TALLER 2 CLUSTERING

ANDRES FELIPE ORREGO QUINTERO

C.C. 1041204802

1. A. In which cases might it be more useful to apply?

R/

Non-Convex Clusters: Spectral clustering's ability to handle non-convex clusters makes it a valuable choice. In many real-world datasets, clusters can have complex shapes and may not be well represented by convex geometries. Being able to identify such clusters is crucial for accurate clustering.

Manifold Structures: Spectral clustering's effectiveness in discovering data that lies on low-dimensional manifolds is highly relevant in various domains. Many real-world datasets exhibit complex structures that are more effectively captured by spectral clustering than traditional methods like K-Means.

Image Segmentation: Spectral clustering is commonly used in image segmentation tasks, where it can help partition an image into meaningful regions based on pixel similarities. Image segmentation is essential in computer vision applications, medical image analysis, and more.

B. What are the mathematical fundamentals of it?

R/

Spectral clustering is based on linear algebra and graph theory principles. Its mathematical fundamentals can be summarized as follows:

Affinity Matrix (Similarity Matrix): Spectral clustering starts with the construction of an affinity matrix (also known as a similarity matrix) that encodes pairwise similarities or distances between data points. The affinity matrix is typically denoted as W and is of size NxN, where N is the number of data points. Common affinity measures include Gaussian (RBF) kernel similarity, k-nearest neighbors, or other similarity measures depending on the problem.

Graph Representation: Spectral clustering can be viewed as a graph-based approach. The affinity matrix defines the edge weights of a graph, where each data point corresponds to a node (vertex), and the similarity between data points determines the strength of the edges between nodes.

Graph Laplacian Matrix: The next step is to compute the graph Laplacian matrix, often denoted as L. There are different variations of the Laplacian matrix, but the most common ones used in spectral clustering are the unnormalized Laplacian, the normalized Laplacian, and the random walk Laplacian. The Laplacian matrix captures the graph's structure and plays a crucial role in spectral clustering.

Eigenvalue Decomposition: The Laplacian matrix L is decomposed into its eigenvectors and eigenvalues. Performing eigenvalue decomposition on L yields a set of eigenvectors and corresponding eigenvalues. These eigenvectors represent different modes or patterns in the data.

Dimension Reduction: Spectral clustering often involves selecting a subset of the top eigenvectors corresponding to the smallest eigenvalues. This reduces the dimensionality of the data while preserving its essential structure.

Clustering: After dimension reduction, the reduced data points (represented by the selected eigenvectors) are clustered using a traditional clustering algorithm. K-Means is a common choice, but other algorithms can also be used. The number of clusters is typically determined by examining the eigenvectors or using techniques like the elbow method or the eigengap heuristic.

Assigning Data Points: Once the clusters are obtained, each data point is assigned to one of the clusters based on its representation in the reduced eigenvector space.

C. What is the algorithm to compute it?

R/

The algorithm to compute spectral clustering involves several steps, including the construction of an affinity matrix, the computation of the graph Laplacian, eigenvalue decomposition, dimensionality reduction, and clustering. Here is a step-by-step algorithm for spectral clustering:

Affinity Matrix Construction: Given a dataset with N data points, construct an N×N affinity matrix (similarity matrix) W, where each entry W[i, j] represents the similarity or distance between data points i and j. Common affinity measures include the Gaussian (RBF) kernel similarity, k-nearest neighbors, or other similarity metrics based on the problem domain.

Graph Representation: Treat the affinity matrix W as the adjacency matrix of an undirected weighted graph. Each data point corresponds to a node (vertex) in the graph, and the edge weights represent the strength of connections between nodes.

Graph Laplacian Matrix: Compute the graph Laplacian matrix L from the affinity matrix W. There are different variants of the Laplacian matrix:

Unnormalized Laplacian: L = D - W, where D is a diagonal matrix with D[i, i] representing the sum of row i in W (the degree of node i).

Normalized Laplacian: L = I - D^(-1/2) \* W \* D^(-1/2), where I is the identity matrix, and D^(-1/2) is the inverse square root of the diagonal degree matrix.

Eigenvalue Decomposition: Perform eigenvalue decomposition on the Laplacian matrix L to obtain a set of eigenvectors (u\_1, u\_2, ..., u\_N) and corresponding eigenvalues (λ\_1, λ\_2, ..., λ\_N). These eigenvectors represent different modes or patterns in the data.

