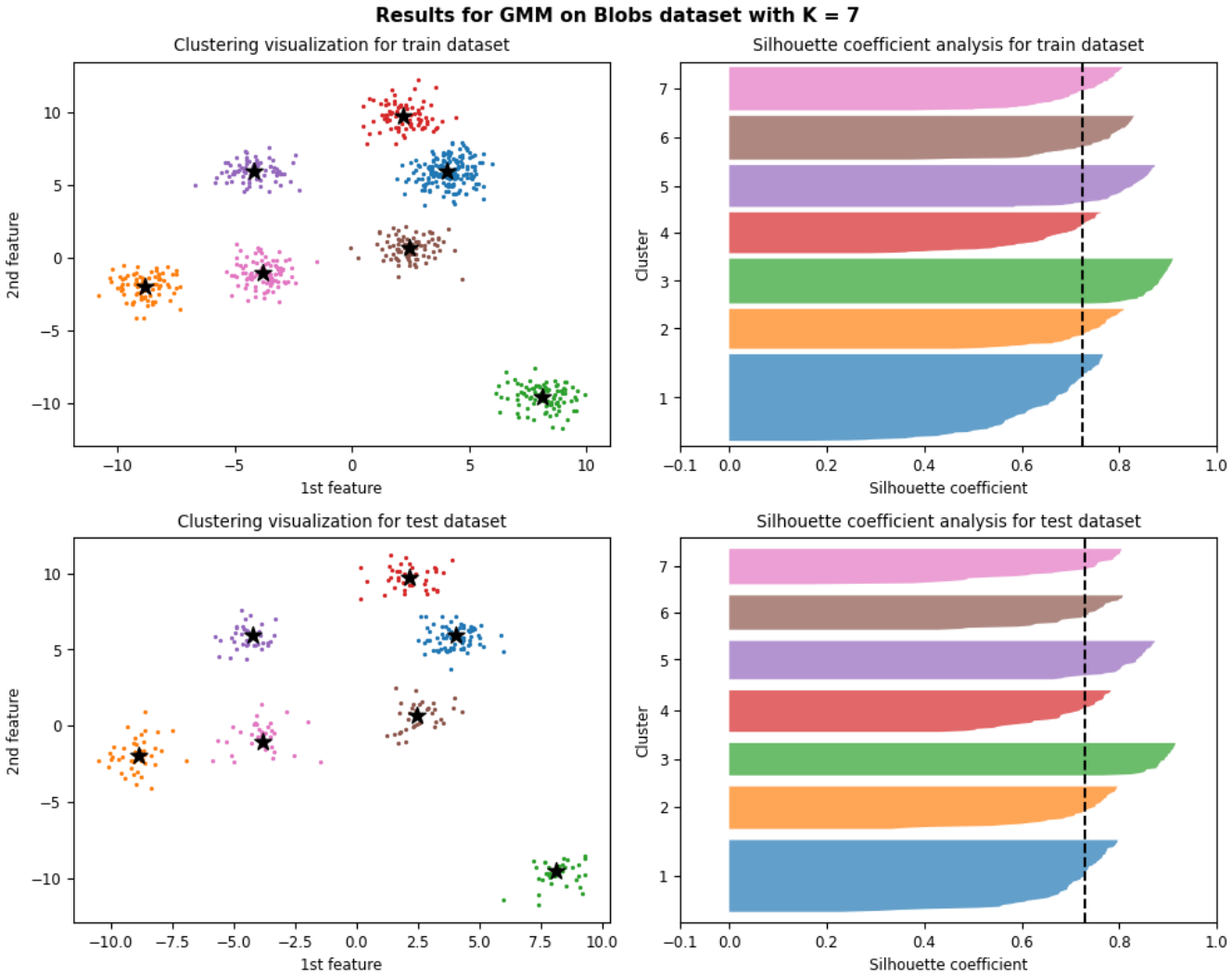


Figure 88: Results and silhouette analysis for running GMM on Blobs dataset with K = 7

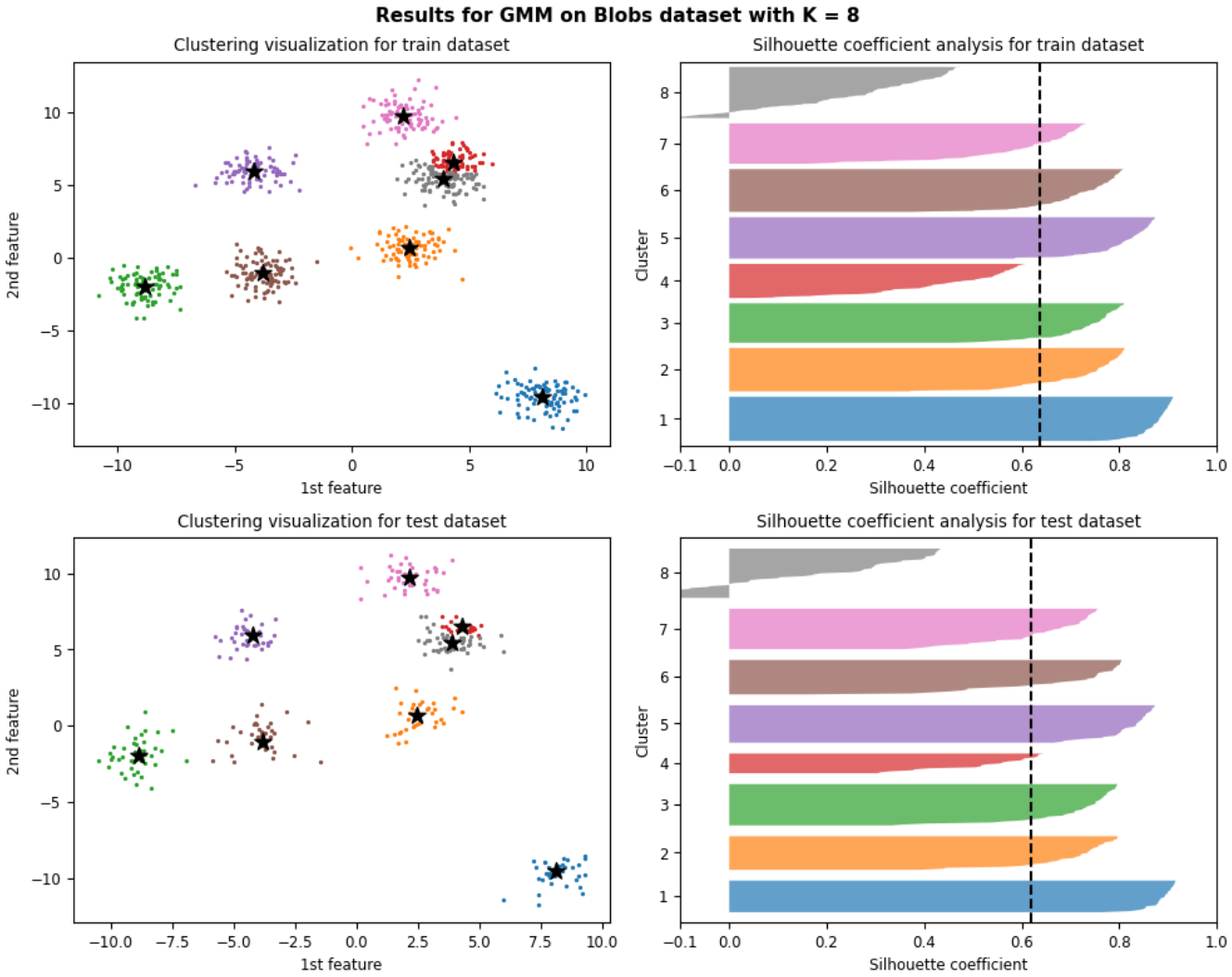


Figure 89: Results and silhouette analysis for running GMM on Blobs dataset with K = 8

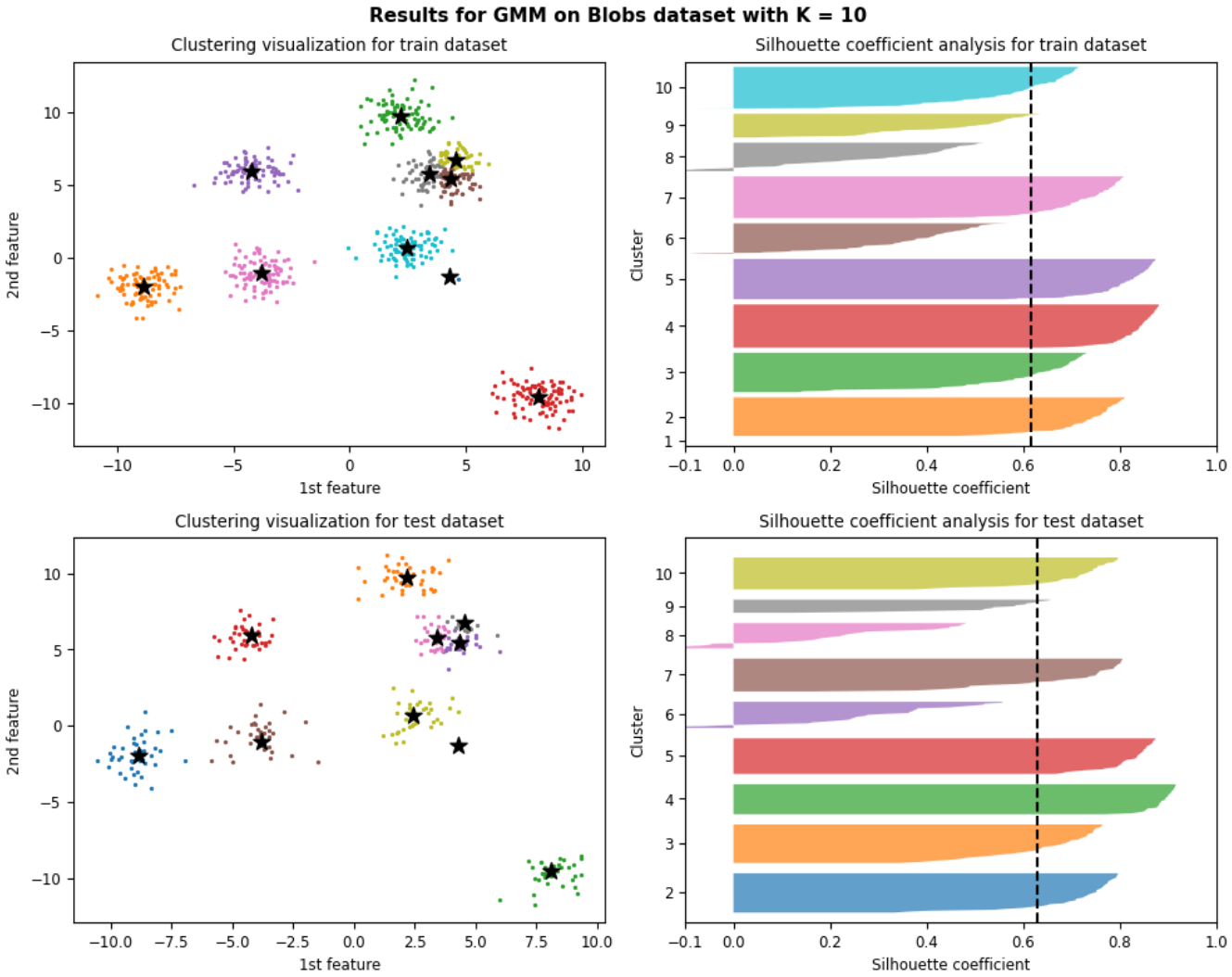
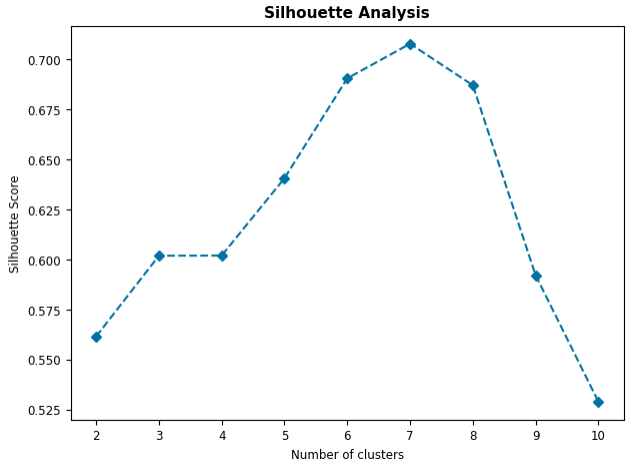
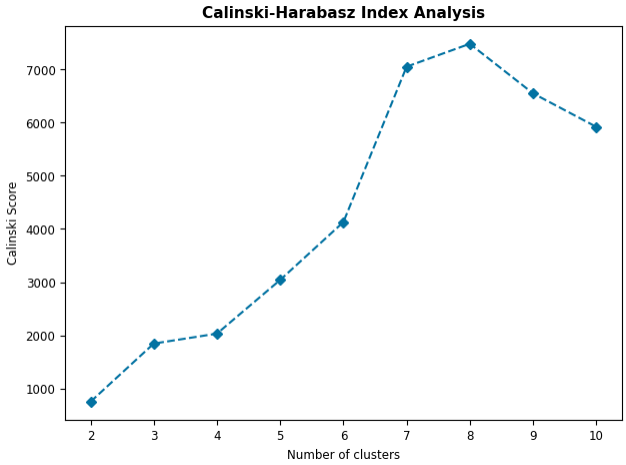


Figure 90: Results and silhouette analysis for running GMM on Blobs dataset with K = 10

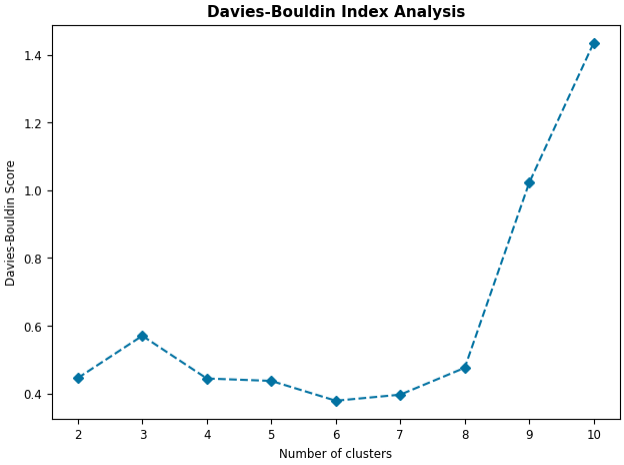
3.3.2.2 Choosing the best K



(a)



(b)



(c)

Figure 91: metrics result (a) Average Silhouette score analysis. (b) Calinski-Harabasz Index Analysis. (c) Davies-Bouldin Index Analysis

According Silhouette analysis results K = 7 gaussian components is the best choice since it stands well against all the three measuring criteria for silhouette analysis where all clusters' plot is beyond average Silhouette score, with mostly uniform thickness and do not have wide fluctuations in the size. It also has the highest average Silhouette score.

According to Calinski-Harabasz index & Davies-Bouldin index analysis results and their definition, K = 8 and K = 6 are best choices in order. But in both of these metrics K = 7 gaussian components is so close to best values we chose, and can be a sub-optimal answer in terms of these metrics.

3.3.3 Elliptical dataset

3.3.3.1 Clustering results and silhouette analysis

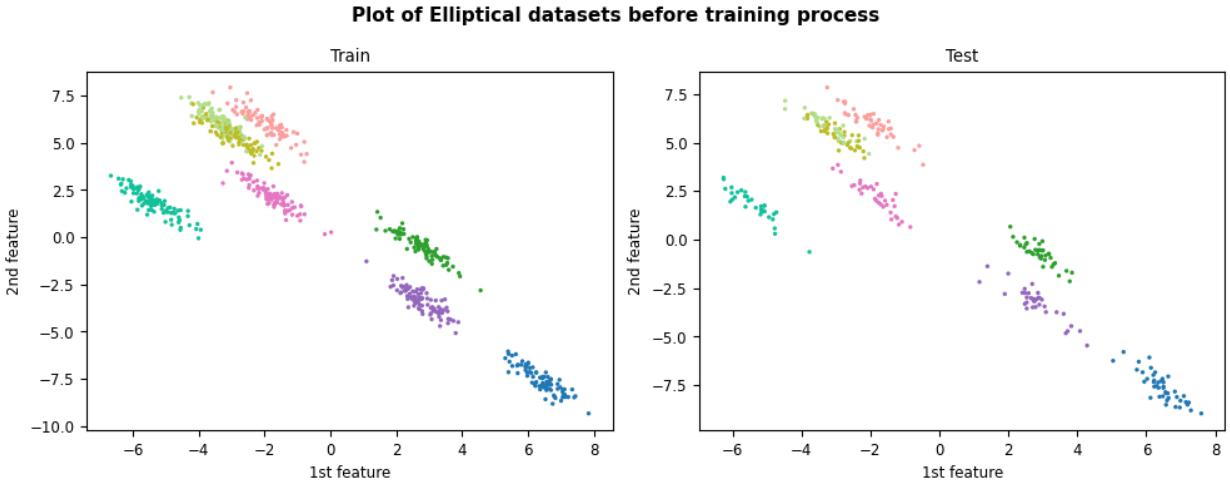


Figure 92: Plot of train & test datasets for Elliptical dataset

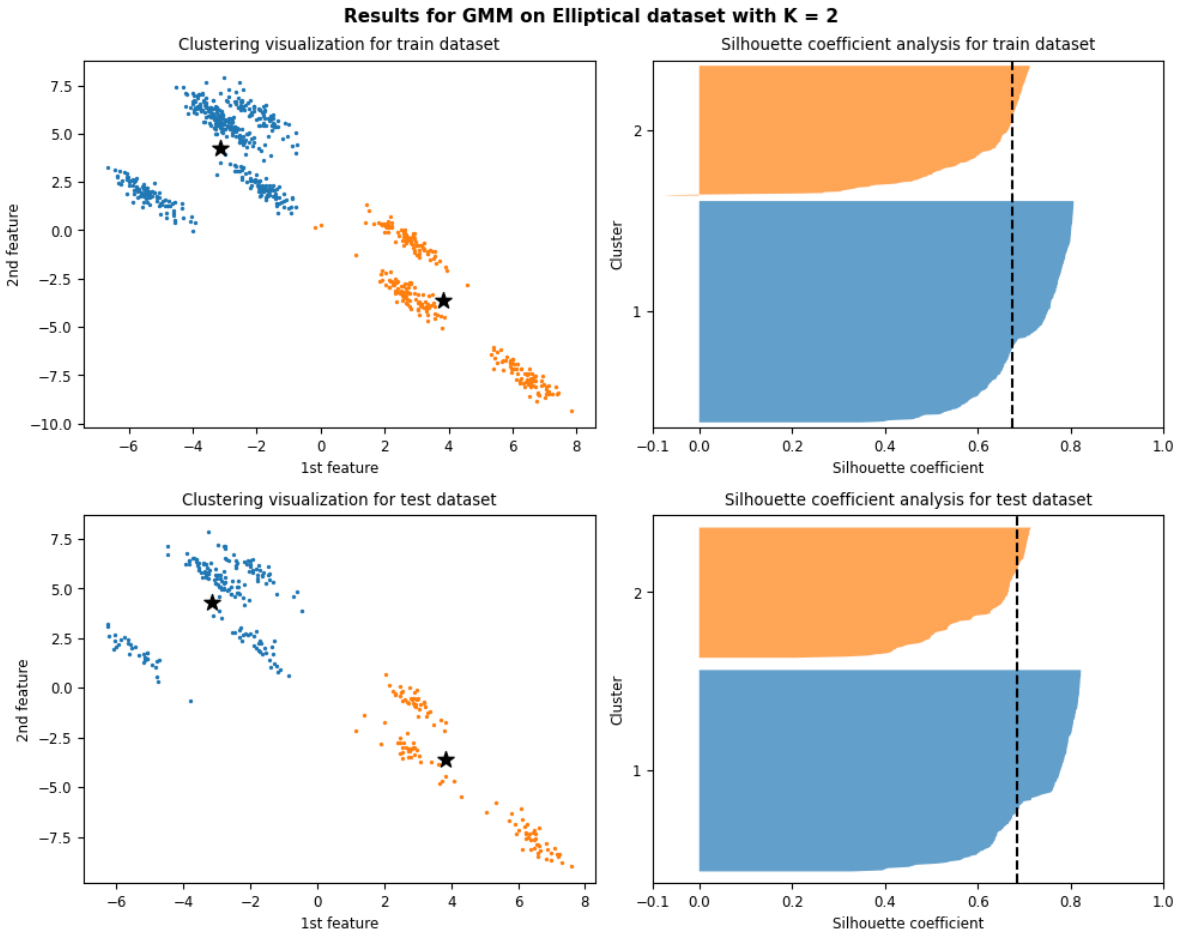


Figure 93: Results and silhouette analysis for running GMM on Elliptical dataset with K = 2

