DA5400: Foundations of Machine Learning

Assignment 1: PCA and K-Means

Student Name: Adithya Sai .L
Student ID: DA25S008
Code: github repository. Please find the updated report in the Github repository

Date: September 22, 2025

1 You are given a dataset with 1000 data points each in R^2.

1.a Write a piece of code to run the PCA algorithm on this data-set. How much of the variance in the data-set is explained by each of the principal components?

Refer pca.py or Question1.ipynb
Variance explained by the first Principal Component $= \frac{\lambda_1}{\lambda_1+\lambda_2} = 0.6539 = 65.39\%$
Variance explained by the second Principal Component $= \frac{\lambda_2}{\lambda_1+\lambda_2} = 0.3461 = 34.61\%$

1.b Write a piece of code to implement the Kernel PCA algorithm on this dataset. Explore various kernels discussed in class. For each Kernel, plot the projection of each point in the dataset onto the top-2 principal components. Use one plot for each kernel - In case of RBF kernel, use a different plot for each value of σ that you use.

Refer pca.py or Question1.ipynb

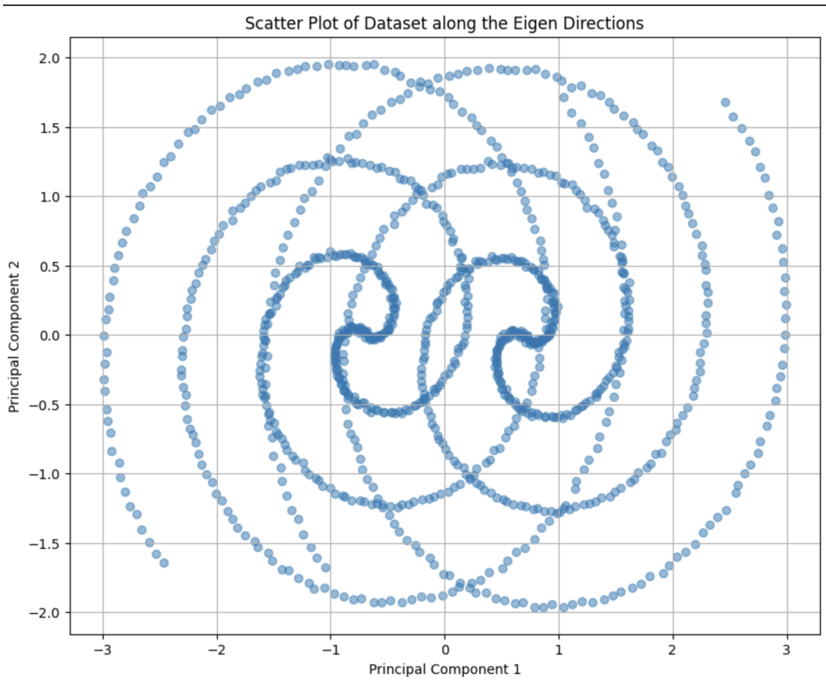

Figure 1: Linear kernel

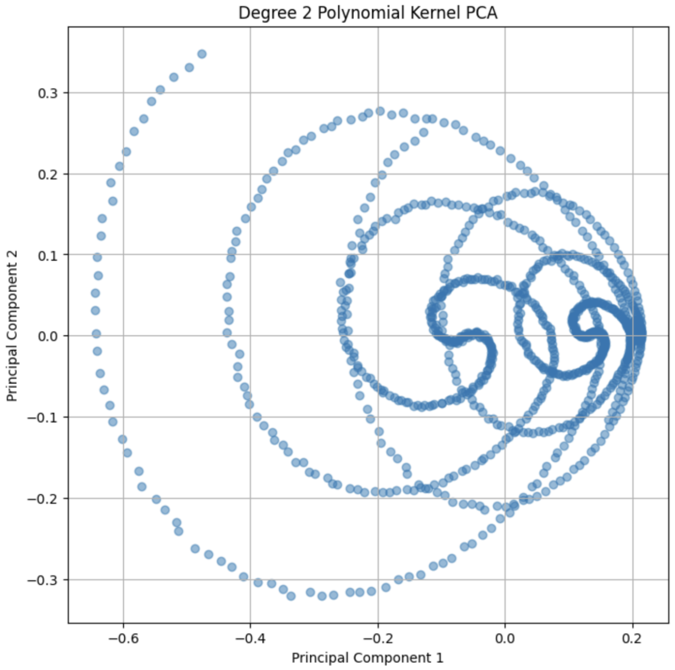

Figure 2: Quadratic kernel

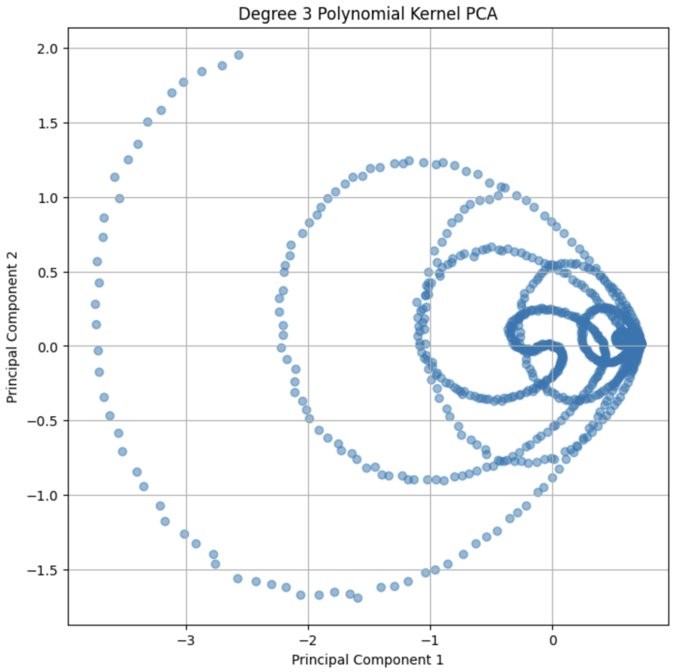

Figure 3: Cubic kernel

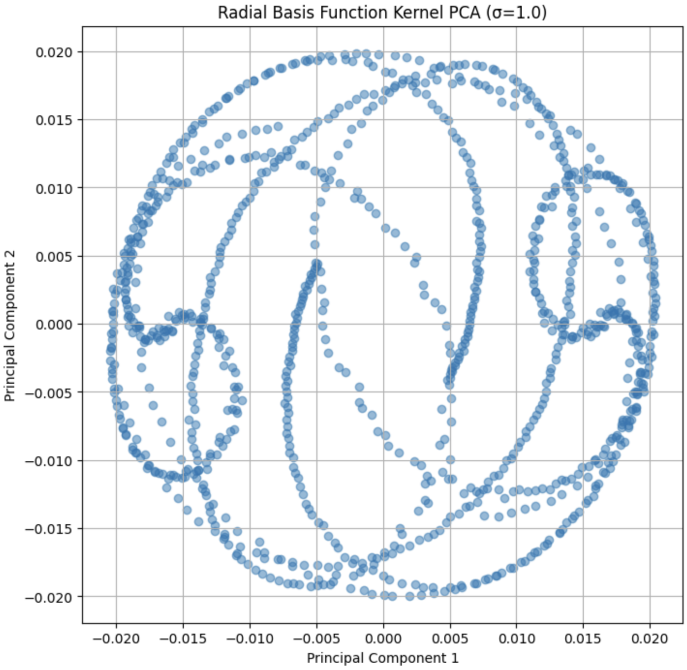

Figure 4: RBF kernel $\sigma = 1$

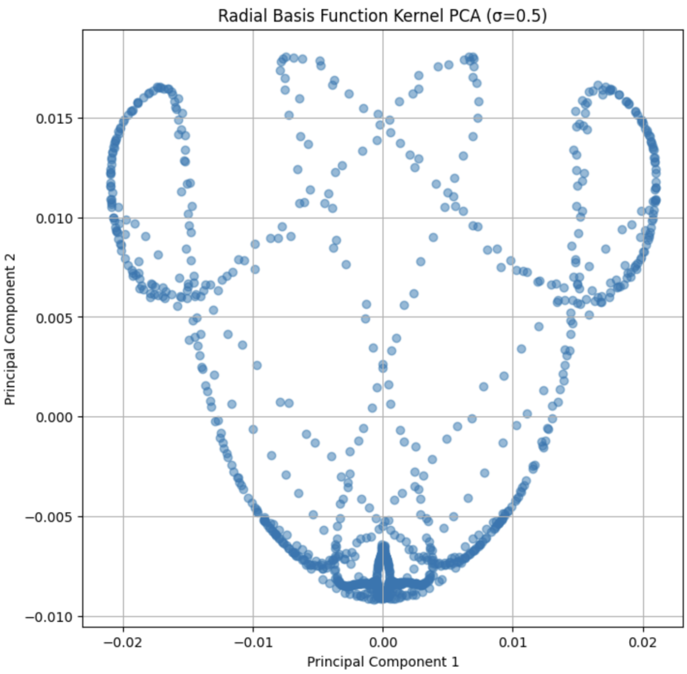

Figure 5: RBF kernel $\sigma = 0.5$

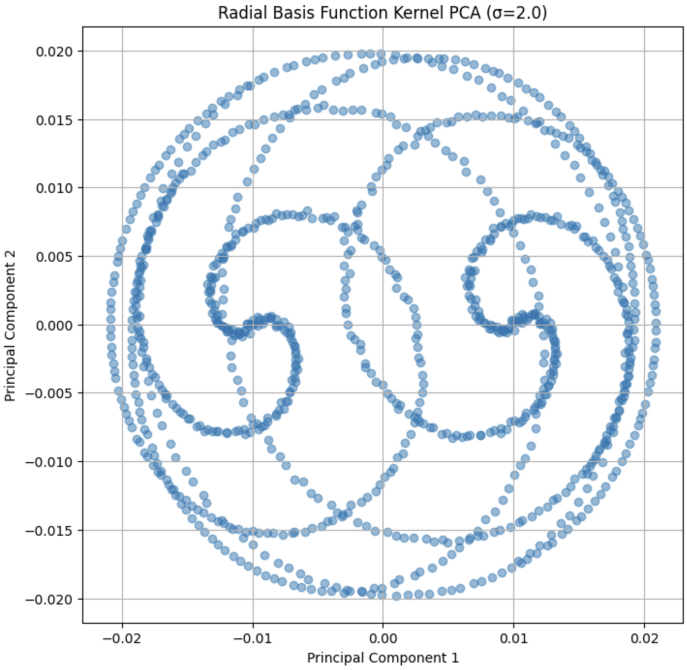

