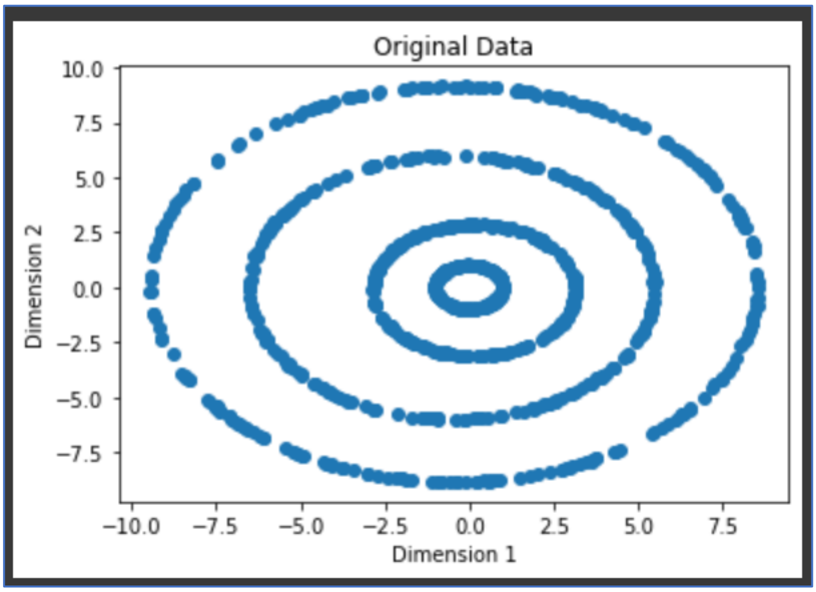
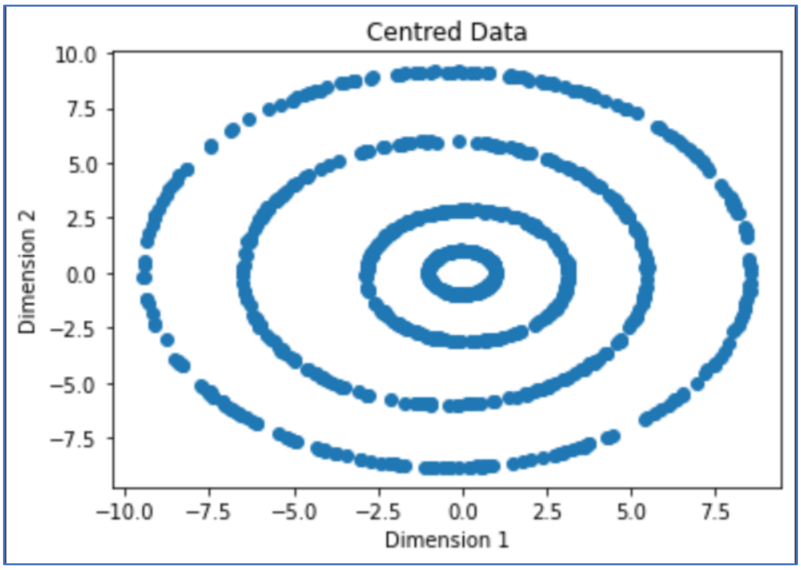
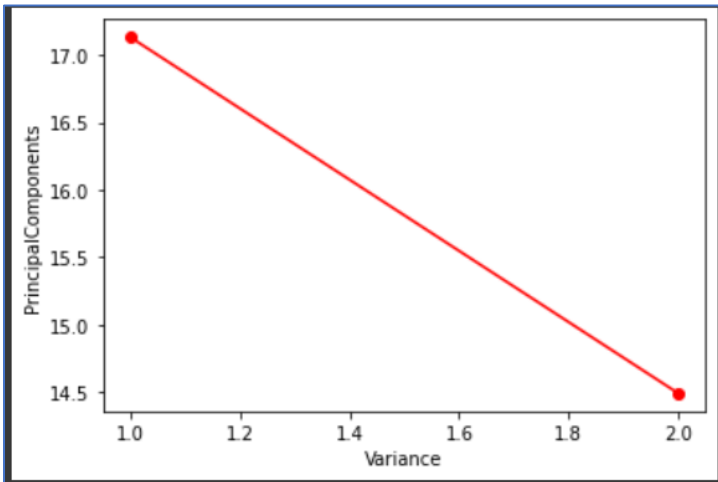
Question (1)

i) PCA after centring on dataset

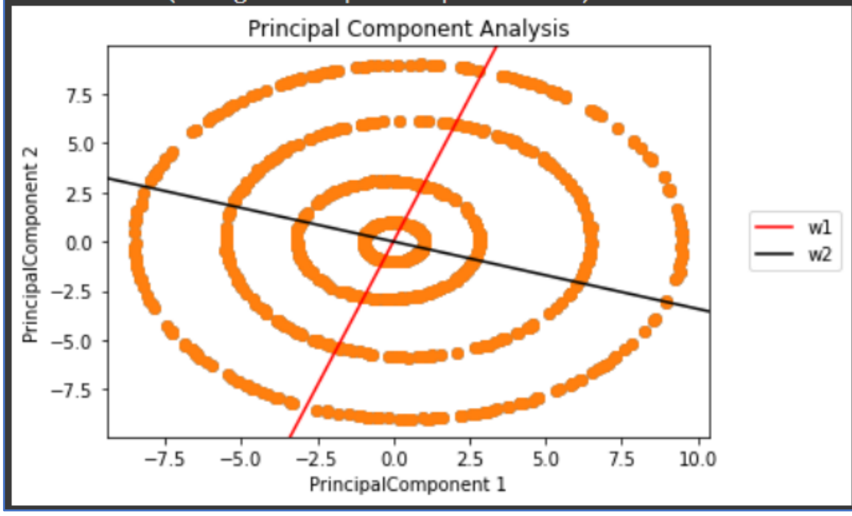
Source code file name: - **first_part_i.py**





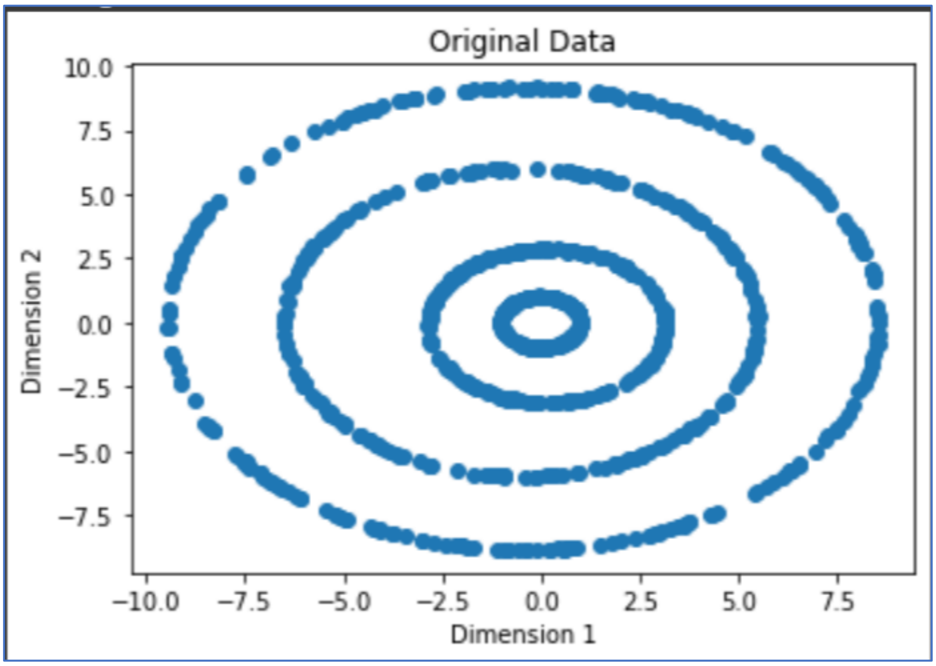


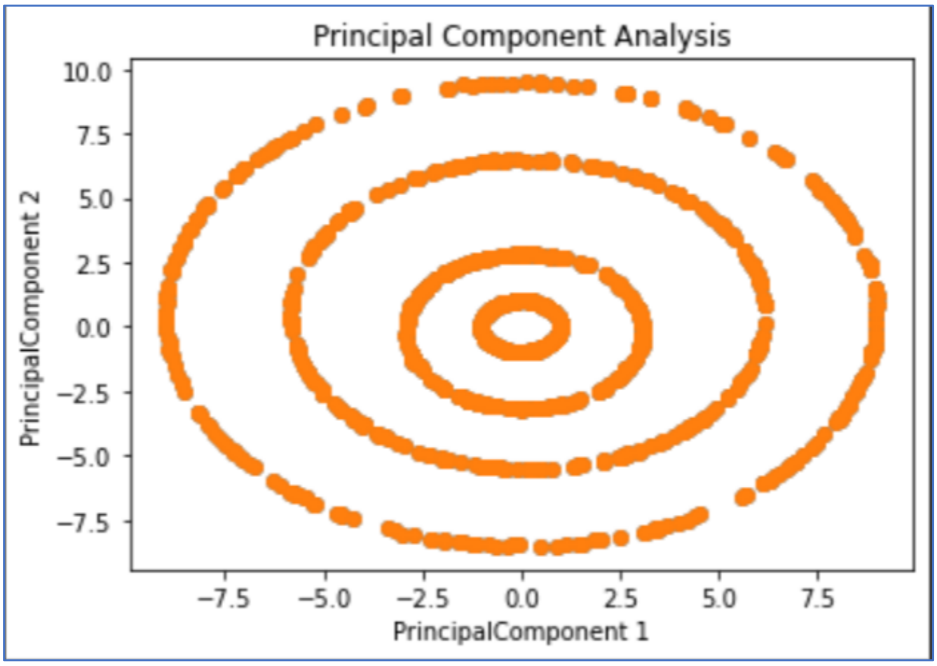

```
Variance 1 (along Principal Component  1 ) = 54.17802452885222 %
Variance 2 (along Principal Component  2 ) = 45.82197547114778 %
```



ii) PCA without centring on dataset

Source code file name: - **first_part_ii.py**





Observations: Results obtained are quite same as the part i of this question.