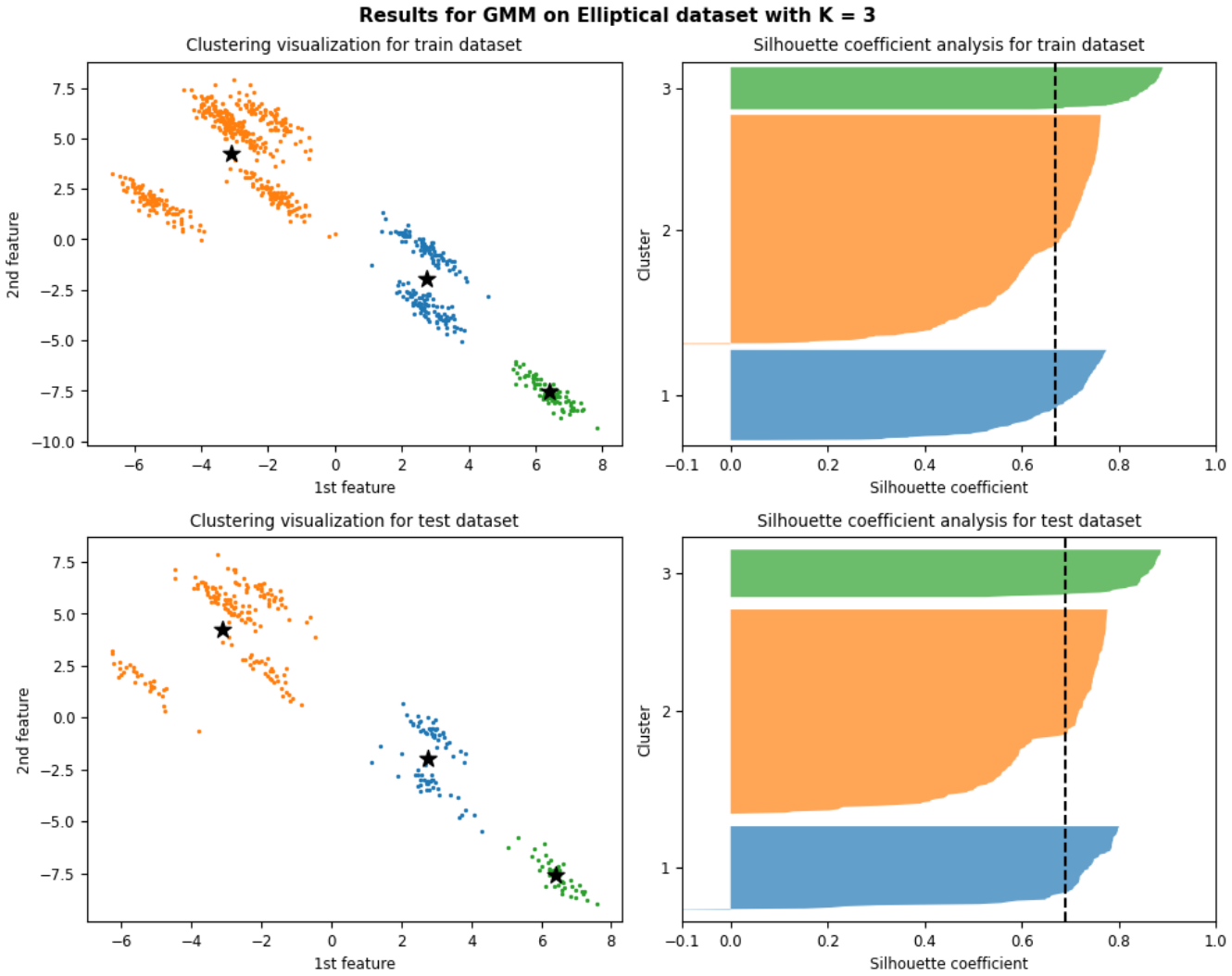


Figure 94: Results and silhouette analysis for running GMM on Elliptical dataset with K = 3

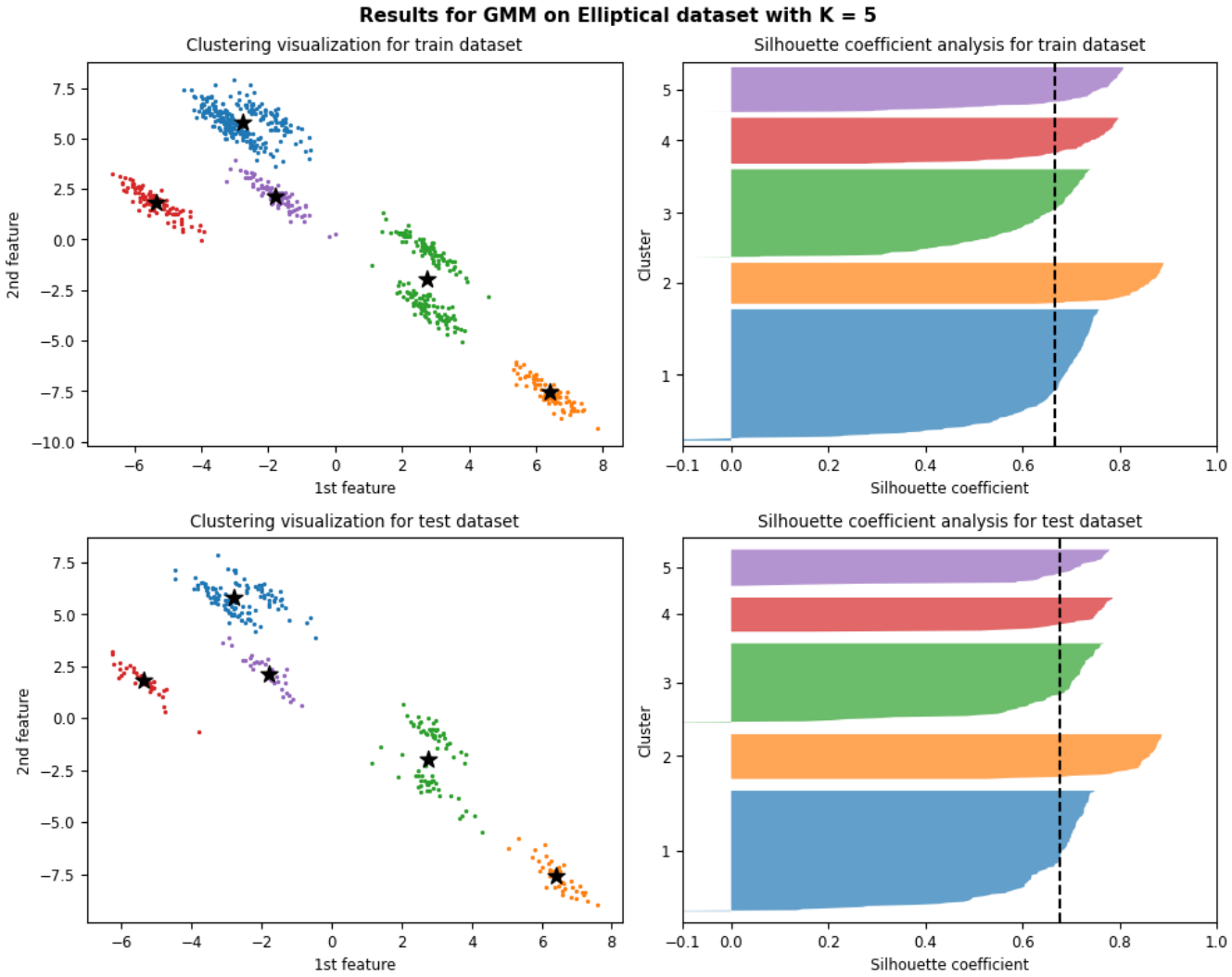


Figure 95: Results and silhouette analysis for running GMM on Elliptical dataset with K = 5

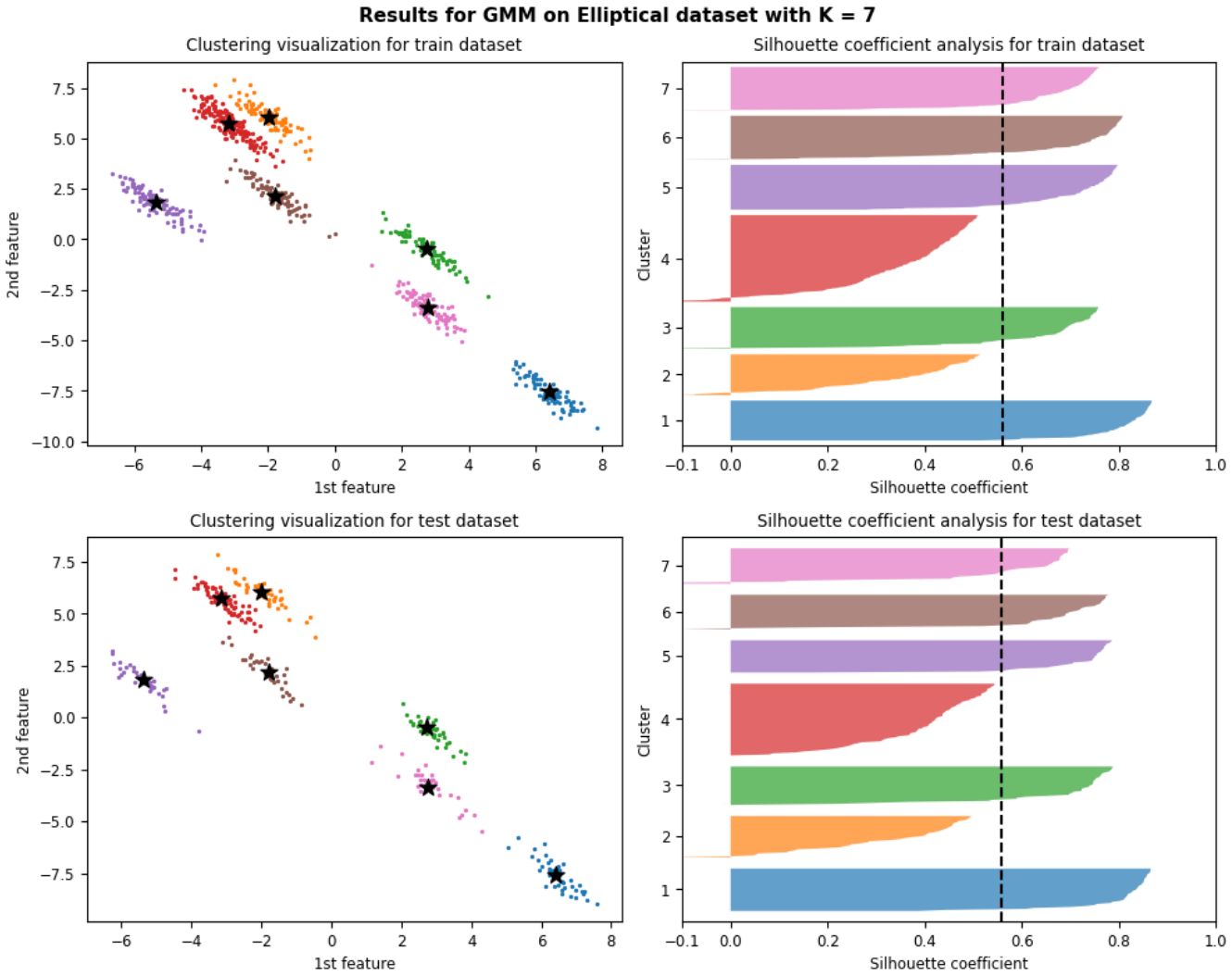


Figure 96: Results and silhouette analysis for running GMM on Elliptical dataset with K = 7

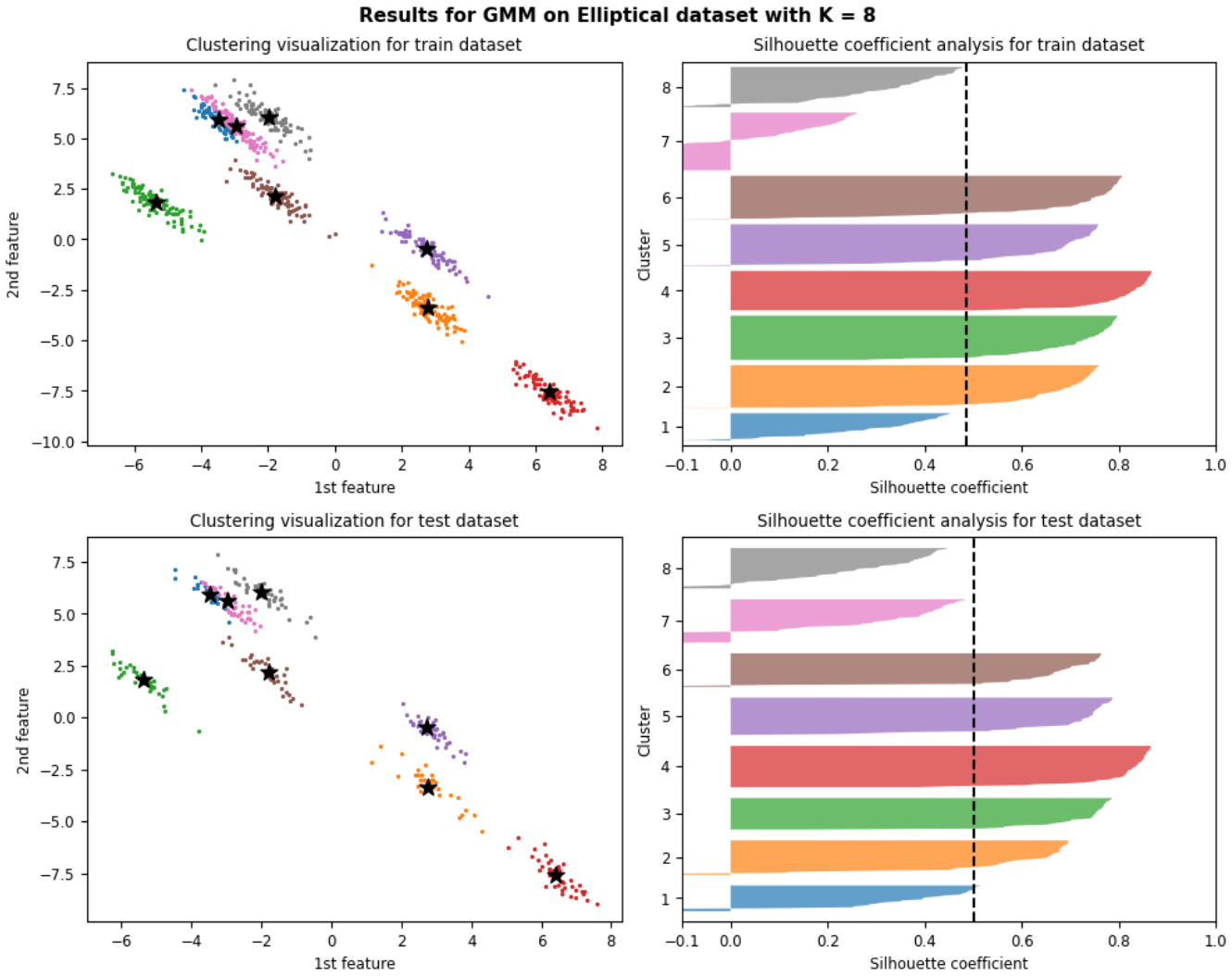


Figure 97: Results and silhouette analysis for running GMM on Elliptical dataset with K = 8

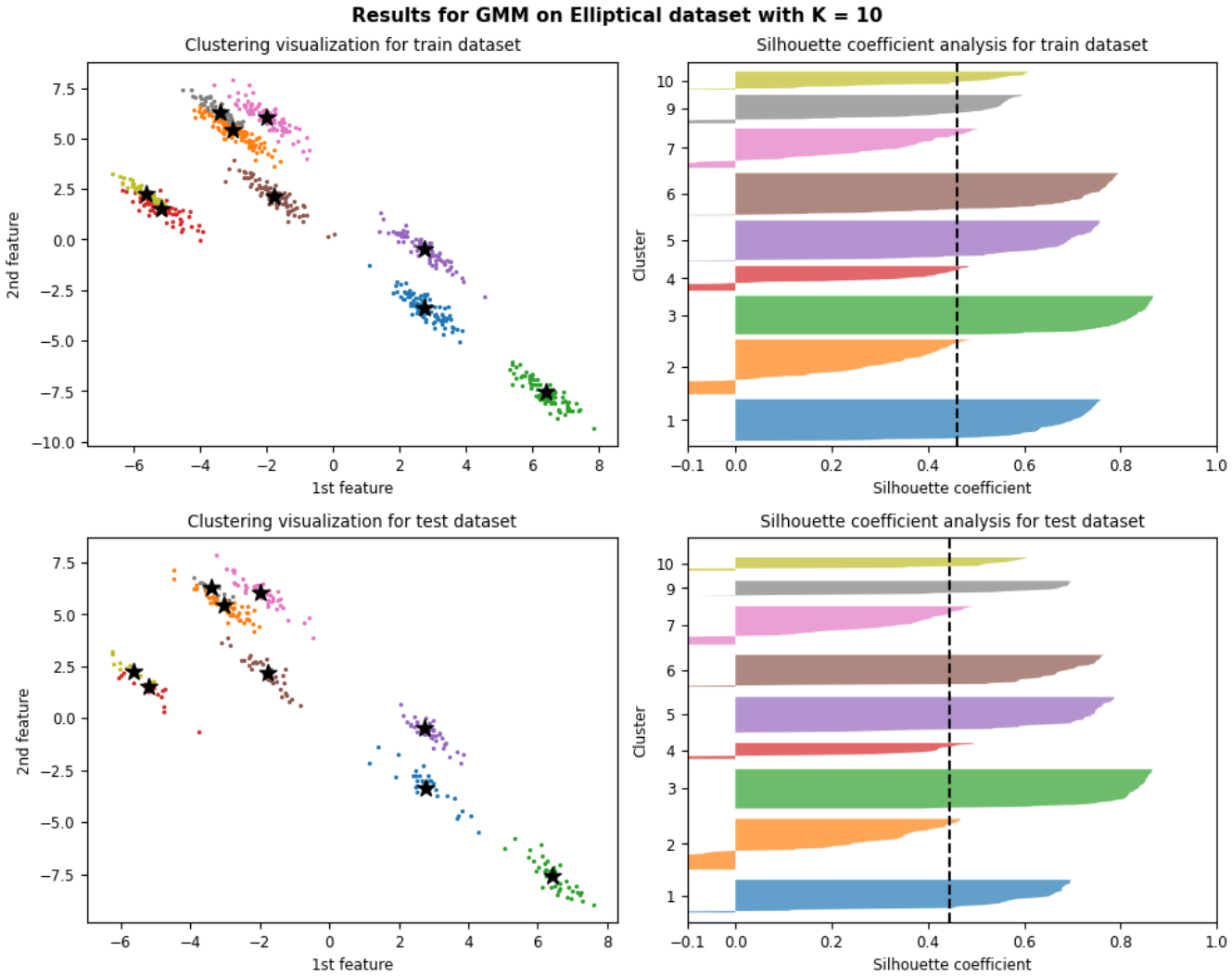
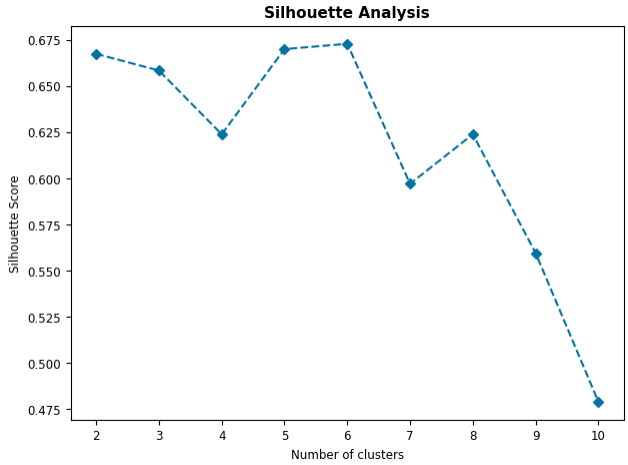
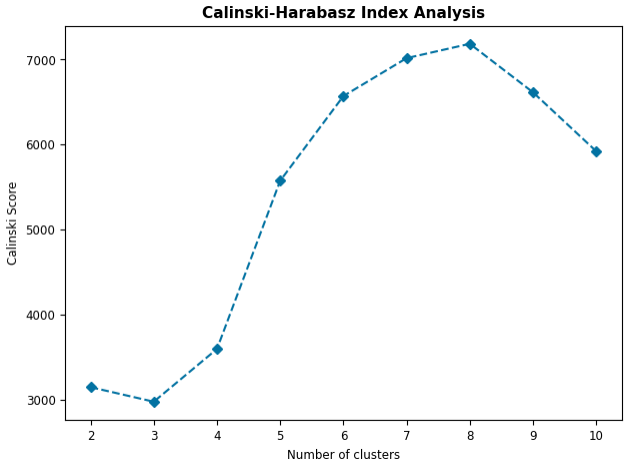


Figure 98: Results and silhouette analysis for running GMM on Elliptical dataset with K = 10

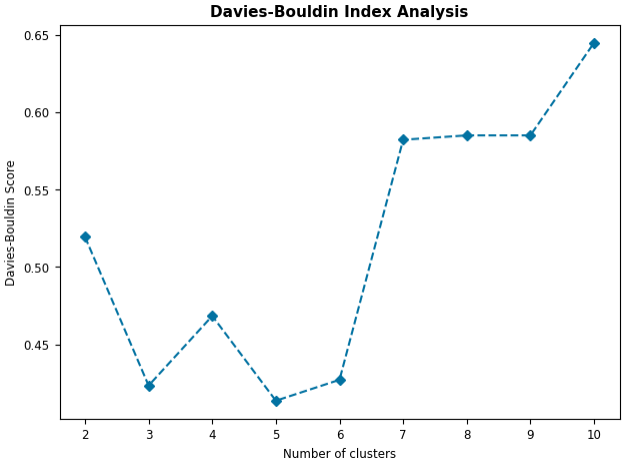
3.3.3.2 Choosing the best K



(a)



(b)



(c)

Figure 99: metrics result (a) Average Silhouette score analysis. (b) Calinski-Harabasz Index Analysis. (c) Davies-Bouldin Index Analysis

By just looking at Figure 96, K = 7 gaussian components had the best results for clustering but in terms of silhouette analysis because cluster 4 and 9 fall before the average score, it is considered a sub optimal number of K. K = 6 & 5 are also a sub optimal in terms of silhouette analysis choice since thickness of clusters is not uniform.

