Pattern Recognition And Machine Learning – Assignment 1

Question 1. Part I. 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?

Solution: Observation after running the PCA algorithm with data centering on the given dataset were as follows:

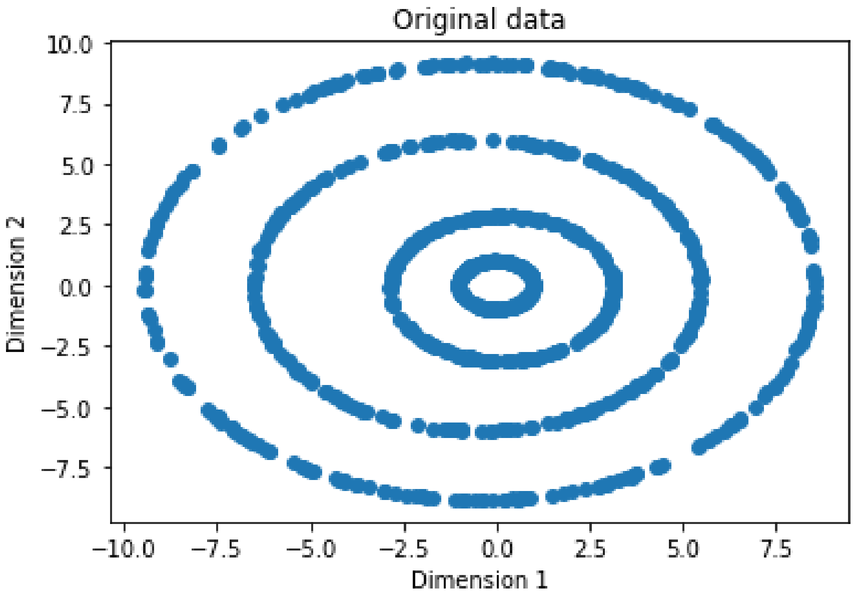
·Data had mean of 4.07e-07 along the first dimension and mean of 2.22e-07 along the second dimension prior to data centering process

· After centering the data and computing the top two eigen vector for the dataset the following variance were recorded:

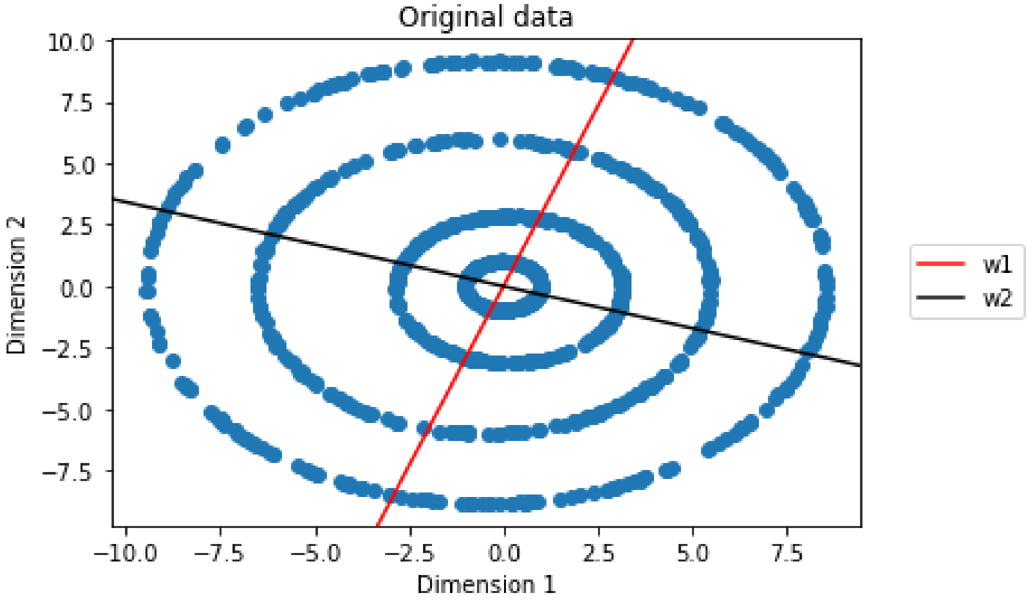
- Variance along the first principal component 54.17 %
- Variance along the second principal component 45.82%

The following is the plot obtained for the dataset given:

Original Dataset:



Original Dataset with principal components drawn:



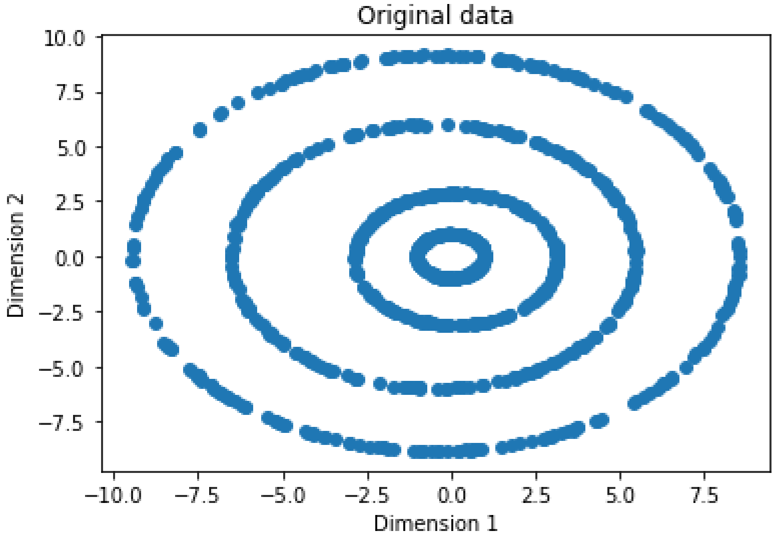
Question 1. Part II. Study the effect of running PCA without centering the data-set. What are your observations? Does Centering help?

Solution: The effect of running PCA algorithm on the given dataset without centering was not much pronounced as the mean of the dataset was very small already. The mean originally recorded was 4.07e-07 and 2.22e-07 respectively which were quite close to zero so data centering in this had a negligible effect.

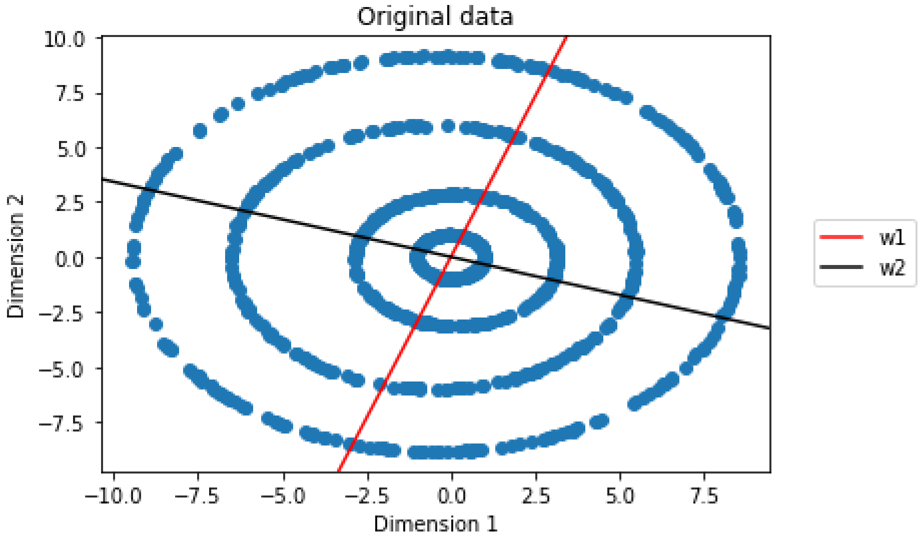
In general Data Centering is a crucial preprocessing step in order to run PCA algorithm the reason being PCA assumes that the new dimension(line) will always pass through the origin and by doing data centering we shift the origin and now the line given by pca will pass through the origin of the shifted data.

The following plot was observed without data centering:

Original Dataset:



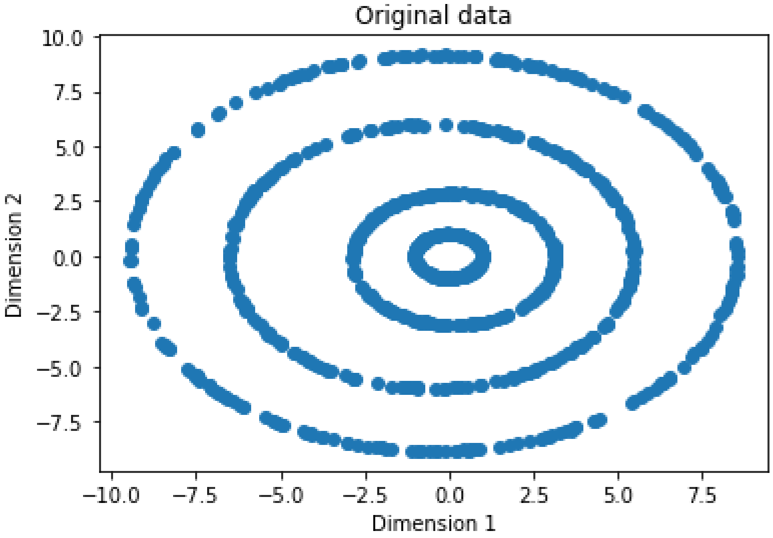
Dataset after performing PCA on uncentered data:



Comparing the figure obtained in part i and ii we can say that data centering did not help much.
Variance along the first principal component 54.17 %
Variance along the second principal component 45.82 %
The variance recorded were also similar.

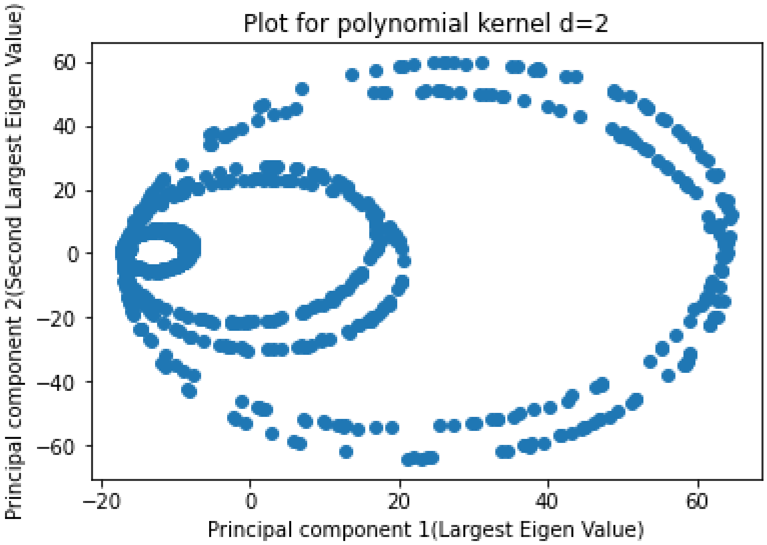
Question 1. Part III. Write a piece of code to implement the Kernel PCA algorithm on this dataset. Use the following kernels : A. κ(x, y) = (1 + x T y) d for d = {2, 3} B. κ(x, y) = exp −(x−y) T (x−y) 2σ2 for σ = {0.1, 0.2, . . . , 1} Plot the projection of each point in the dataset onto the top-2 components for each kernel. Use one plot for each kernel and in the case of (B), use a different plot for each value of σ.

Solution: Original Data:

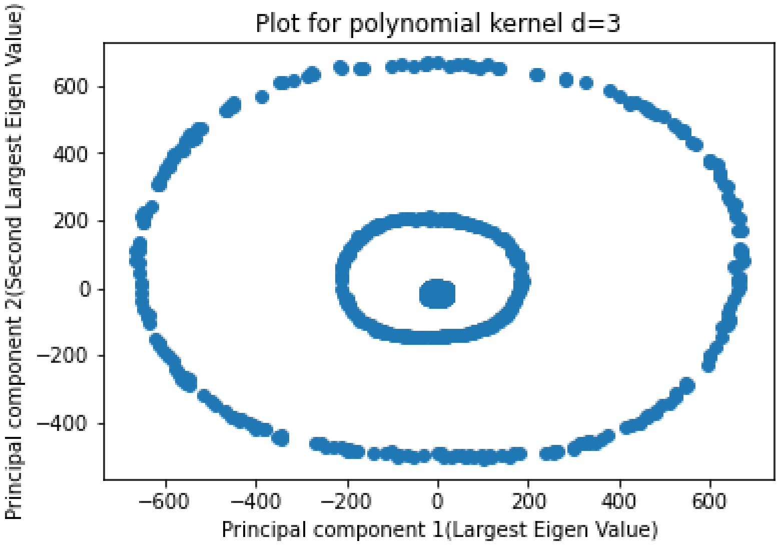


Part A - Polynomial Kernel

For d=2:

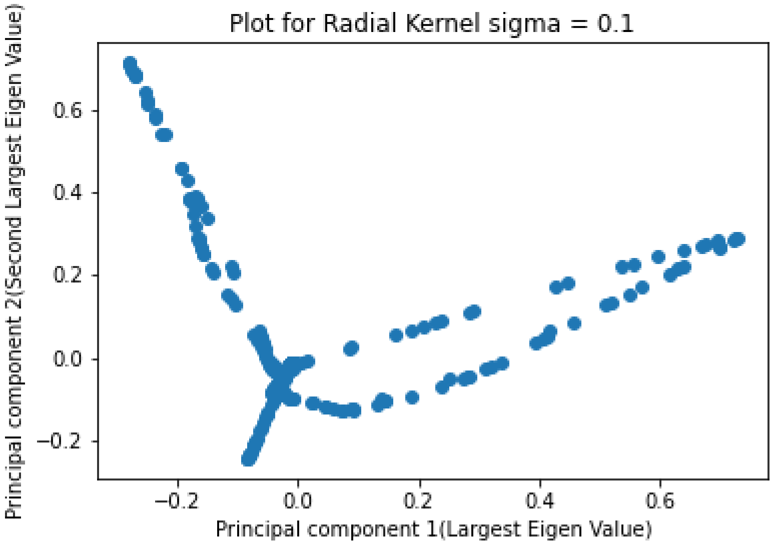


For d=3

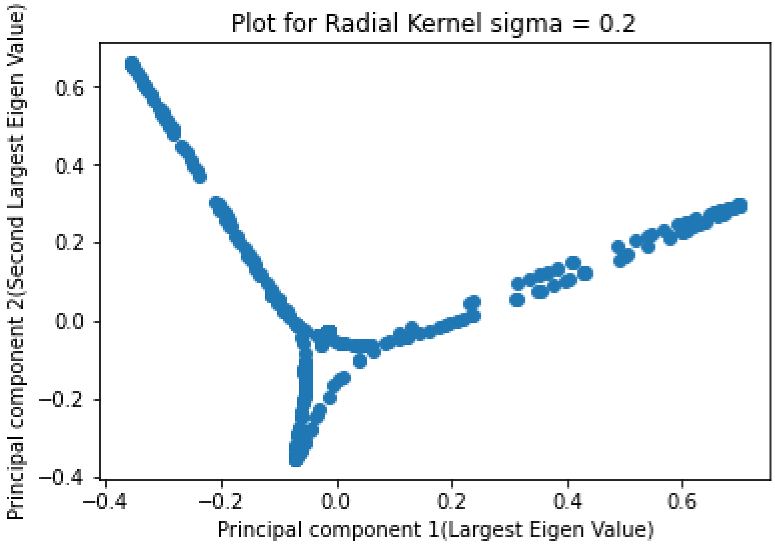


Part B - Radial Basis Kernel

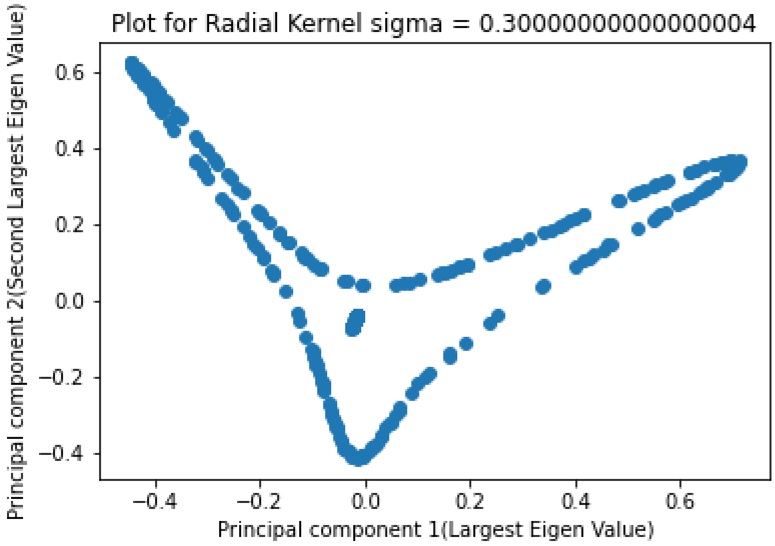
For sigma=0.1



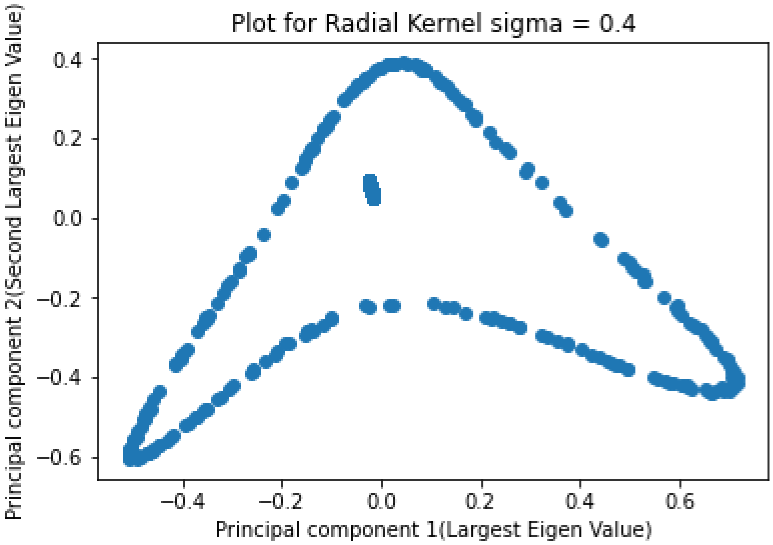
For sigma =0.2



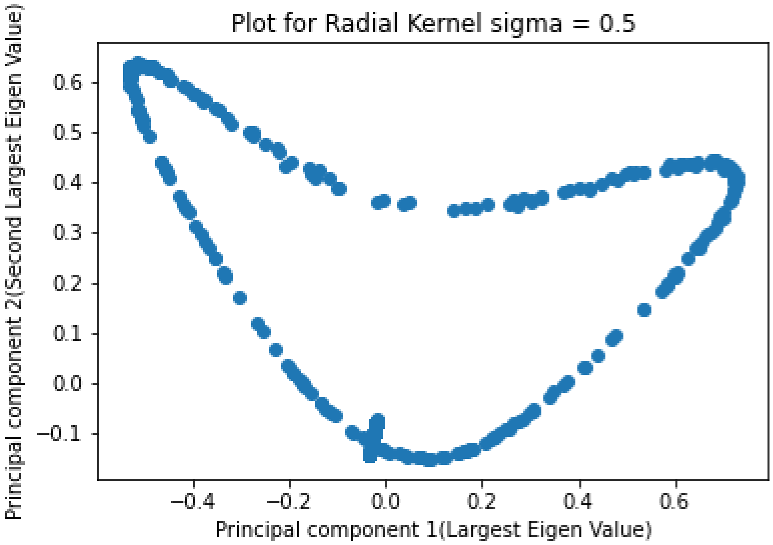
For sigma=0.3



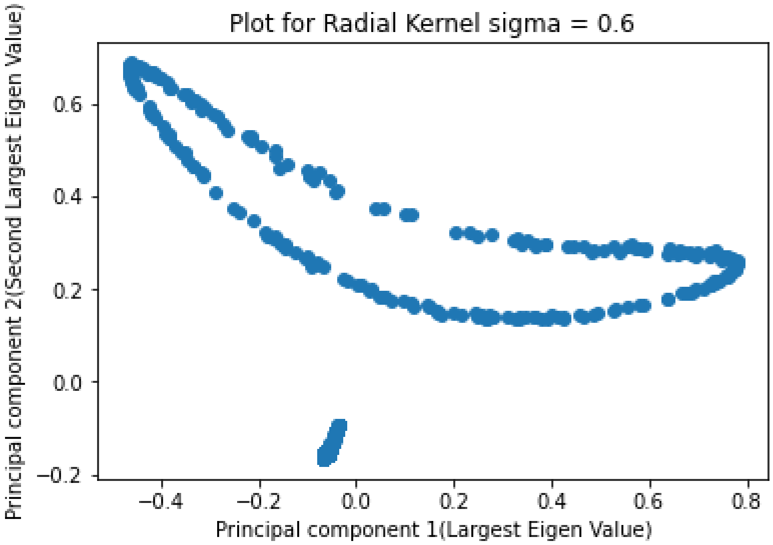
For sigma=0.4



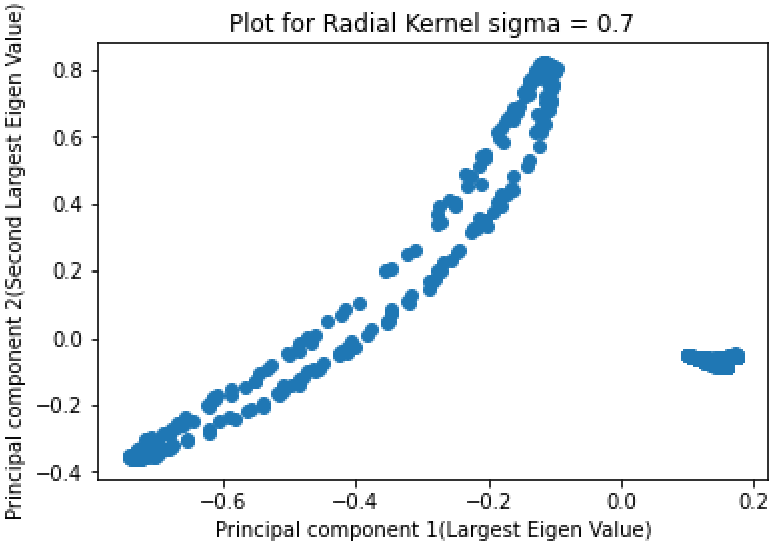
For sigma=0.5



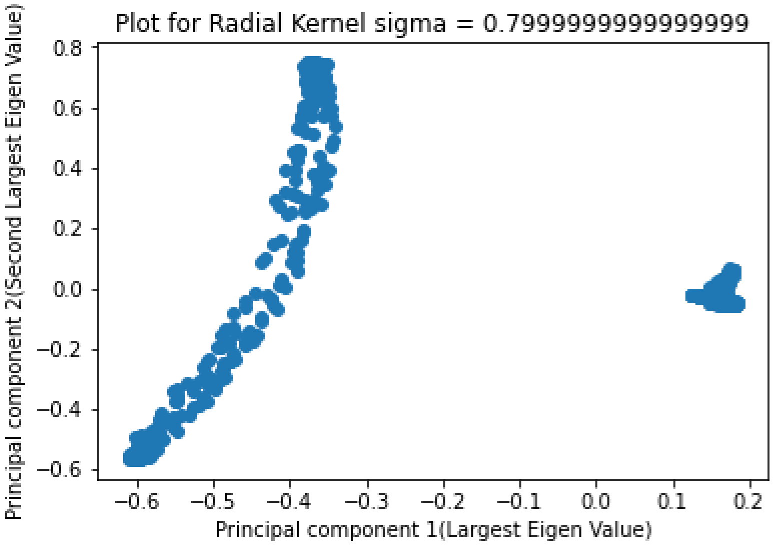
For sigma=0.6



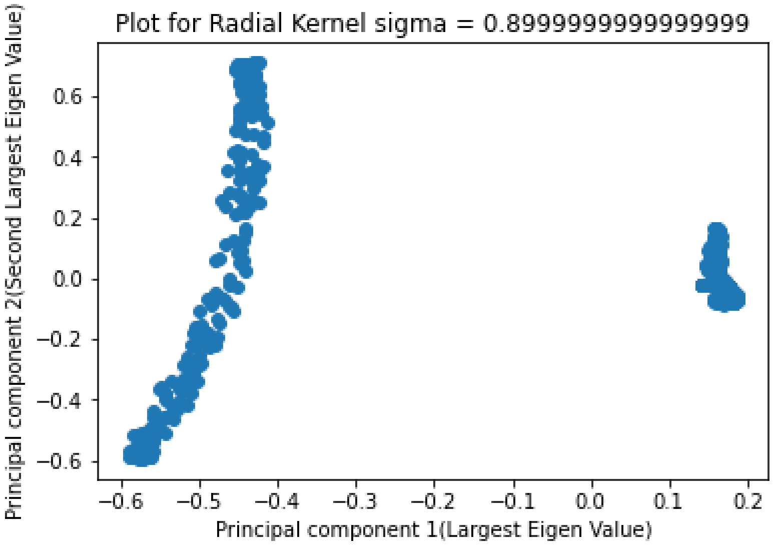
For sigma=0.7



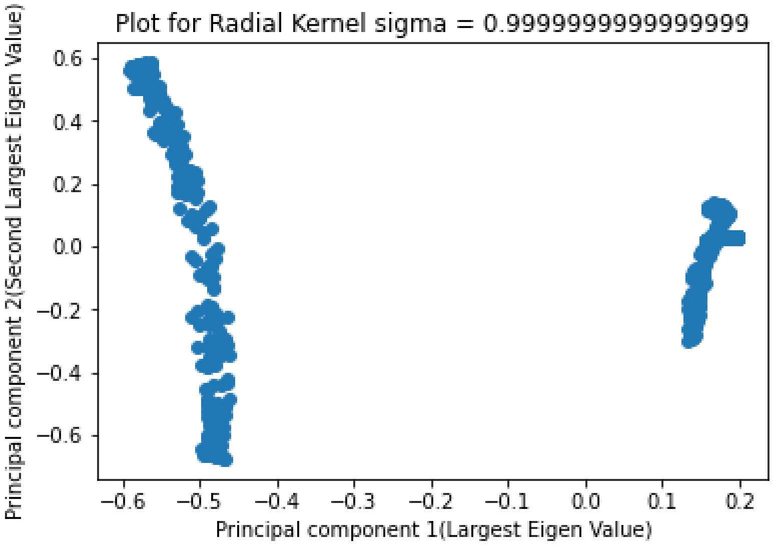
For sigma=0.8



For sigma=0.9



For sigma=1


Plot for Radial Kernel sigma = 0.9999999999999999
Principal component 2(Second Largest Eigen Value)
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.6
−0.5
−0.4
−0.3
−0.2
−0.1
0.0
0.1
0.2
Principal component 1(Largest Eigen Value)

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

Solution: According to me Polynomial Kernel with d=3 is best suited for the given dataset.