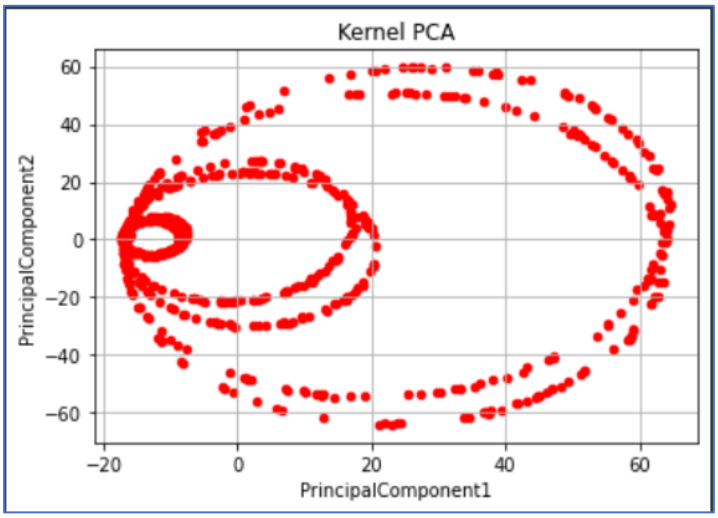
Since the mean of the data set are very small as compared to the dataset hence there is almost no effect of centring the dataset in our case.

iii) Kernel PCA on dataset

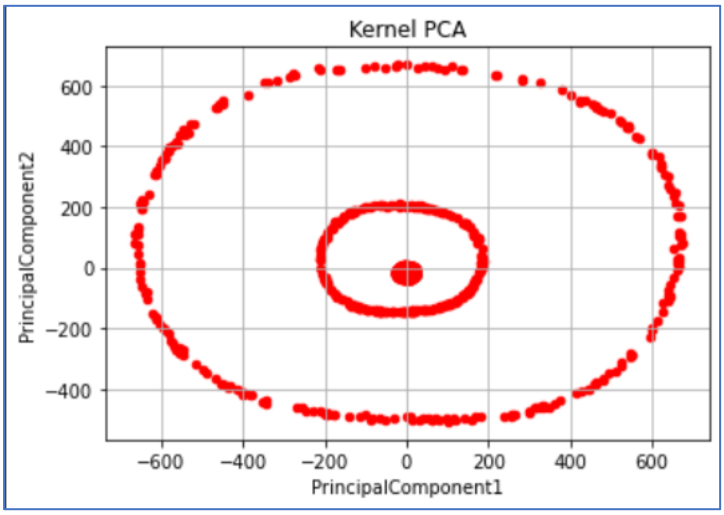
A. $\kappa(x, y) = (1 + x^T y)^d$ for $d = \{2, 3\}$

Source code file name: - **first_part_iii.py**

d=2



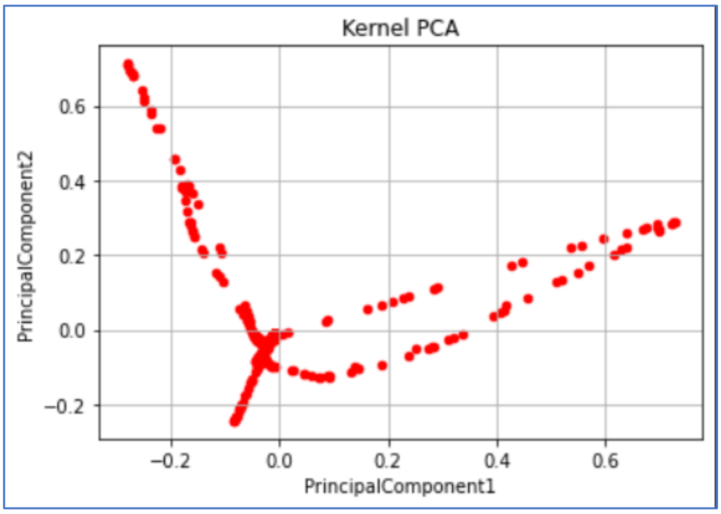
d=3



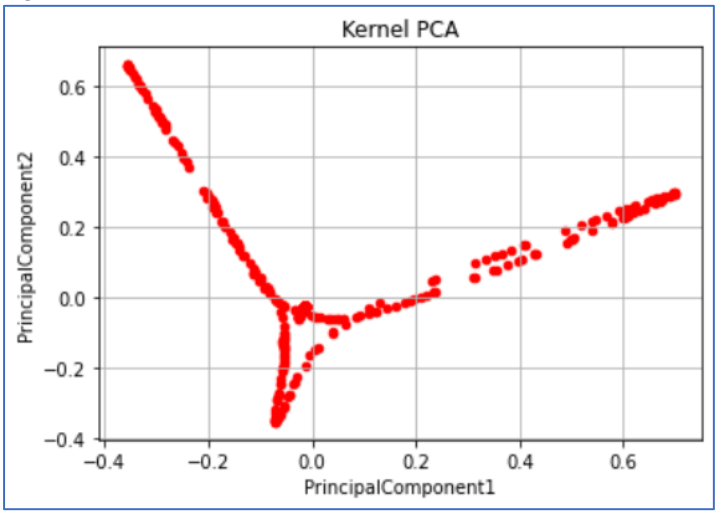
B. $\kappa(x, y) = \exp \frac{-(x-y)^T (x-y)}{2\sigma^2}$ for $\sigma = \{0.1, 0.2, \ldots, 1\}$