According to Davies-Bouldin index analysis results and is definition, K = 5 is an optimal choice. For Calinski-Harabasz the highest value is K = 8 which we believe is not a rational choice for this dataset. Still K = 5 was chosen two of metrics but we believe K = 7 is best choice by just looking at plots and shows the performance of GMM for these dense clusters.

3.3.4 Elliptical dataset

3.3.4.1 Clustering results and silhouette analysis

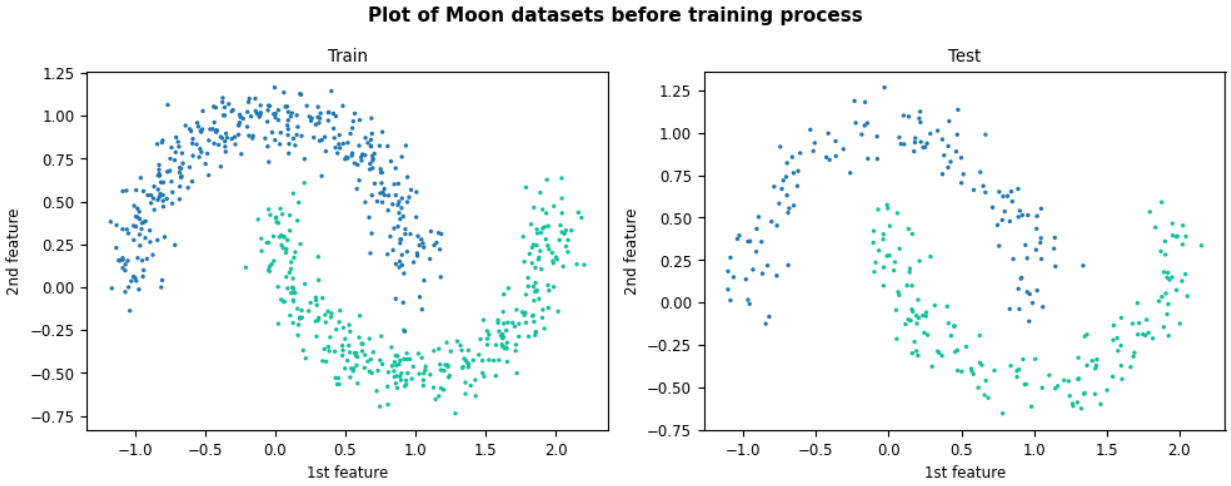


Figure 100: Plot of train & test datasets for Elliptical dataset

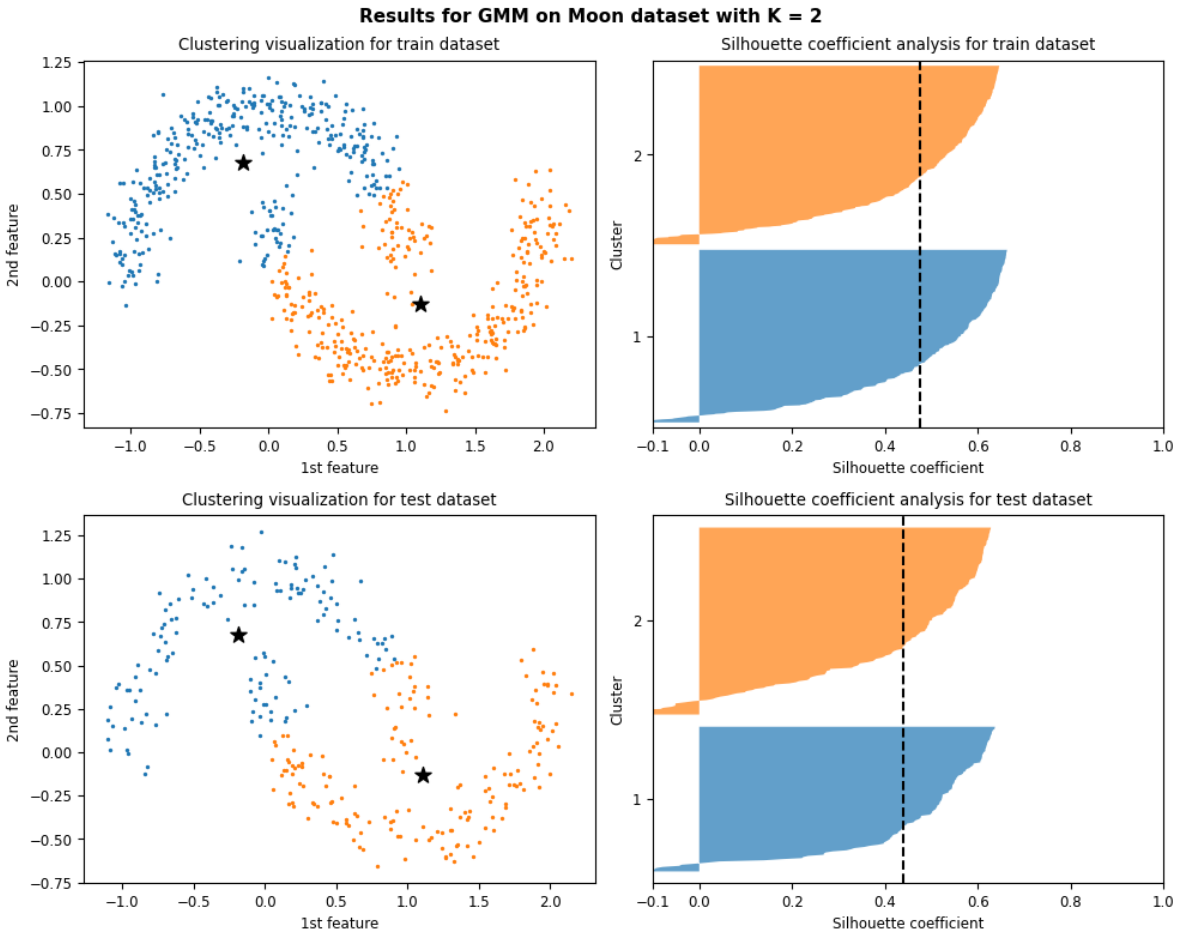


Figure 101: Results and silhouette analysis for running GMM on Elliptical dataset with K = 2

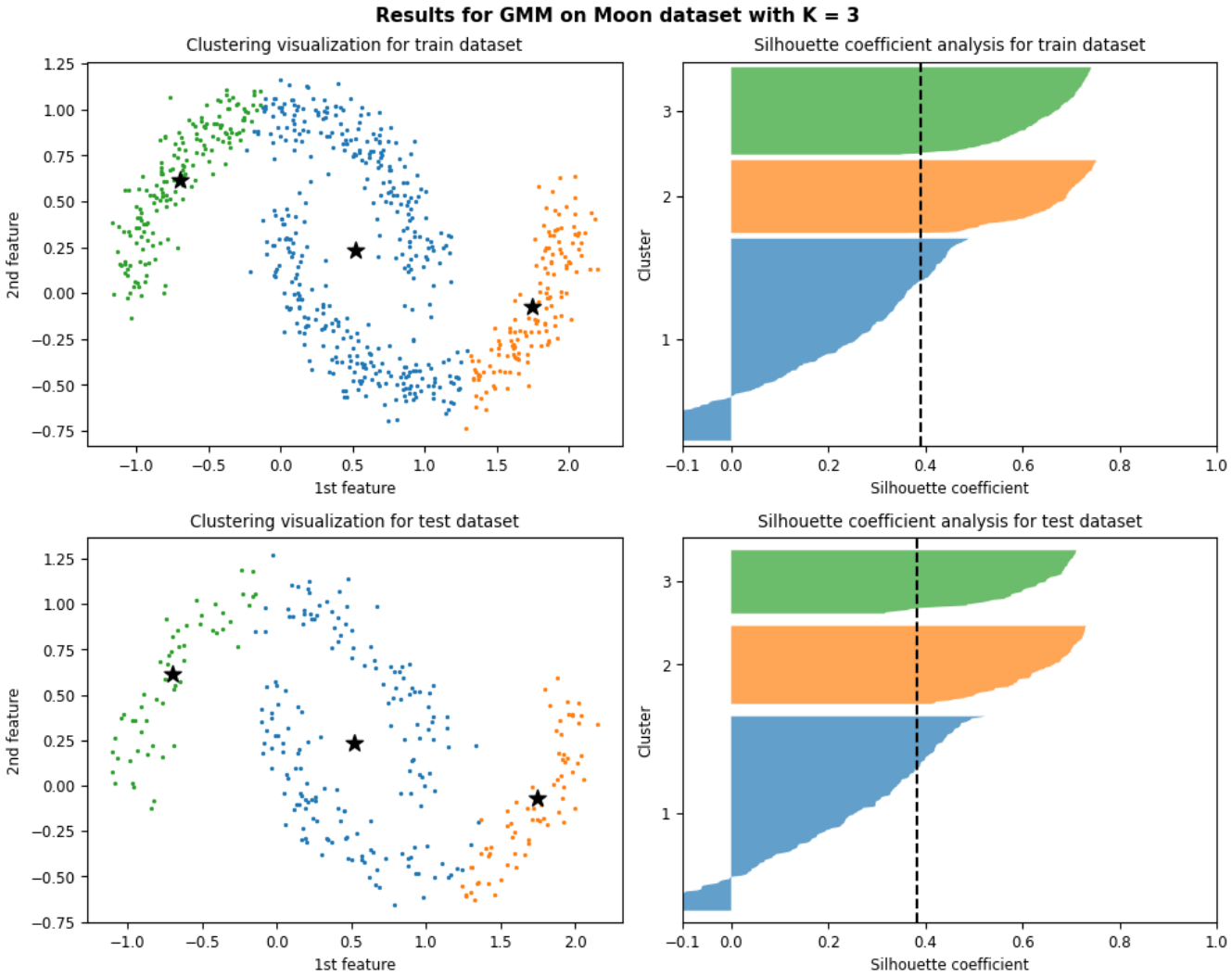


Figure 102: Results and silhouette analysis for running GMM on Elliptical dataset with K = 3

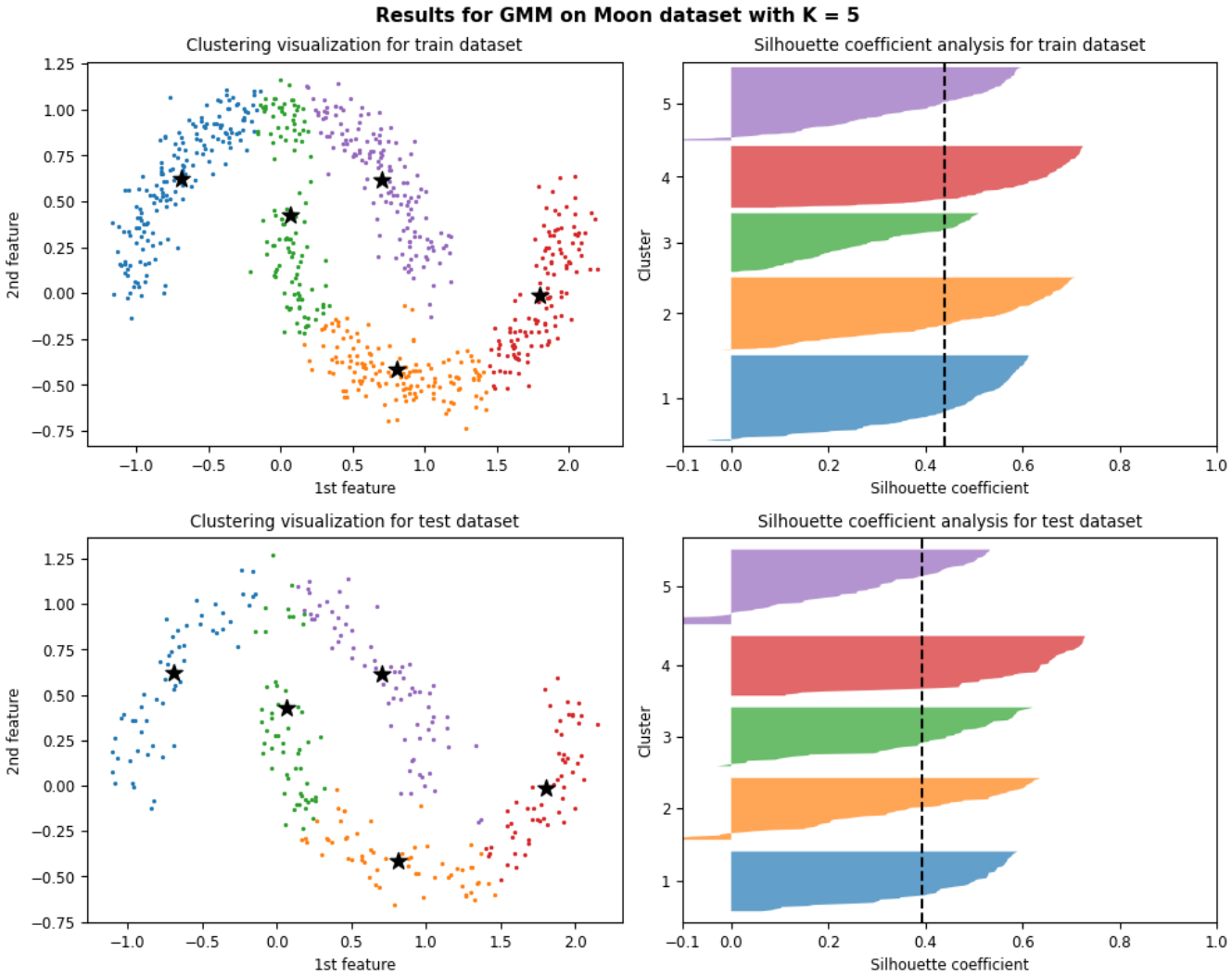


Figure 103: Results and silhouette analysis for running GMM on Elliptical dataset with K = 5

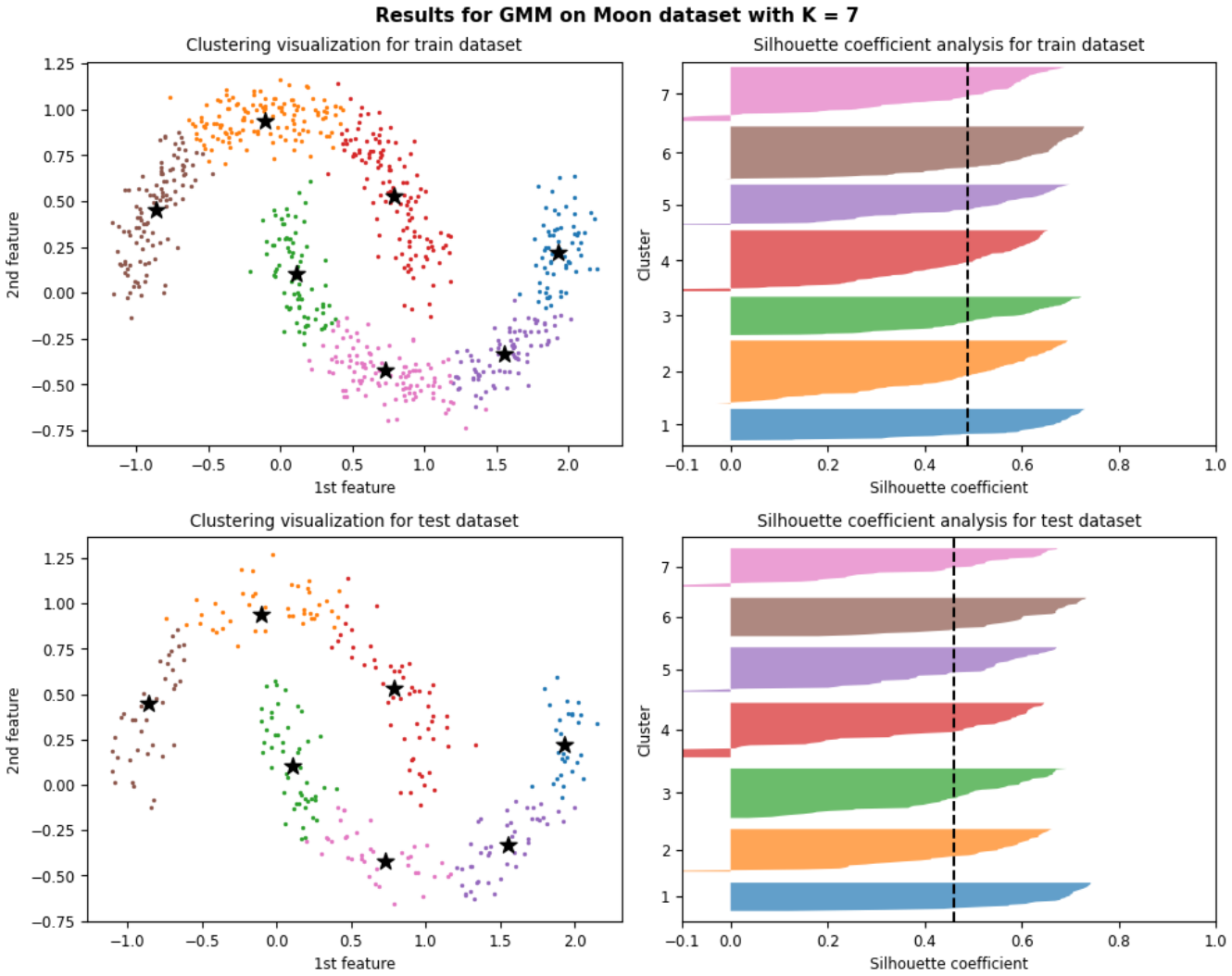


Figure 104: Results and silhouette analysis for running GMM on Elliptical dataset with K = 7

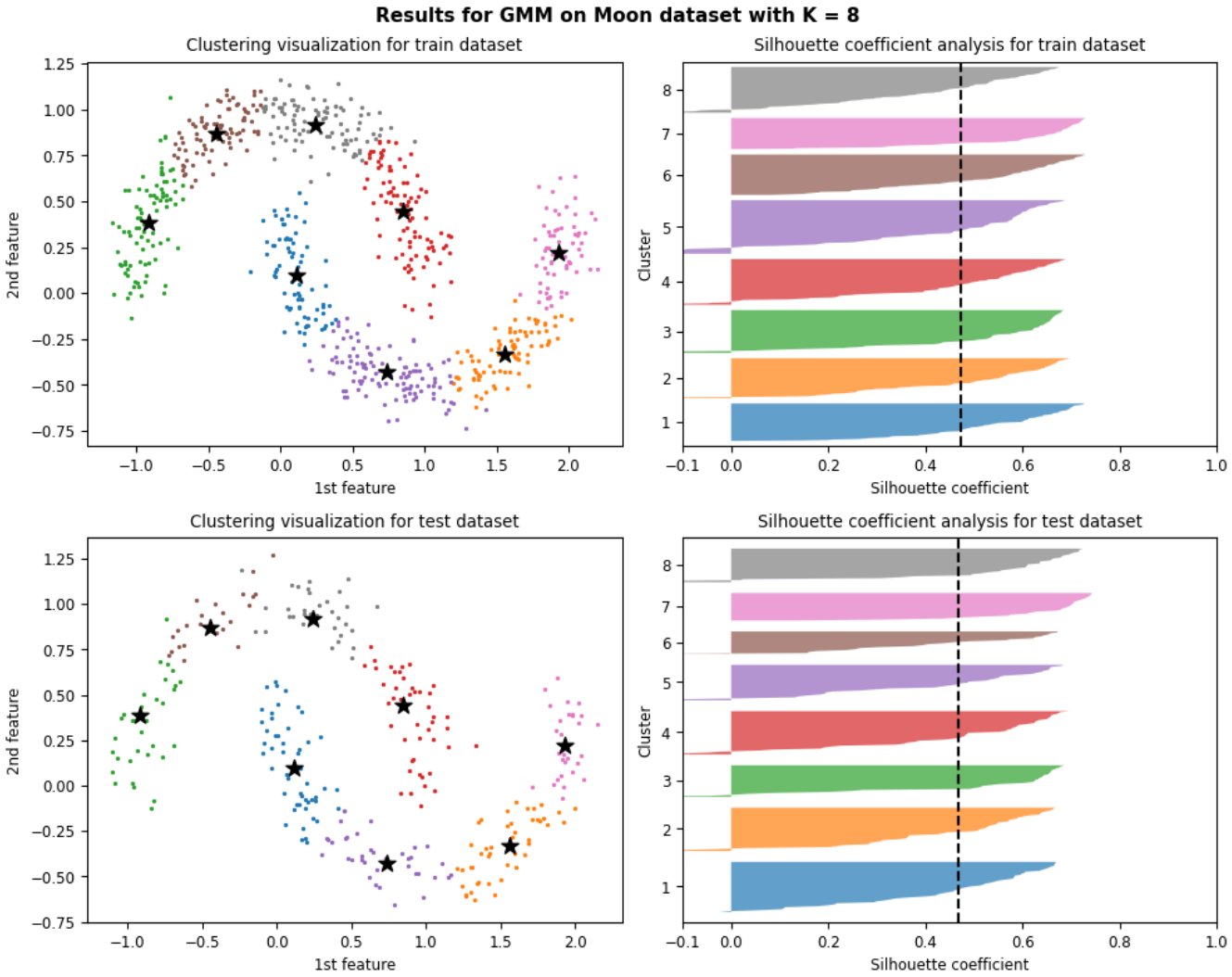


Figure 105: Results and silhouette analysis for running GMM on Elliptical dataset with K = 8