Argument:

I. For Polynomial kernel d=2:
The total variance retained by the top two component for the given choice of kernel was close to 68.53%

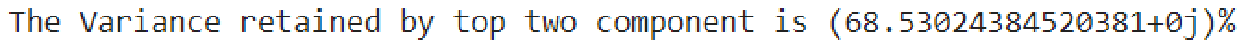
II. For PolynomialKernel d=3:
The total variance retained by the top two component for the given choice of kernel was 73.41%

III. After running radial basis function over values of sigma from 0.1 to 1.0, The maximum variance retained by top two component was 15.64% which was recorded for sigma = 1.0

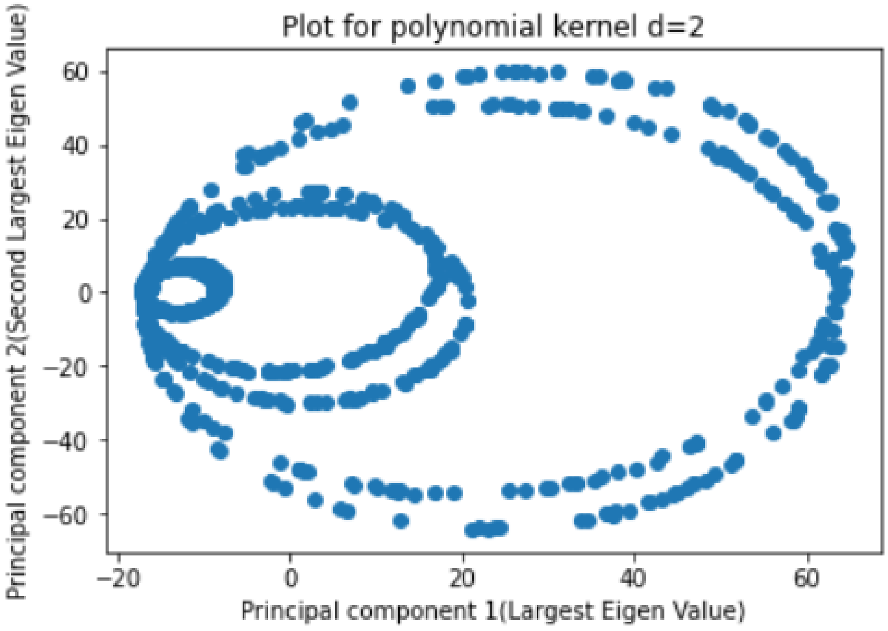
Conclusion:

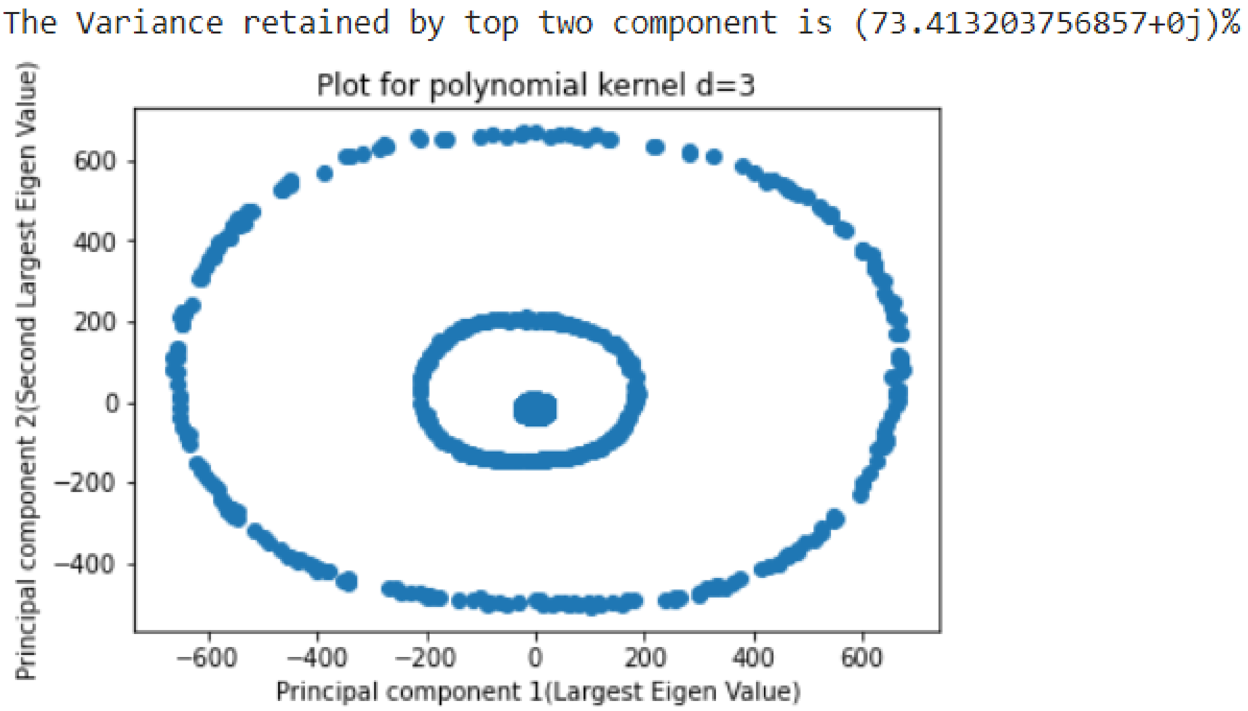
From the above argument we can conclude that Polynomial kernel with d=3 was best suited for the dataset.

Proof:



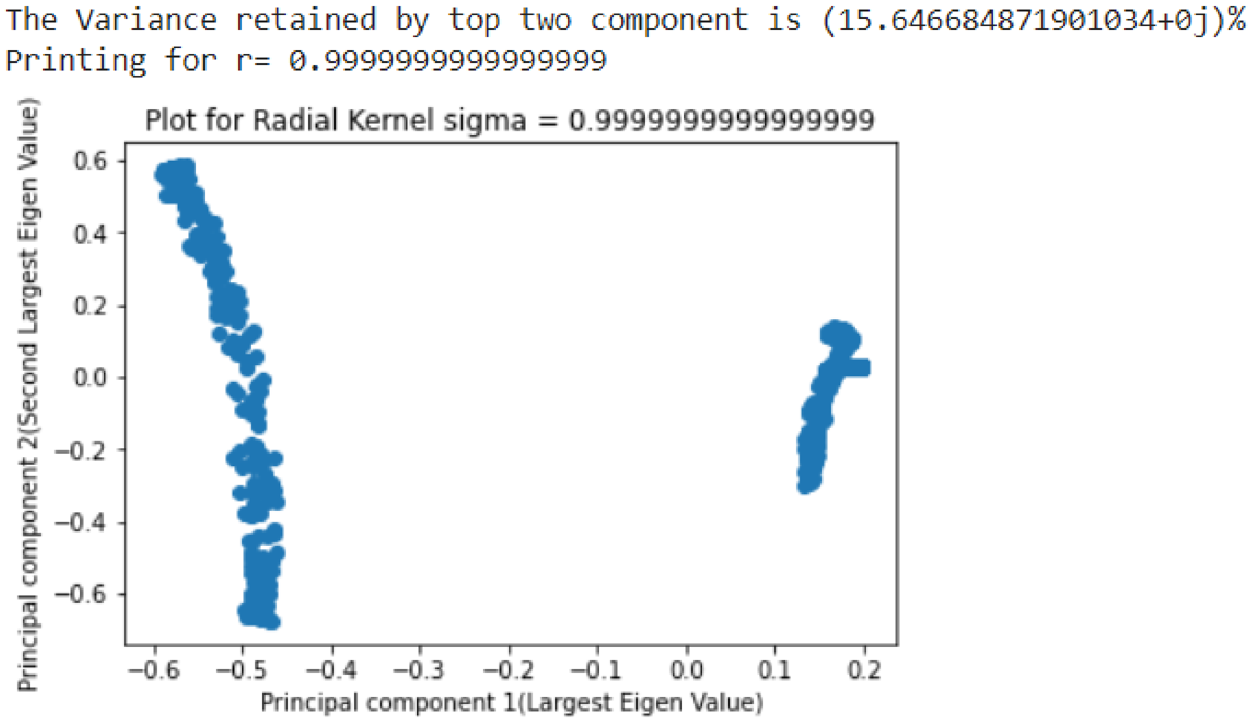
```
The Variance retained by top two component is (68.53024384520381+0j)%
```





```
The Variance retained by top two component is (73.413203756857+0j)%
```

Plot for polynomial kernel d=3
Principal component 2(Second Largest Eigen Value)
600
400
200
0
−200
−400
−600
−400
−200
0
200
400
600
Principal component 1(Largest Eigen Value)

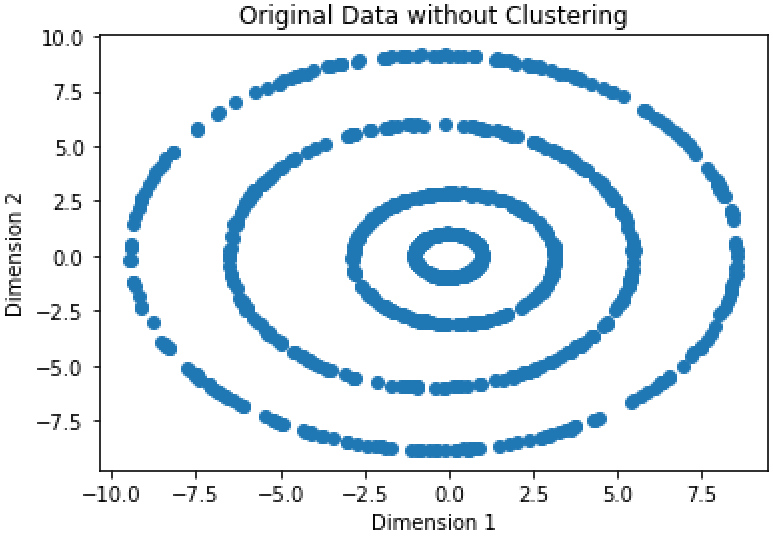


```
The Variance retained by top two component is (15.646684871901034+0j)%
Printing for r= 0.9999999999999999
```

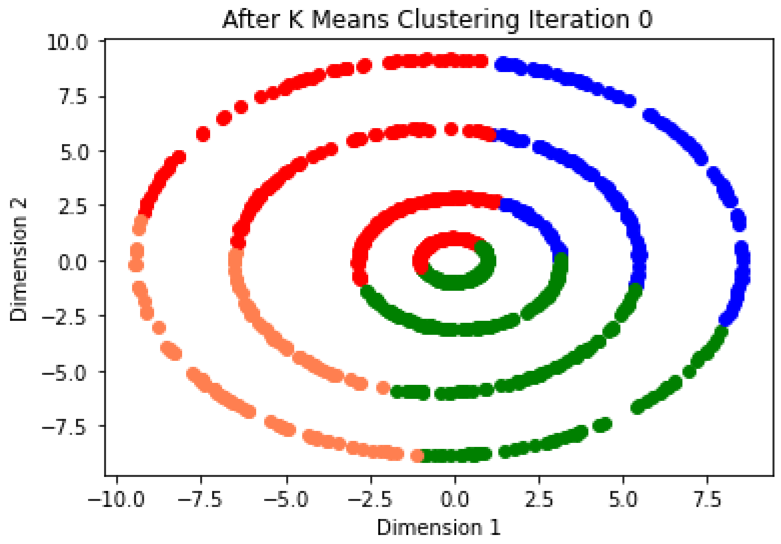
Plot for Radial Kernel sigma = 0.9999999999999999
Principal component 2(Second Largest Eigen Value)
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.6
−0.5
−0.4
−0.3
−0.2
−0.1
0.0
0.1
0.2
Principal component 1(Largest Eigen Value)

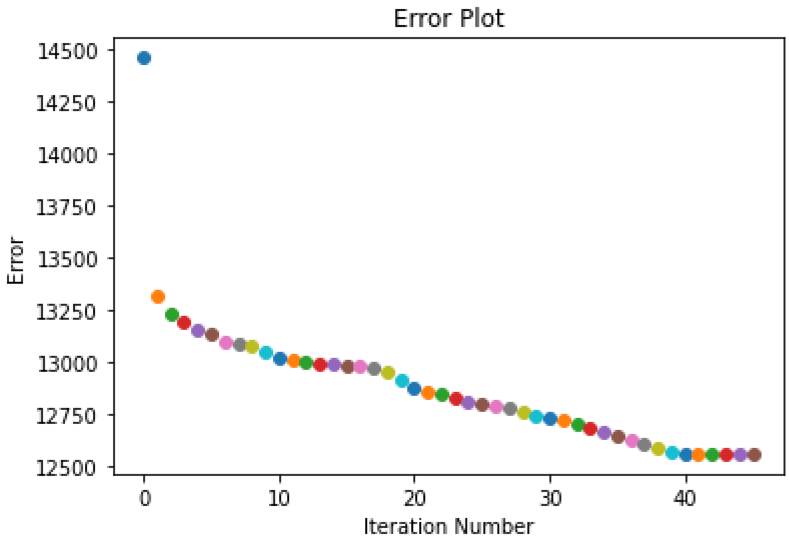
Question 2. Part I. Write a piece of code to run the algorithm studied in class for the K-means problem with k = 4 . Try 5 different random initialization and plot the error function w.r.t iterations in each case. In each case, plot the clusters obtained in different colors.

Solution: Original Dataset

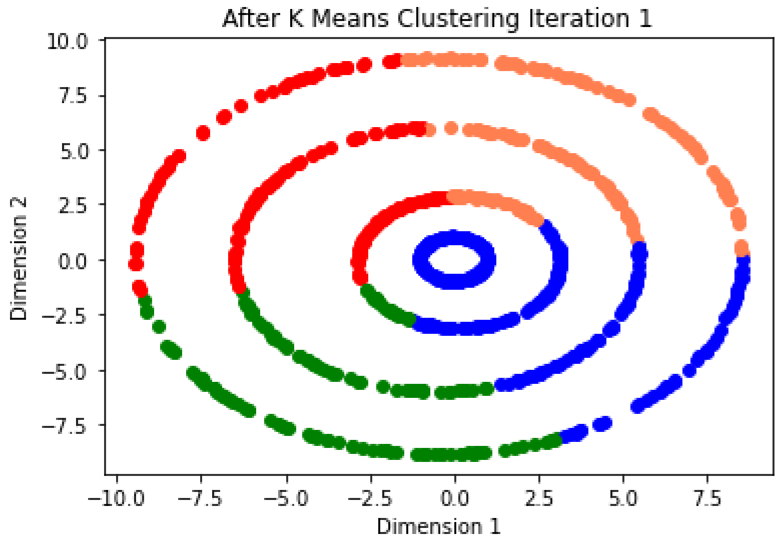


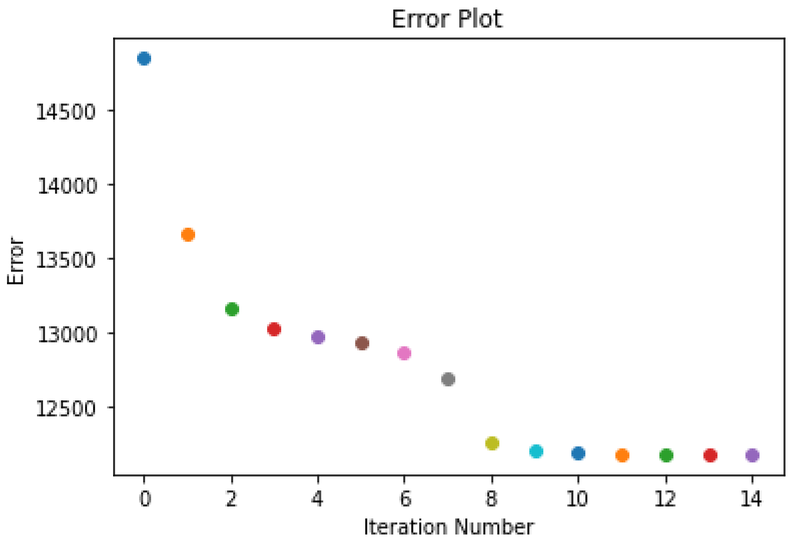
Random Initialization :: The iteration is 0
Steps to converge :: 44