Figure 6: RBF kernel $\sigma = 2$

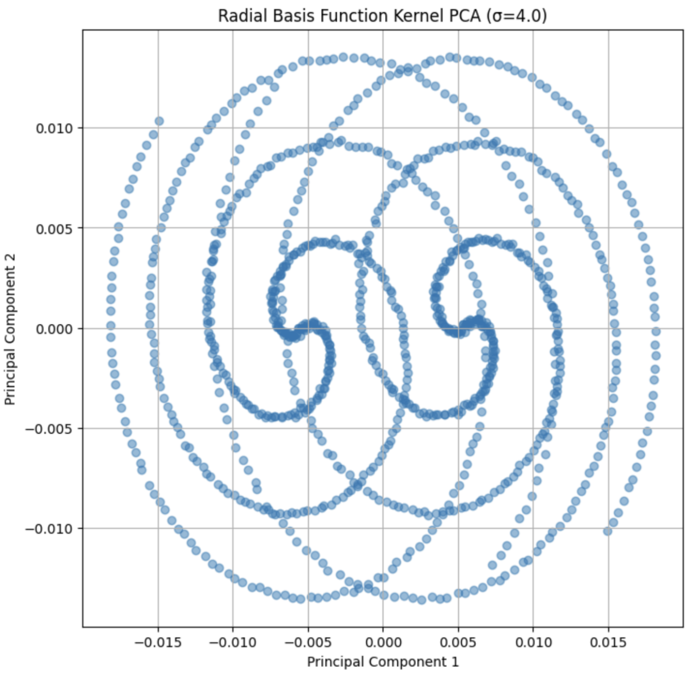

Figure 7: RBF kernel $\sigma = 4$

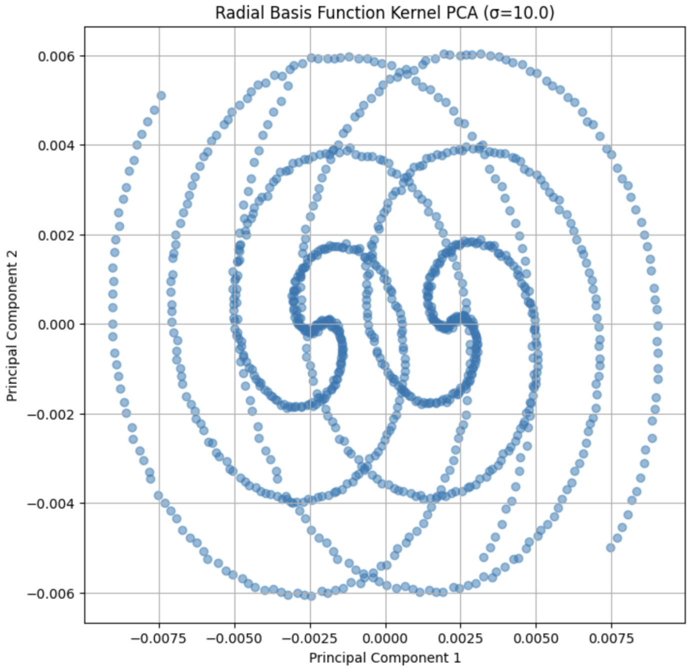

Figure 8: RBF kernel $\sigma = 10$

1.c Which Kernel do you think is best suited for this dataset and why?

The answer depends on what we the downstream application is. If we want linearly boundary between the data when projected onto the first 2 principal components, it seems $\sigma = 0.2$ performs well. However, if we consider the downstream task of classification between the 2 spirals, it seems $\sigma = 0.1$ performs best. Finally if we want to reconstruct the data using the first 2 principal components, a high value such as $\sigma = 100$ almost recovers the entire data.