Dimension Reduction: Select a subset of the top k eigenvectors (e.g., u\_1, u\_2, ..., u\_k) corresponding to the smallest eigenvalues. This reduces the dimensionality of the data from N to k.

Clustering: Apply a clustering algorithm (e.g., K-Means) to the reduced data points in the eigenvector space to obtain the final clusters. The number of clusters, k, can be specified beforehand or determined using techniques like the elbow method, eigengap heuristic, or other clustering validation methods.

Assigning Data Points: Assign each data point to one of the clusters based on its representation in the reduced eigenvector space.

D. Does it hold any relation to some of the concepts previously mentioned in class? Which, and how?

R/ yes, in class we have seen some fundamentals of lineal algebra, and also the use of eigenvalues, also k-means algorithm is used in this method.

1. A. Research about the DBSCAN method, and answer the following questions:

In which cases might it be more useful to apply?

R/

Arbitrary-Shaped Clusters: DBSCAN is highly valuable when dealing with data that contains clusters of irregular or arbitrary shapes. Many real-world datasets exhibit non-convex cluster structures, and DBSCAN's ability to discover such clusters without any prior assumptions about their shapes makes it a top choice.

Unknown Number of Clusters: When you don't know in advance how many clusters exist in your data, DBSCAN can automatically determine the number of clusters based on the data's density structure. This adaptability is especially useful when you're exploring unfamiliar datasets or when the number of clusters varies across different parts of the data.

Robust to Noise: DBSCAN's robustness to noise and its ability to label data points as outliers or noise make it indispensable when dealing with datasets that contain irregularities, anomalies, or noisy data points. It can help you identify and separate meaningful clusters from noisy or irrelevant data.

B. what are the mathematical fundamentals of DBSCAN

R/

Core Point: A data point p is considered a core point if there are at least min\_samples data points (including p itself) within a distance of ε (epsilon) from p. In mathematical terms:

Core Point Definition: For a data point p, it is a core point if and only if |{q ∈ D | dist(p, q) ≤ ε}| ≥ min\_samples, where D is the dataset and dist(p, q) is the distance metric used.

Directly Density-Reachable: Two data points p and q are said to be directly density-reachable if q is within a distance of ε from p, and p is a core point. In mathematical terms:

Directly Density-Reachable Definition: Data point q is directly density-reachable from data point p if dist(p, q) ≤ ε and p is a core point.

Density-Reachable: A data point q is said to be density-reachable from another data point p with respect to ε and min\_samples if there exists a sequence of data points p1, p2, ..., pn such that p1 = p, pn = q, and for each pi, pi+1 is directly density-reachable from pi. In other words, q can be reached from p by connecting a series of core points.

Density-Connected: Two data points p and q are density-connected with respect to ε and min\_samples if there exists a data point o such that both p and q are density-reachable from o with respect to ε and min\_samples. Density-connected points belong to the same cluster.

Noise and Border Points: Data points that are neither core points nor density-reachable from core points are considered noise points or outliers. Core points form the core of clusters, while border points are on the fringes of clusters but are not core points themselves.

Clustering: DBSCAN clusters data points by grouping together all data points that are density-connected. Each cluster typically contains a set of core points and their associated border points. Noise points are not assigned to any cluster.

Parameter Setting: DBSCAN requires two main parameters: ε (epsilon), which defines the radius of the neighborhood around each data point, and min\_samples, which specifies the minimum number of data points required to form a core point. These parameters significantly influence the clustering results and should be chosen carefully based on the data and problem domain.

C. Is there any relation between DBSCAN and Spectral Clustering? If so, what is it?

R/

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) and Spectral Clustering are two distinct clustering algorithms with different underlying principles and methods. However, there are some indirect connections and relationships between the two in the context of clustering:

Density-Based Clustering vs. Graph-Based Clustering:

DBSCAN: DBSCAN is a density-based clustering algorithm that forms clusters based on the density of data points in the feature space. It identifies clusters as regions of high data point density separated by areas of lower density. DBSCAN operates directly on the original data points and their pairwise distances or similarities.

Spectral Clustering: Spectral clustering, on the other hand, is a graph-based clustering algorithm. It constructs a graph representation of the data, typically using an affinity matrix that encodes pairwise similarities or distances between data points. Spectral clustering then leverages the graph structure and spectral decomposition techniques to perform clustering in a transformed space.