After K Means Clustering Iteration 0

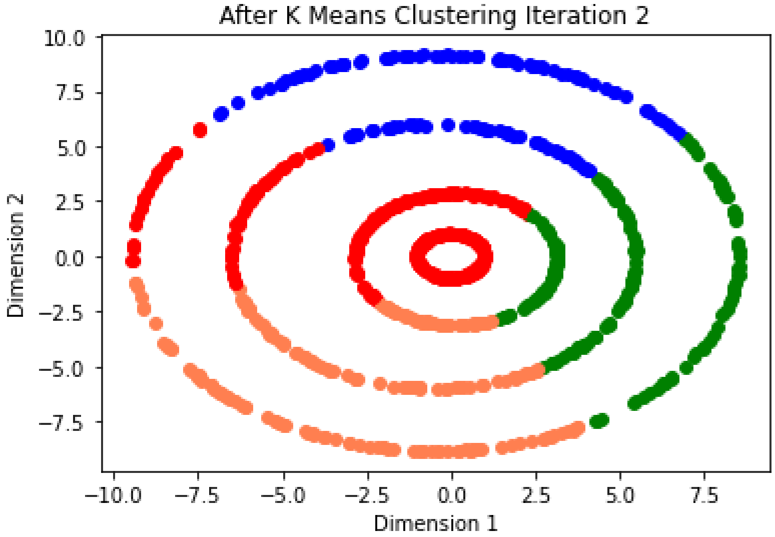

Error Plot

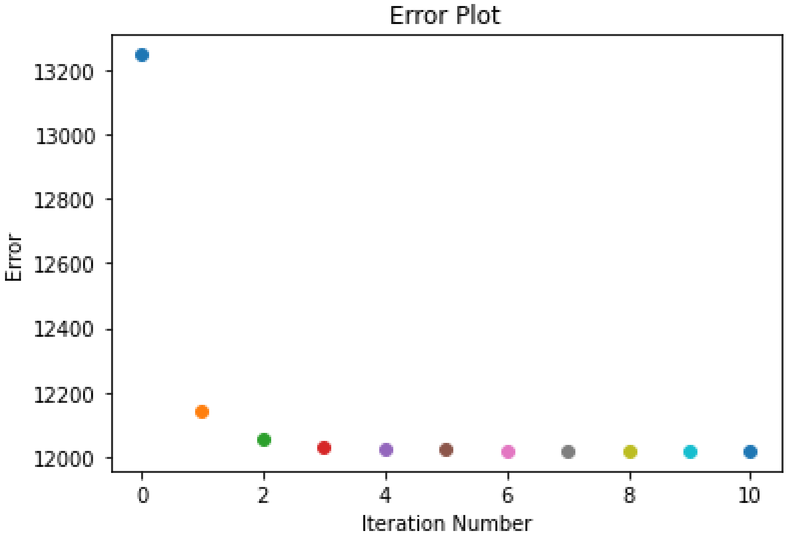
```
Random Initialization :: The iteration is  1
Steps to converge ::  13
```



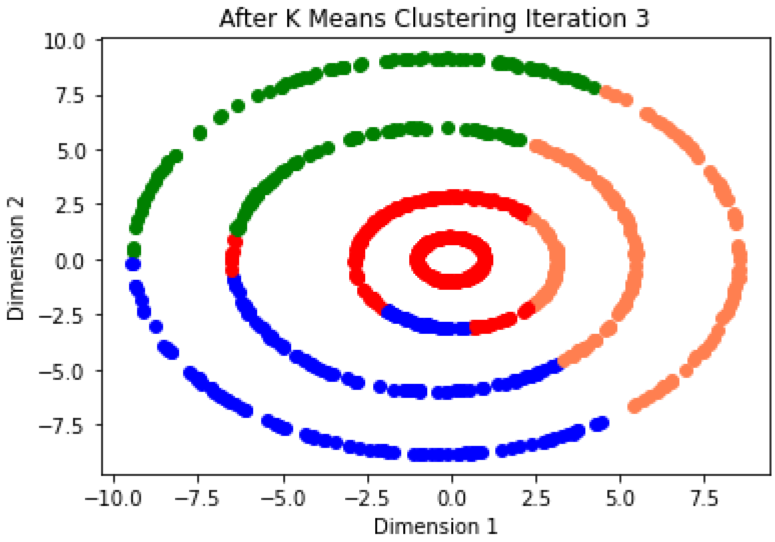


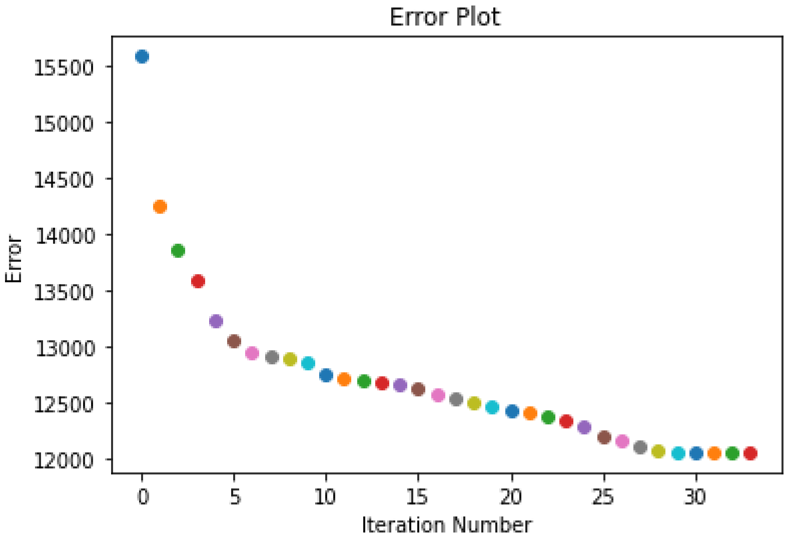
```
Random Initialization :: The iteration is  2
Steps to converge ::  9
```



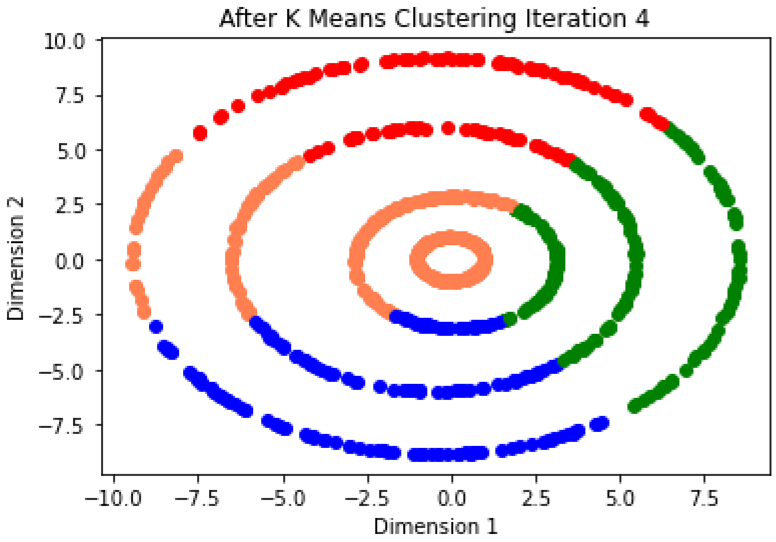


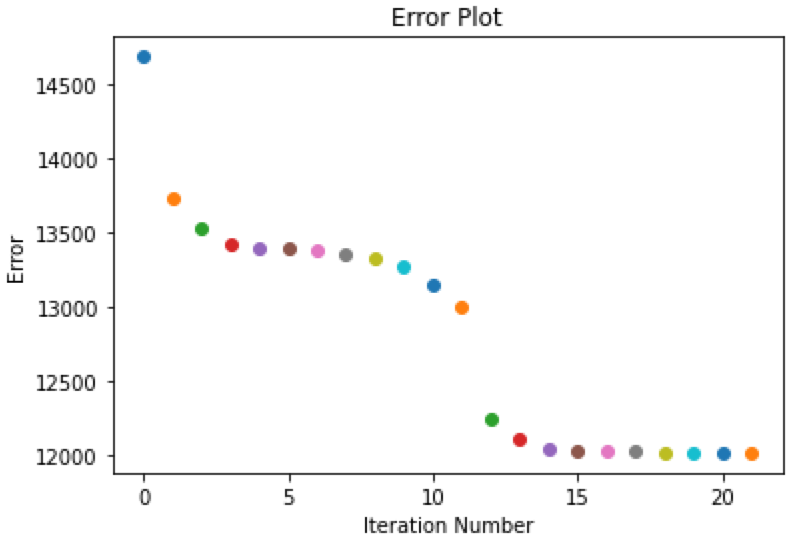
```
Random Initialization :: The iteration is  3
Steps to converge ::  32
```





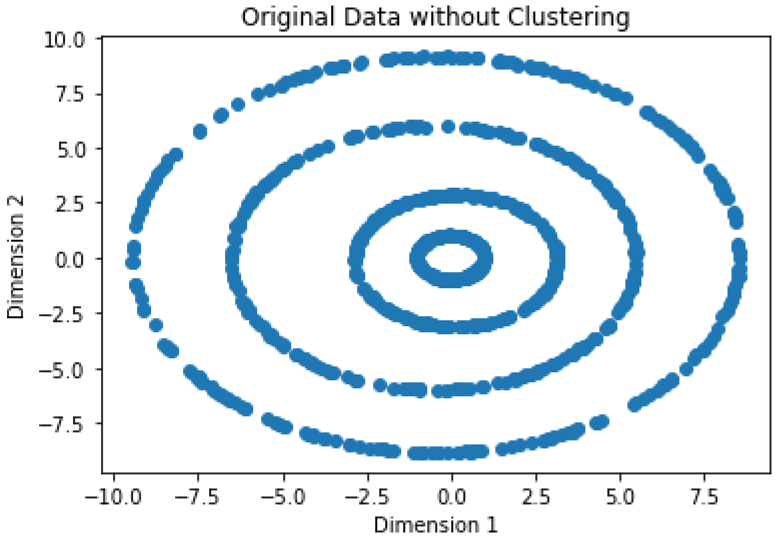
```
Random Initialization :: The iteration is  4
Steps to converge ::  20
```



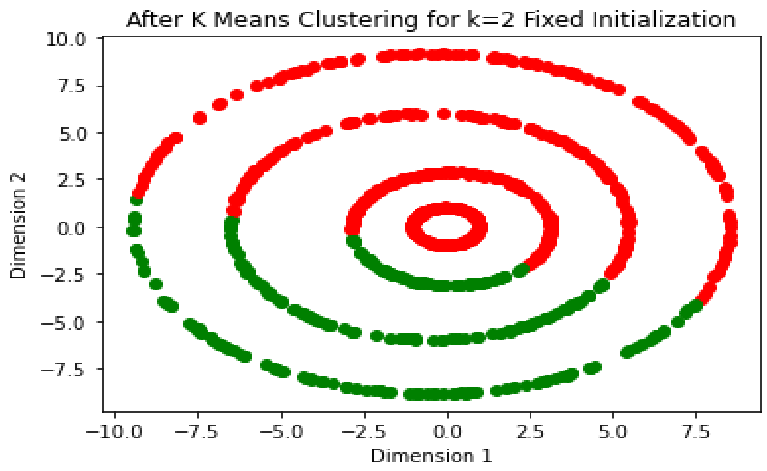


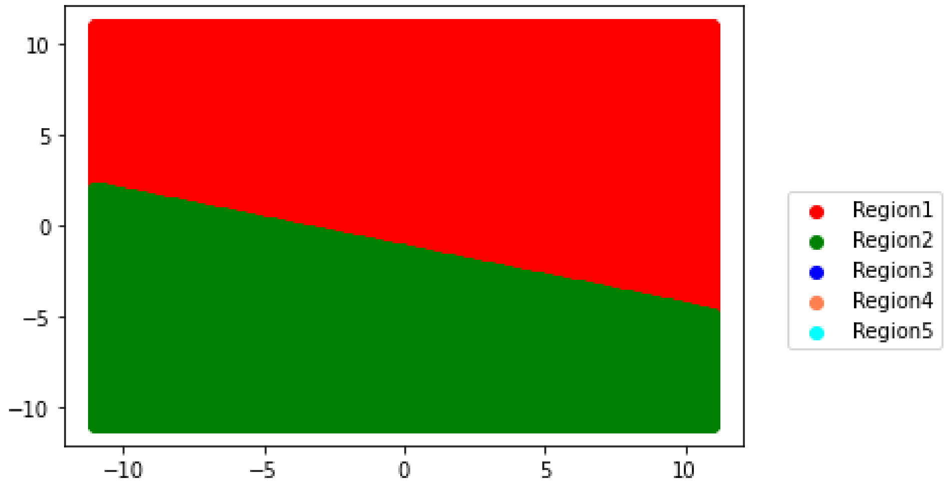
Question 2. Part II. Fix a random initialization. For K = {2, 3, 4, 5}, obtain cluster centers according to K-means algorithm using the fixed initialization. For each value of K, plot the Voronoi regions associated to each cluster center. (You can assume the minimum and maximum value in the data-set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

Solution: Original Dataset

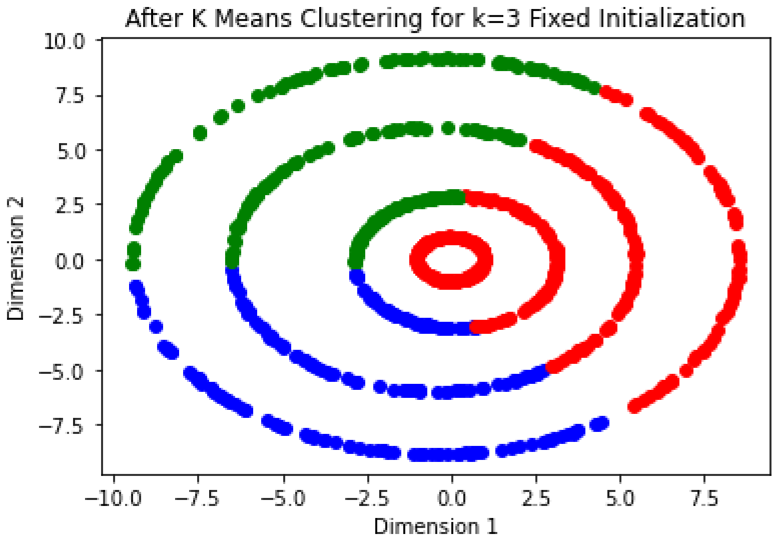


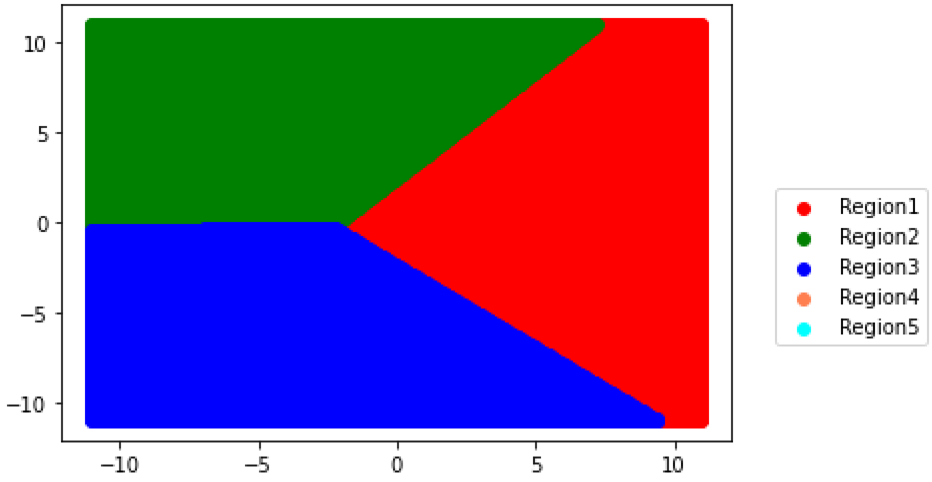
```
Running for the value of k as ::  2
Steps to converge ::  39
```