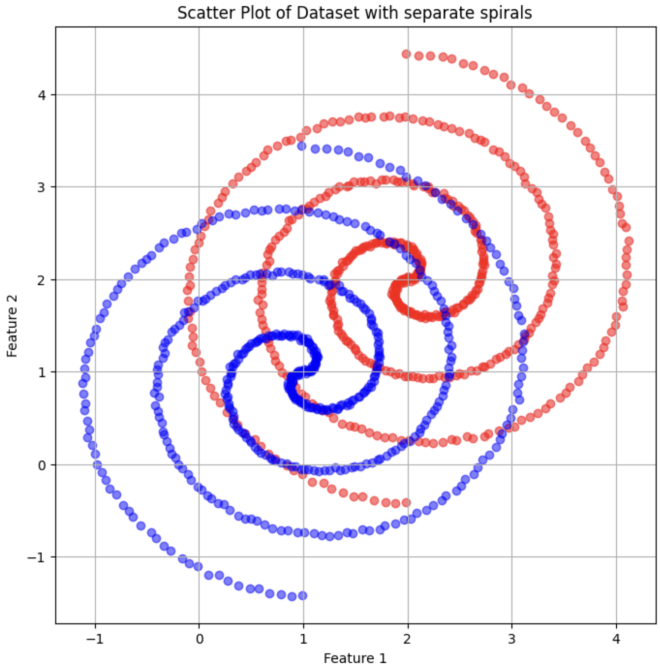

Figure 9: Dataset Separated

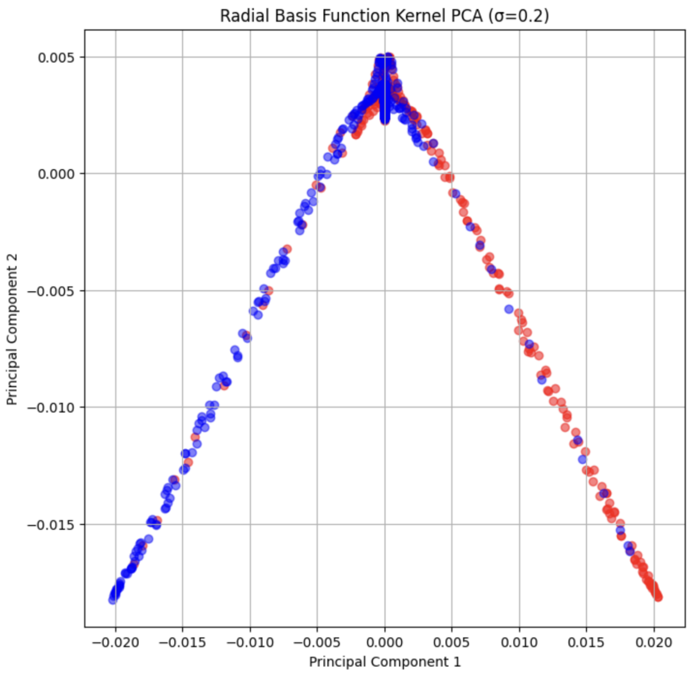

Figure 10: RBF kernel $\sigma = 0.2$ projected

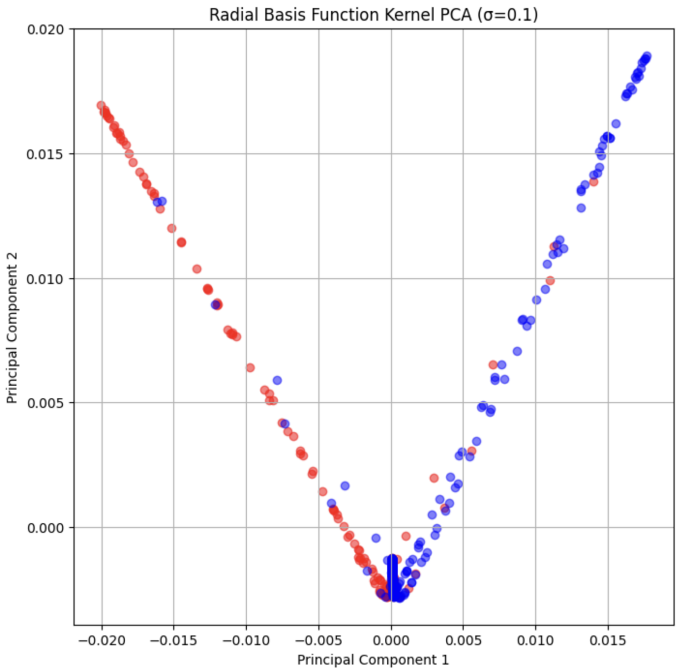

Figure 11: RBF kernel $\sigma = 0.1$ projected

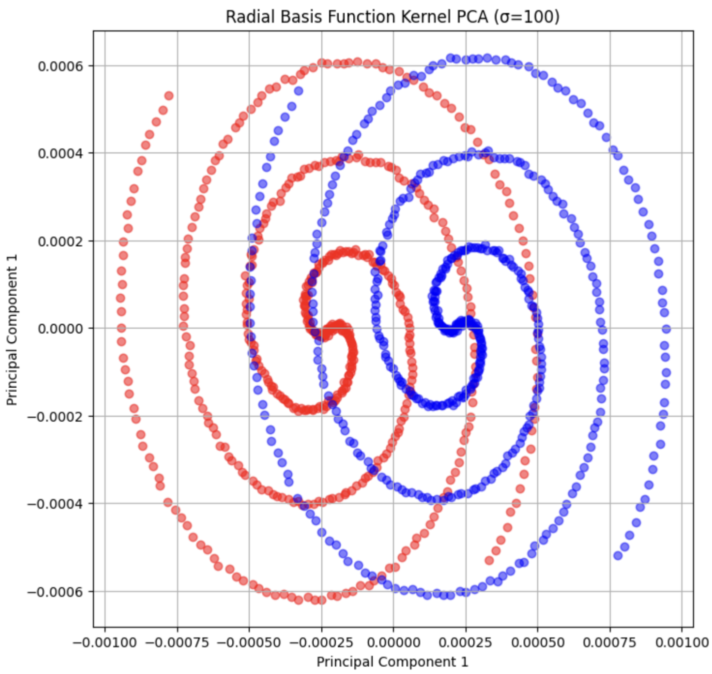

Figure 12: RBF kernel $\sigma = 100$ projected

2 You are given a data-set with 1000 data points each in R^2.

2.a Write a piece of code to run the algorithm studied in class for the K-means problem with k = 4. Try 5 different random initialization and plot the error function w.r.t. iterations in each case. In each case, plot the clusters obtained in different colors.