Graph Representation:

DBSCAN: While DBSCAN itself is not inherently graph-based, it can be applied to data that is represented as a graph. For example, you can use DBSCAN to cluster data points in a graph if you have a suitable graph structure.

Spectral Clustering: Spectral clustering is explicitly graph-based. It constructs a graph from the data's affinity matrix and then operates on the eigenvalues and eigenvectors of the Laplacian matrix of this graph. Spectral clustering leverages the spectral properties of the graph to find clusters.

Data Characteristics:

DBSCAN: DBSCAN is particularly suitable for datasets with varying densities and complex geometric structures, as it identifies clusters based on local density properties. It is well-suited for finding clusters with irregular shapes.

Spectral Clustering: Spectral clustering is effective for finding clusters based on the data's underlying structure, such as identifying low-dimensional manifolds within the data. It is valuable when data points may not be naturally separated by density but exhibit intrinsic structures in a transformed space.

Parameter Considerations:

DBSCAN: DBSCAN requires the specification of parameters like ε (epsilon) and min\_samples to control the neighborhood size and core point definition. The choice of these parameters can impact clustering results.

Spectral Clustering: Spectral clustering involves parameter choices related to the affinity matrix, kernel function, number of clusters, and the type of Laplacian matrix used.

In summary, while DBSCAN and Spectral Clustering are different algorithms with distinct methodologies, they can be applied in different contexts based on the characteristics of your data and the problem at hand. In some cases, you may use DBSCAN to pre-cluster your data based on density properties and then apply spectral clustering to further analyze the resulting clusters in a transformed space. The choice between these algorithms ultimately depends on the nature of your data and your clustering objectives.

1. What is the elbow method in clustering? And which flaws does it pose to assess quality?

R/

The elbow method is a heuristic technique used to determine the optimal number of clusters in a dataset for clustering algorithms like K-Means. The method involves plotting the explained variation as a function of the number of clusters and looking for an "elbow point" in the plot, where the explained variation starts to level off. The number of clusters at the elbow point is often considered a reasonable choice for the dataset's optimal cluster count.

Here's how the elbow method works:

Fit a clustering algorithm (e.g., K-Means) to the data for a range of cluster counts (e.g., from 1 to some maximum value).

For each cluster count, calculate a measure of variation or distortion within the clusters (e.g., the sum of squared distances from data points to their cluster centroids).

Plot the cluster count against the corresponding variation measure.

Examine the resulting plot. The "elbow point" is where the variation starts to level off, indicating that adding more clusters does not significantly reduce the variation.

The number of clusters at the elbow point is often chosen as the optimal cluster count.

However, the elbow method has some flaws and limitations:

Subjectivity: Determining the exact elbow point can be subjective and may vary depending on the dataset and the choice of variation measure. Different analysts may interpret the plot differently.

Noisy Data: In datasets with a lot of noise or overlapping clusters, the elbow method may not yield a clear elbow point, making it challenging to determine the optimal number of clusters.

Dependence on Initialization: The result of K-Means clustering depends on the initial placement of cluster centroids. Running K-Means multiple times with different initializations may yield different results, affecting the elbow method's interpretation.

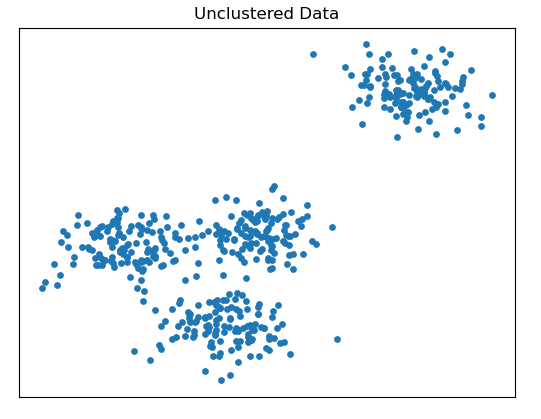
Metric Dependency: The choice of variation measure (e.g., sum of squared distances) can influence the results. Different variation measures may lead to different elbow points.

Other Cluster Structures: The elbow method assumes that the data exhibits a distinct "elbow" in the variation plot. In some cases, the data may not have a well-defined elbow, or there may be multiple elbows.