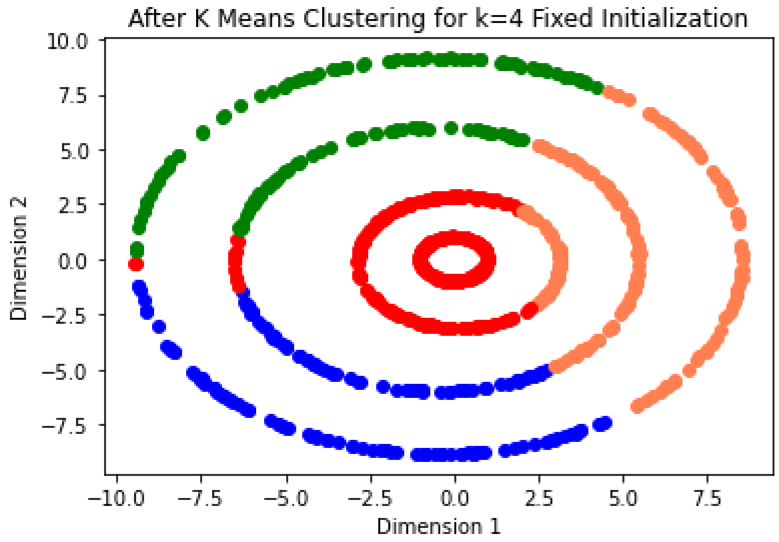


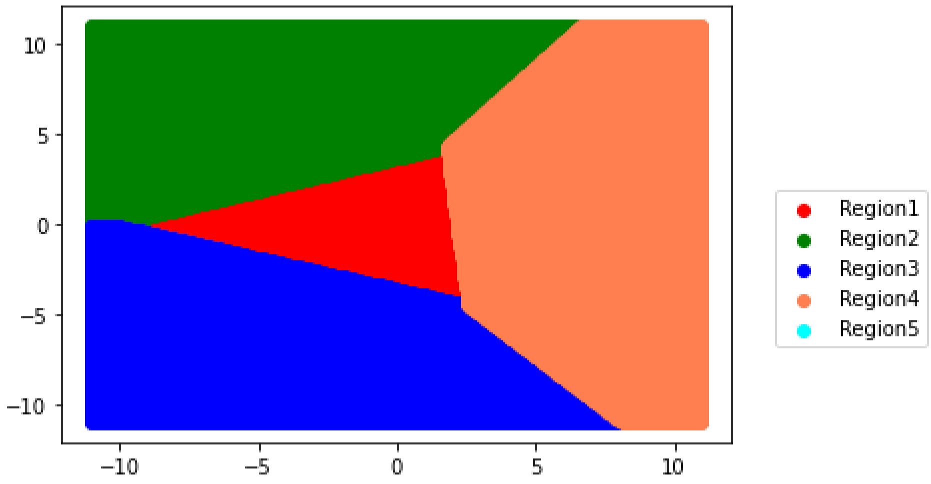
```
Running for the value of k as ::  3
Steps to converge ::  25
```



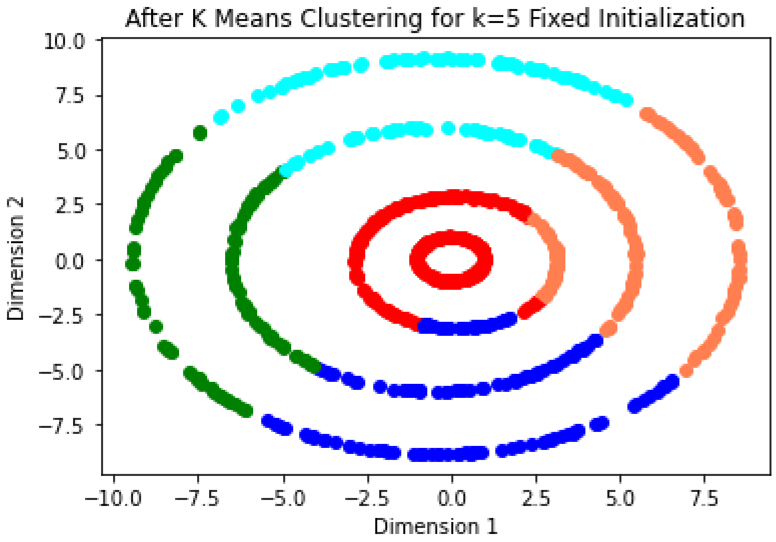


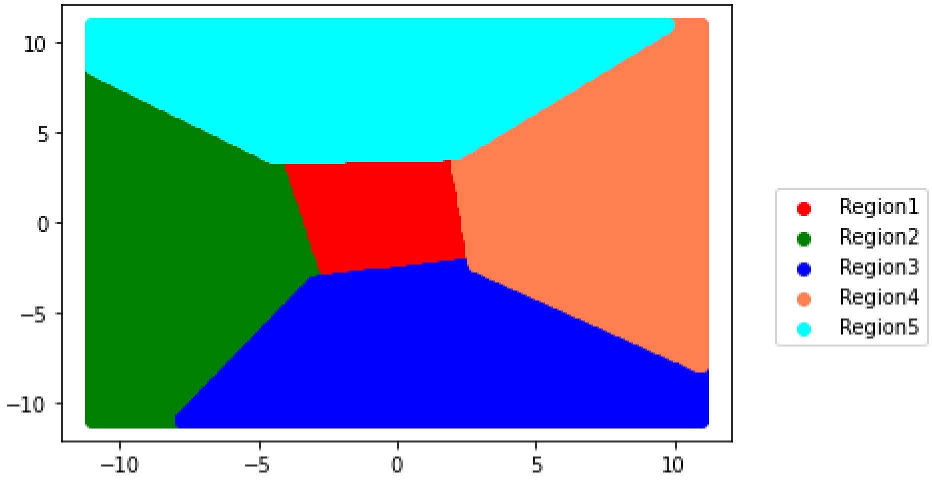
```
Running for the value of k as ::  4
Steps to converge ::  21
```





```
Running for the value of k as ::  5
Steps to converge ::  8
```

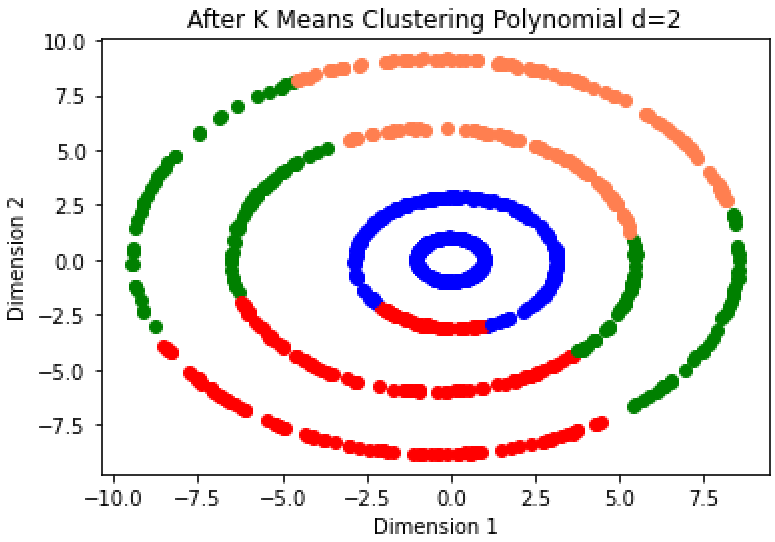




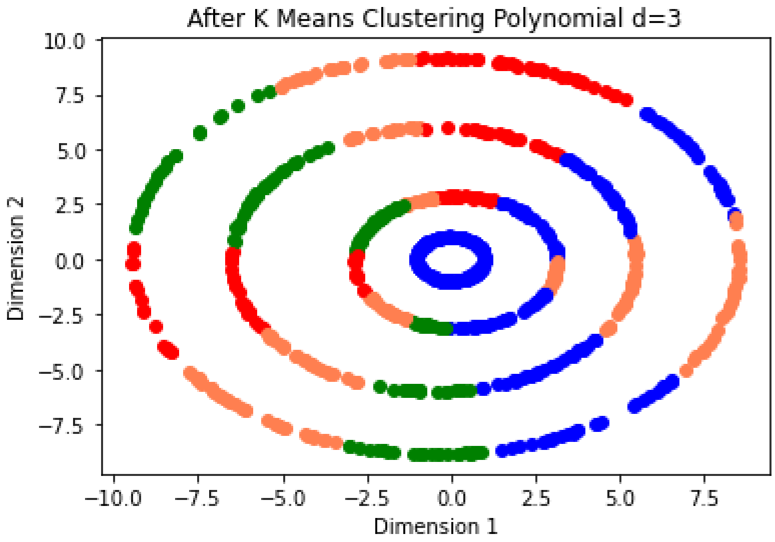
Question 3. Part III. Run the spectral clustering algorithm (spectral relaxation of K-means using KernelPCA) k = 4. Choose an appropriate kernel for this data-set and plot the clusters obtained in different colors. Explain your choice of kernel based on the output you obtain.

Solution: According to me polynomial kernel with d=2 is a good choice compared to the rest as the ideal output should be separate rings as separate clusters and observing the output k means with polynomial kernel d=2 seems to be the best option.

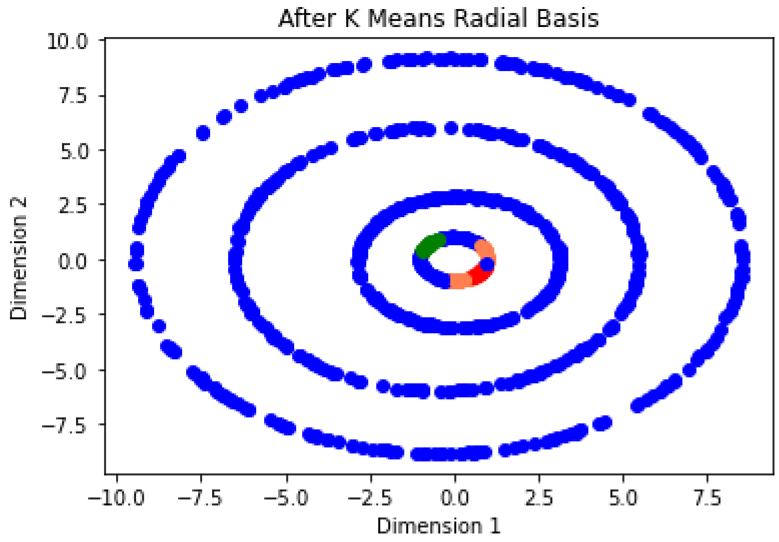
For Polynomial Kernel: d=2

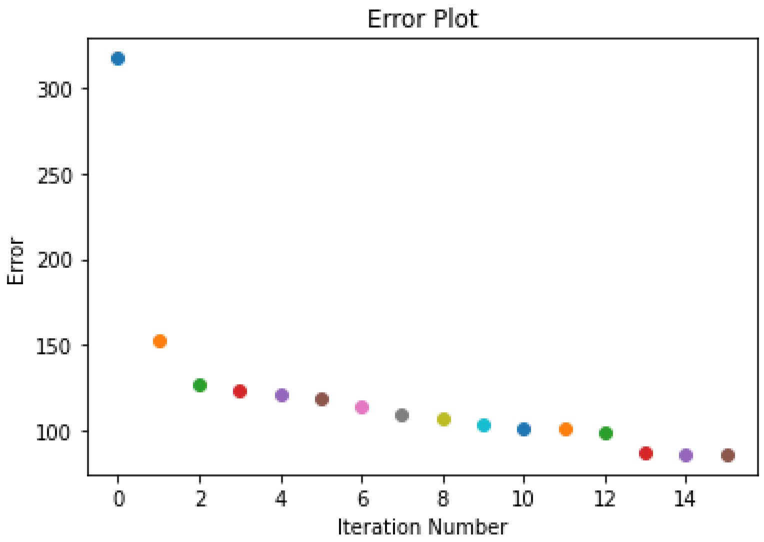


For Polynomial Kernel: d=3

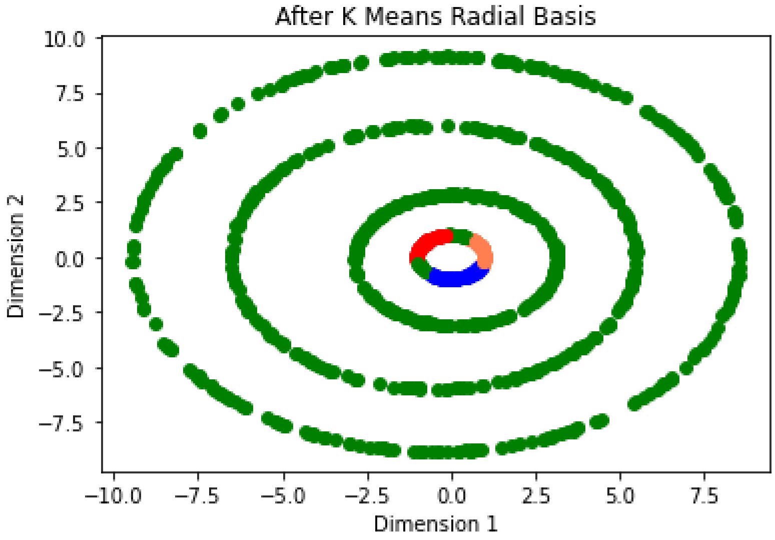


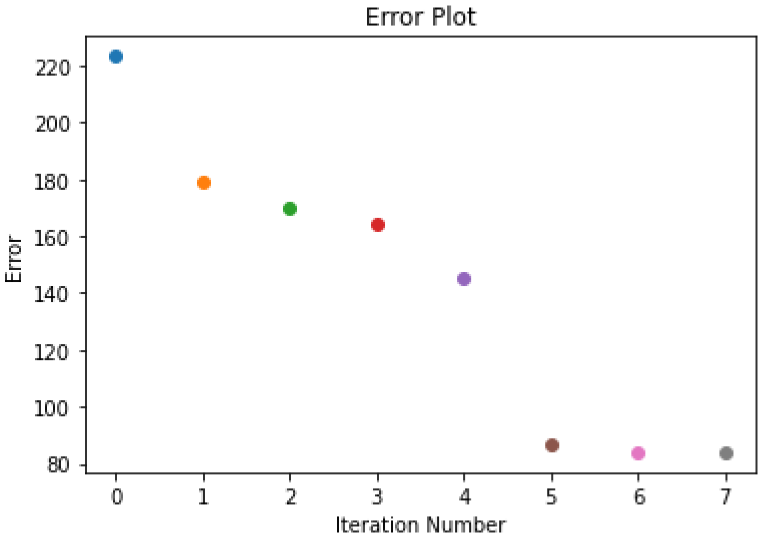
For Radial Basis:
sigma=0.1


After K Means Radial Basis
Dimension 2
Dimension 1
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5