Source code file name: - **first_part_iii.py**

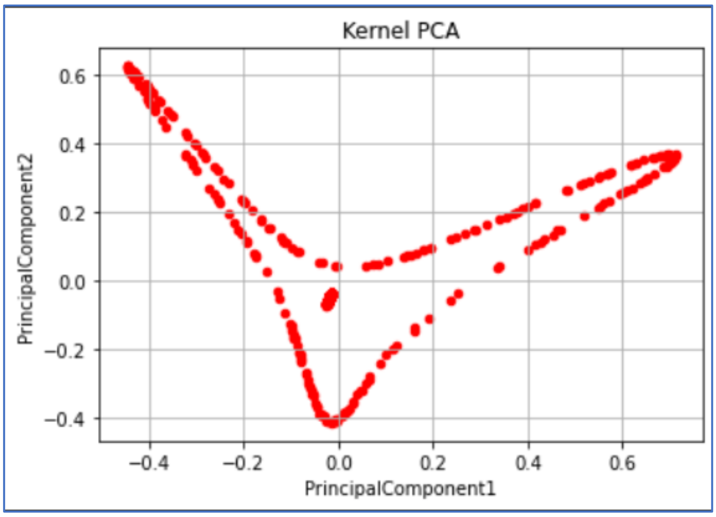
Sigma=0.1



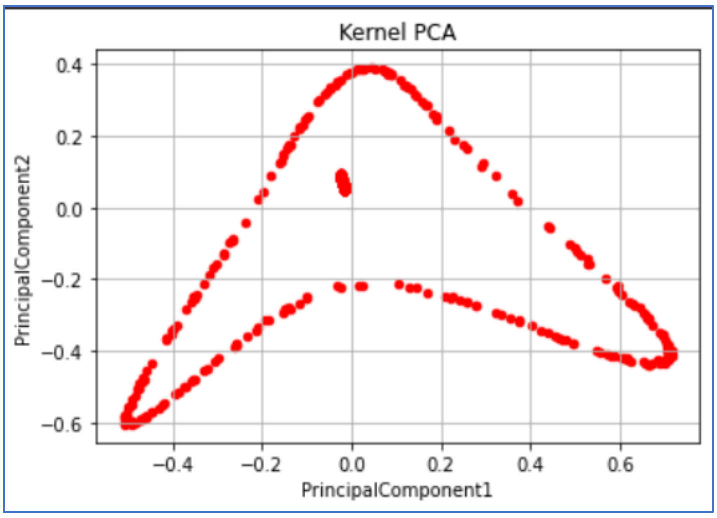
Sigma=0.2



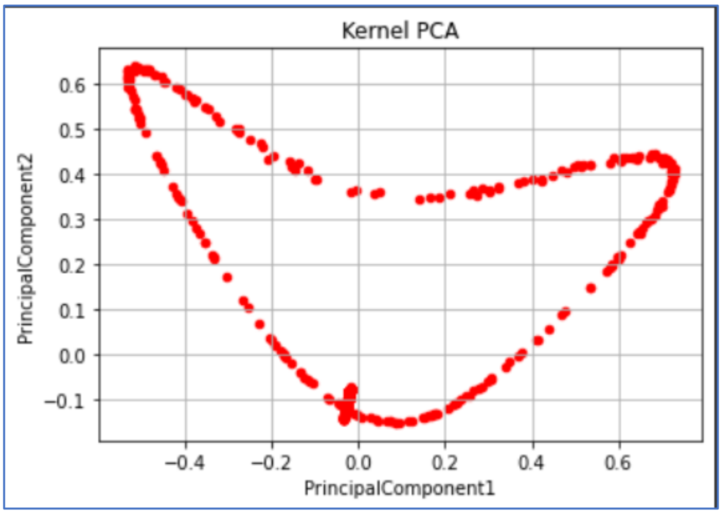
Sigma=0.3



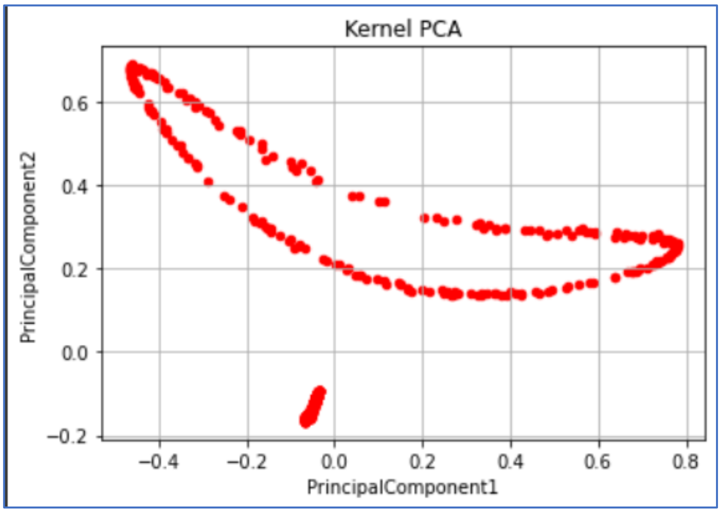
Sigma=0.4



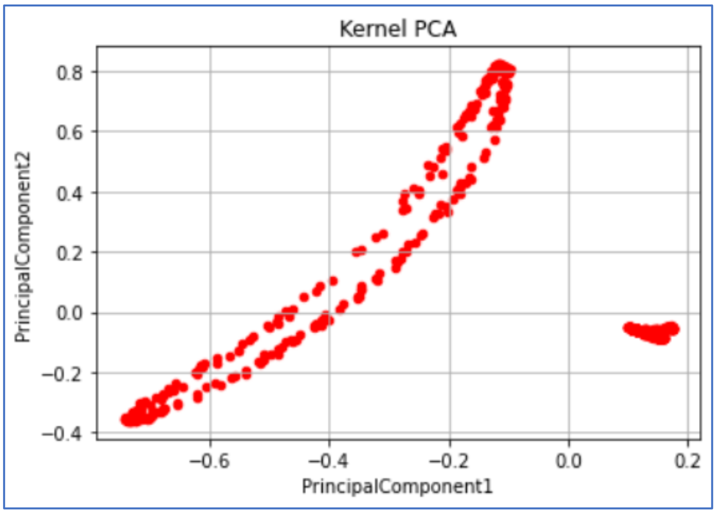
Sigma=0.5



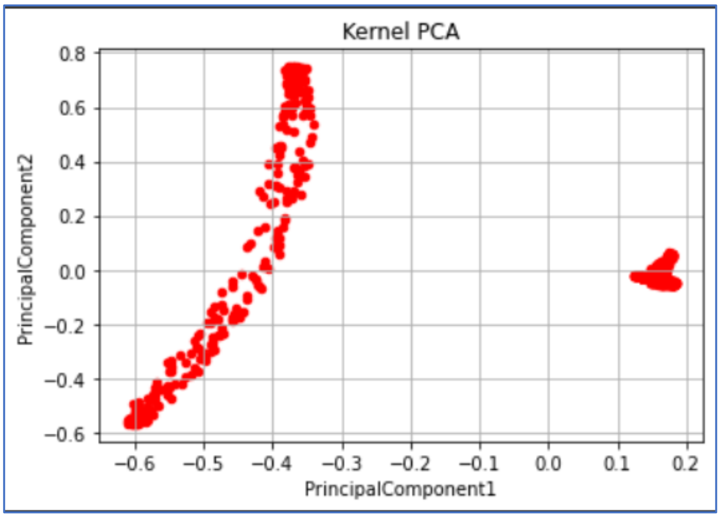
Sigma=0.6



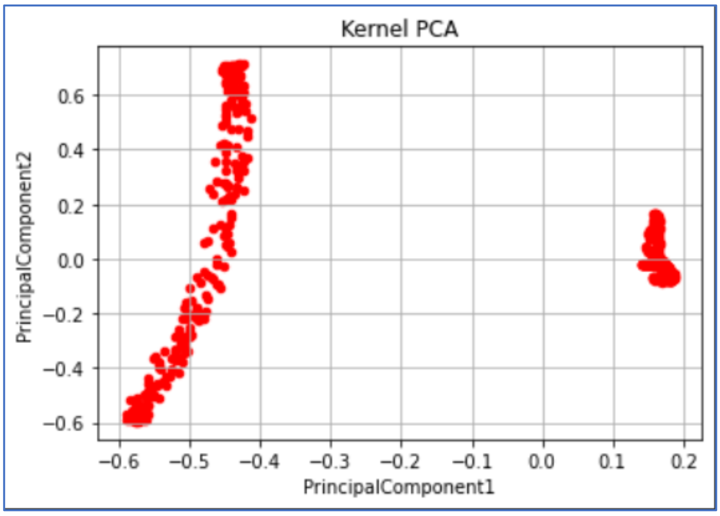
Sigma=0.7



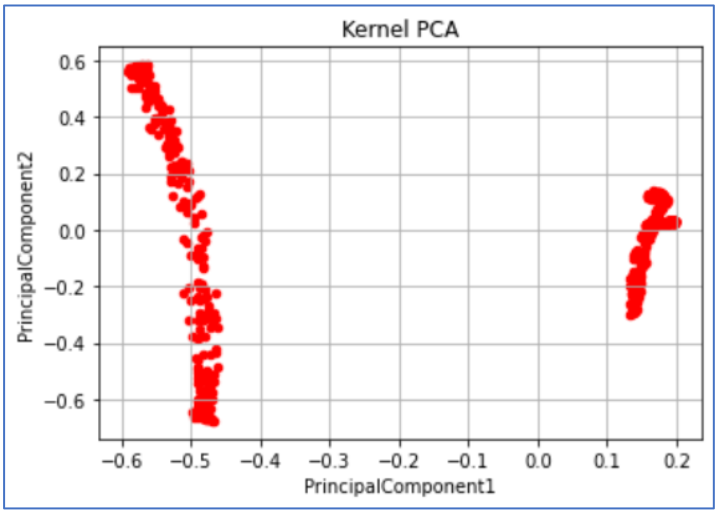
Sigma=0.8



Sigma=0.9



Sigma=1.0

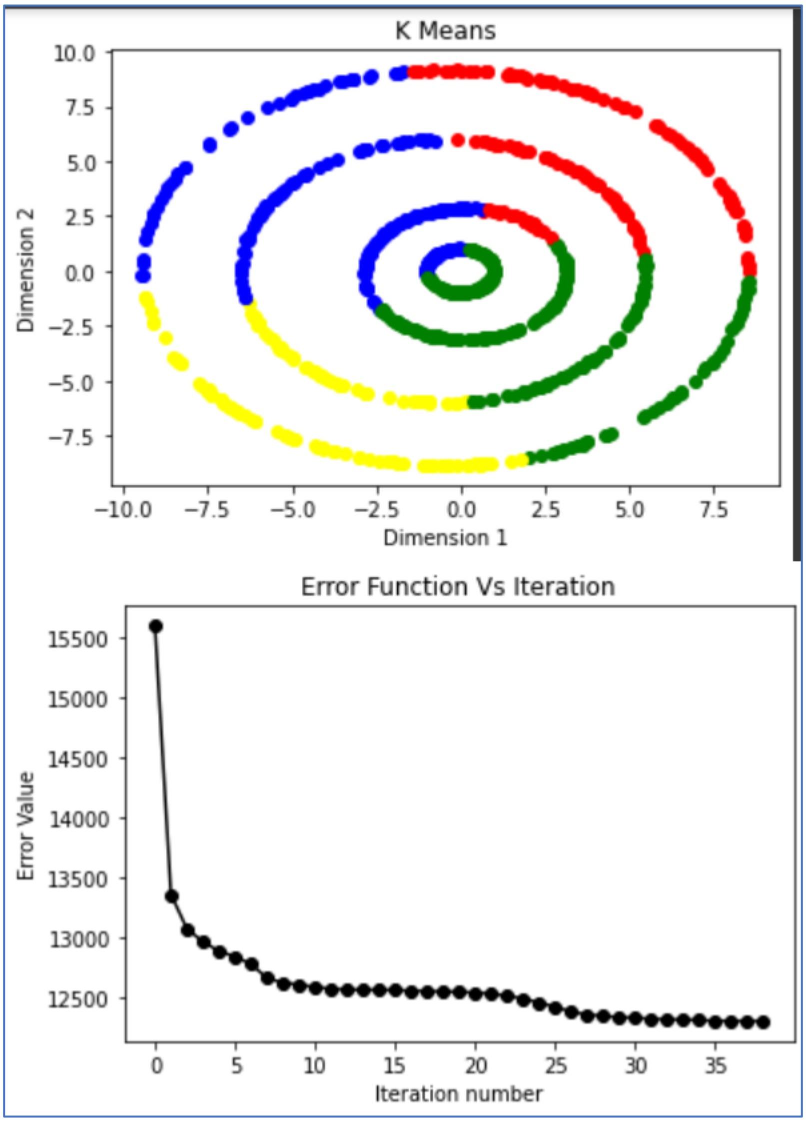


iv) By Observation from the above plots of kernel, degree 3 polynomial kernel is best suited for this dataset. Its variance in the top two components of the kernel is maximum which is approx. 73%. Other variance for degree 2 poly kernel is approx. 68% and the maximum variance for exponential kernel holds for sigma=1.0 and is approx. 15%
So, degree 3 polynomial kernel should be chosen for plotting.

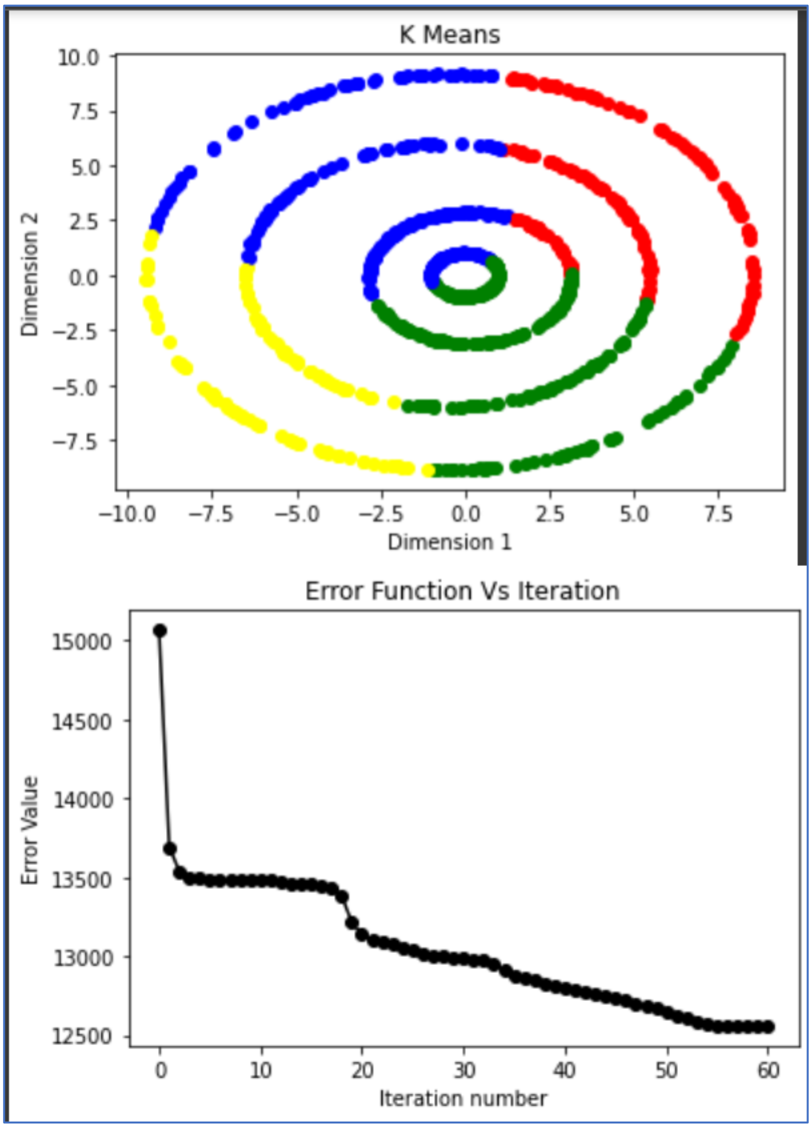
Question (2)