Refer kmeans.py or Question2.ipynb

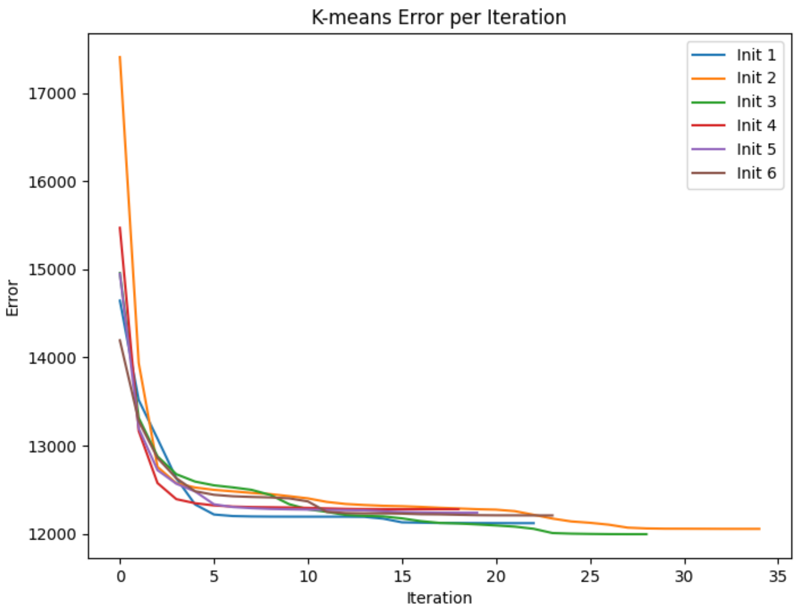

Figure 13: Error convergence

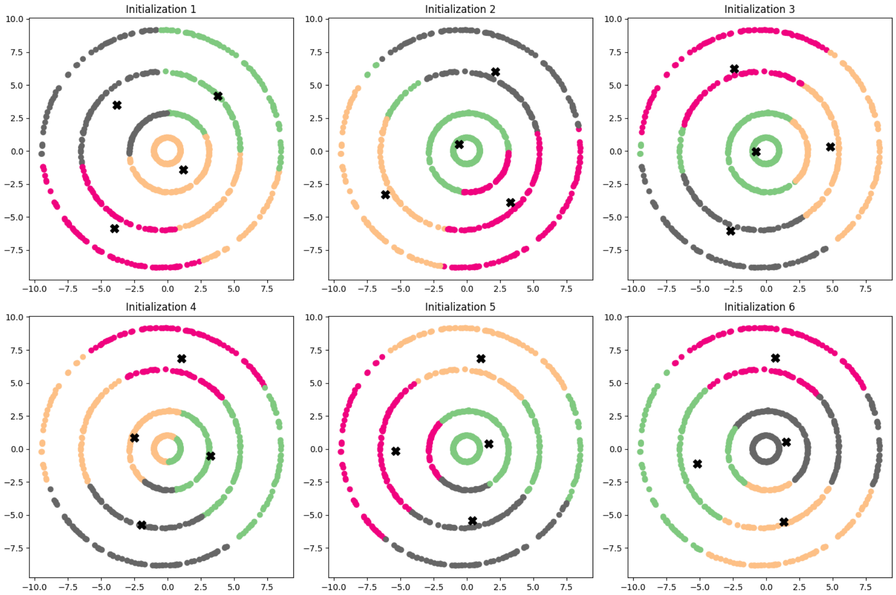

Figure 14: Clusters obtained

2.b Fix a random initialization. For K= {2,3,4,5}, obtain cluster centers according to K-means algorithm using the fixed initialization. For each value of K, plot the Voronoi regions associated to each cluster center. (You can assume the minimum and maximum value in the data-set to be the range for each component of R2).