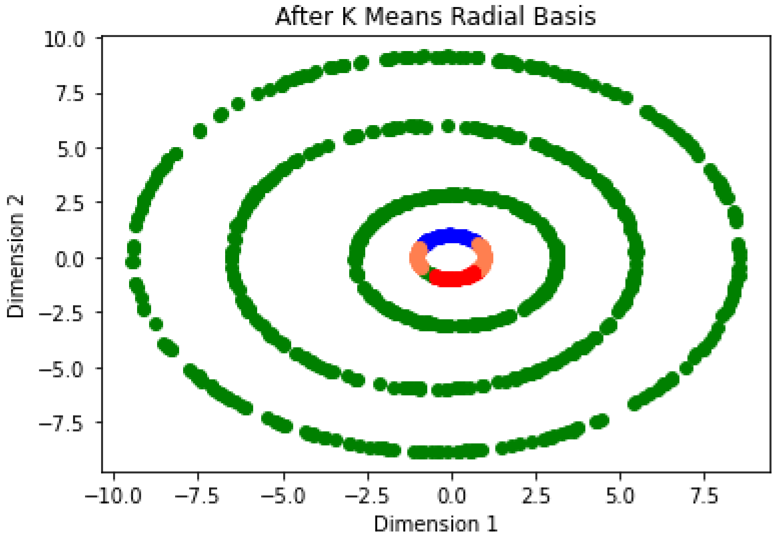

Error Plot
Error
Iteration Number
300
250
200
150
100
0
2
4
6
8
10
12
14

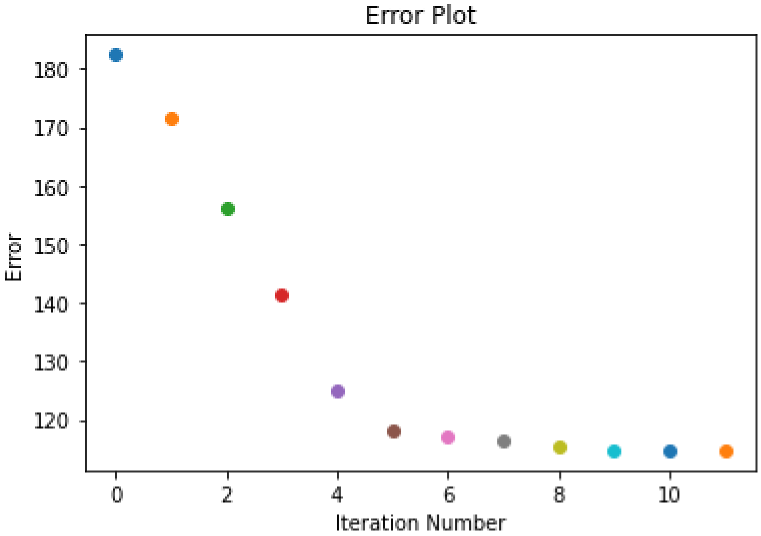
sigma=0.2


After K Means Radial Basis
Dimension 2
Dimension 1
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5

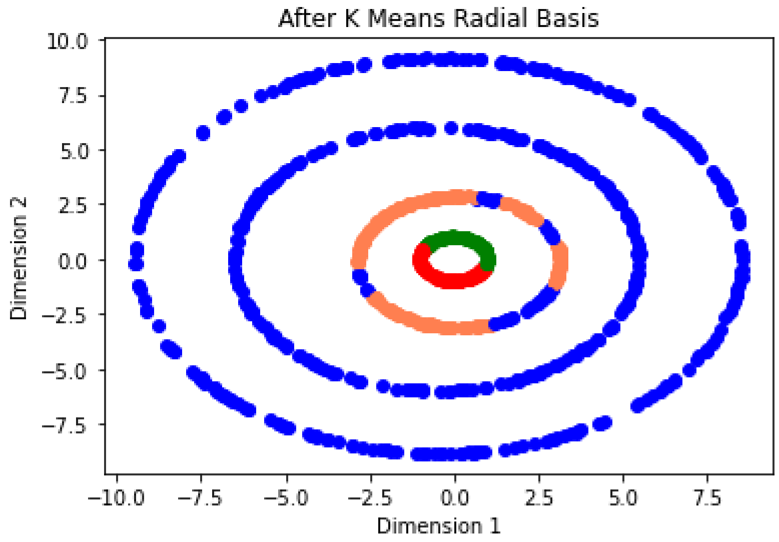

Error Plot
Error
Iteration Number
220
200
180
160
140
120
100
80
0
1
2
3
4
5
6
7

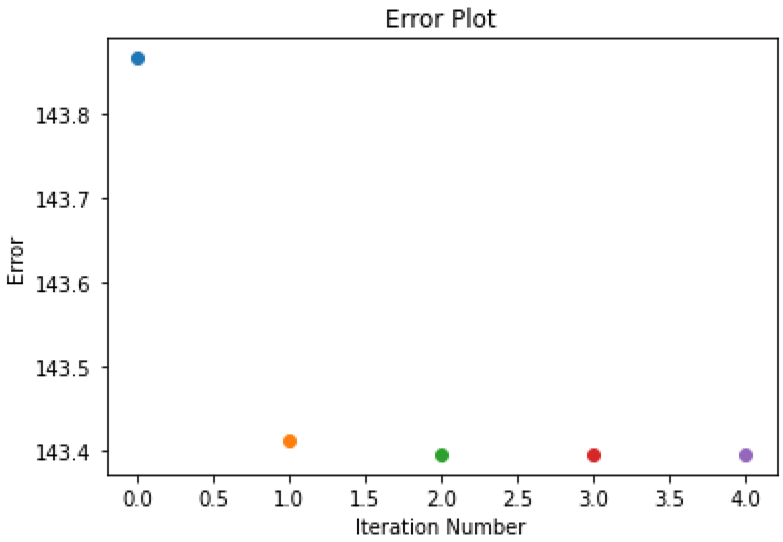
sigma=0.3


After K Means Radial Basis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1

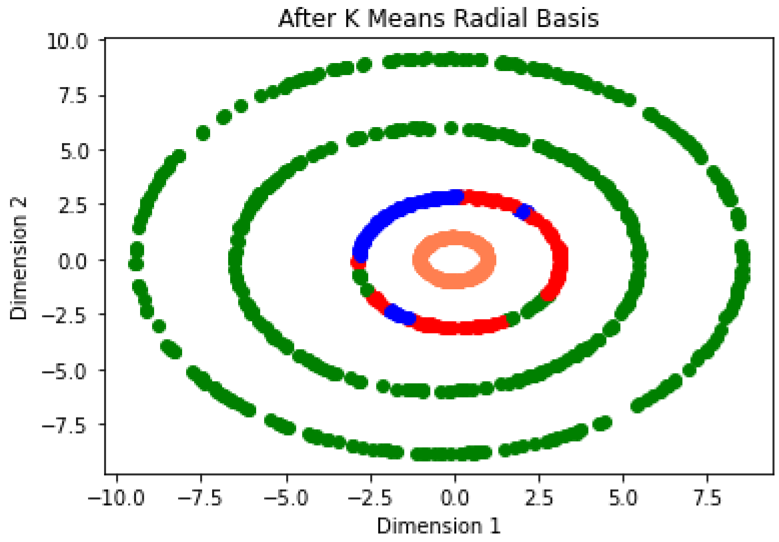

Error Plot
180
170
160
150
140
130
120
Error
0
2
4
6
8
10
Iteration Number

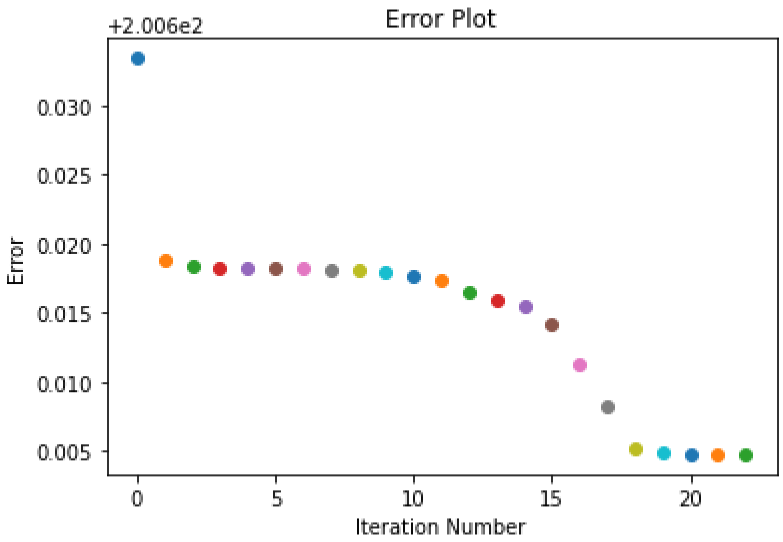
sigma=0.4


After K Means Radial Basis
Dimension 2
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1

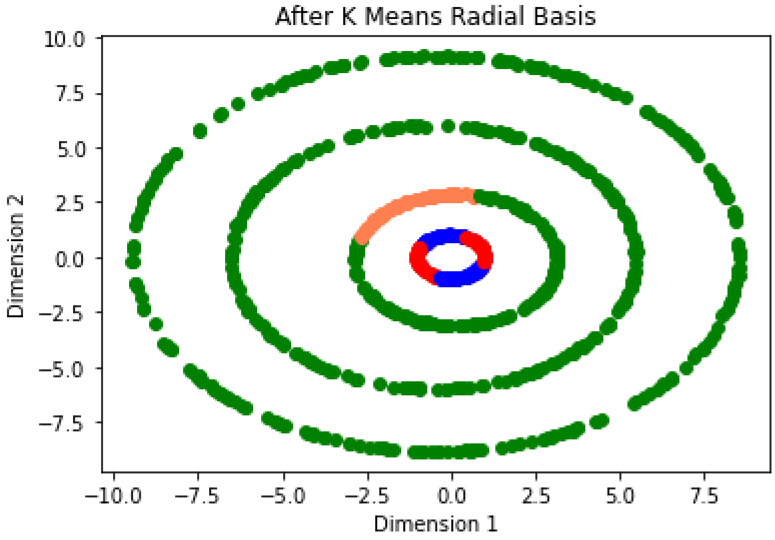

Error Plot
Error
143.8
143.7
143.6
143.5
143.4
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Iteration Number

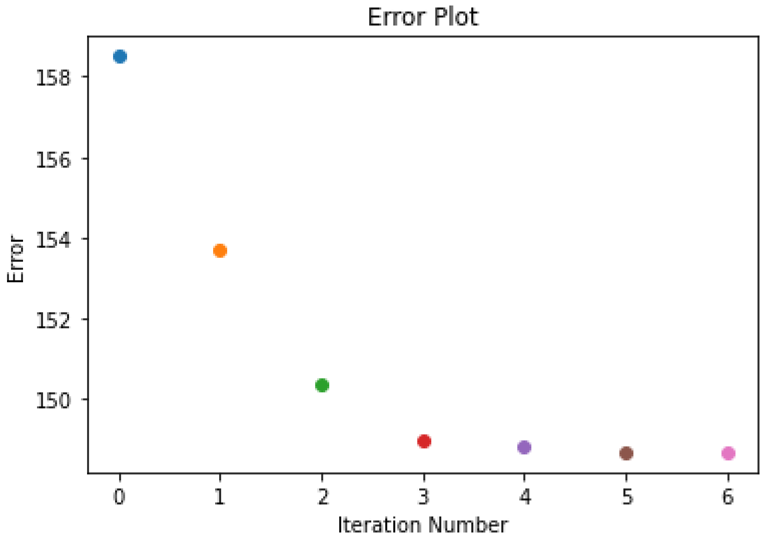
sigma=0.5


After K Means Radial Basis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1

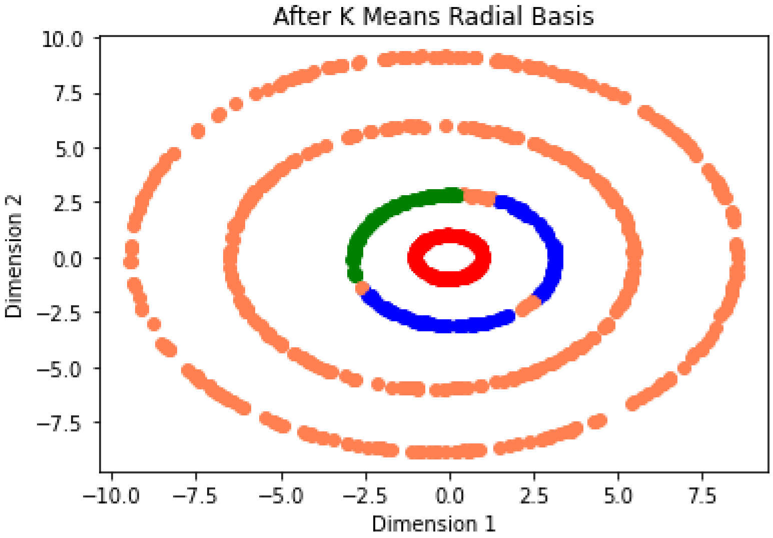

Error Plot
+2.006e2
0.030
0.025
0.020
0.015
0.010
0.005
Error
0
5
10
15
20
Iteration Number