Not Suitable for All Algorithms: The elbow method is primarily designed for algorithms like K-Means, which optimize for cluster compactness. It may not be appropriate for assessing the optimal number of clusters in other clustering algorithms with different objectives or structures.

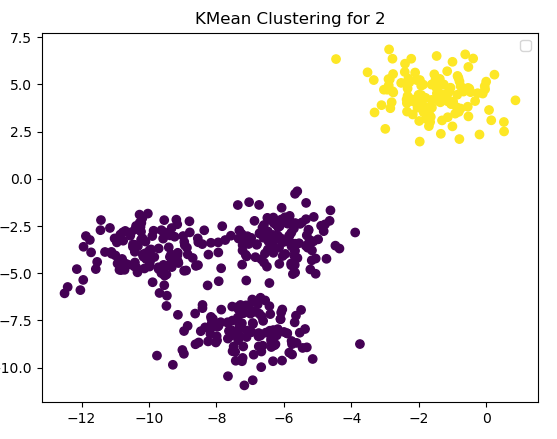
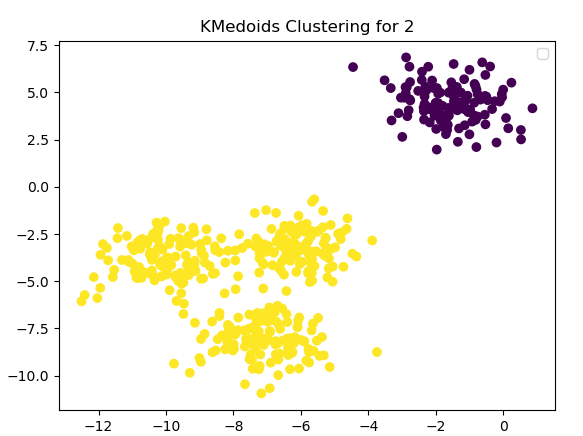
Given these limitations, the elbow method should be used as a preliminary guideline for estimating the number of clusters rather than as a definitive answer. It's often a good starting point for exploring clustering solutions, but additional analysis and validation, such as silhouette analysis or domain knowledge, should be used to make a final decision on the number of clusters.

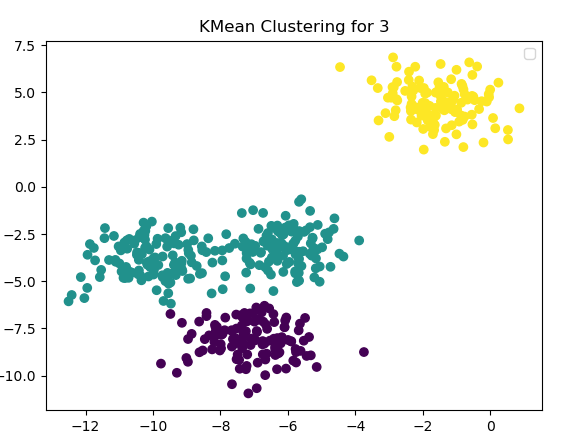
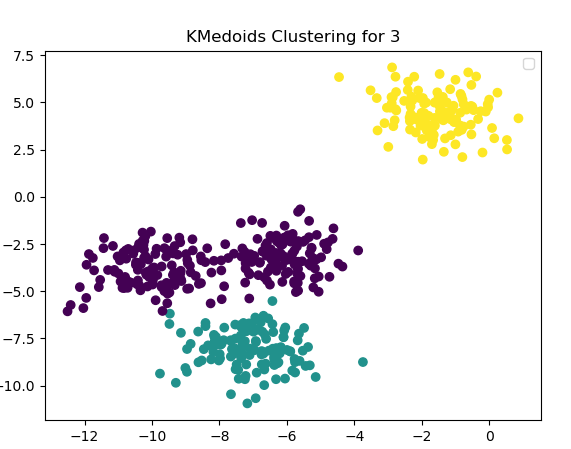
1. This point is in the file named library.
2. B.