i) Source code file name: - **second_part_i.py**

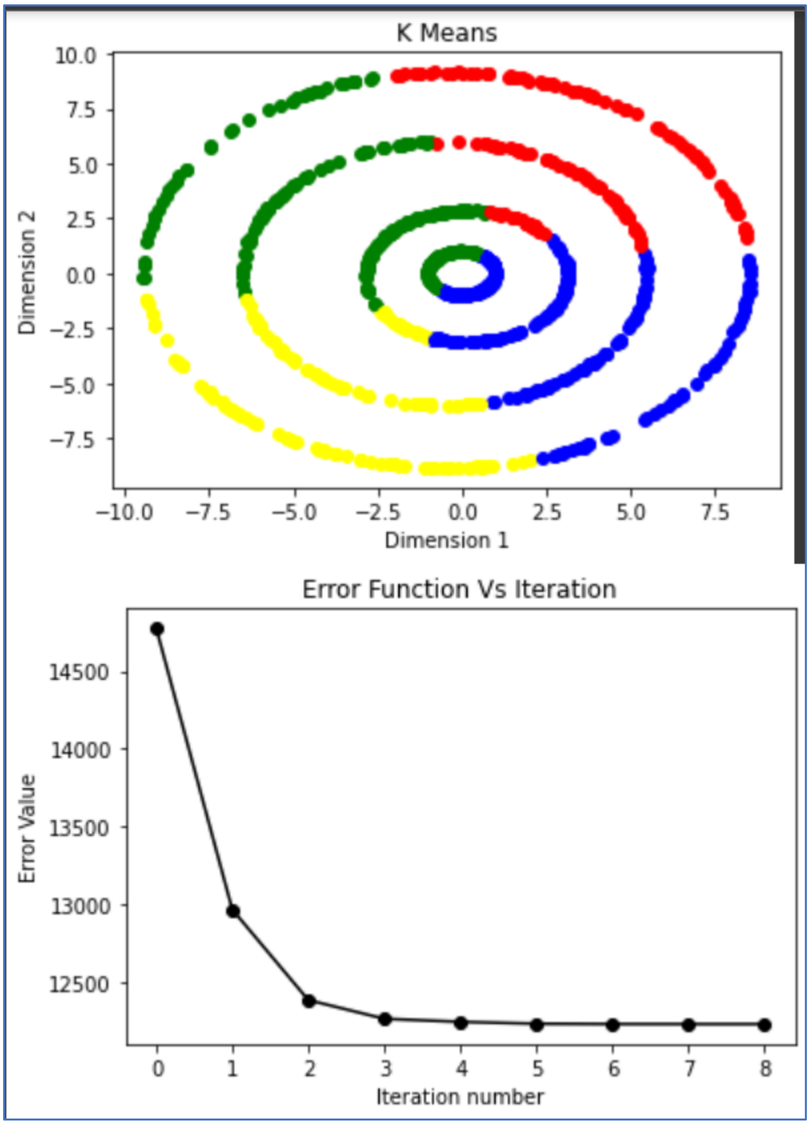
Random Initialization 1:



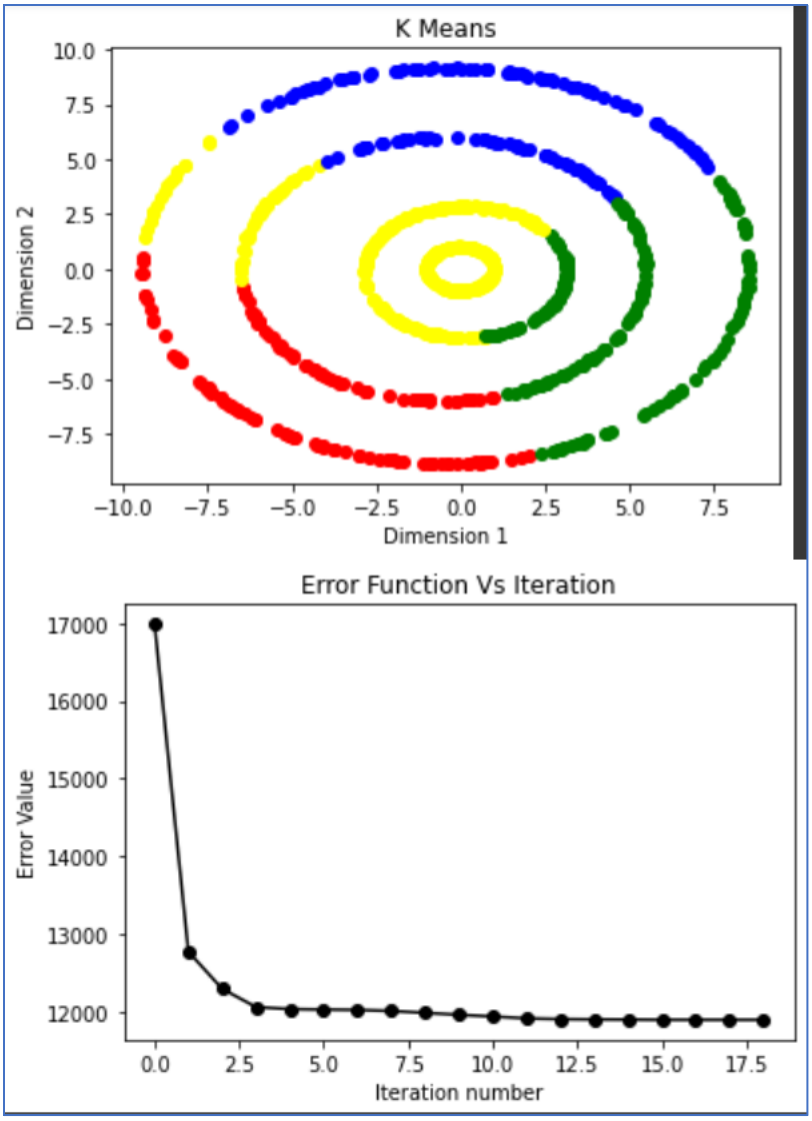
Random Initialization 2:



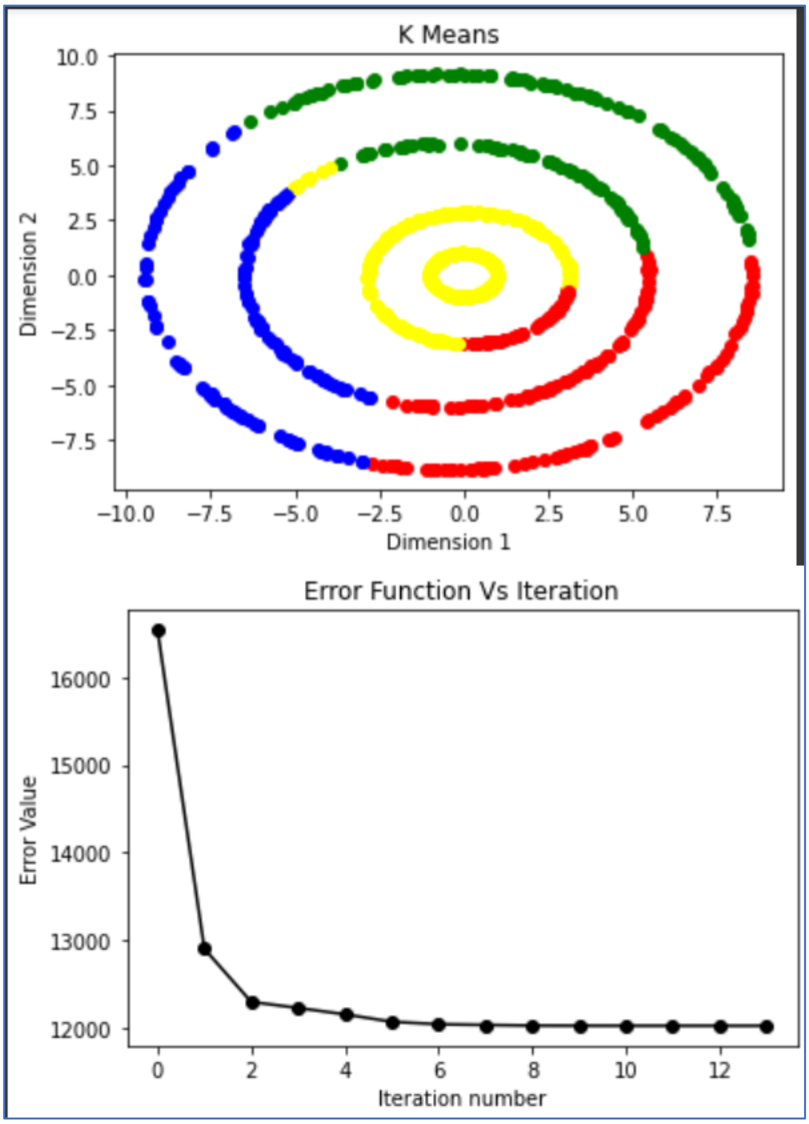
Random Initialization 3:



Random Initialization 4:

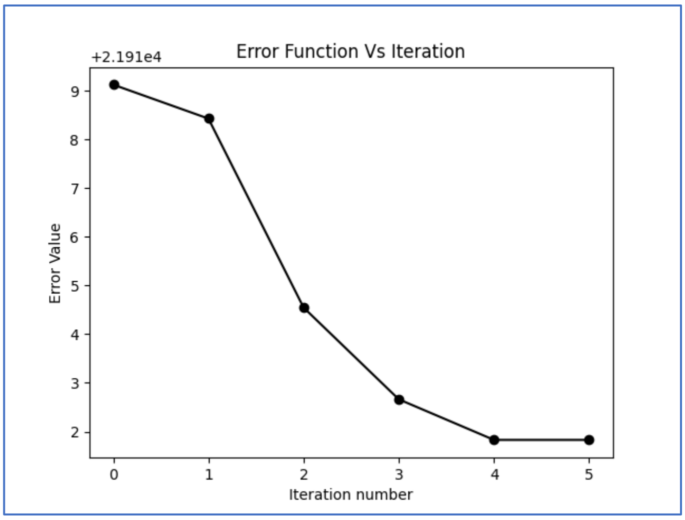


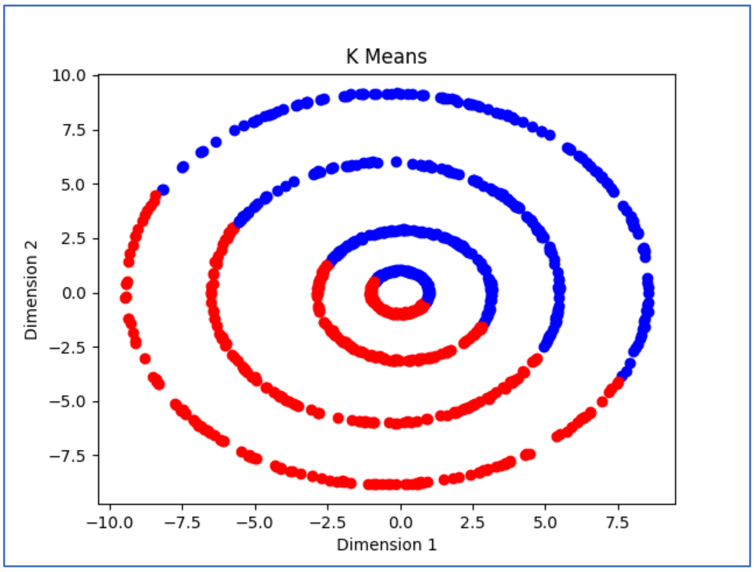
Random Initialization 5:

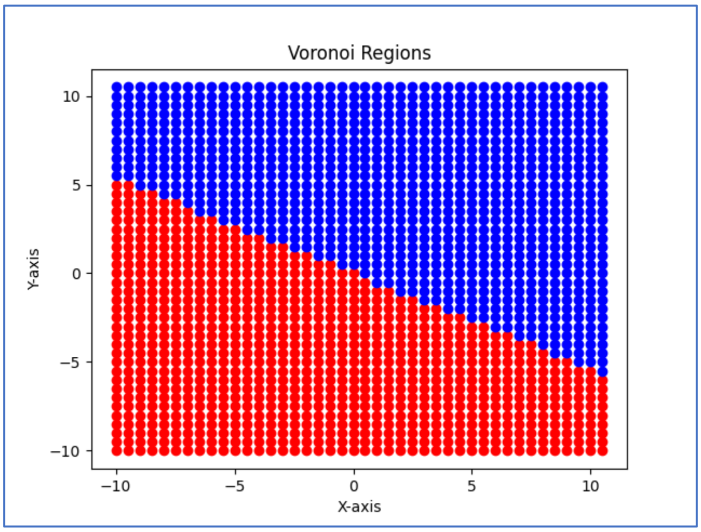


ii) Source code file name: - **second_part_ii.py**

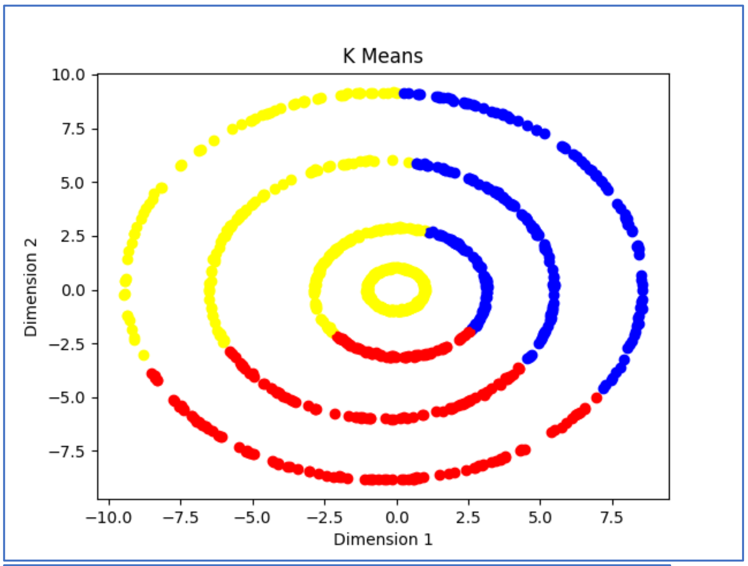
K=2

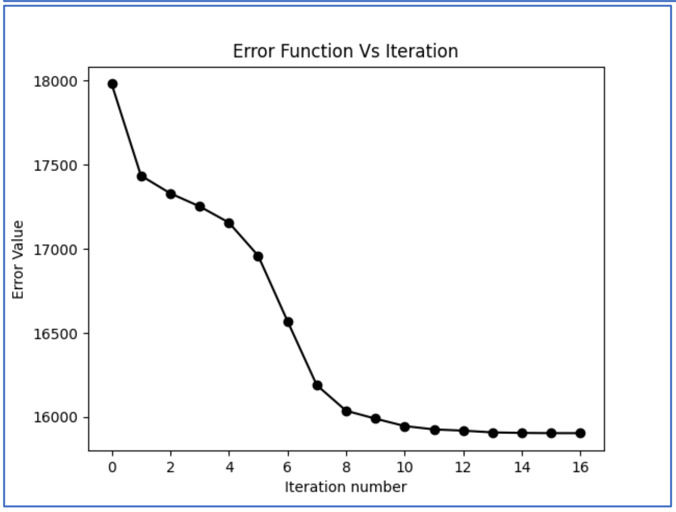


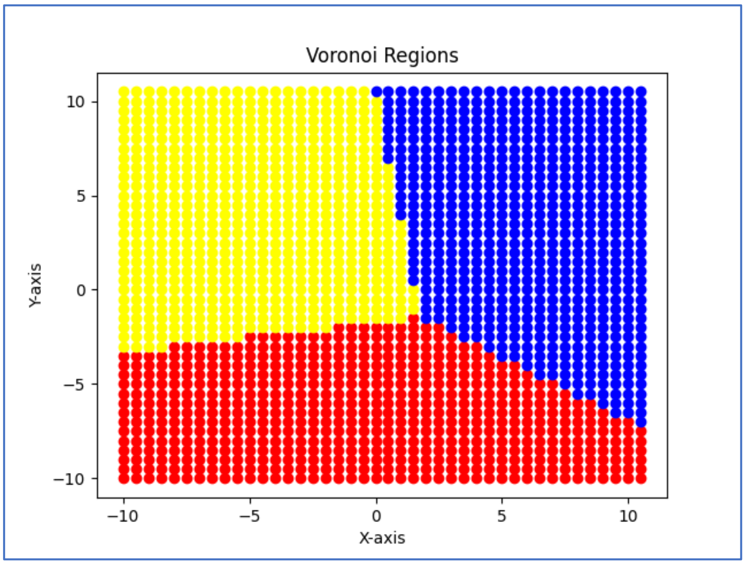




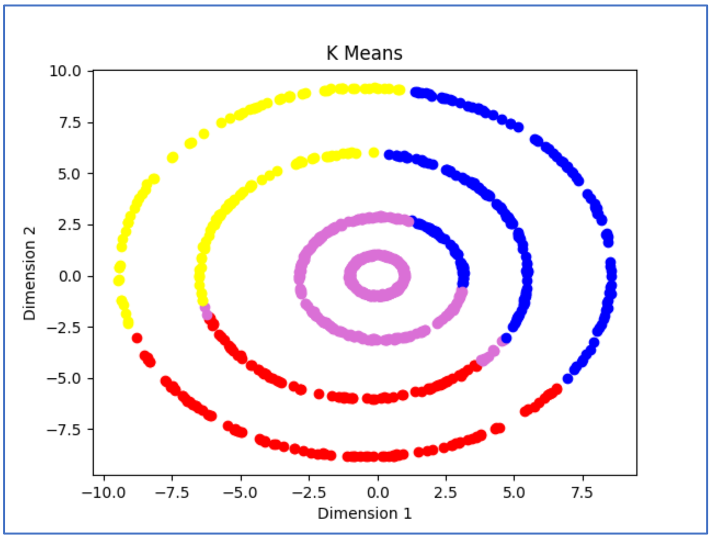
K=3

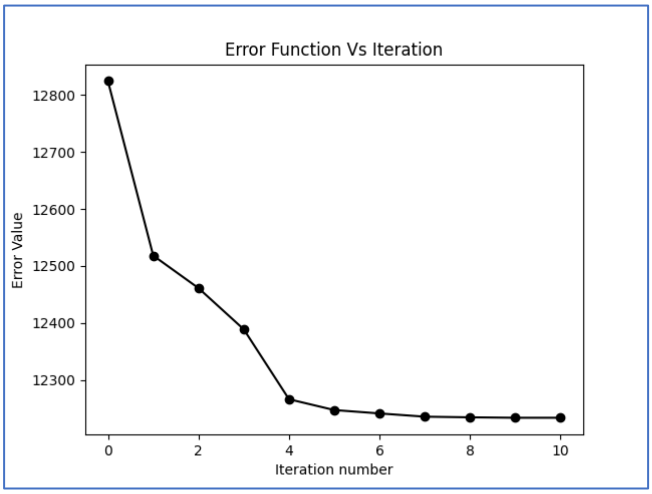

K Means
Dimension 2
Dimension 1
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5

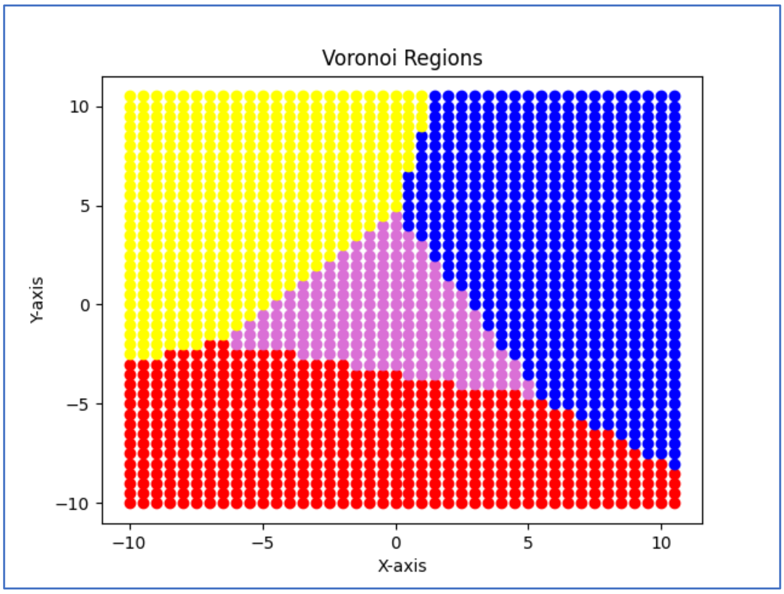

Error Function Vs Iteration
Error Value
Iteration number
18000
17500
17000
16500
16000
0
2
4
6
8
10
12
14
16