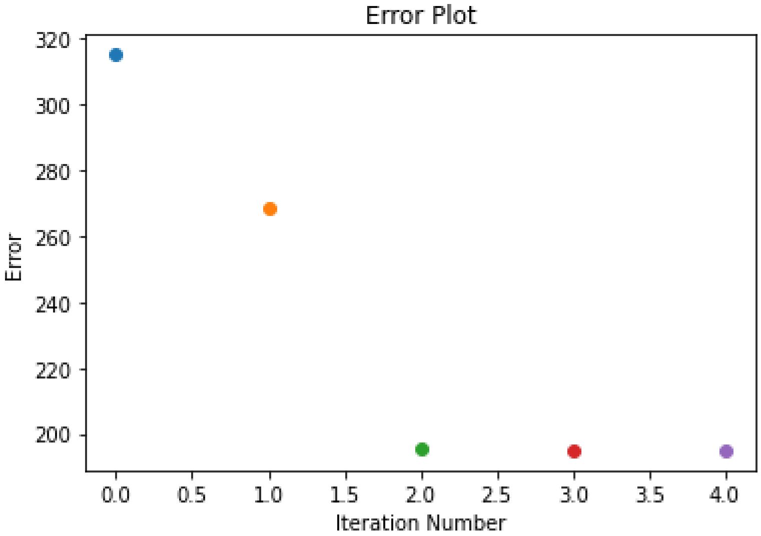
sigma=0.6


After K Means Radial Basis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1

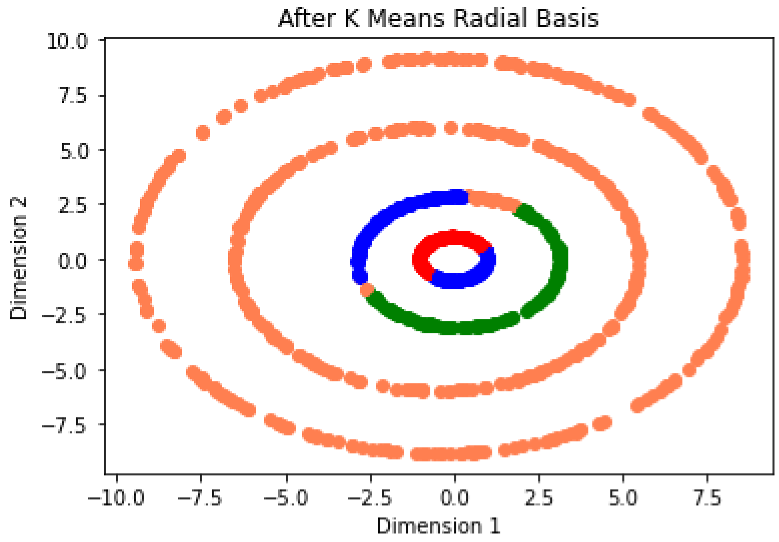

Error Plot
158
156
154
152
150
Error
0
1
2
3
4
5
6
Iteration Number

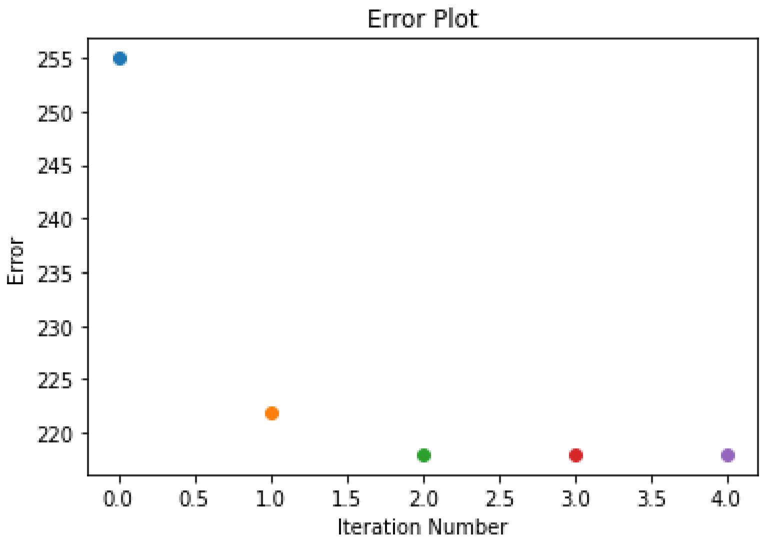
sigma=0.7


After K Means Radial Basis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1

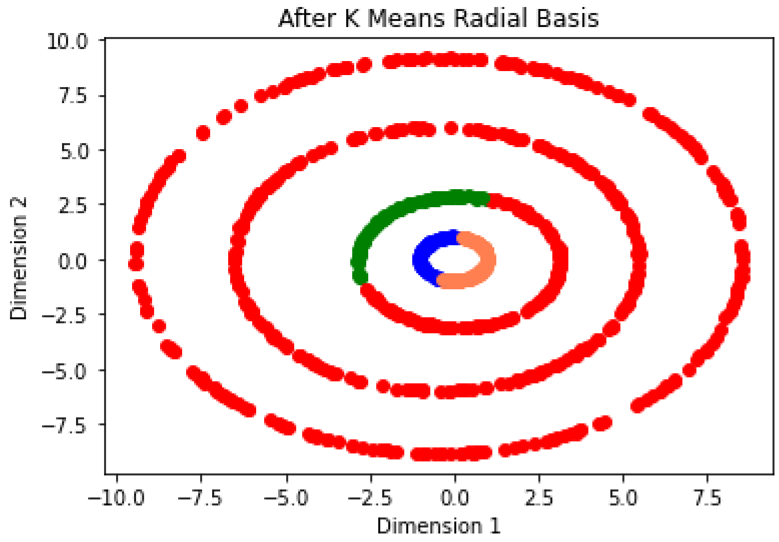

Error Plot
320
300
280
260
240
220
200
Error
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Iteration Number

sigma=0.8

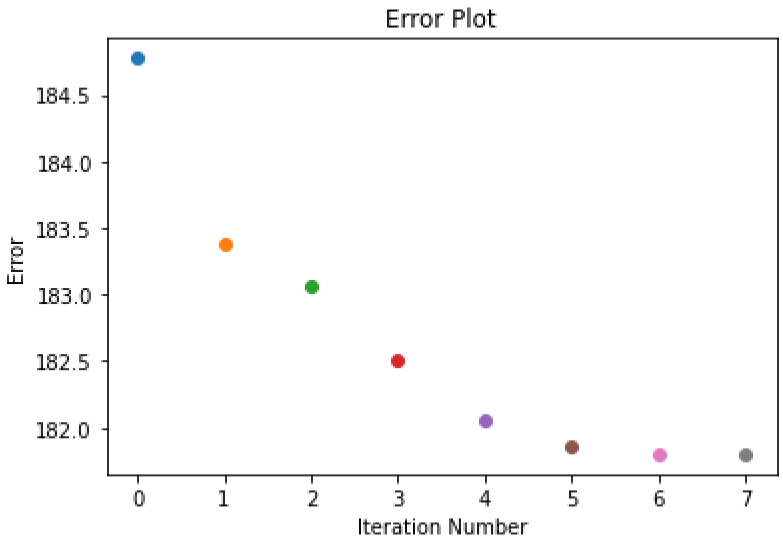

After K Means Radial Basis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1


Error Plot
255
250
245
240
235
230
225
220
Error
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Iteration Number

sigma=0.9

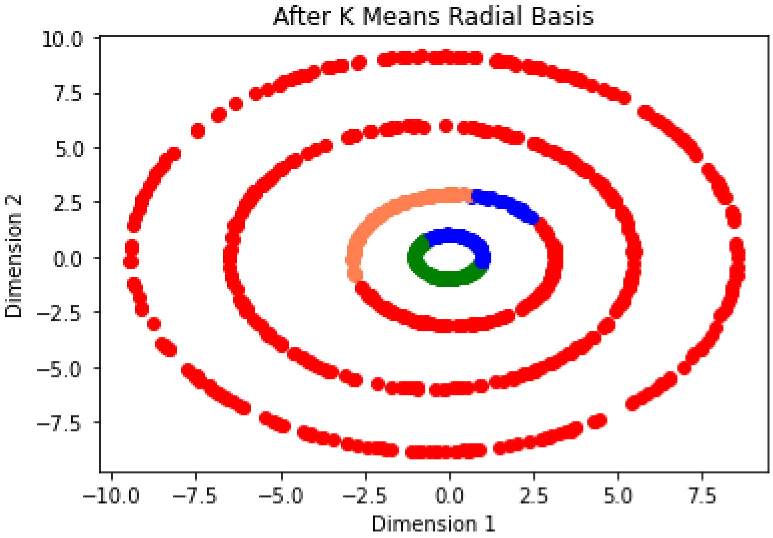


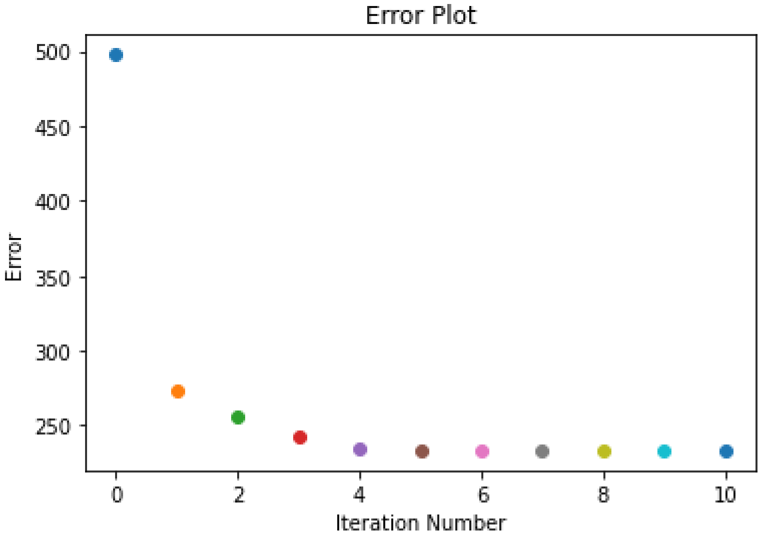
After K Means Radial Basis



Error Plot

sigma=1


After K Means Radial Basis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1


Error Plot
500
450
400
350
300
250
Error
0
2
4
6
8
10
Iteration Number

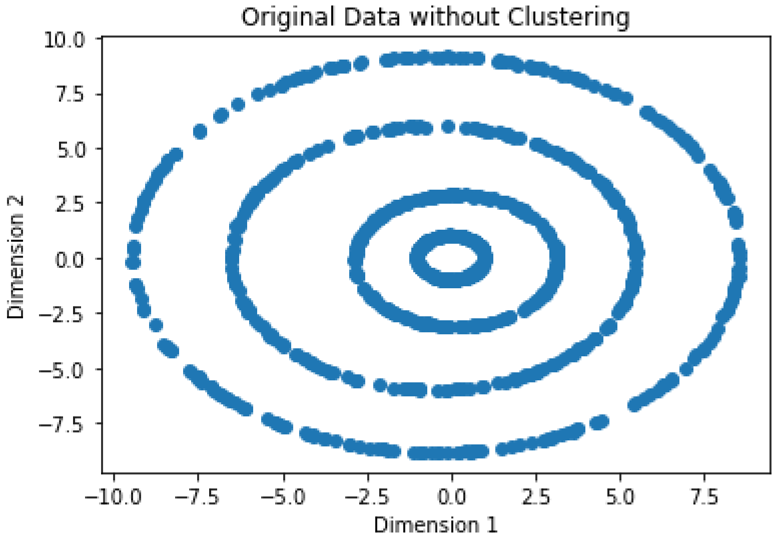
Question 2. Part IV. Instead of using the method suggested by spectral clustering to map eigenvectors to cluster assignments, use the following method: Assign data point i to cluster ` whenever ` = arg max j=1,...,k v j i where v j ∈ R n is the eigenvector of the Kernel matrix associated with the j-th largest eigenvalue. How does this mapping perform for this dataset?. Explain your insights.

Solution: According to me this mapping performs quite well as compared to random initialization but this approach is not as good as running the lloyd algorithm using the normalized eigen vector approach as the data.

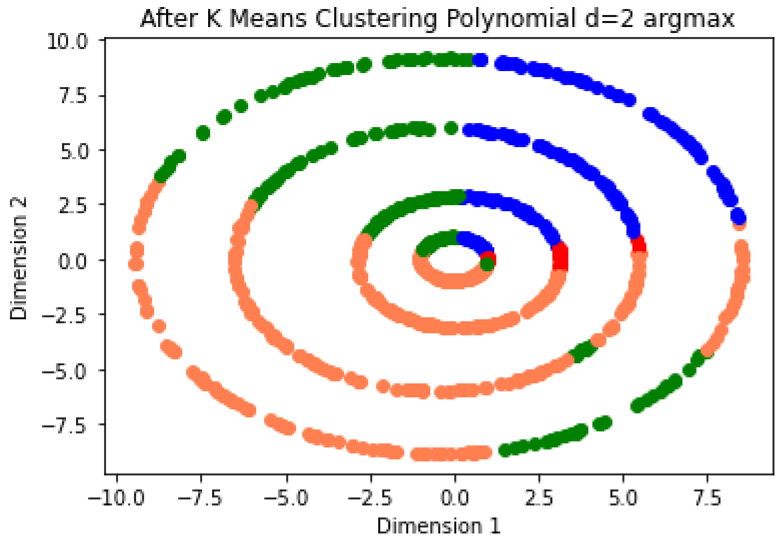
The Reason this approach is good is that, the H matrix i.e 1000*k matrix we get after taking the top k eigen vectors of the Kernel Matrix actually contains the importance of particular data point for a particular cluster.

If a data point has highest value corresponding H[n][k], then this data point n is most likely to be in kth cluster.