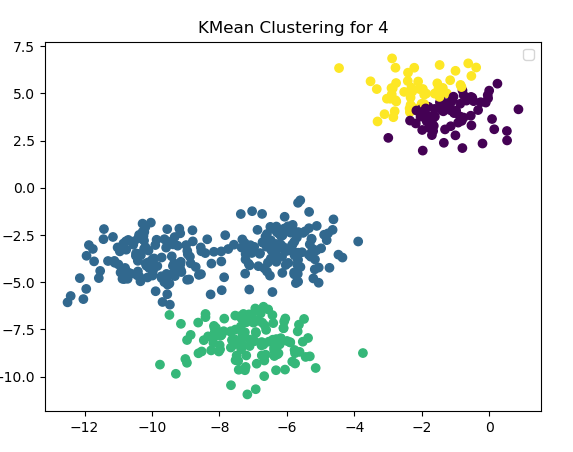
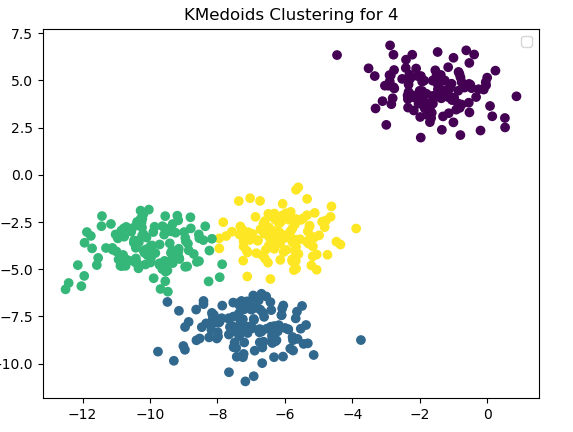


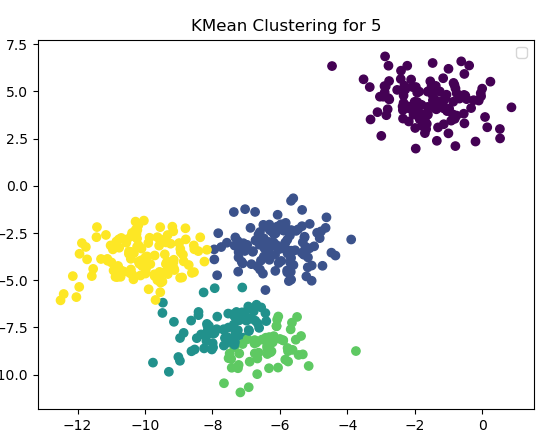
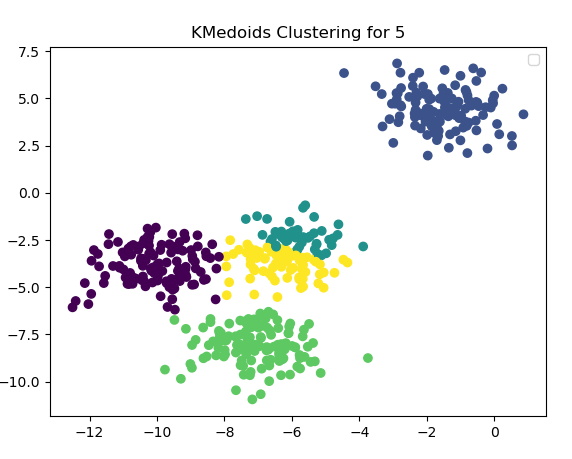
We can see 4 clusters, 1 is too far from other clusters and 3 clusters are quite close.

C. this are the results

With this result of silhouette

the silhouette average for 2 clusters in Kmeans is 0.7049787496083262

the silhouette average for 2 clusters in kmedoids is 0.7049787496083262

the silhouette average for 3 clusters in Kmeans is 0.5640395813218549

the silhouette average for 3 clusters in kmedoids is 0.5873430979447513

the silhouette average for 4 clusters in Kmeans is 0.4589412264184558

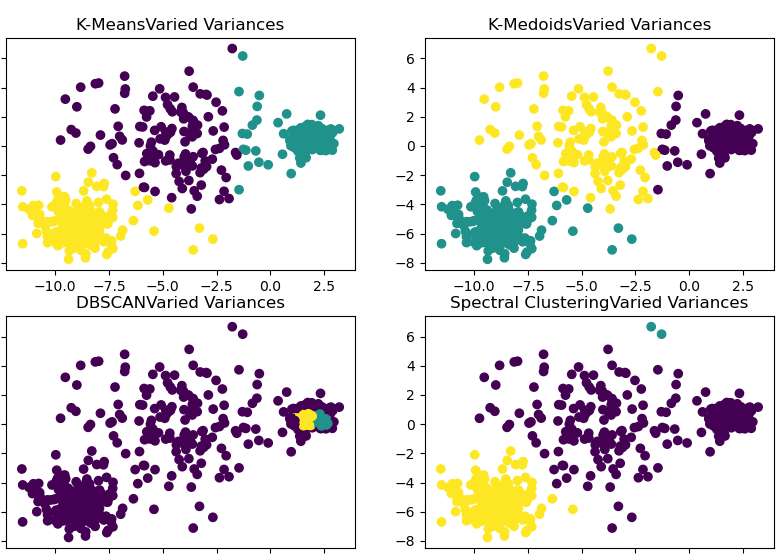
the silhouette average for 4 clusters in kmedoids is 0.6505186632729437