Refer kmeans.py or Question2.ipynb

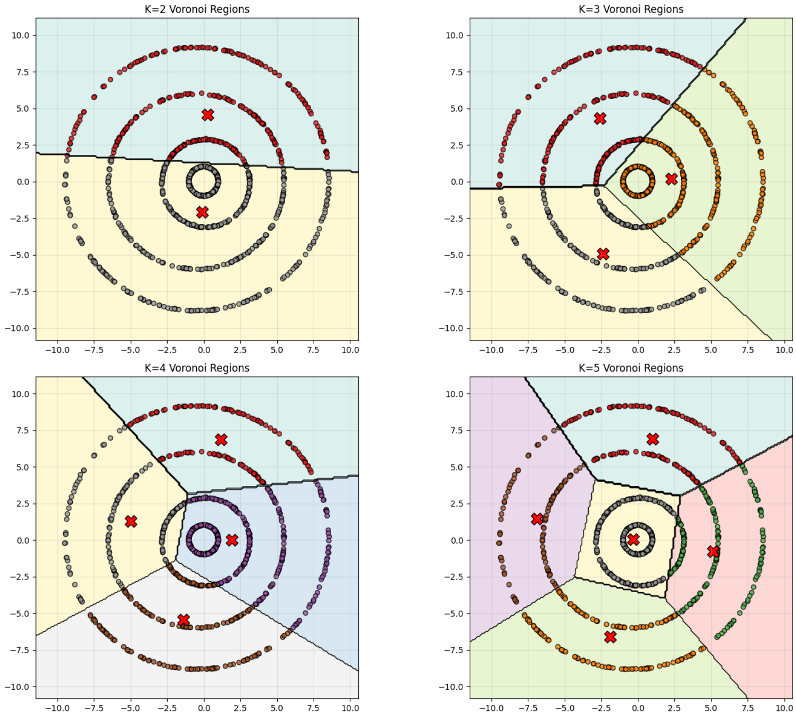

Figure 15: Voronoi regions

2.c Run the spectral clustering algorithm (spectral relaxation of K-means using Kernel- PCA) k = 4. Choose an appropriate kernel for this data-set and plot the clusters obtained in different colors. Explain your choice of kernel based on the output you obtain.