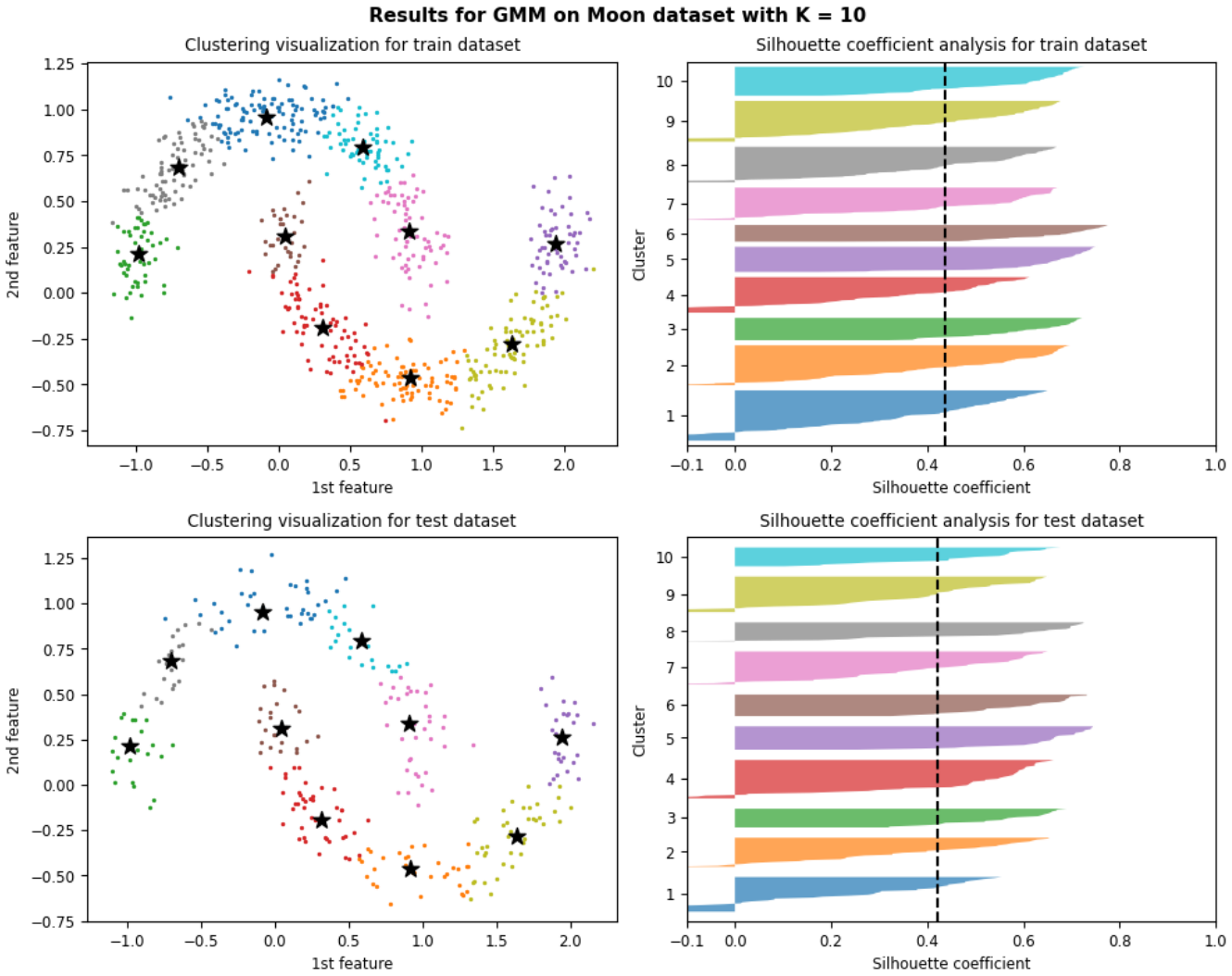
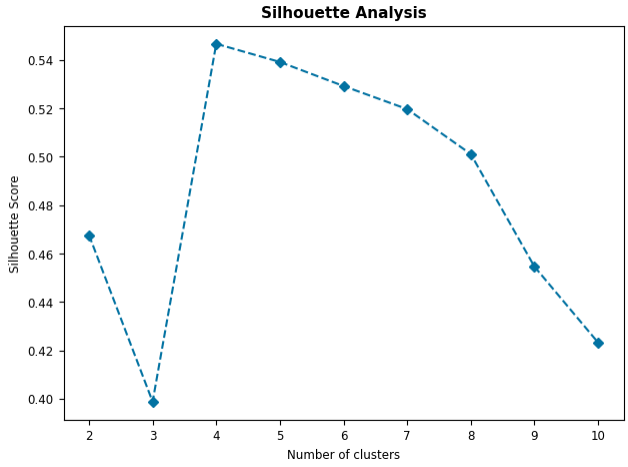
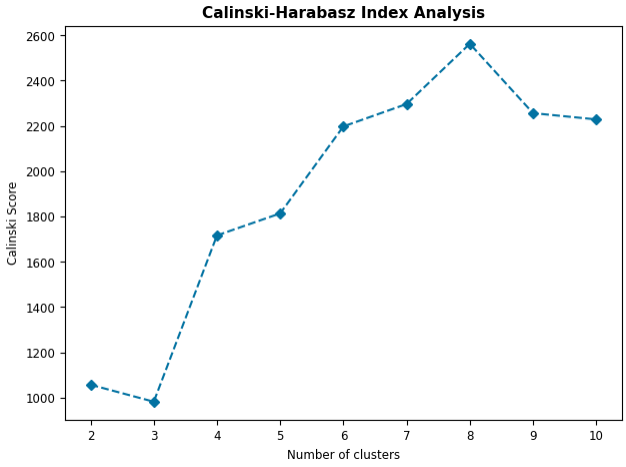


Figure 106: Results and silhouette analysis for running GMM on Elliptical dataset with K = 10

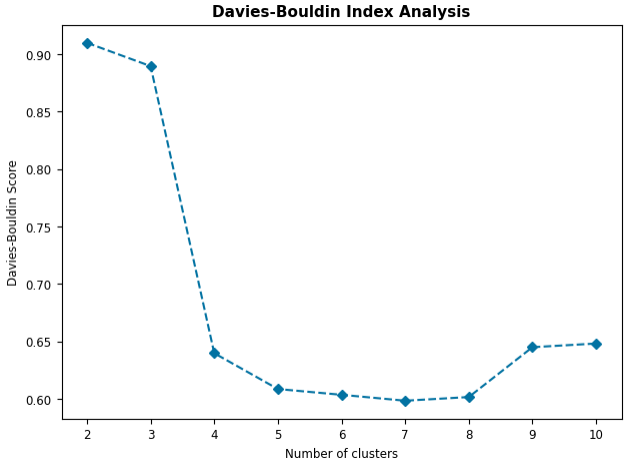
3.3.4.2 Choosing the best K



(a)



(b)



(c)

Figure 107: metrics result (a) Average Silhouette score analysis. (b) Calinski-Harabasz Index Analysis. (c) Davies-Bouldin Index Analysis

Just like k-means case we cannot determine an optimal number of K = gaussian components using Silhouette analysis since nearly for every K, all the criteria are satisfied.

Using Davies-Bouldin index analysis to choose an optimal number of K is also ambiguous since the score value for K = 5, 6, 7 & 8 are so close to each other.

According to Calinski-Harabasz Index analysis results and definition K = 8 can be an optimal answer but still comparing it to original dataset and it's 2 classes, this choice is not the right one. In fact, GMM just like K-means didn't perfume very well on this dataset.

3.3.5 Circle dataset

3.3.5.1 Clustering results and silhouette analysis

Figure 108: Plot of train & test datasets for Circle dataset

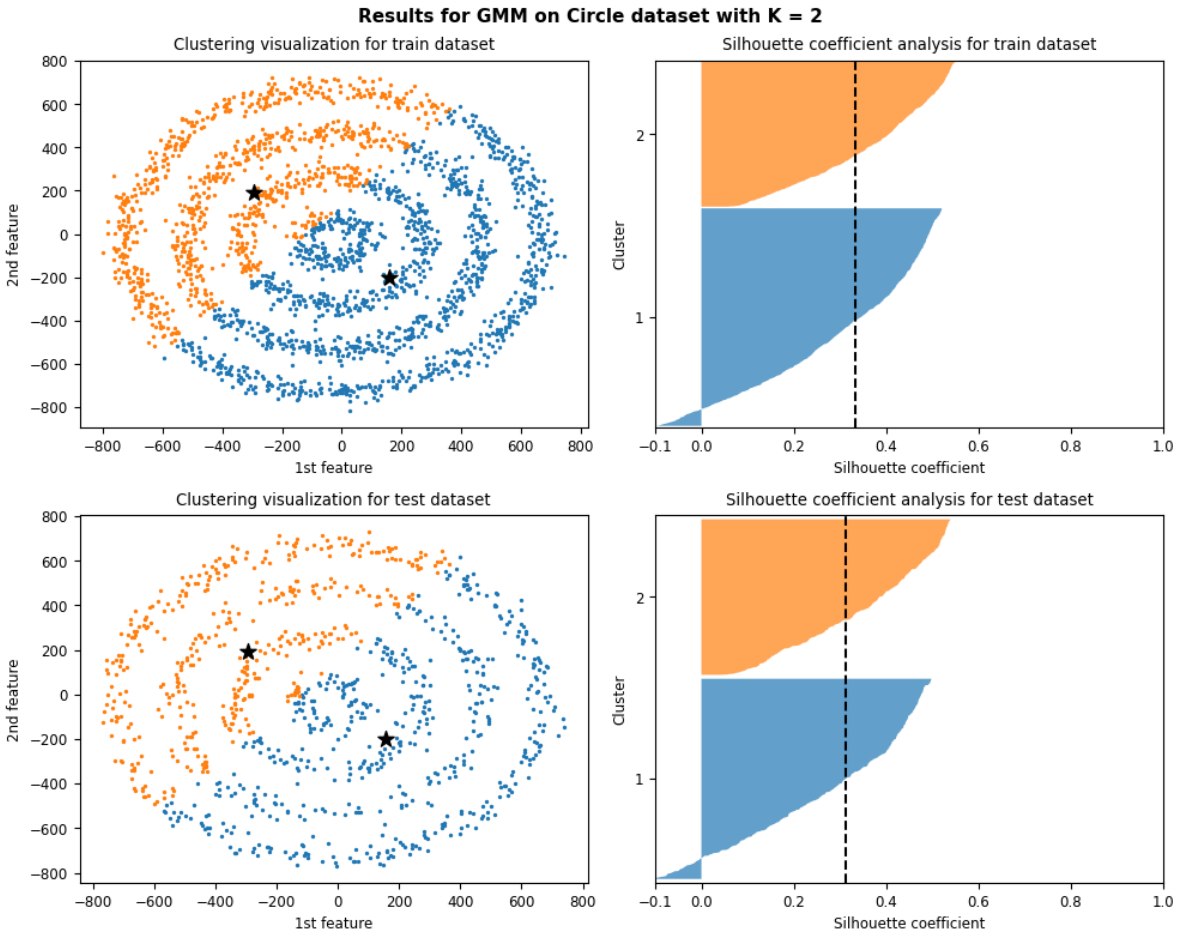


Figure 109: Results and silhouette analysis for running GMM on Circle dataset with K = 2

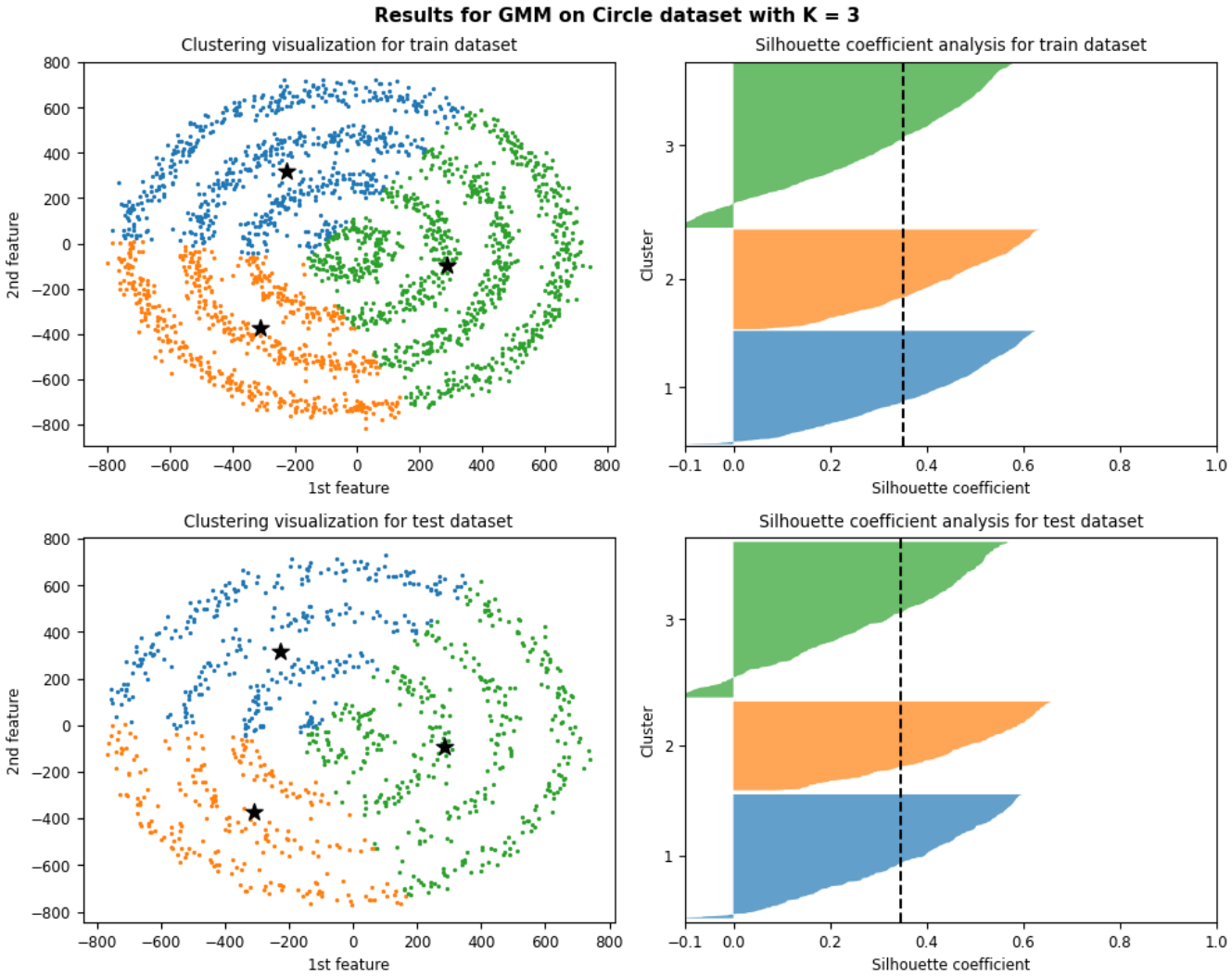


Figure 110: Results and silhouette analysis for running GMM on Circle dataset with K = 3

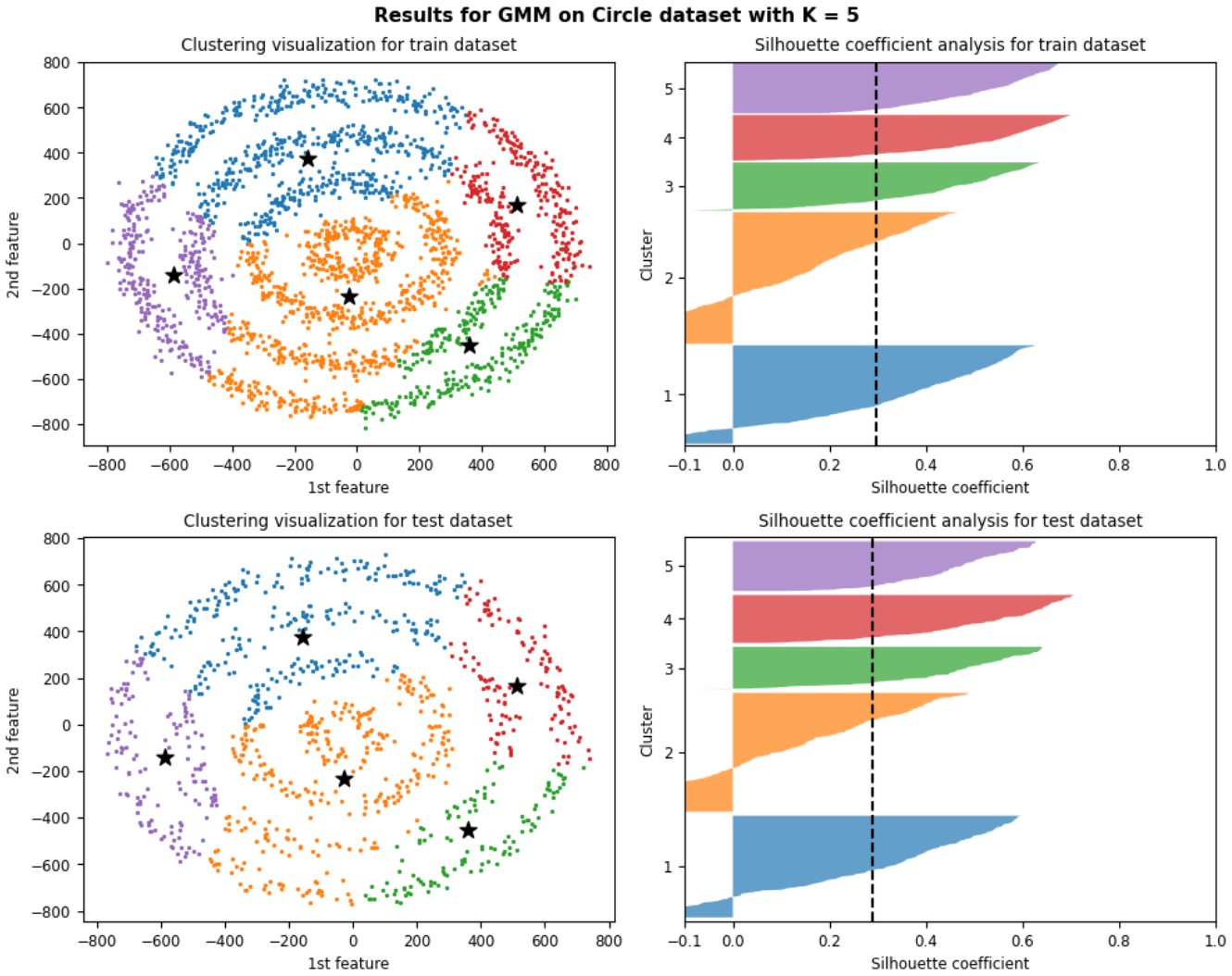


Figure 111: Results and silhouette analysis for running GMM on Circle dataset with K = 5

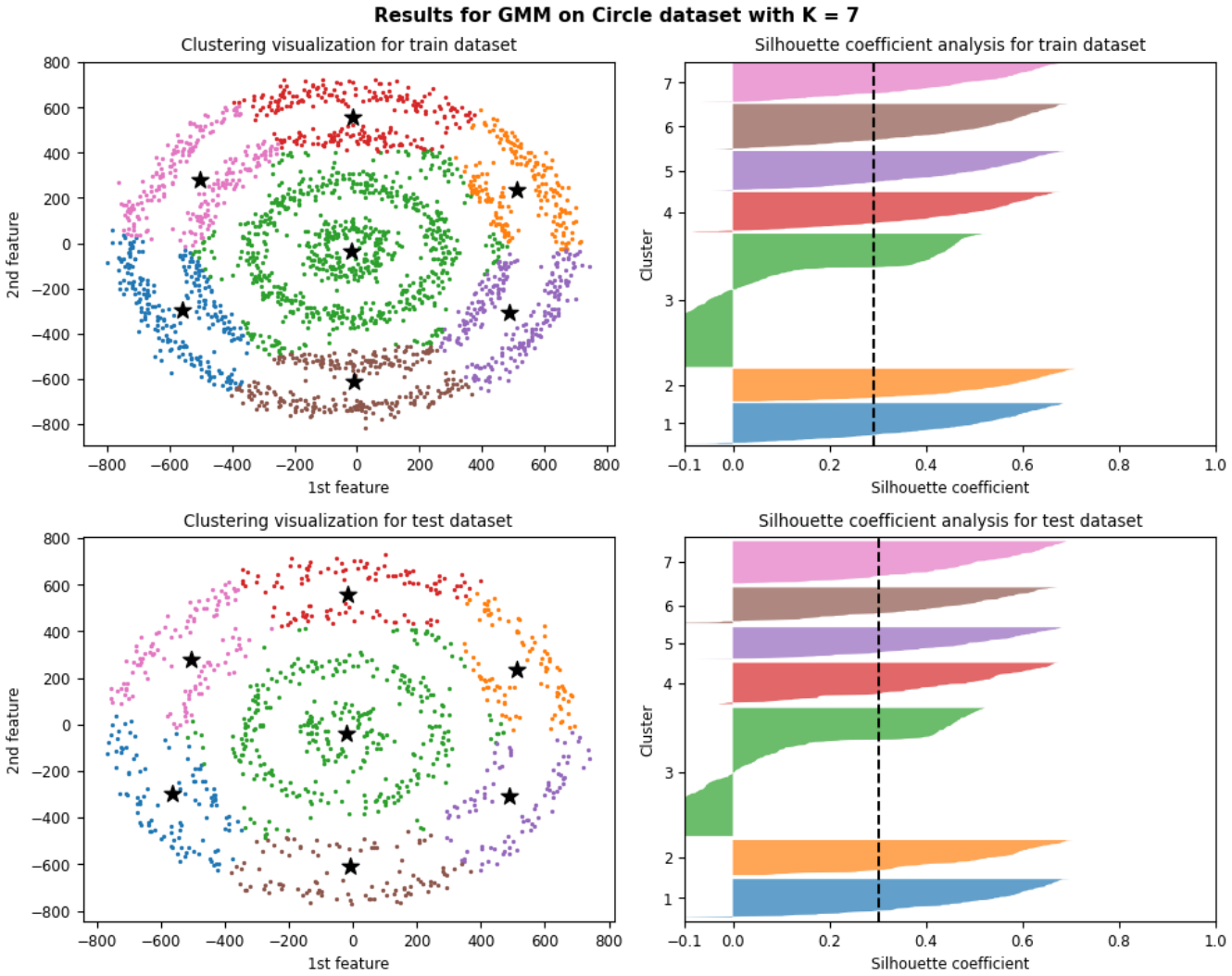


Figure 112: Results and silhouette analysis for running GMM on Circle dataset with K = 7