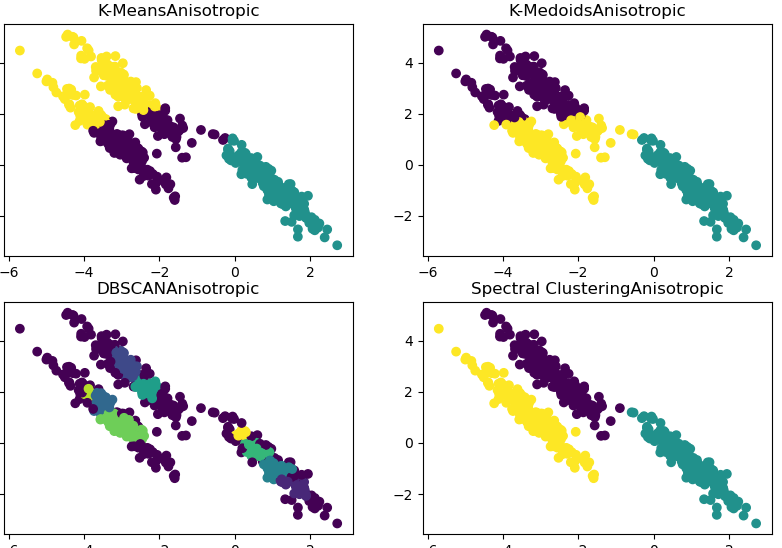
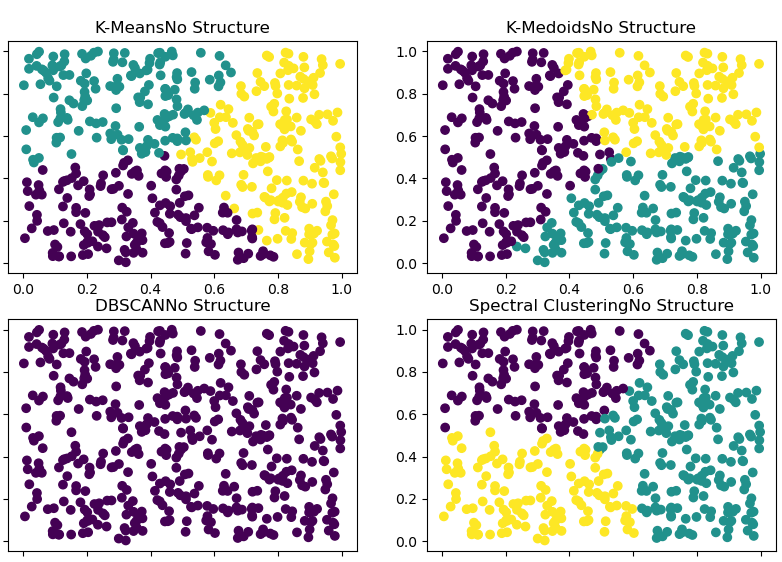
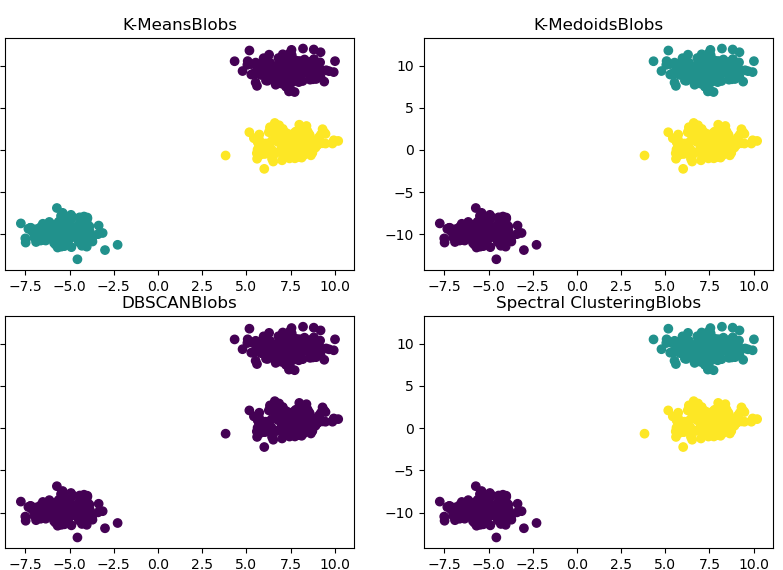
the silhouette average for 5 clusters in Kmeans is 0.5748362978270065