Refer kmeans.py or Question2.ipynb

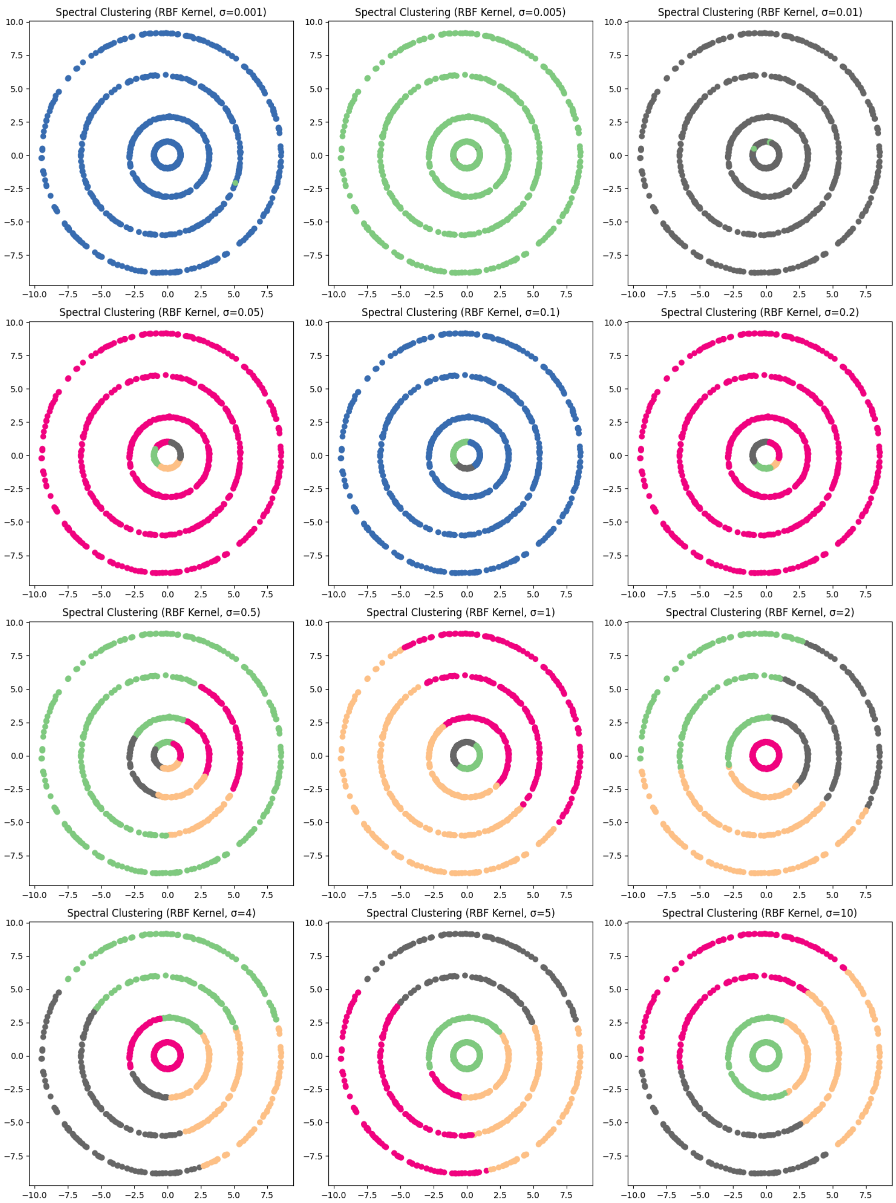

Figure 16: Spectral Clustering with RBF kernel and different sigmas.

The quadratic kernel should be enough to form clusters separating the data as belonging to different concentric circles. However, it doesn't seem to perform as expected, although the performance is better when taking the Laplacian of the Kernel matrix. Further, perhaps centering the kernel matrix would help separate the clusters. I have also found out that the kernel matrix, despite the kernel function being a valid map, leads to non-trivial negative eigen values when using np.linalg.eigh function. Further investigation is required to make the spectral clustering behave as expected.

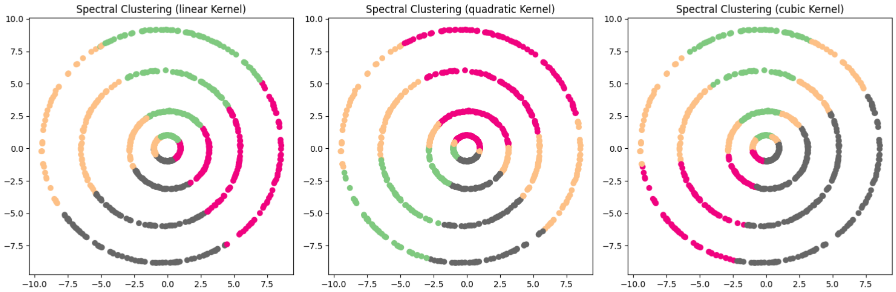

Figure 17: Spectral Clustering with different polynomial kernels.

2.d Instead of using the method suggested by spectral clustering to map eigenvectors to cluster assignments, use the following method: Assign data point i to cluster ℓ whenever $\ell = \arg\max_{j=1,\ldots,k} v_{ji}$ where $v_j \in R^n$ is the eigenvector of the Kernel matrix associated with the j-th largest eigenvalue. How does this mapping perform for this dataset? Explain your insights.

Performance is better after this modification, particularly after taking the Laplacian of the kernel matrix.

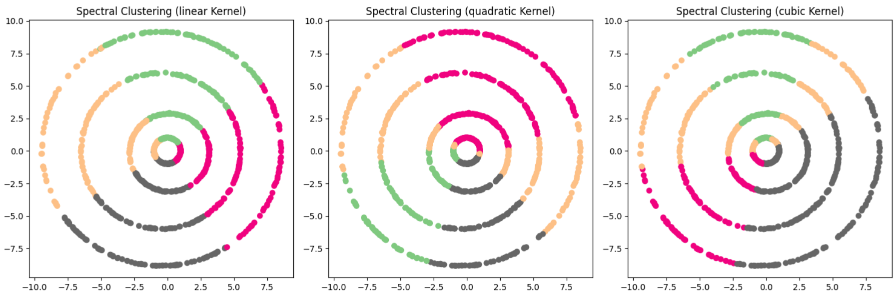

Figure 18: Spectral Clustering modified