Voronoi Regions
Y-axis
X-axis
10
5
0
−5
−10
−10
−5
0
5
10

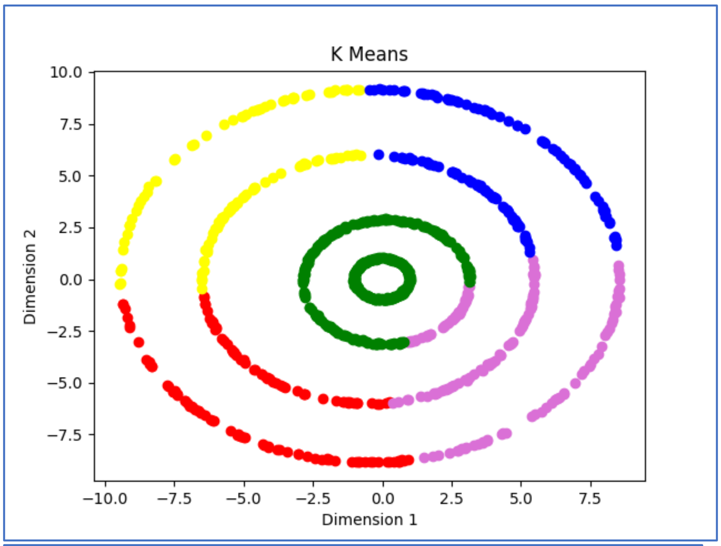
K=4

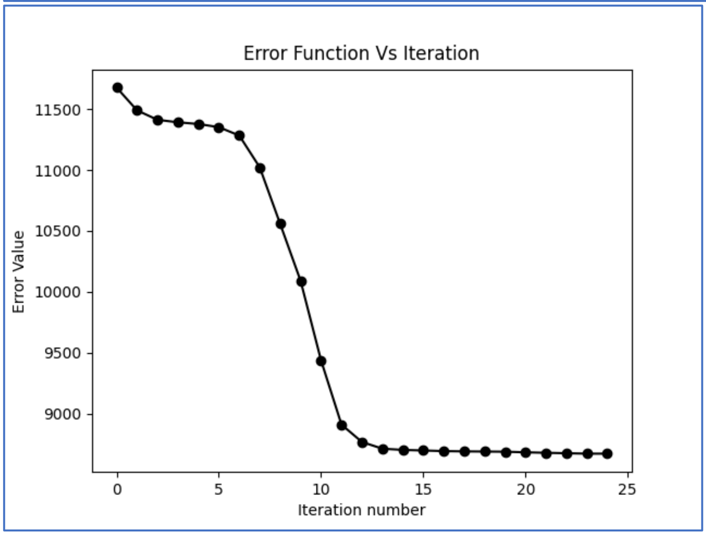

K Means
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
Dimension 2
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
Dimension 1

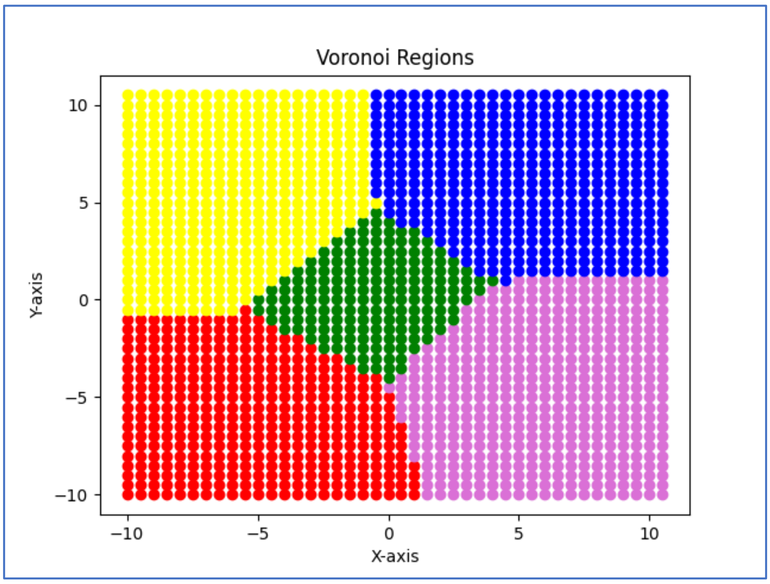

Error Function Vs Iteration
12800
12700
12600
12500
12400
12300
Error Value
0
2
4
6
8
10
Iteration number


Voronoi Regions
10
5
0
−5
−10
Y-axis
−10
−5
0
5
10
X-axis

K=5


K Means
Dimension 1
Dimension 2

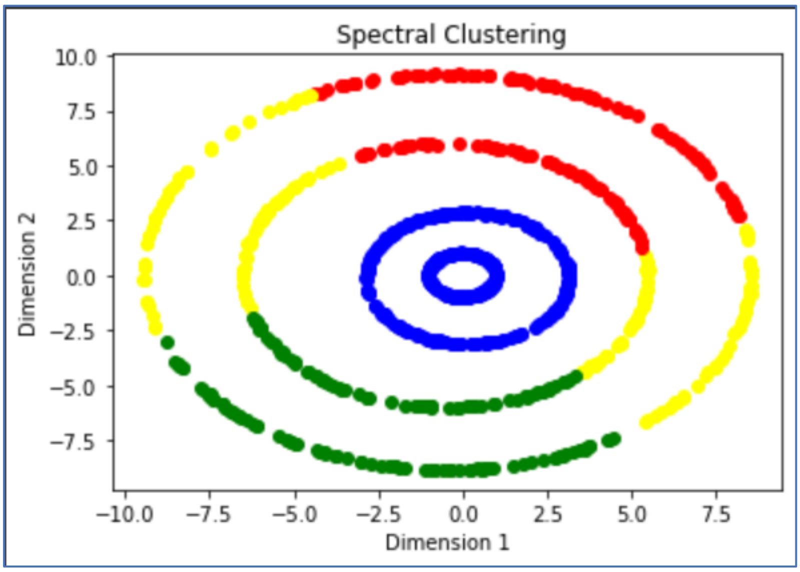

Error Function Vs Iteration
Iteration number
Error Value


Voronoi Regions
X-axis
Y-axis

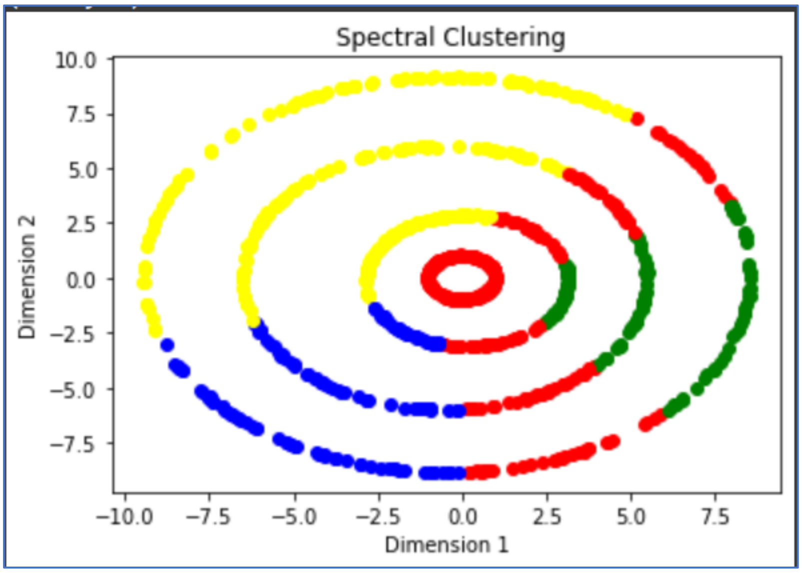
iii) Source code file name: - **second_part_iii.py**

Polynomial Kernel with degree 2 is the closest to the expected output which is rings clustered from outwards to inwards.

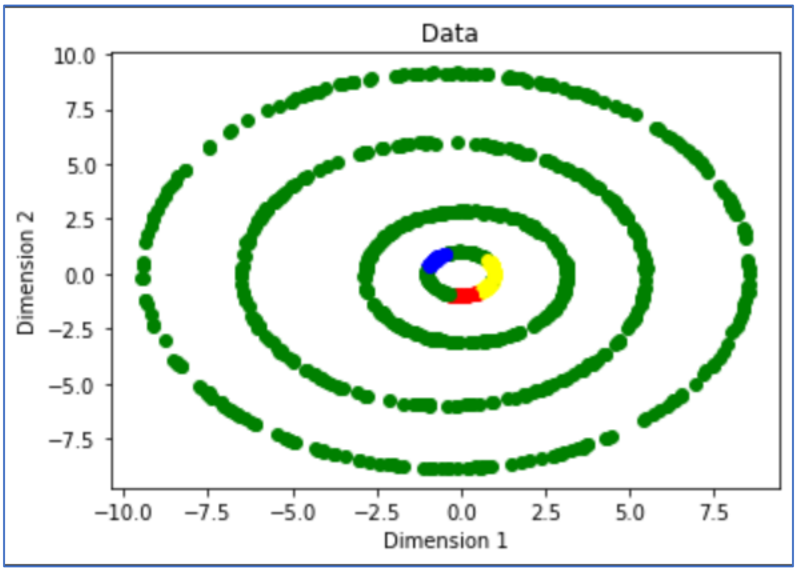
Kernel degree 2 polynomial from Q1



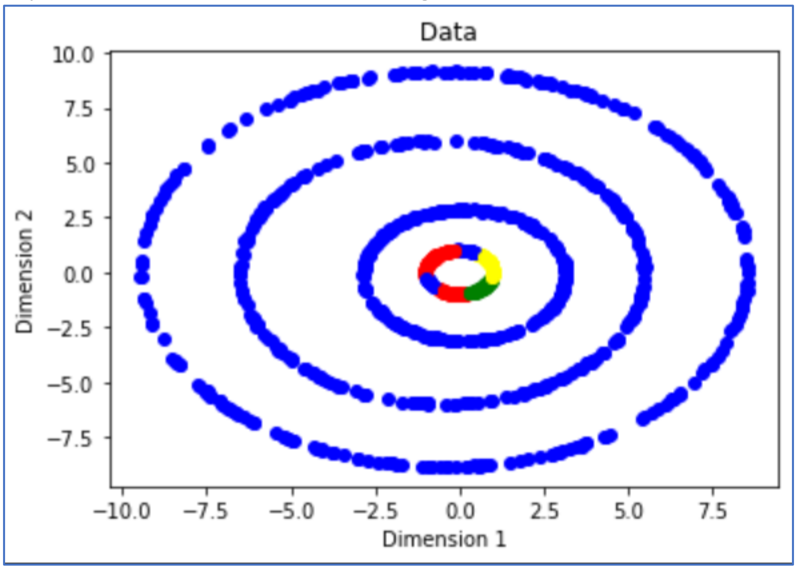
Kernel degree 3 polynomial from Q1



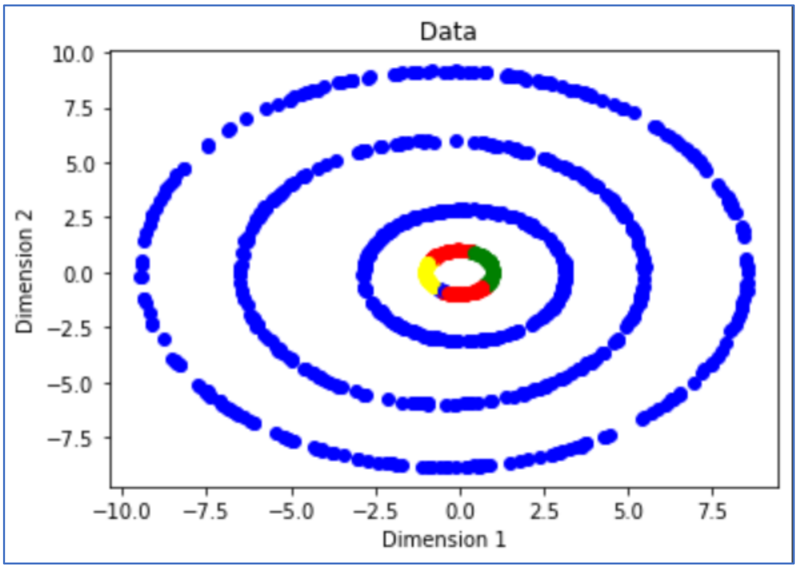
Exponential Kernel function with Sigma=0.1 from Q1