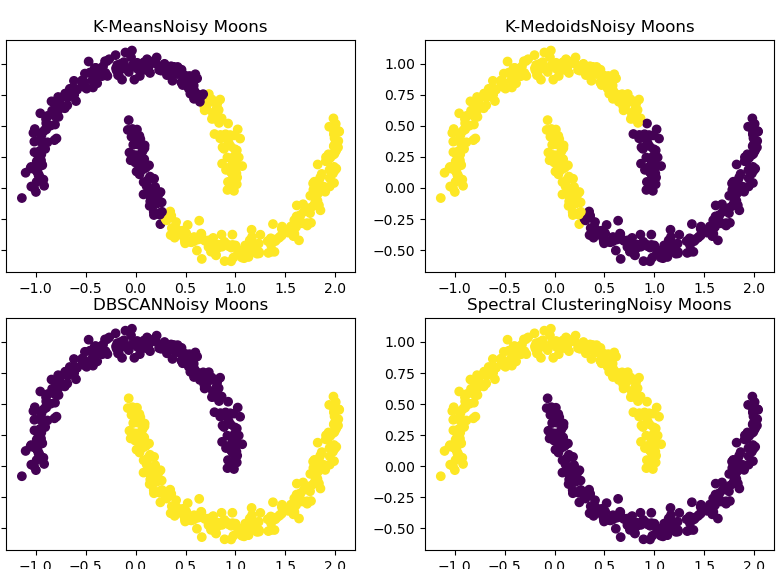
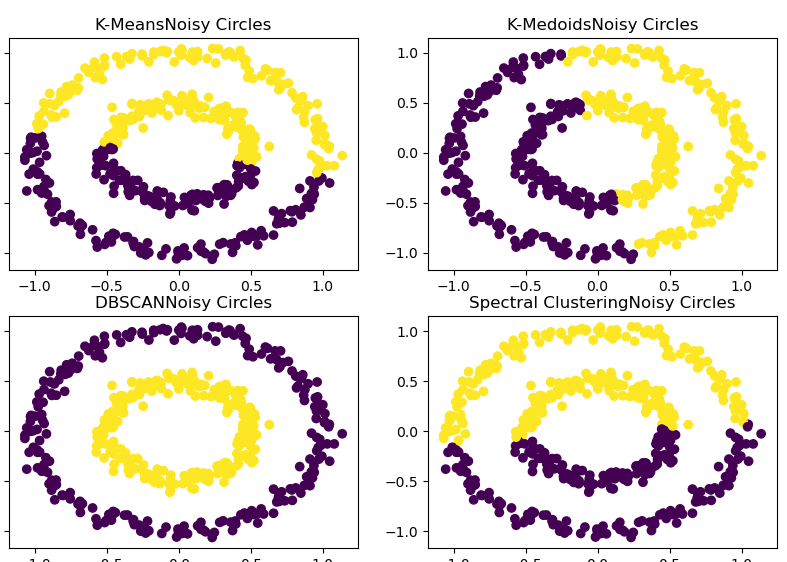
the silhouette average for 5 clusters in kmedoids is 0.5579110311006575

D. the best results were with 2 cluster; it was expected because 1 cluster was too far from other 3 clusters, the result of k-medians with 4 clusters was good.

1. These are the results



I can say that spectral clustering is the best in main part of the cases except in the last one, you can change the parameter gamma to get better results, ig gamma is too big is because every graph is far away from other, thats why I set gamma very big in the last 2 cases.