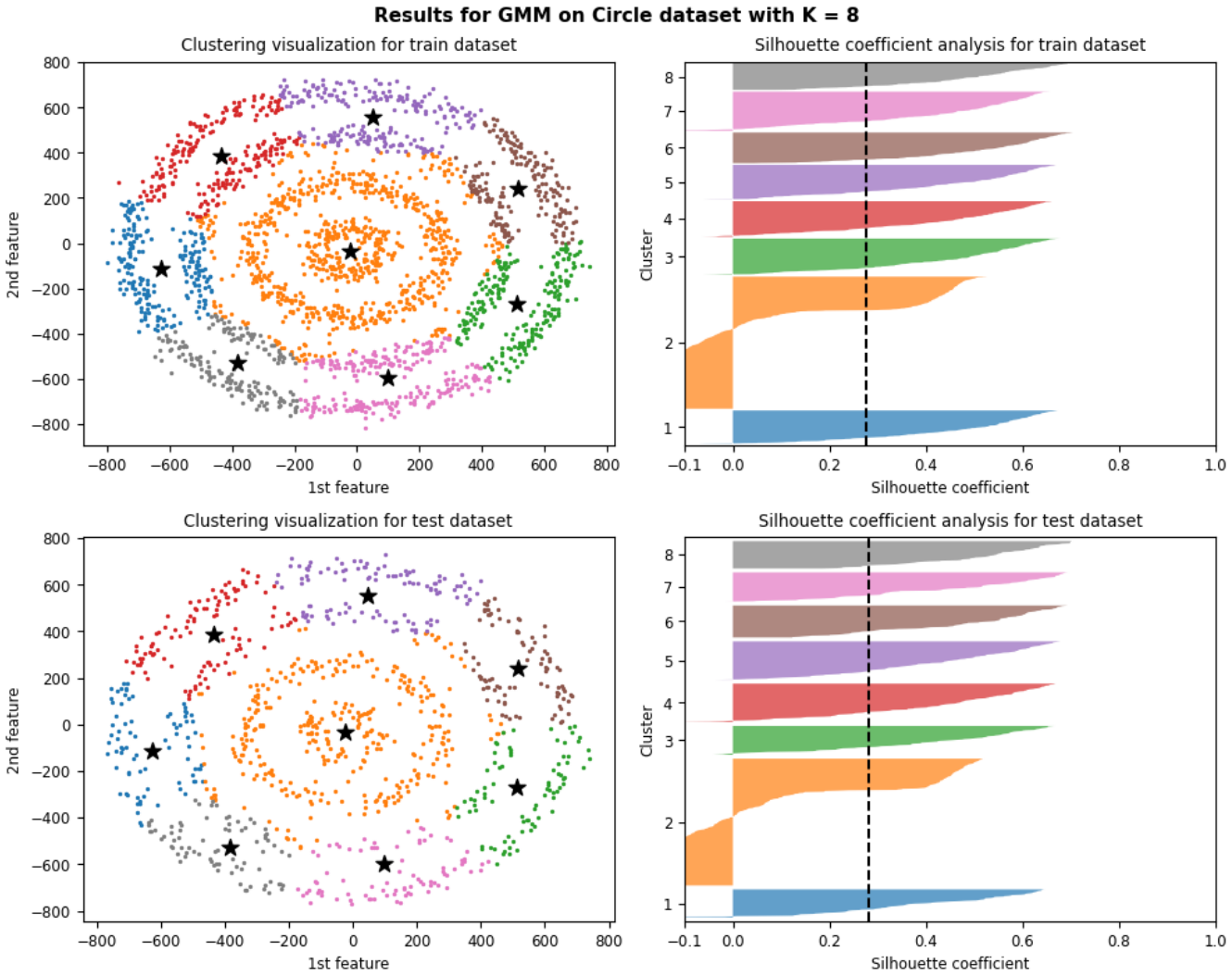


Figure 113: Results and silhouette analysis for running GMM on Circle dataset with K = 8

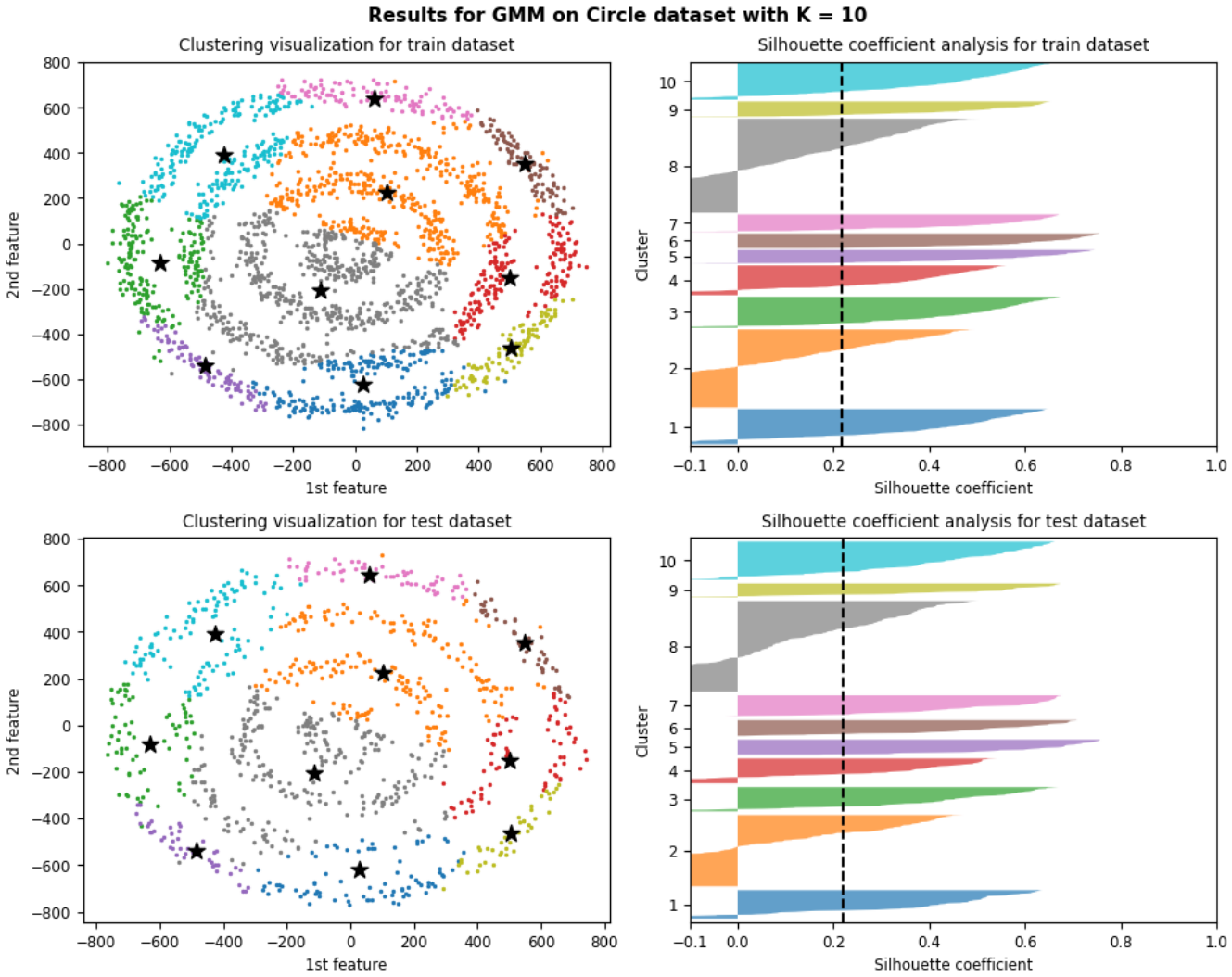


Figure 114: Results and silhouette analysis for running GMM on Circle dataset with K = 10

3.3.5.2 Choosing the best K

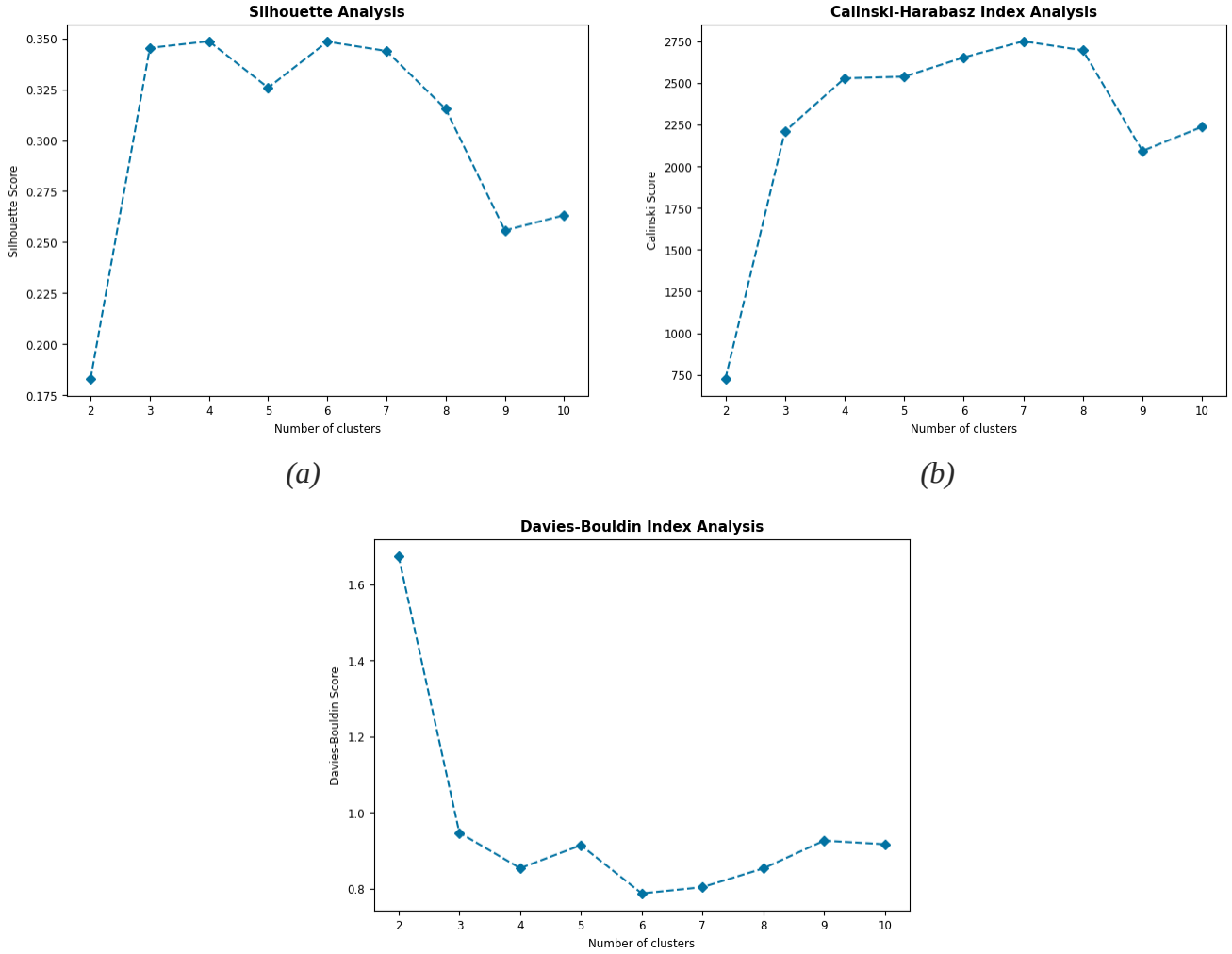


Figure 115: metrics result (a) Average Silhouette score analysis. (b) Calinski-Harabasz Index Analysis. (c) Davies-Bouldin Index Analysis

Just like previous case GMM didn't perform very well in terms of clustering this dataset. None of the metrics could select the right number of clusters for this dataset. We can also see in Silhouette analysis results for different Ks, we have a huge number of negative values which clearly shows some of samples of this cluster might belong to adjacent clusters.

3.4 GMM vs K-Means

We saw that K-means had the best performance on Blobs dataset and it struggled in terms on clustering other datasets. But GMM performed very well not only on Blobs dataset but on Elliptical dataset as well where K-means struggled because of some dense and close clusters. GMM also performed much better than K-means on TSNV dataset in compare to K-means results.

Still, both of these algorithms struggled to cluster samples in Moon and Circle datasets.

We concluded that Gaussian mixture models can handle even very oblong clusters. In K-means, data point is deterministically assigned to one and only one cluster, but in reality, there may be overlapping between the cluster GMM provide us the probabilities of the data point belonging to each of the possible clusters.

4 SVM

A supervised machine learning model called a support vector machine (SVM) employs classification techniques to solve two-group classification problems. An SVM model can classify new text after being given sets of labeled training data for each category.

They offer two key advantages over more recent algorithms like neural networks: incredible speed and improved performance with fewer samples (in the thousands). As a result, the approach is excellent for text classification issues, where it's typical to only have access to a dataset with a few thousand tags on each sample.

This part aims to classify two datasets with SVM with linear and non-linearly.

4.1 Linear SVM implementation

To implement linear svm, *SVMclassifier* class has been created. This class has a constructor method to initiate the “C, W, and b” parameters. After that, we have the *Calcloss* method to calculate the amount of our loss or error.

To train our model, we have a *fit* method, which gets X, Y, size, alpha, and iterations as arguments and will introduce our linear SVM model based on mathematic formulas.

In the end, it returns weights and bias, and losses.

We use the *hypothesis* function to predict the labels of the data; this function uses the sigmoid method to predict.

We used numbers 1, 100, and 1000 for C hyperparameters to model our train data.