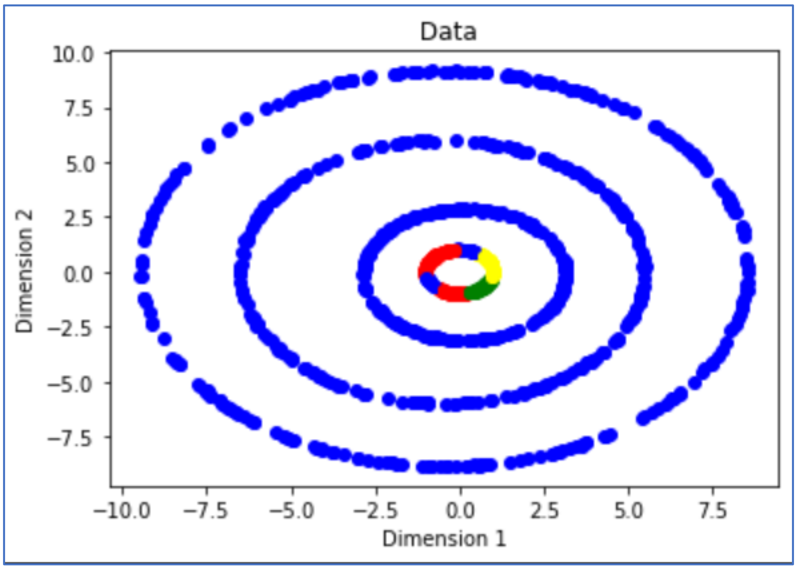
Exponential Kernel function with Sigma=0.2 from Q1



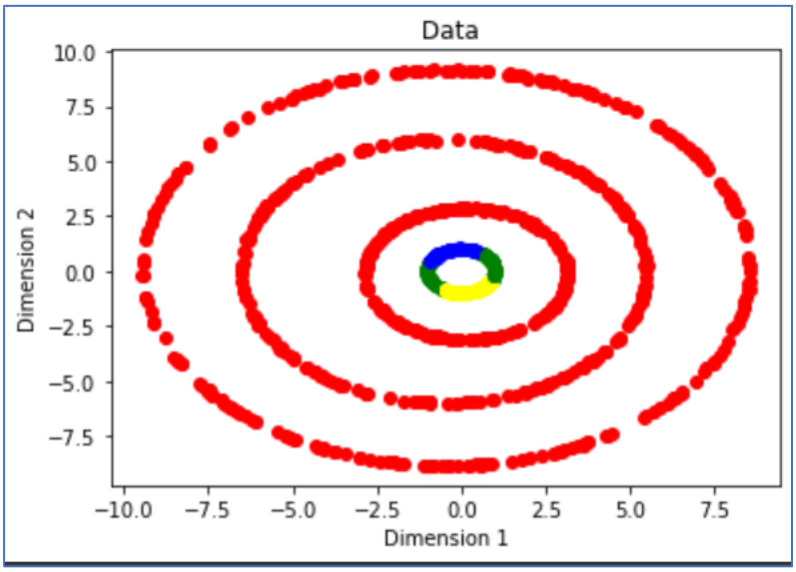
Exponential Kernel function with Sigma=0.3 from Q1



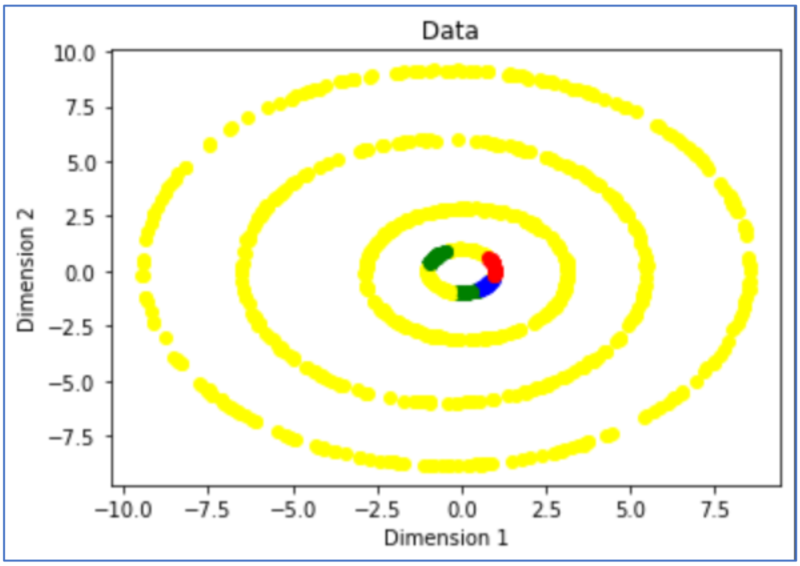
Exponential Kernel function with Sigma=0.4 from Q1



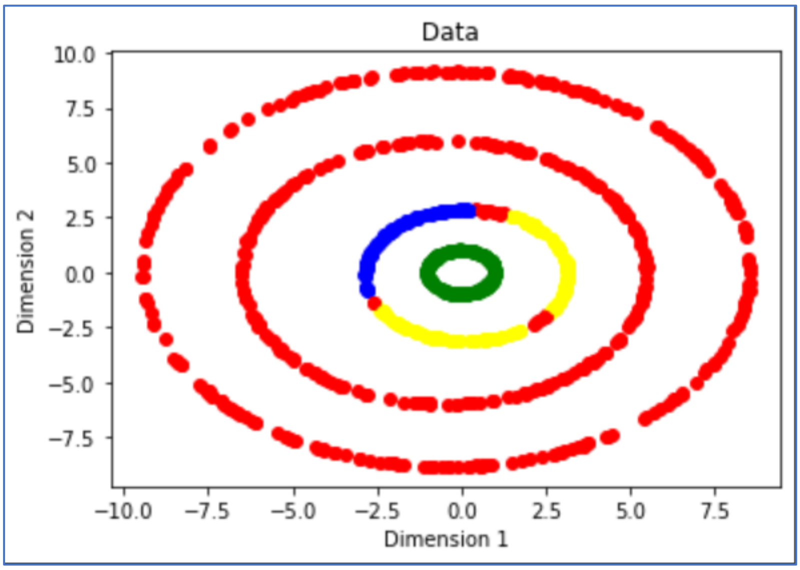
Exponential Kernel function with Sigma=0.5 from Q1



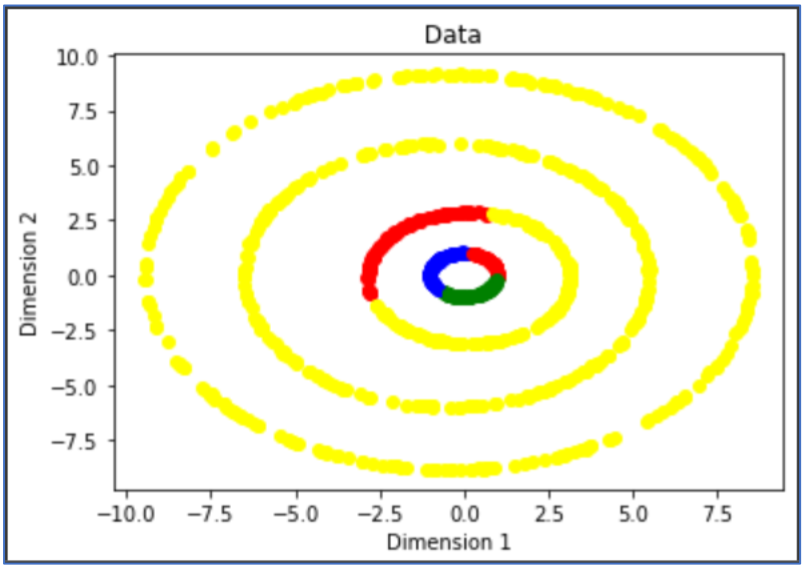
Exponential Kernel function with Sigma=0.6 from Q1



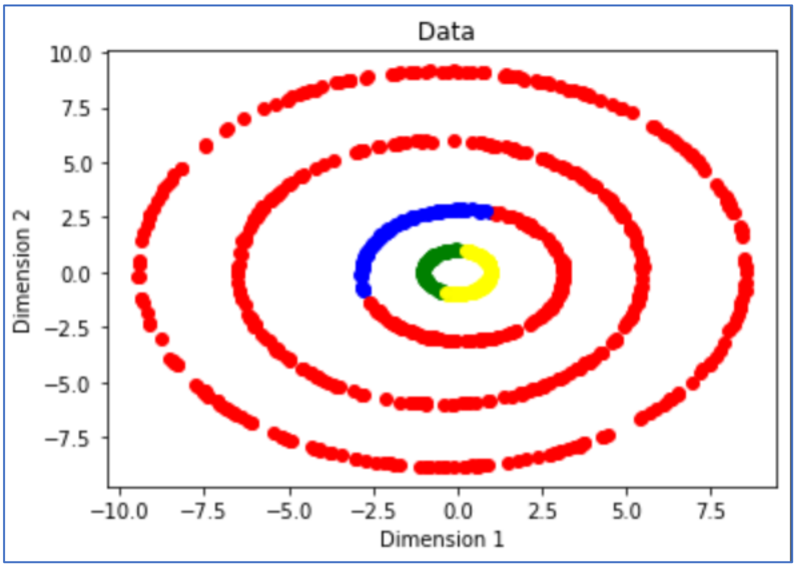
Exponential Kernel function with Sigma=0.7 from Q1