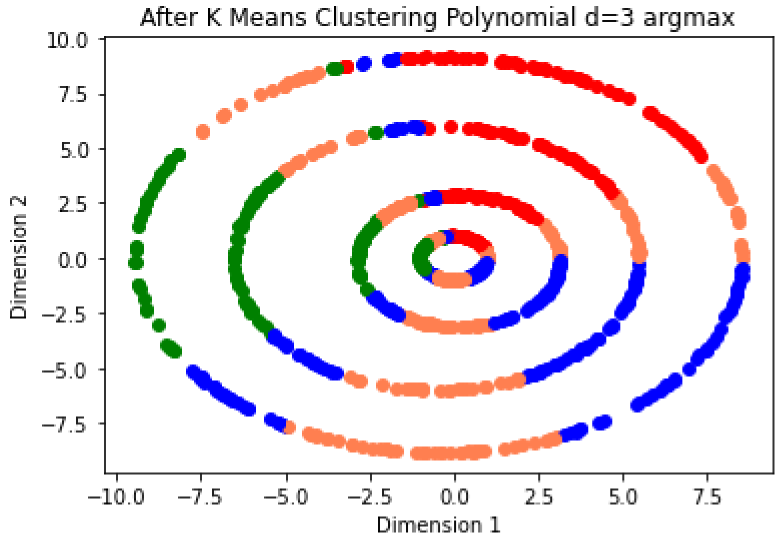
Here are some screenshots of of the argmax approach:



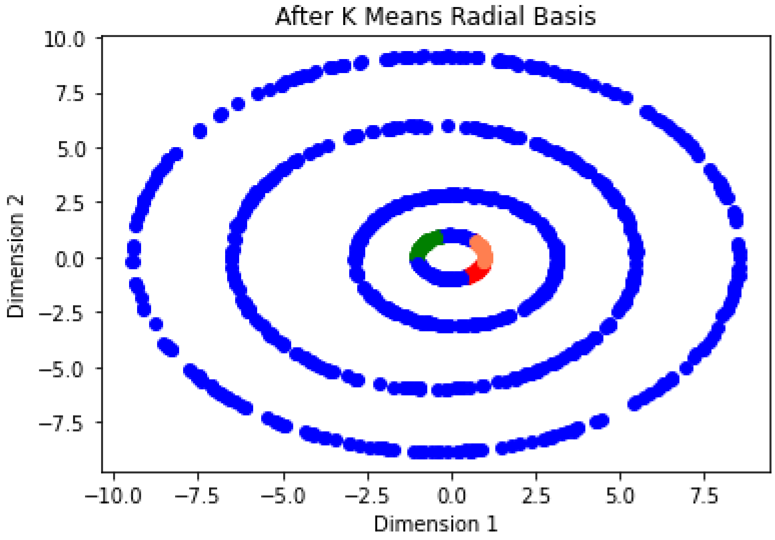
For Polynomial Kernel d=2



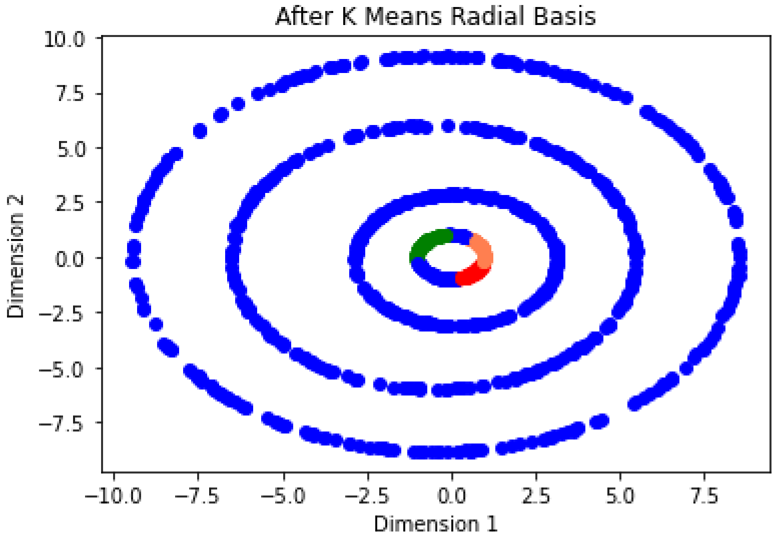
For Polynomial Kernel d=3



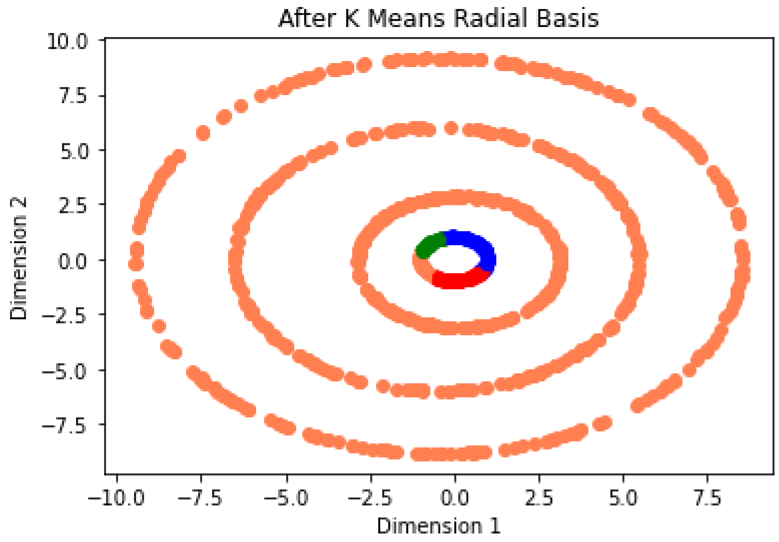
For Radial Basis:
sigma =0.1



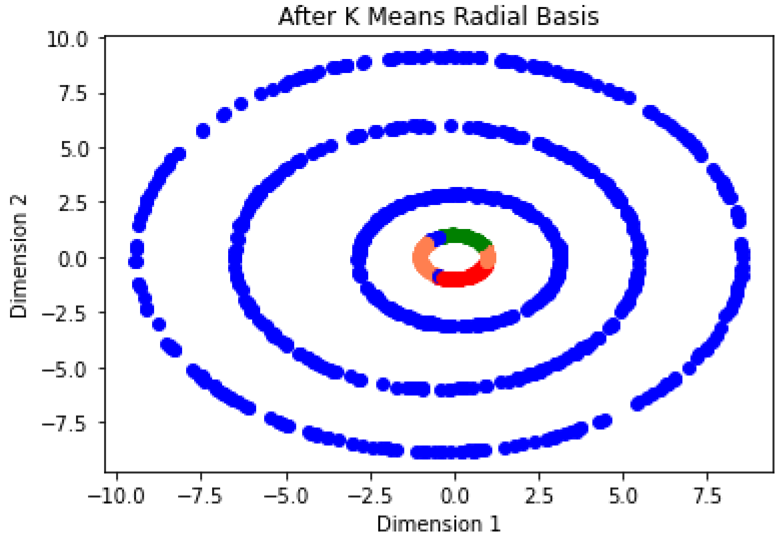
sigma =0.2



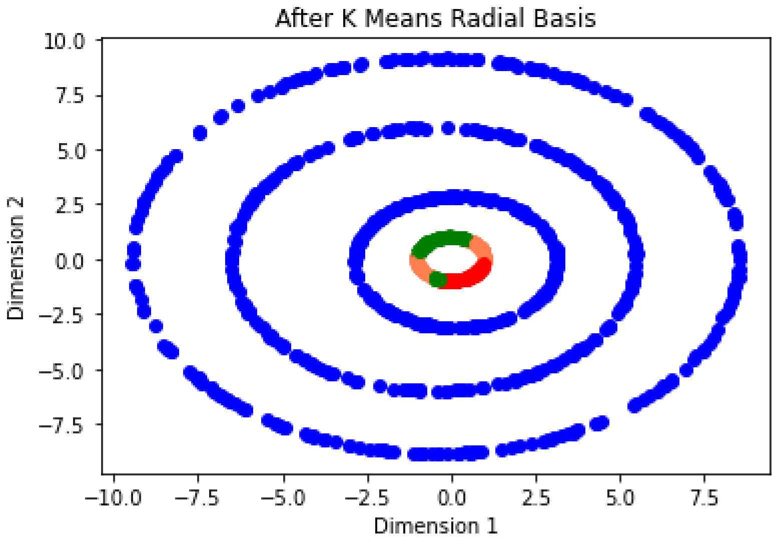
sigma=0.3



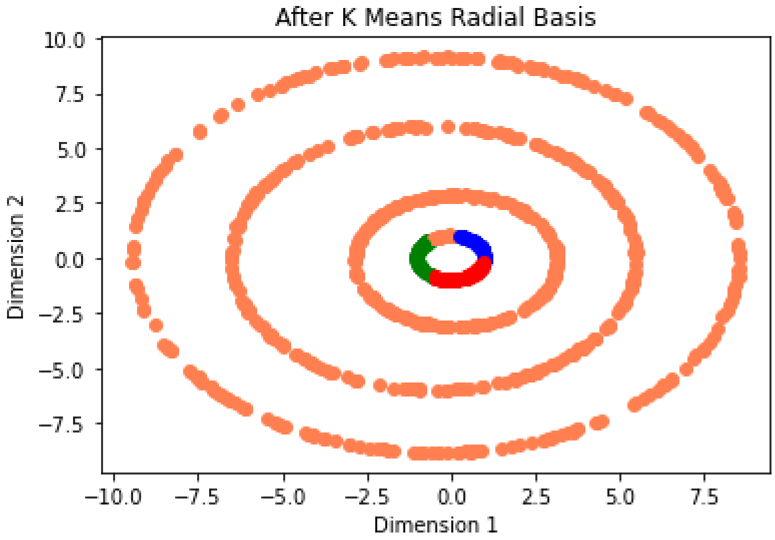
sigma =0.4



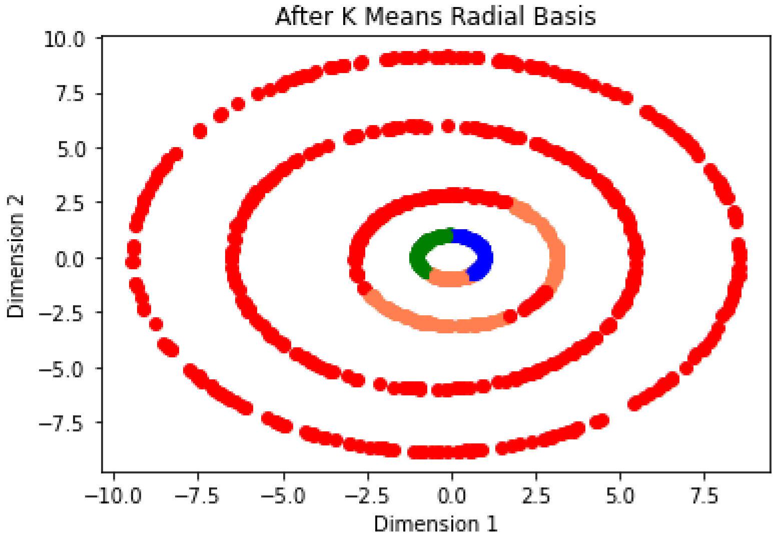
sigma=0.5



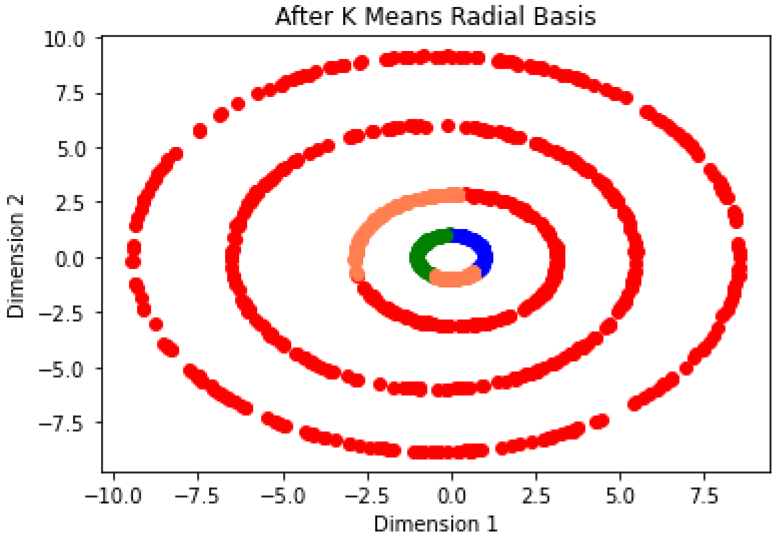
sigma=0.6



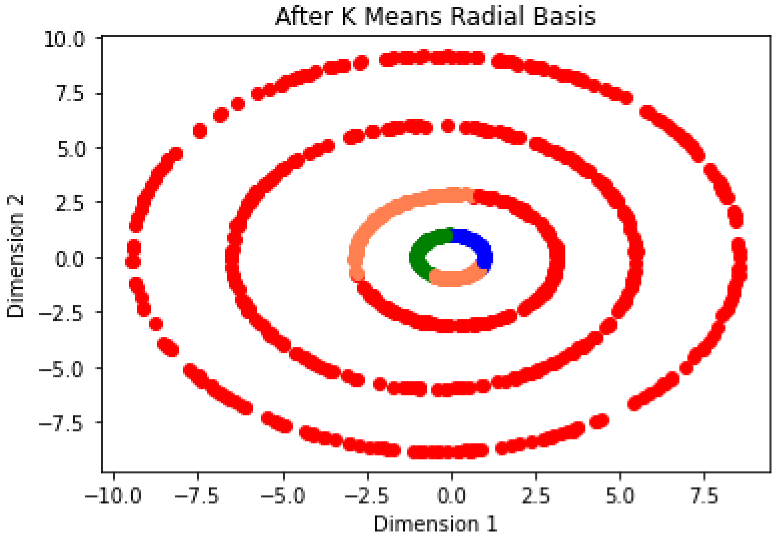
sigma=0.7



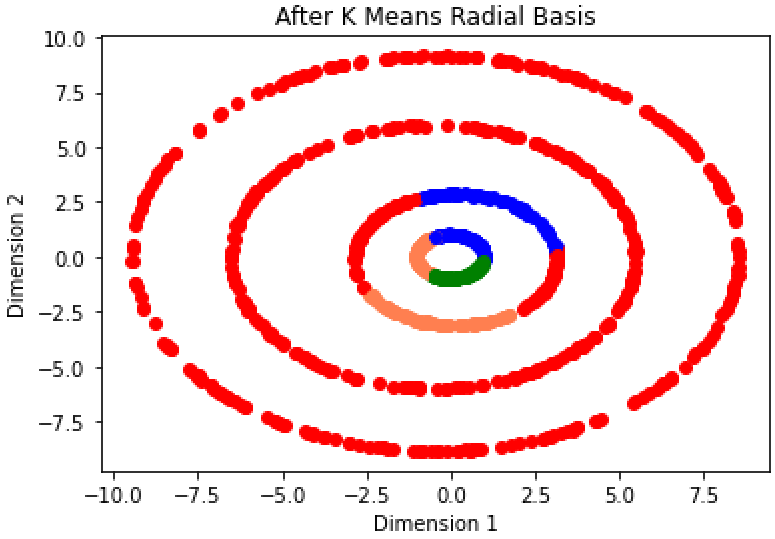
sigma=0.8



sigma=0.9


After K Means Radial Basis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1

sigma=1


After K Means Radial Basis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1