The results are as follows

4.2 C = 1

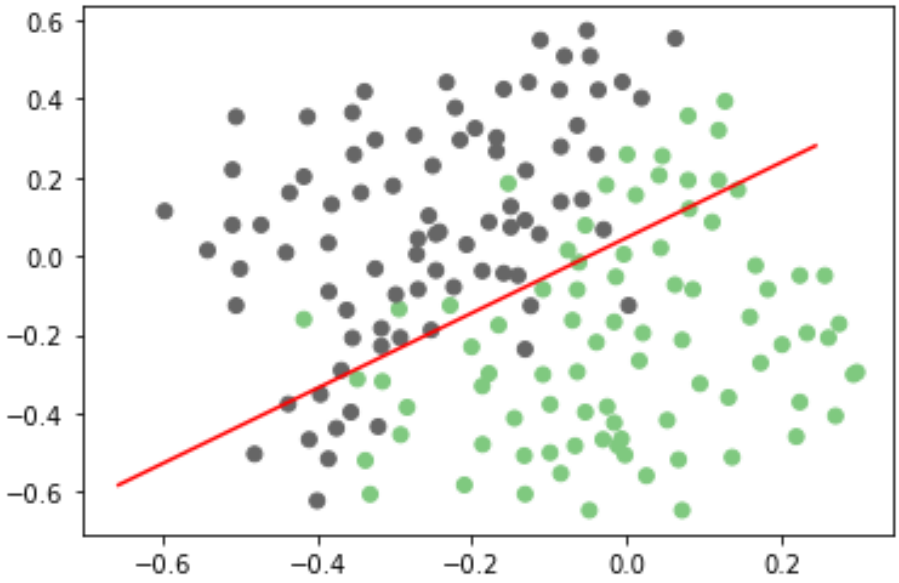


Figure 116: C=1 train plot

The accuracy for c1 in training data is 83.43 %

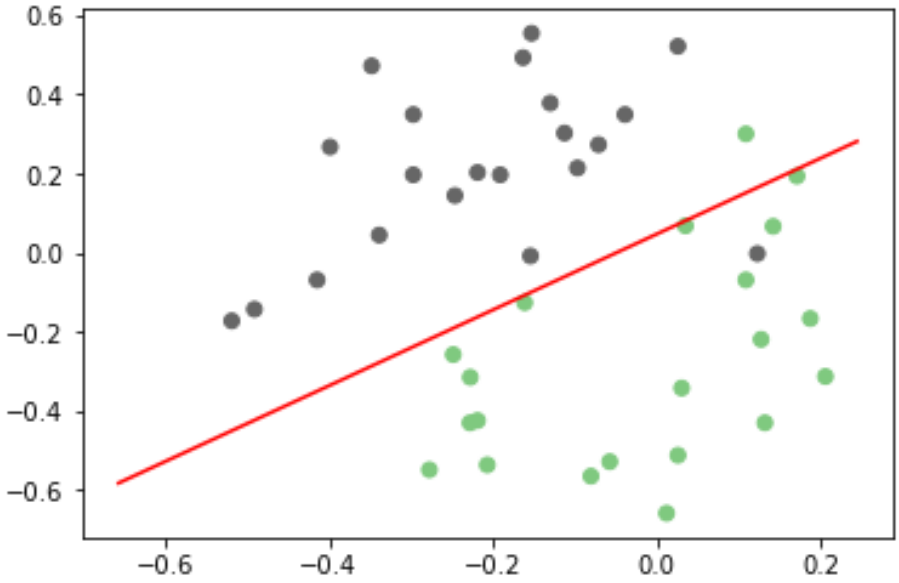


Figure 117: C=1 test plot

The accuracy for c=1 in test data is 95.23%%

4.3 C = 10

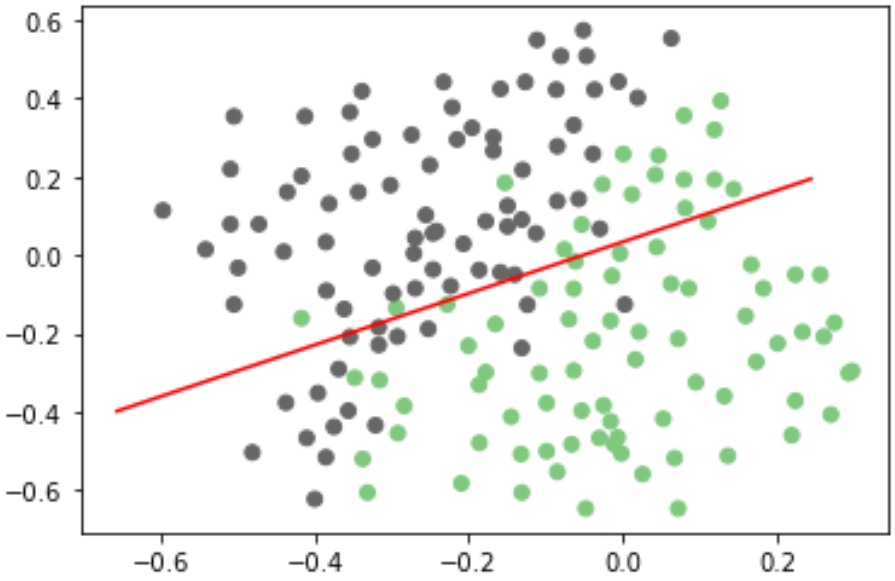


Figure 118: C=10 train plot

The accuracy for c=10 in training data is 79.28 %

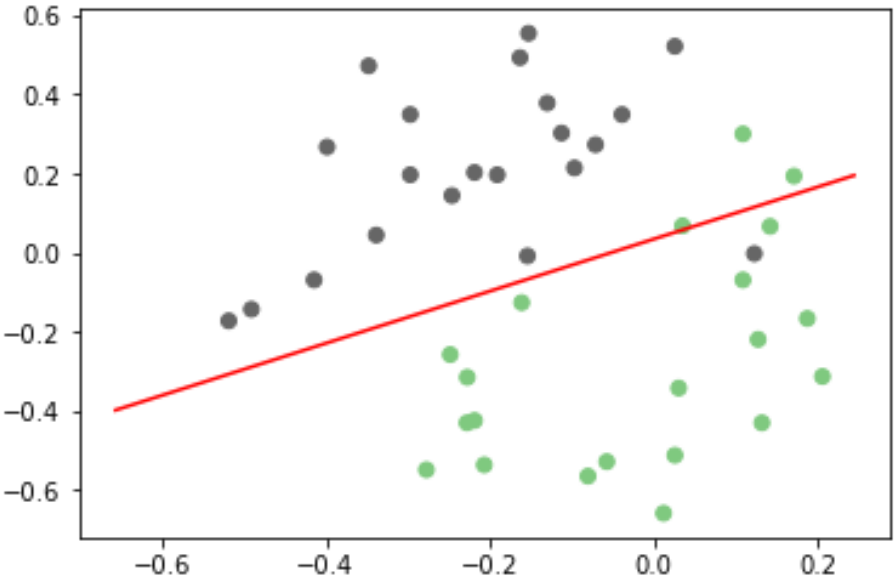


Figure 119: C=10 test plot

The accuracy for c=10 in test data is 90.47 %

4.4 C = 100

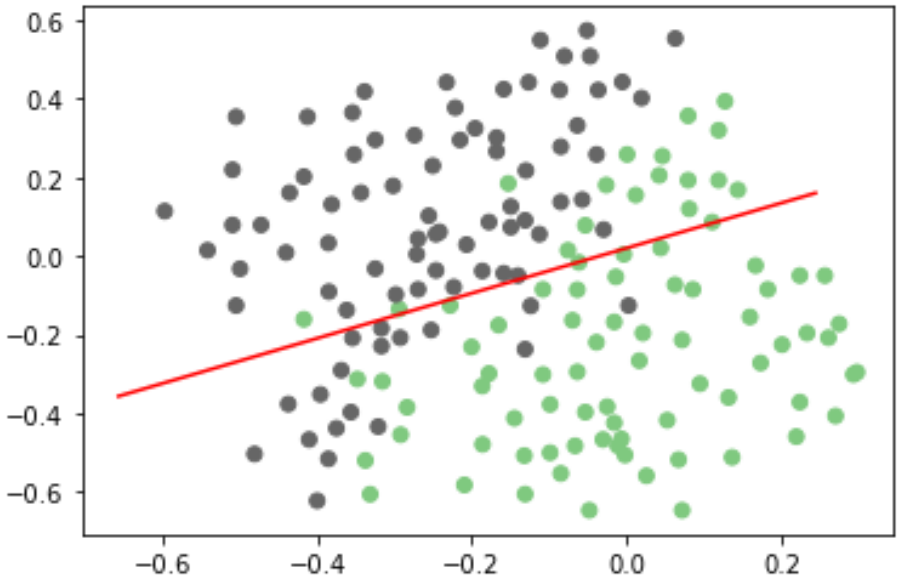


Figure 120: C=100 train plot

The accuracy for c=100 in training data is 78.69 %

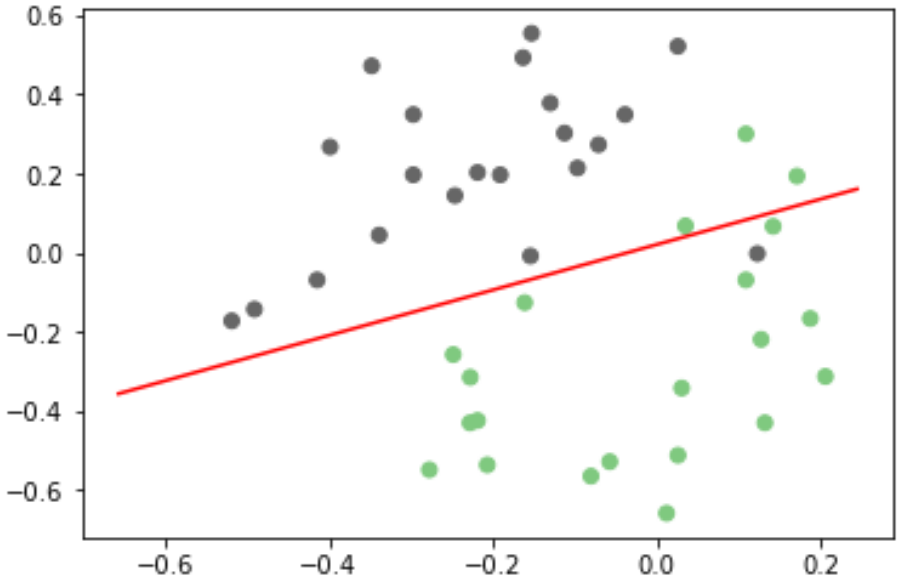
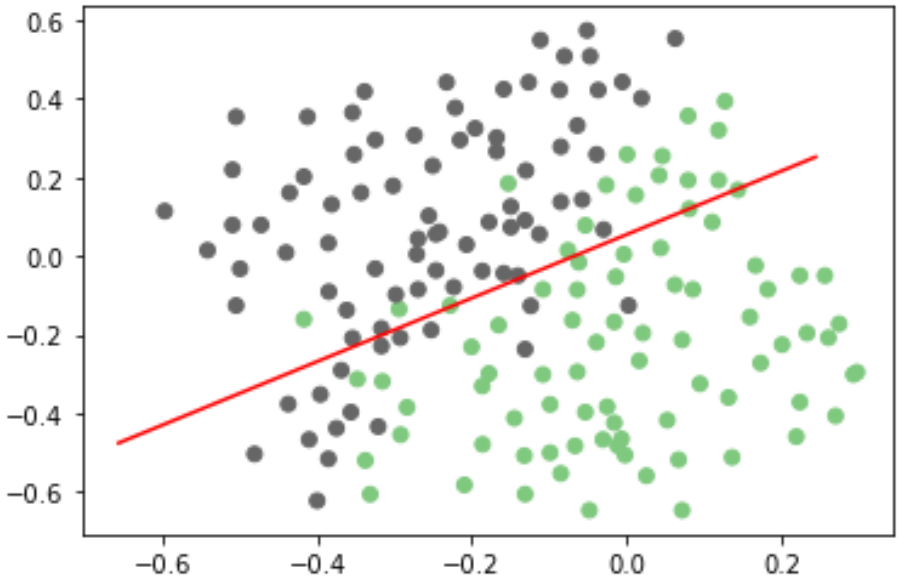


Figure 121: C=100 test plot

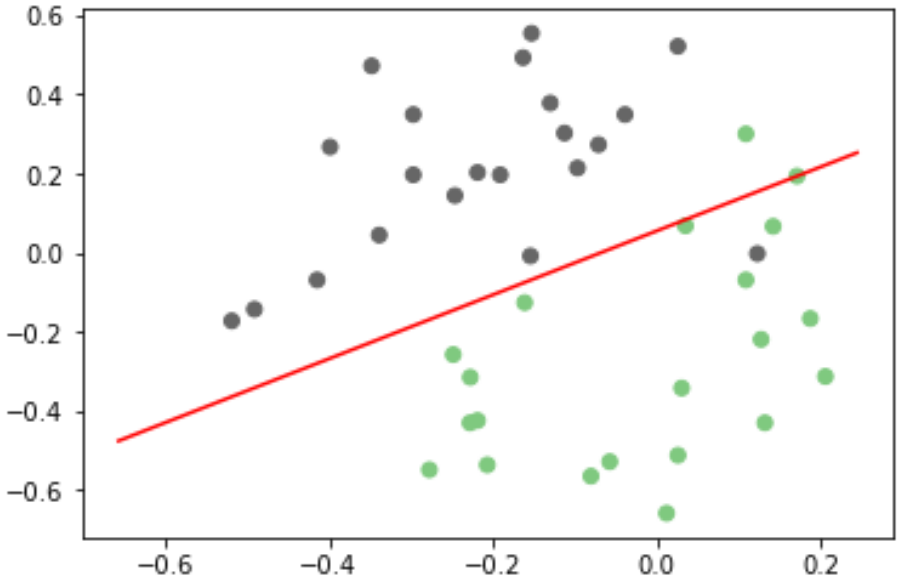
The accuracy for c=100 in test data is 90.4 %

4.5 C = 1000



C=1000 train plot

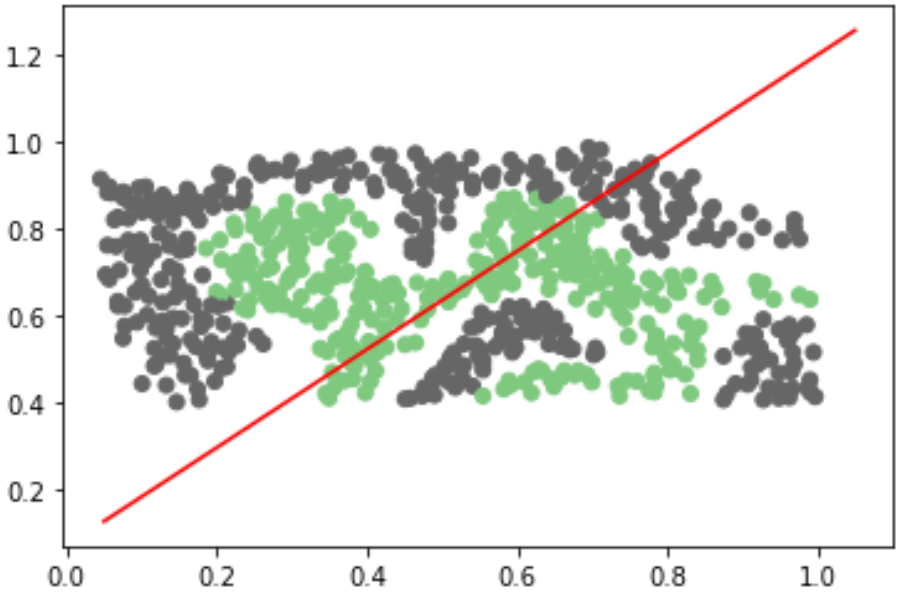
The accuracy for c=1000 in training data is 81.06 %



C=1000 test plot

The accuracy for c=1000 in test data is 92.85 %

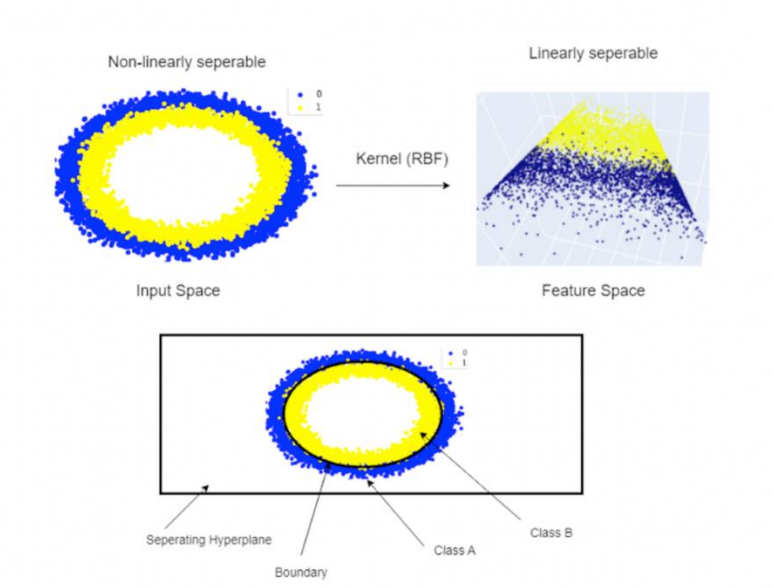
4.5.1 Linear SVM on Dataset2



Linear SVM is not an excellent choice to use for this dataset, so we use non-linear SVM.

4.6 Non-linear SVM implementation

RBF, short for **Radial Basis Function Kernel,** is a very powerful kernel used in SVM. Unlike linear or polynomial kernels, RBF is more complex and efficient because it can combine multiple polynomial kernels multiple times of different degrees to project the non-linearly separable data into higher dimensional space so that it can be separable using a hyperplane.



To implement the RBF kernel SVM, the *SVM* class has been created. This class has a constructor method to initialize "C, kernel, sigma, iterations number.".

The class uses the *CalcLoss* method to calculate the loss in our model. And like any other model, it has a *fit* method to train our model, and it will calculate the weights, bias, and loss.

The prediction method predicts the labels of data and uses the *decide* model.

4.6.1 Get parameters

We used numbers "0.01,0.03,0.1,0.3,1,3,10,30" for our C and sigma, and we used our data to model on each c and sigma to get the best number for c and sigma for each dataset.

```
46- C = 3 ,Sigma = 3, train accuracy = 82.03 ,test accuracy = 82.66 ,objective =  206.02
47- C = 3 ,Sigma = 10, train accuracy = 92.17 ,test accuracy = 92.49 ,objective =  230.9
48- C = 3 ,Sigma = 30, train accuracy = 98.55 ,test accuracy = 99.42 ,objective =  247.68
49- C = 10 ,Sigma = 0.01, train accuracy = 58.99 ,test accuracy = 61.27 ,objective =  150.89
50- C = 10 ,Sigma = 0.03, train accuracy = 59.57 ,test accuracy = 61.85 ,objective =  152.34
51- C = 10 ,Sigma = 0.1, train accuracy = 66.81 ,test accuracy = 70.52 ,objective =  172.59
52- C = 10 ,Sigma = 0.3, train accuracy = 78.7 ,test accuracy = 82.08 ,objective =  201.82
53- C = 10 ,Sigma = 1, train accuracy = 80.58 ,test accuracy = 81.5 ,objective =  202.83
54- C = 10 ,Sigma = 3, train accuracy = 83.91 ,test accuracy = 82.08 ,objective =  207.03
55- C = 10 ,Sigma = 10, train accuracy = 93.77 ,test accuracy = 94.22 ,objective =  235.1
56- C = 10 ,Sigma = 30, train accuracy = 99.57 ,test accuracy = 98.84 ,objective =  247.83
57- C = 30 ,Sigma = 0.01, train accuracy = 58.84 ,test accuracy = 61.85 ,objective =  151.62
58- C = 30 ,Sigma = 0.03, train accuracy = 62.17 ,test accuracy = 64.74 ,objective =  159.28
59- C = 30 ,Sigma = 0.1, train accuracy = 75.07 ,test accuracy = 78.03 ,objective =  192.12
60- C = 30 ,Sigma = 0.3, train accuracy = 79.57 ,test accuracy = 80.92 ,objective =  200.95
61- C = 30 ,Sigma = 1, train accuracy = 81.16 ,test accuracy = 80.92 ,objective =  202.55
62- C = 30 ,Sigma = 3, train accuracy = 84.2 ,test accuracy = 83.24 ,objective =  209.06
63- C = 30 ,Sigma = 10, train accuracy = 95.36 ,test accuracy = 94.22 ,objective =  236.69
64- C = 30 ,Sigma = 30, train accuracy = 99.57 ,test accuracy = 98.84 ,objective =  247.83
```

Figure 122: Results *of calculating the best sigma and C for Dataset2*

```
40- C = 1 ,Sigma = 30, train accuracy = 94.08 ,test accuracy = 92.86 ,objective =  233.37
41- C = 3 ,Sigma = 0.01, train accuracy = 79.88 ,test accuracy = 88.1 ,objective =  212.02
42- C = 3 ,Sigma = 0.03, train accuracy = 84.62 ,test accuracy = 88.1 ,objective =  216.76
43- C = 3 ,Sigma = 0.1, train accuracy = 86.98 ,test accuracy = 90.48 ,objective =  222.7
44- C = 3 ,Sigma = 0.3, train accuracy = 91.12 ,test accuracy = 92.86 ,objective =  230.41
45- C = 3 ,Sigma = 1, train accuracy = 91.12 ,test accuracy = 95.24 ,objective =  233.98
46- C = 3 ,Sigma = 3, train accuracy = 92.9 ,test accuracy = 95.24 ,objective =  235.76
47- C = 3 ,Sigma = 10, train accuracy = 92.9 ,test accuracy = 92.86 ,objective =  232.19
48- C = 3 ,Sigma = 30, train accuracy = 94.08 ,test accuracy = 92.86 ,objective =  233.37
49- C = 10 ,Sigma = 0.01, train accuracy = 85.21 ,test accuracy = 88.1 ,objective =  217.35
50- C = 10 ,Sigma = 0.03, train accuracy = 86.98 ,test accuracy = 90.48 ,objective =  222.7
51- C = 10 ,Sigma = 0.1, train accuracy = 91.12 ,test accuracy = 95.24 ,objective =  233.98
52- C = 10 ,Sigma = 0.3, train accuracy = 91.72 ,test accuracy = 95.24 ,objective =  234.57
53- C = 10 ,Sigma = 1, train accuracy = 91.72 ,test accuracy = 95.24 ,objective =  234.57
54- C = 10 ,Sigma = 3, train accuracy = 93.49 ,test accuracy = 95.24 ,objective =  236.35
55- C = 10 ,Sigma = 10, train accuracy = 93.49 ,test accuracy = 92.86 ,objective =  232.78
56- C = 10 ,Sigma = 30, train accuracy = 94.08 ,test accuracy = 92.86 ,objective =  233.37
57- C = 30 ,Sigma = 0.01, train accuracy = 86.98 ,test accuracy = 90.48 ,objective =  222.7
58- C = 30 ,Sigma = 0.03, train accuracy = 90.53 ,test accuracy = 95.24 ,objective =  233.39
59- C = 30 ,Sigma = 0.1, train accuracy = 91.72 ,test accuracy = 95.24 ,objective =  234.57
60- C = 30 ,Sigma = 0.3, train accuracy = 92.31 ,test accuracy = 95.24 ,objective =  235.16
61- C = 30 ,Sigma = 1, train accuracy = 93.49 ,test accuracy = 95.24 ,objective =  236.35
62- C = 30 ,Sigma = 3, train accuracy = 94.08 ,test accuracy = 95.24 ,objective =  236.94
63- C = 30 ,Sigma = 10, train accuracy = 92.9 ,test accuracy = 92.86 ,objective =  232.19
64- C = 30 ,Sigma = 30, train accuracy = 94.08 ,test accuracy = 92.86 ,objective =  233.37
```

Figure 123: Results of calculating the best sigma and C for Dataset1

The best C and sigma have been used to run a 10-fold cross-validation on both datasets. The accuracies are as follows.

```
********
fold1 ==> 🛠train accuracy = 94.71 ,🔬test accuracy = 100.0
************
********
fold2 ==> 🛠train accuracy = 95.77 ,🔬test accuracy = 76.19
************
********
fold3 ==> 🛠train accuracy = 94.18 ,🔬test accuracy = 90.48
************
********
fold4 ==> 🛠train accuracy = 93.65 ,🔬test accuracy = 100.0
************
********
fold5 ==> 🛠train accuracy = 93.65 ,🔬test accuracy = 100.0
************
********
fold6 ==> 🛠train accuracy = 93.65 ,🔬test accuracy = 90.48
************
********
fold7 ==> 🛠train accuracy = 93.65 ,🔬test accuracy = 95.24
************
********
fold8 ==> 🛠train accuracy = 93.65 ,🔬test accuracy = 76.19
************
********
fold9 ==> 🛠train accuracy = 93.12 ,🔬test accuracy = 100.0
************
********
fold10 ==> 🛠train accuracy = 95.24 ,🔬test accuracy = 80.95
************
```

Figure 124: Dataset1, best sigma, and C using *10-folds*

Plots for each fold on dataset 2 are as follows:

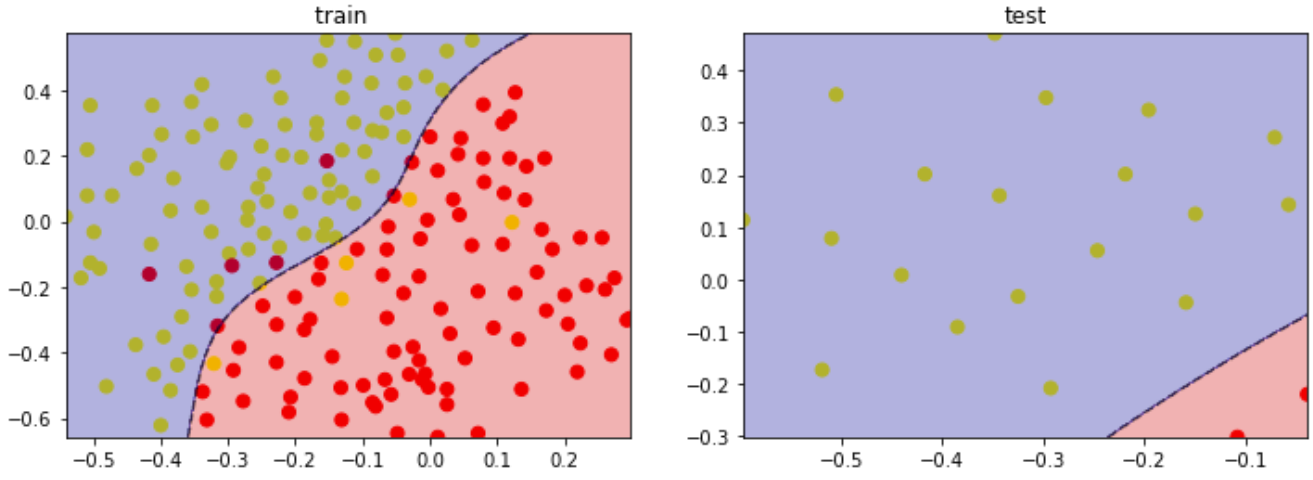


Figure 125: Plots for 1st fold – dataset 1

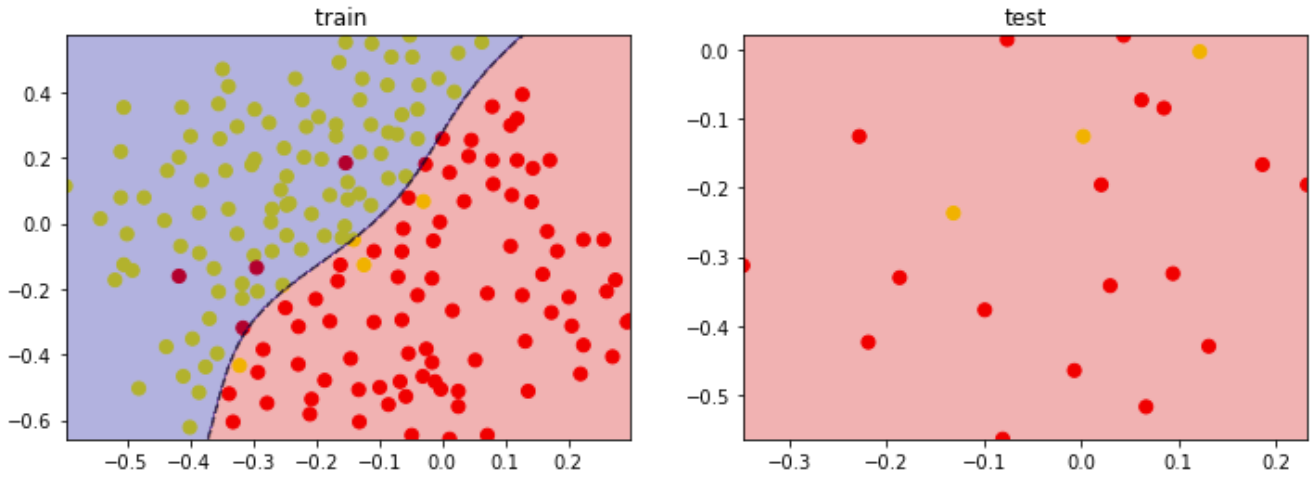


Figure 126: Plots for 2nd fold – dataset 1

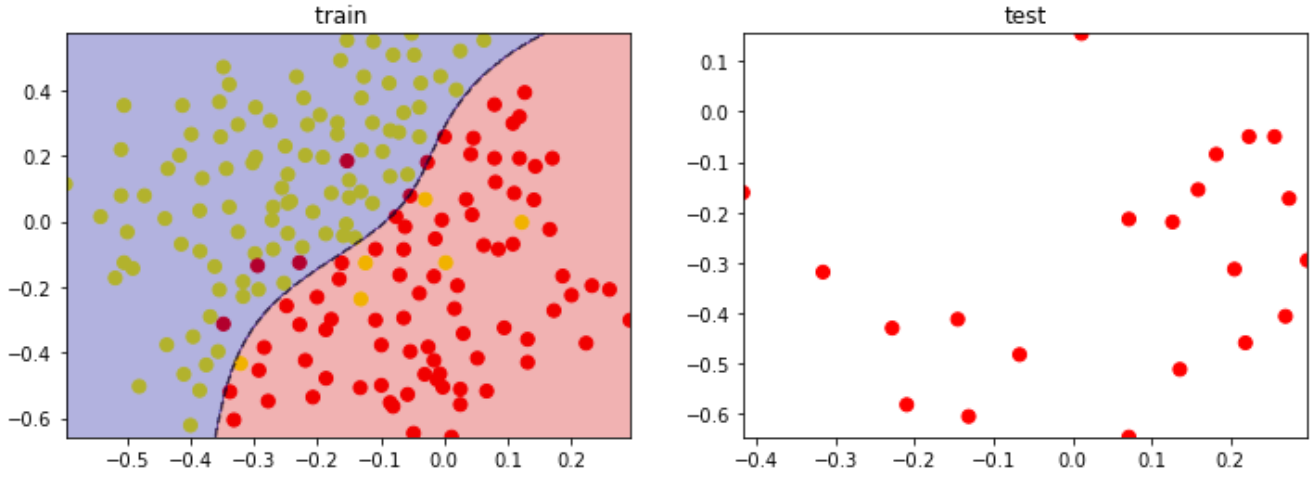


Figure 127: Plots for 3rd fold – dataset 1

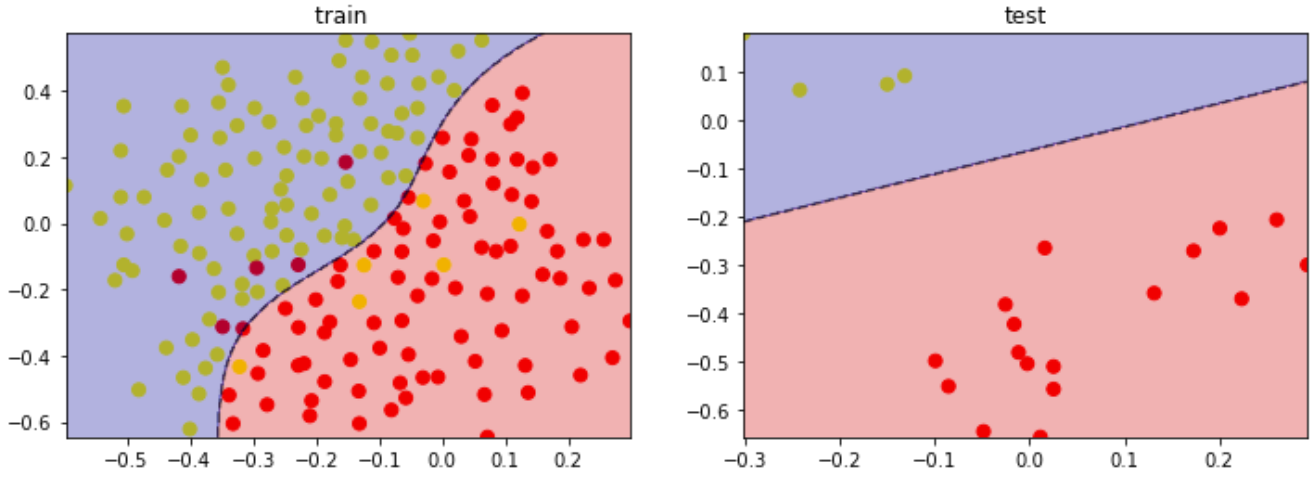


Figure 128: Plots for 4th fold – dataset 1

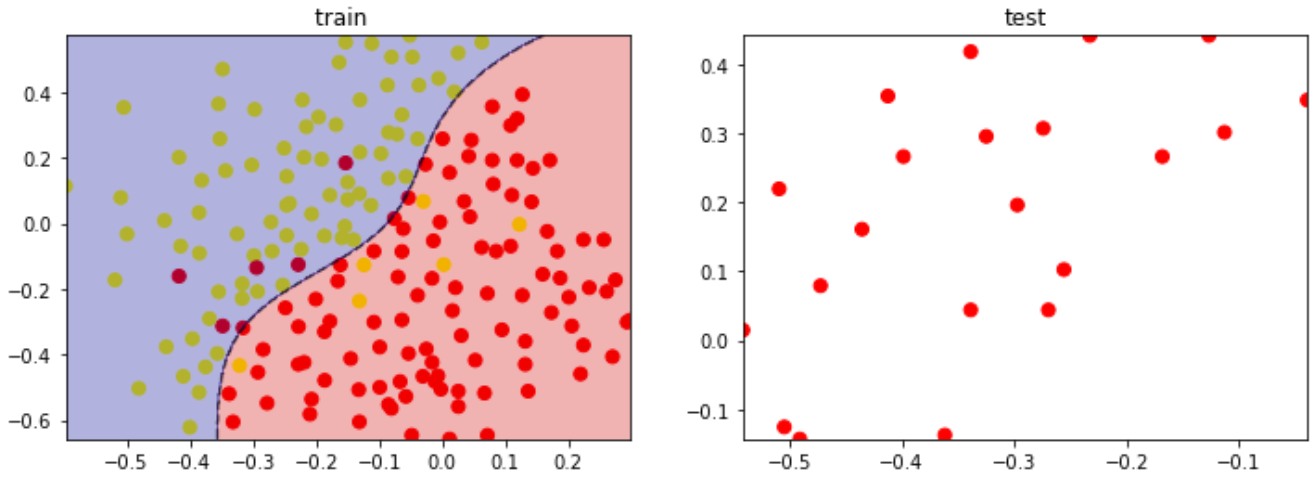


Figure 129: Plots for 5th fold – dataset 1

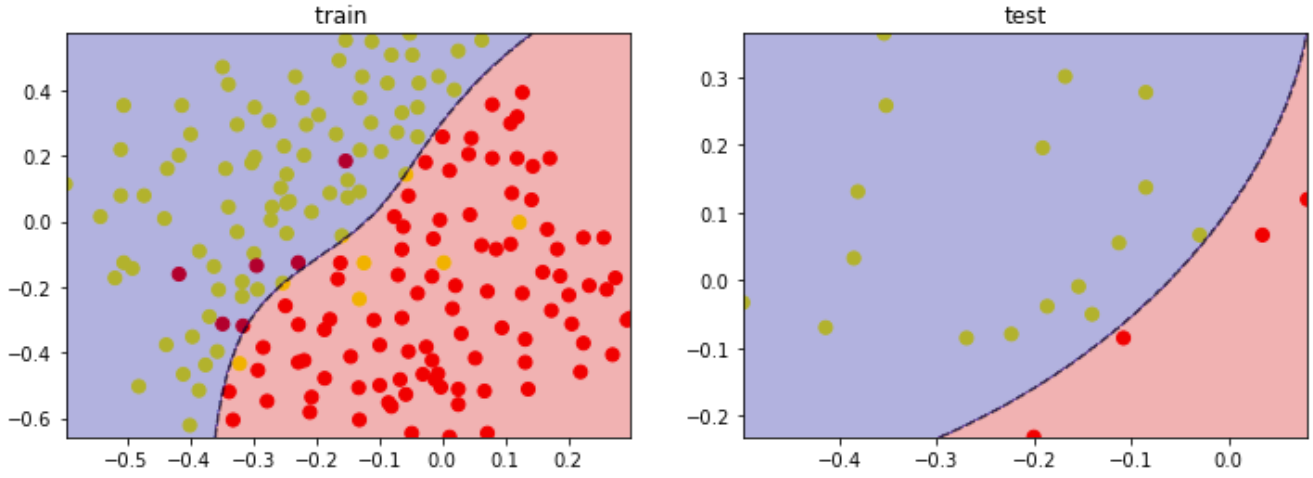


Figure 130: Plots for 6th fold – dataset 1

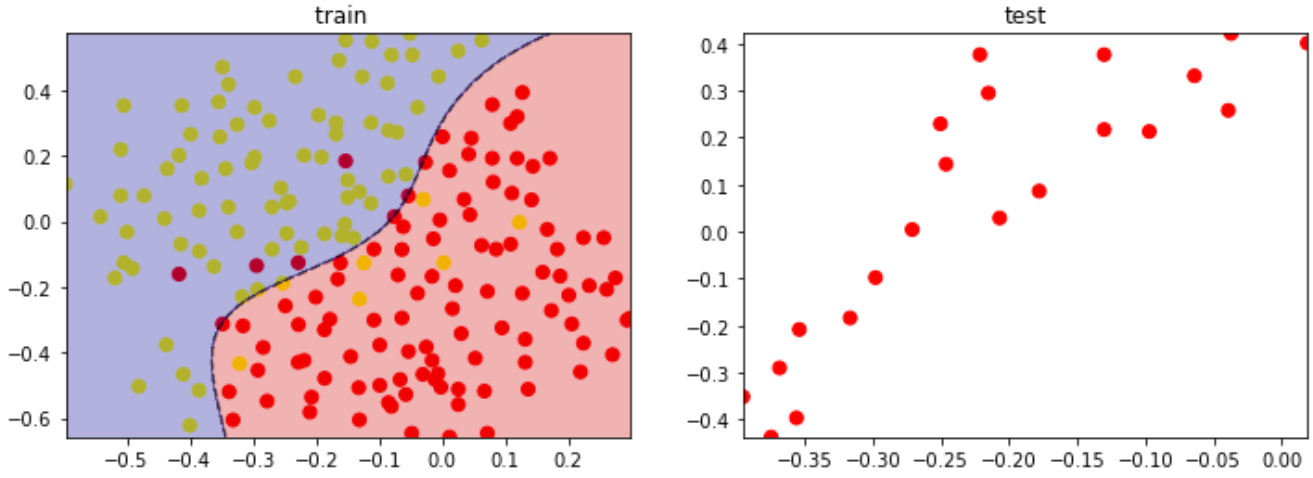


Figure 131: Plots for 7th fold – dataset 1

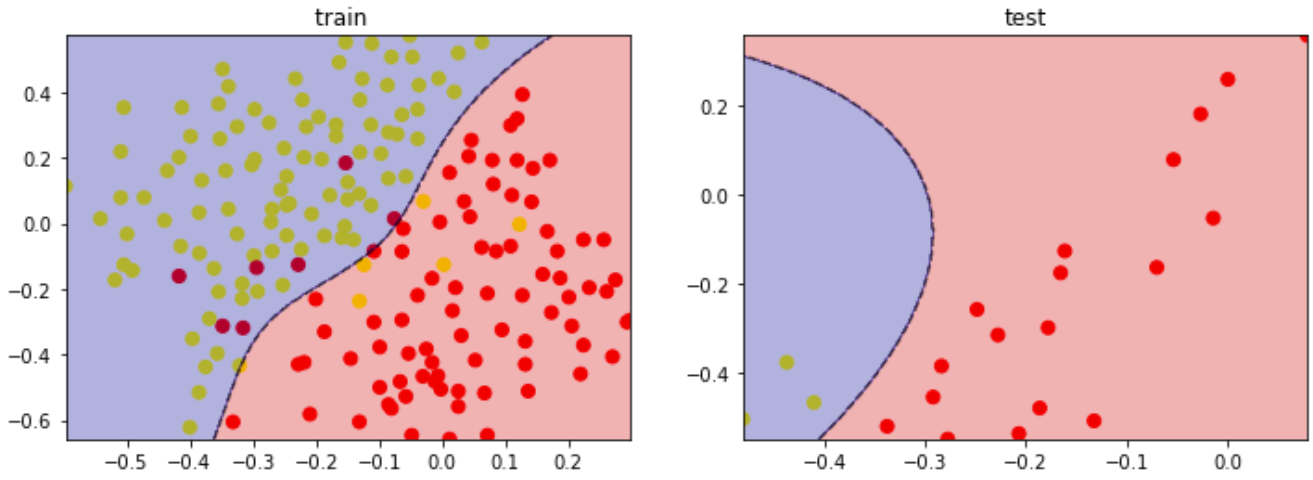


Figure 132: Plots for 8th fold – dataset 1

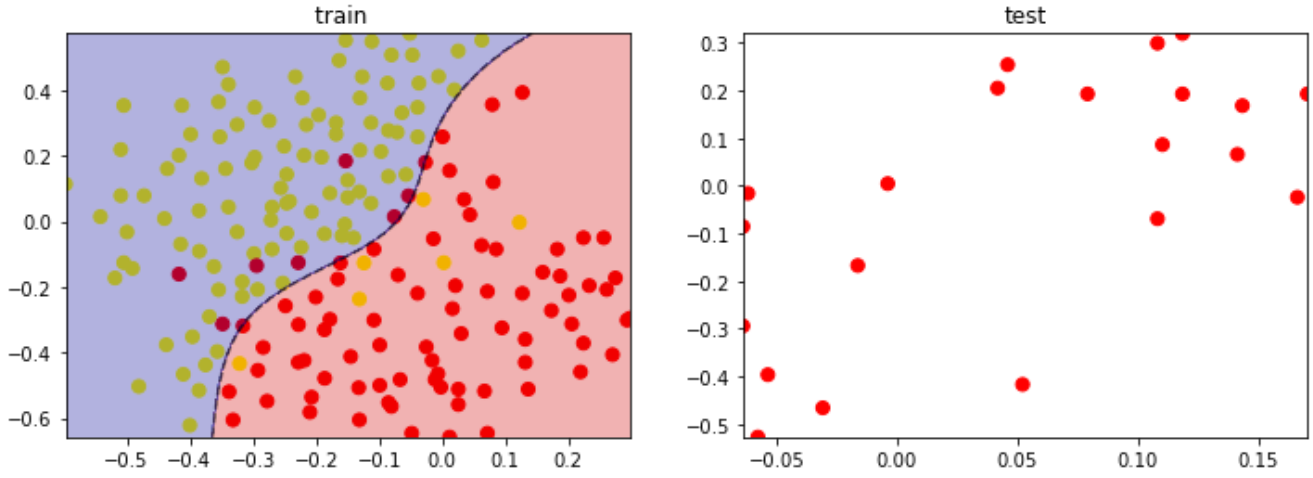


Figure 133: Plots for 9th fold – dataset 1

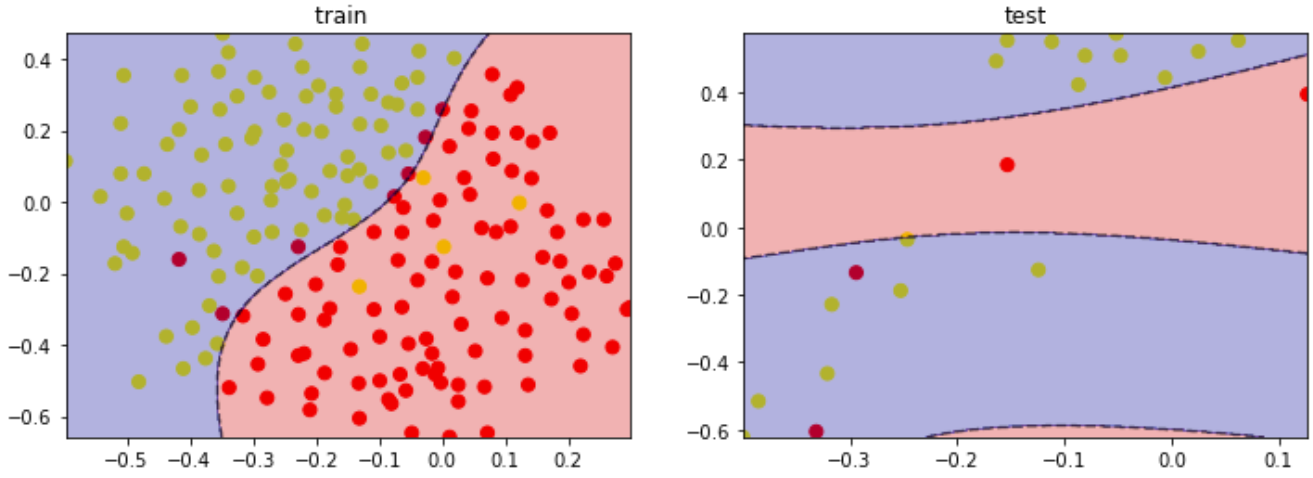


Figure 134: Plots for 10th fold – dataset 1

Dataset two with the best sigma and C using k-fold:

```
********
fold1 ==> 🛠train accuracy = 99.35 ,🔬test accuracy = 100.0
************
********
fold2 ==> 🛠train accuracy = 99.48 ,🔬test accuracy = 100.0
************
********
fold3 ==> 🛠train accuracy = 99.48 ,🔬test accuracy = 100.0
************
********
fold4 ==> 🛠train accuracy = 99.22 ,🔬test accuracy = 100.0
************
********
fold5 ==> 🛠train accuracy = 99.48 ,🔬test accuracy = 97.67
************
********
fold6 ==> 🛠train accuracy = 99.48 ,🔬test accuracy = 97.67
************
********
fold7 ==> 🛠train accuracy = 99.61 ,🔬test accuracy = 97.67
************
********
fold8 ==> 🛠train accuracy = 99.48 ,🔬test accuracy = 98.84
************
********
fold9 ==> 🛠train accuracy = 99.35 ,🔬test accuracy = 100.0
************
********
fold10 ==> 🛠train accuracy = 99.48 ,🔬test accuracy = 97.67
************
```

Figure 135: Dataset2, best sigma, and C using *10-folds*

Plots for each fold on dataset 2 are as follows:

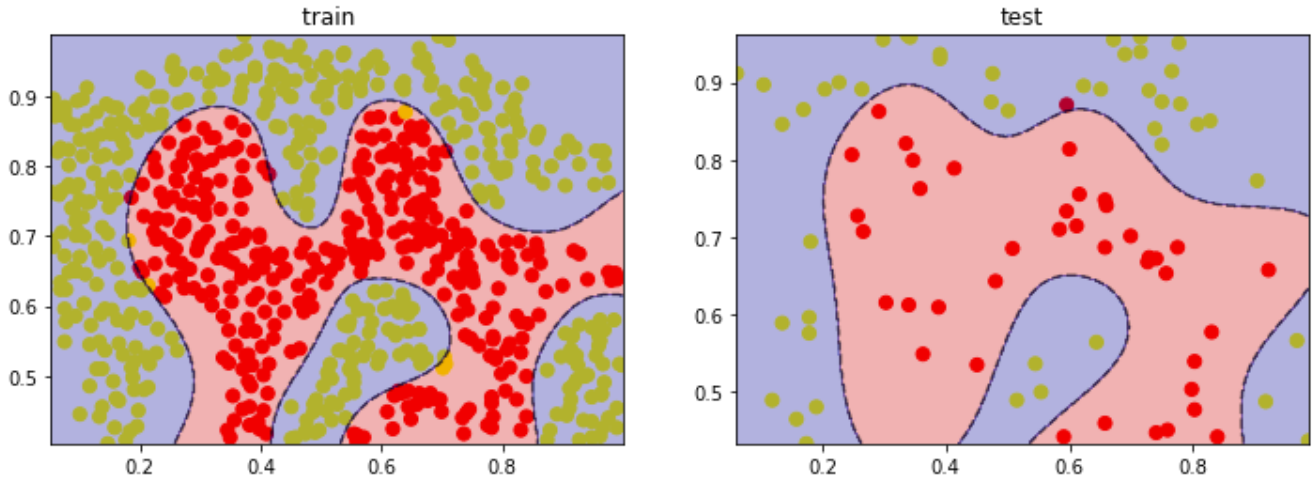


Figure 136: Plots for 1st fold – dataset 2

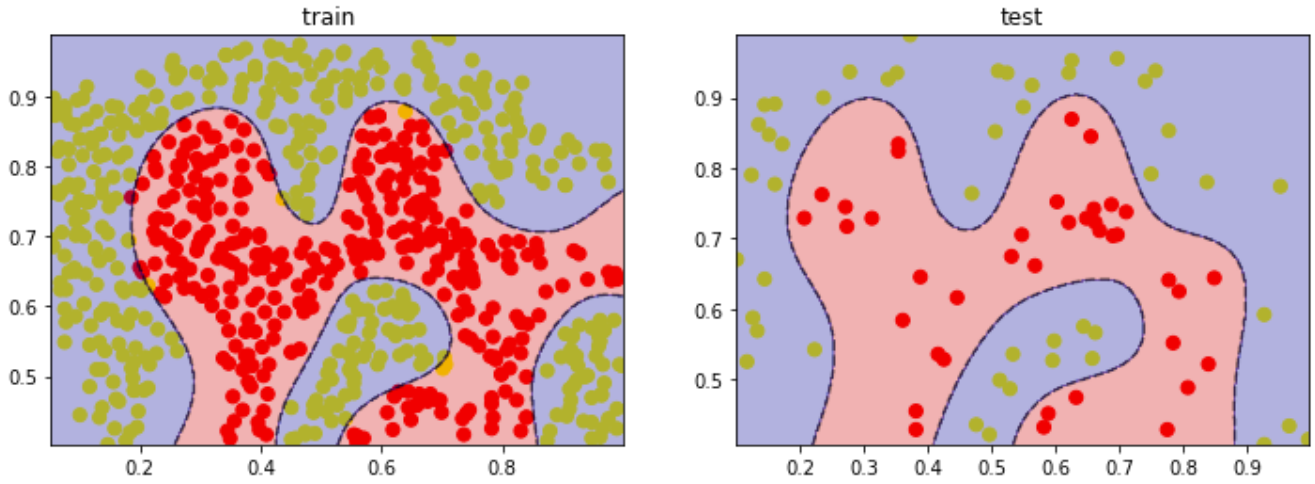


Figure 137: Plots for 2nd fold – dataset 2

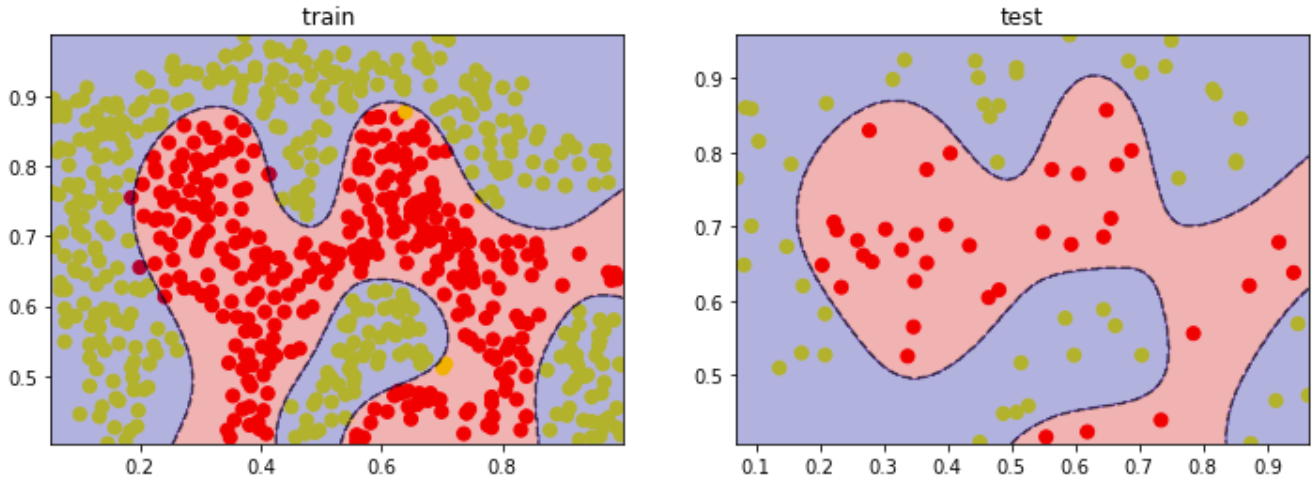


Figure 138: Plots for 3rd fold – dataset 2

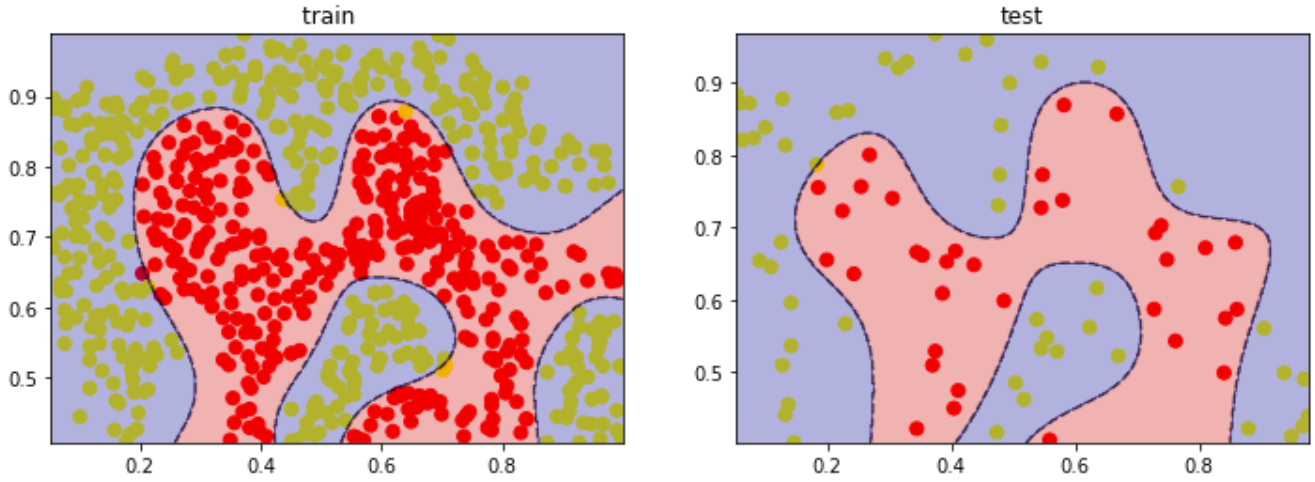


Figure 139: Plots for 4th fold – dataset 2

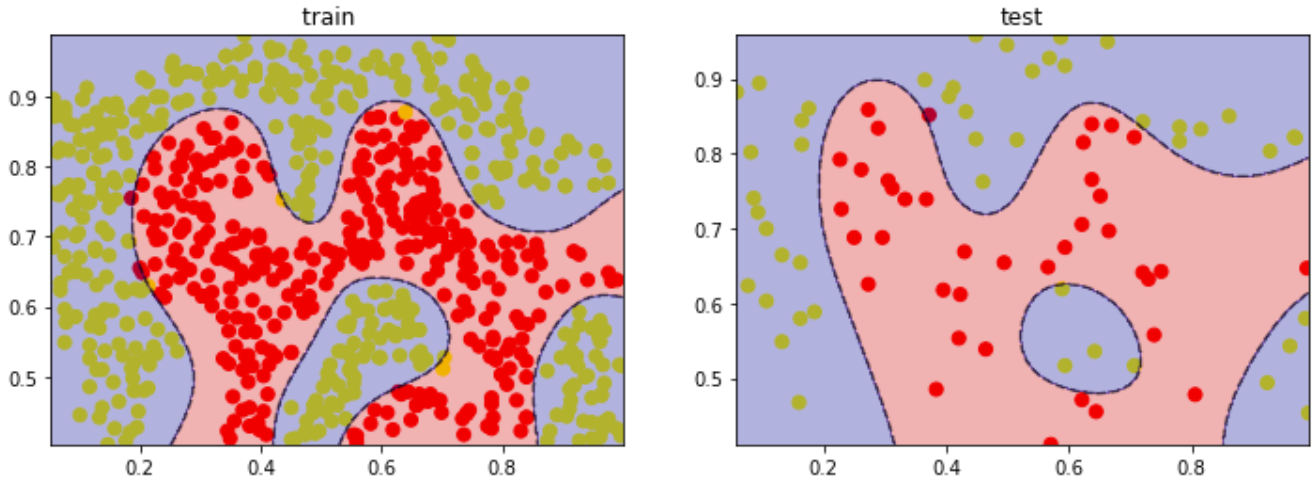


Figure 140: Plots for 5th fold – dataset 2

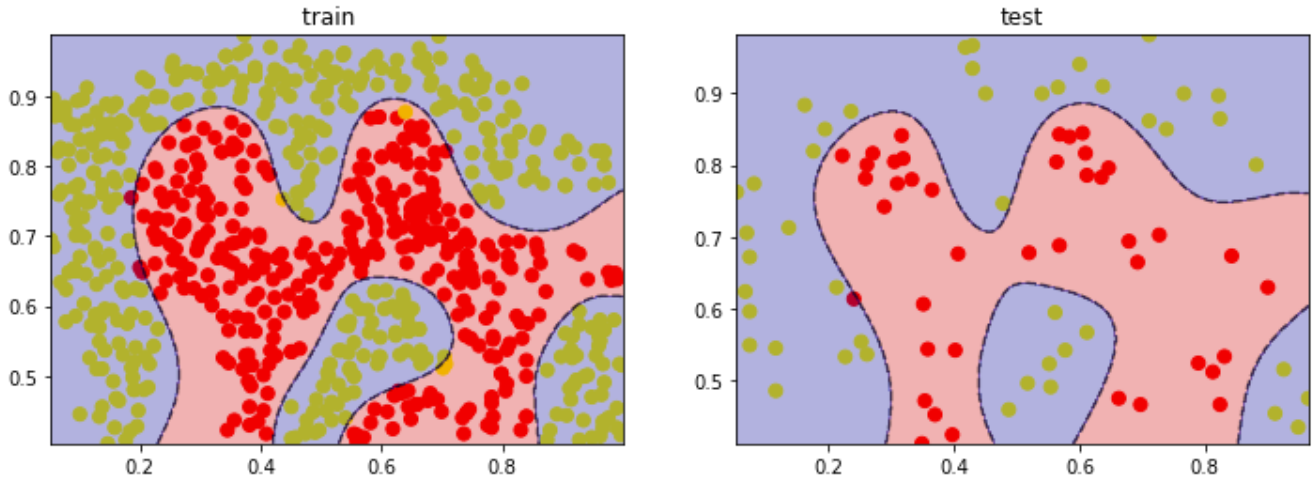


Figure 141: Plots for 6th fold – dataset 2

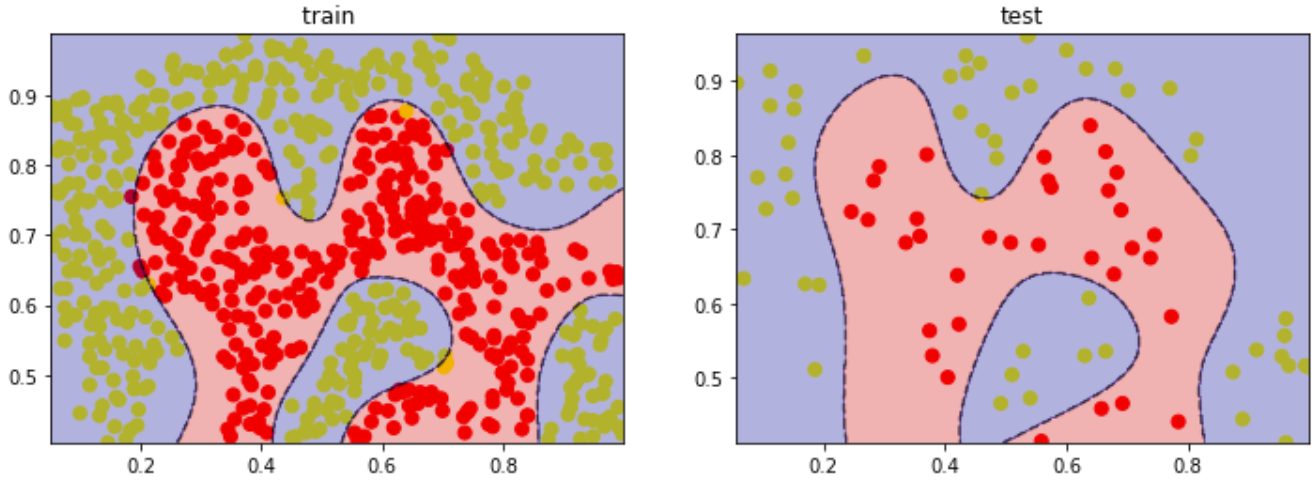


Figure 142: Plots for 7th fold – dataset 2

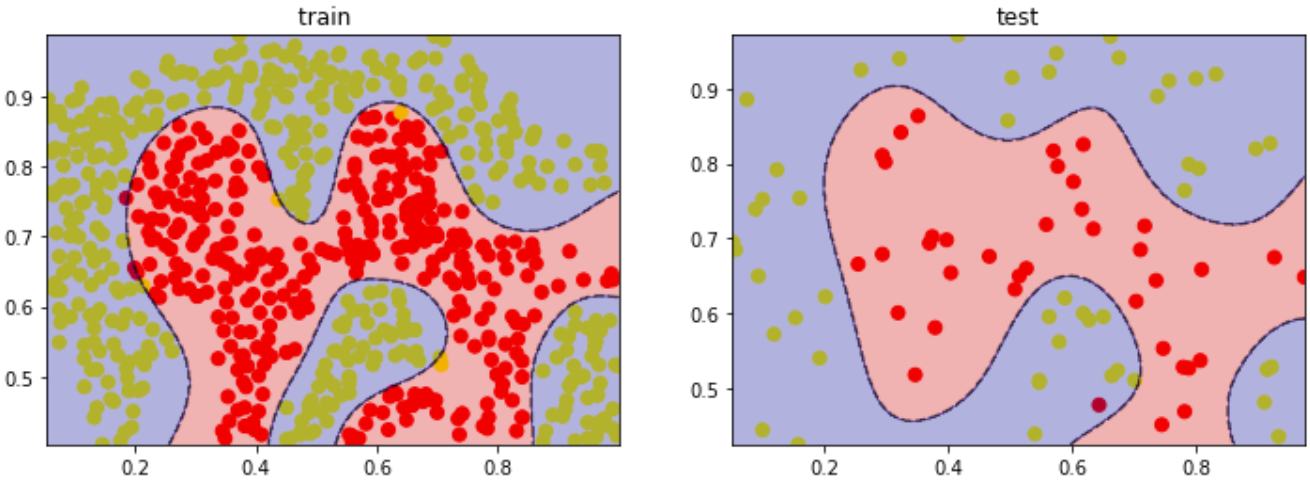


Figure 143: Plots for 8th fold – dataset 2

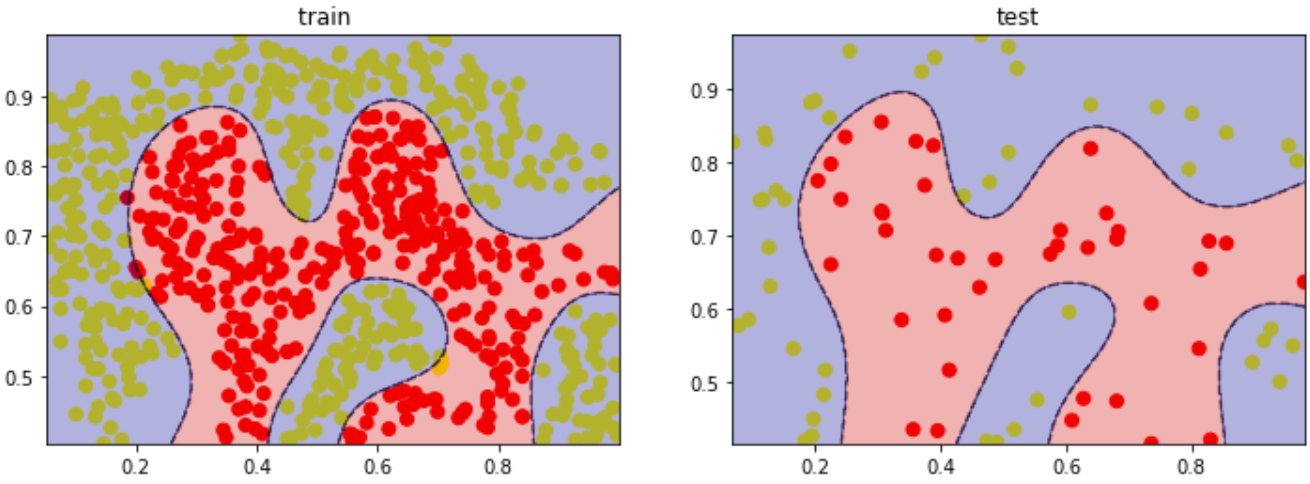


Figure 144: Plots for 9th fold – dataset 2

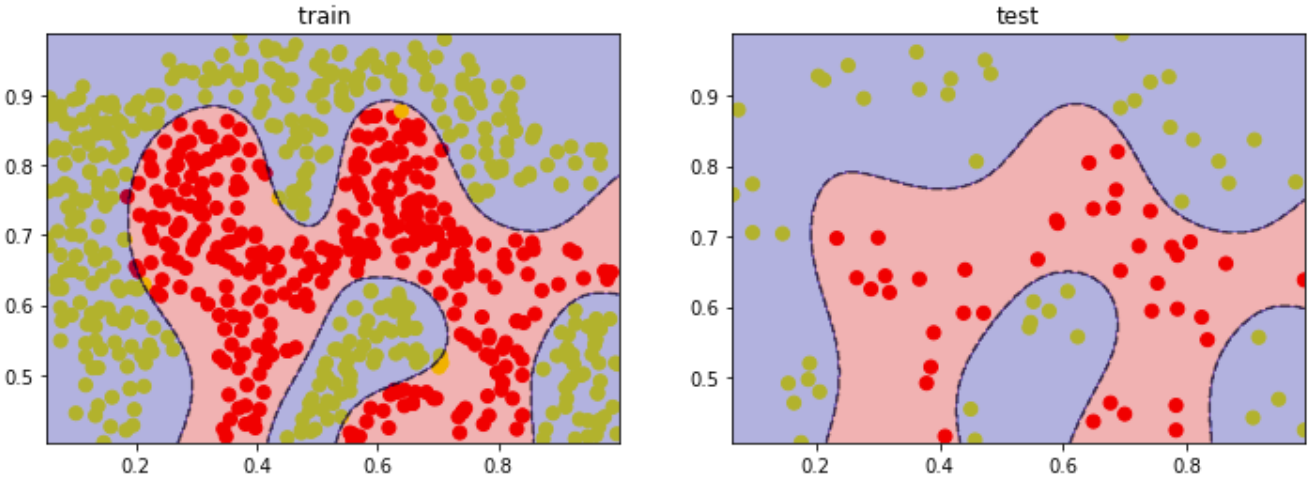


Figure 145: Plots for 10th fold – dataset 2

4.7 Best results plots

Best and worst model (C, sigma) for Dataset1:

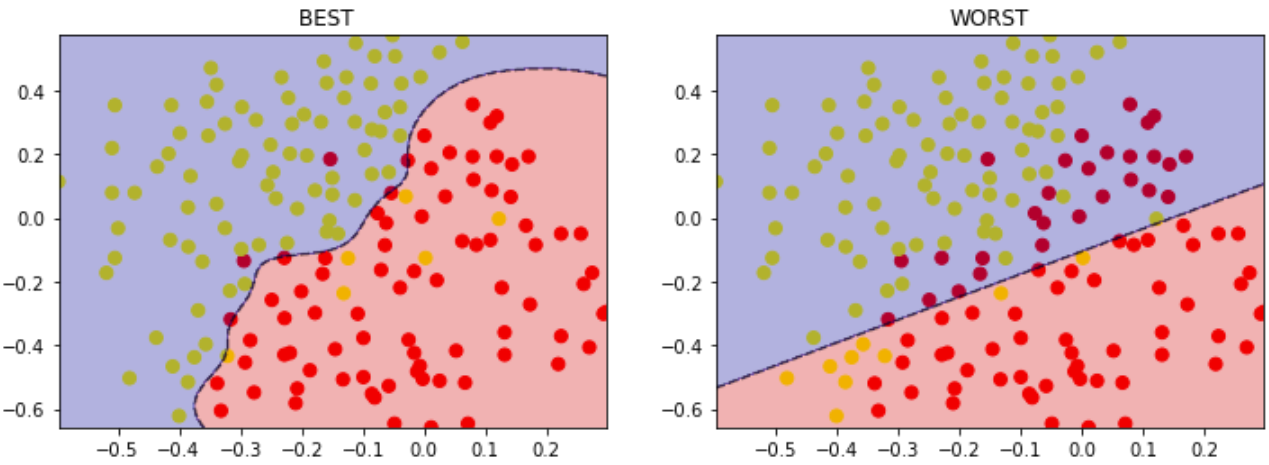


Figure 146: Best = {C = 30, sigma = 3}, Worst = {C = 0.1, Sigma = 0.1}

Best and worst model (C, sigma) for Dataset2:

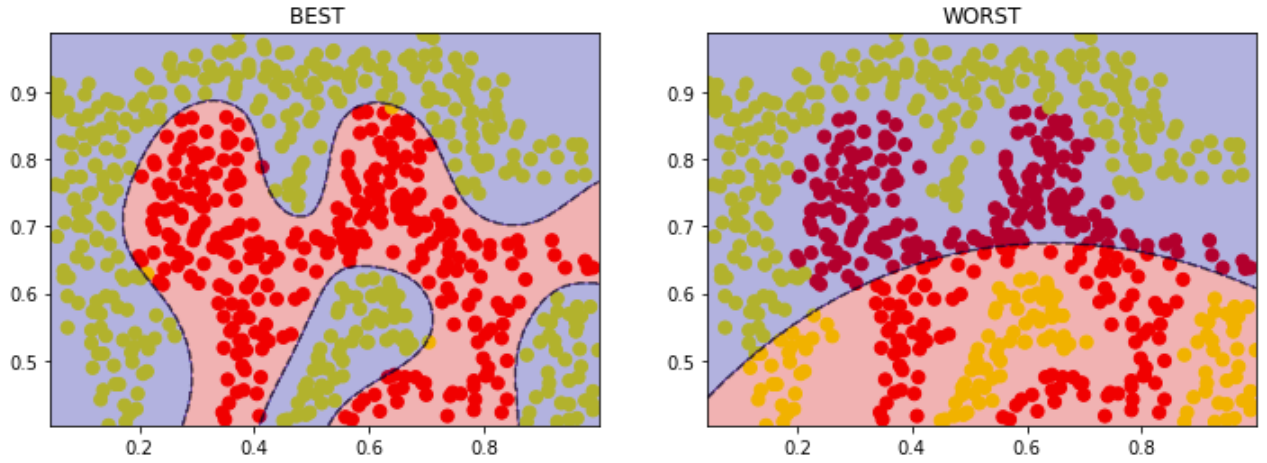


Figure 147: Best = {C = 30, sigma = 30}, Worst = {C = 0.1, Sigma = 0.1}