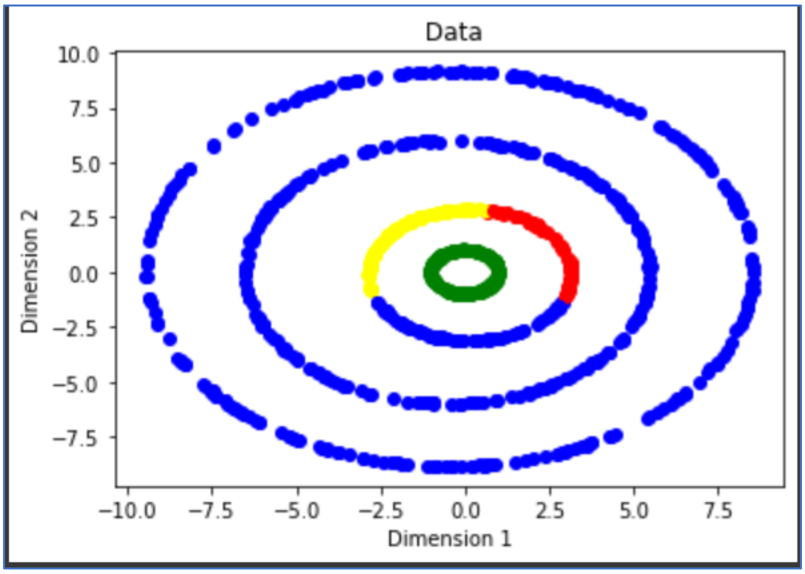
Exponential Kernel function with Sigma=0.8 from Q1



Exponential Kernel function with Sigma=0.9 from Q1



Exponential Kernel function with Sigma=1.0 from Q1



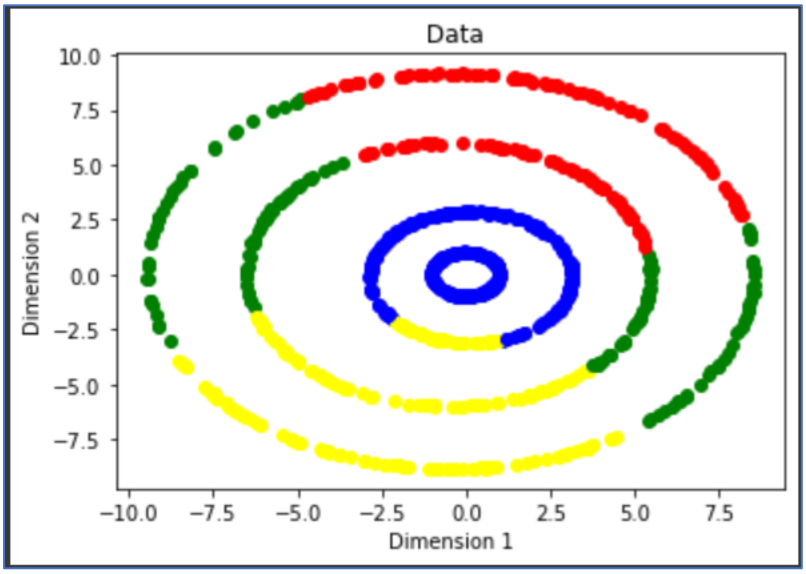
iv) Source code file name: - **second_part_iii.py**

Using arg max function to map eigen vectors to clusters

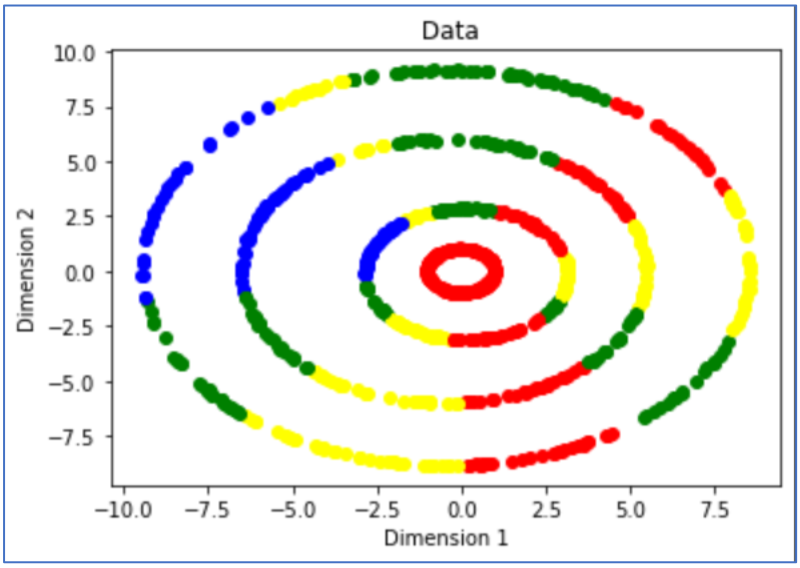
$$\ell = \arg \max_{j=1,\ldots,k} v_i^j$$

Answer: The results obtained are Better than k-mean clustering which is initialized randomly. As here we are moving to higher dimension. But it is not as good as Normalized mean clustering.
H[n][k] denotes the likelihood of point n to go into the kth cluster
So, it performs better as compared to random initialization

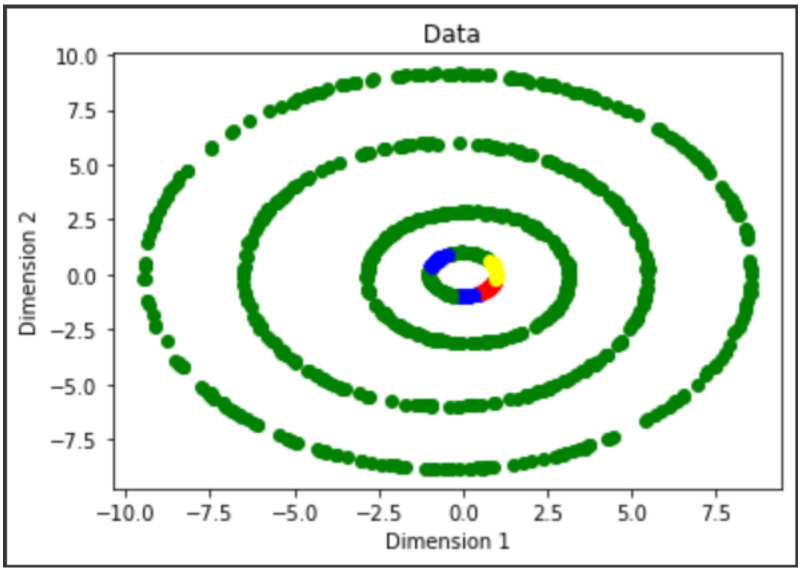
Kernel degree 2 polynomial from Q1



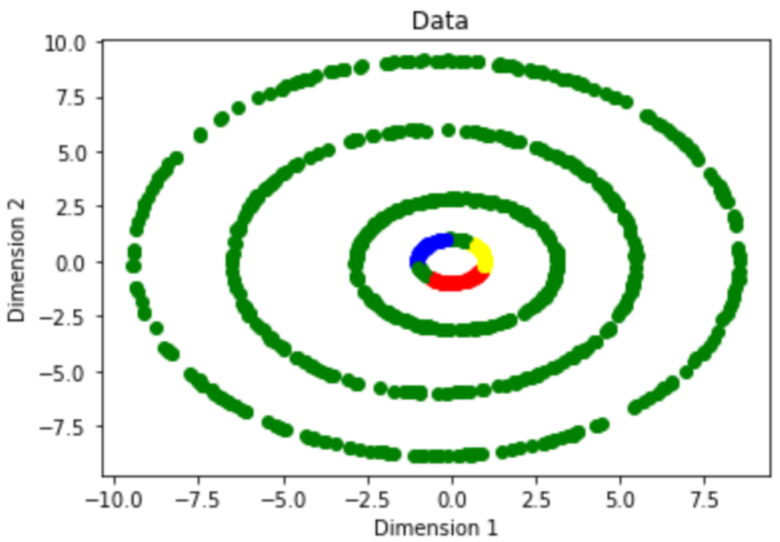
Kernel degree 3 polynomial from Q1



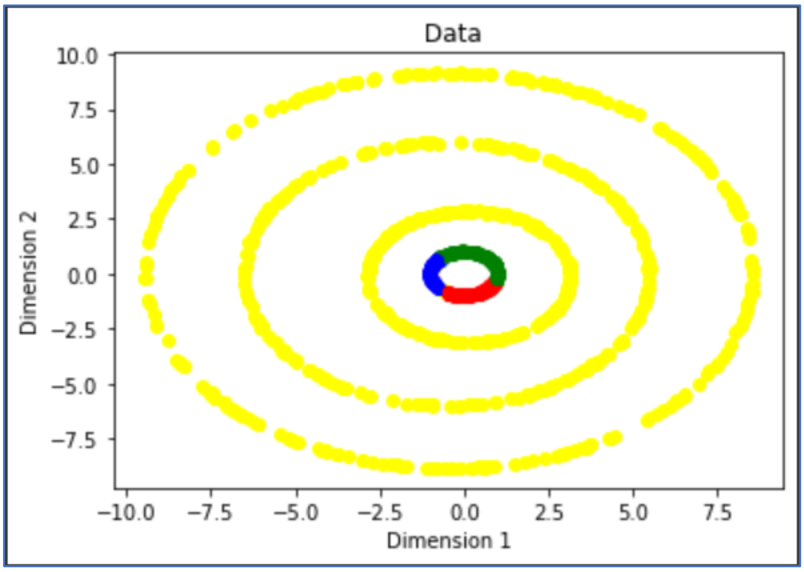
Exponential Kernel function with Sigma=0.1 from Q1



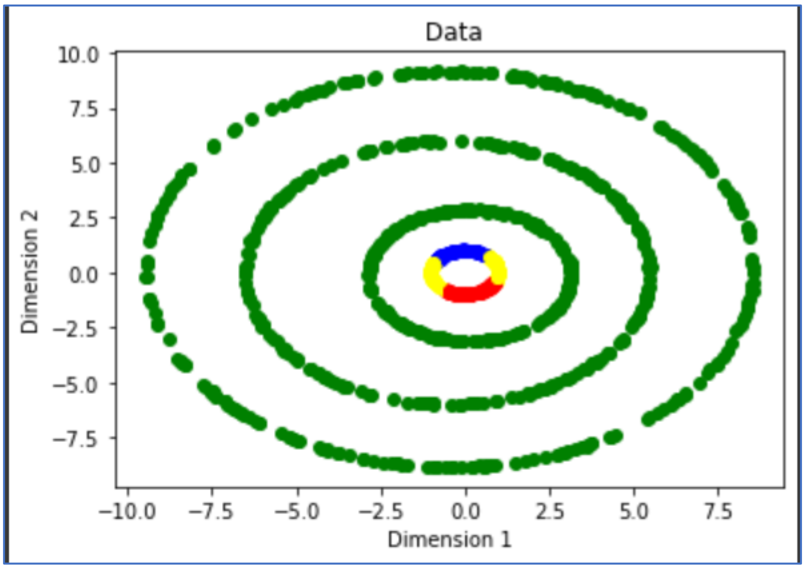
Exponential Kernel function with Sigma=0.2 from Q1



Exponential Kernel function with Sigma=0.3 from Q1



Exponential Kernel function with Sigma=0.4 from Q1


Data
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