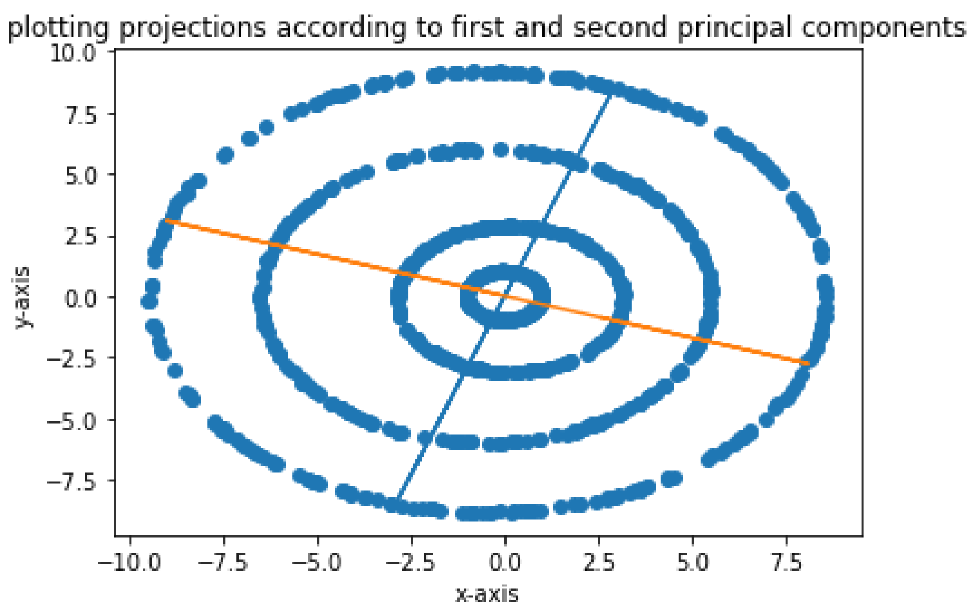
Q1:

i. code to run PCA on dataset:

Question1_(i,ii).ipynb Question1_(i,ii).py

Variance explianed by first principal component: 54.17802452885223

Variance explianed by second principal component: 45.82197547114777



ii. Observation: running PCA without centring does not effect much.
mean along x-axis: 4.0749999996858 58e-07
mean along y-axis: 2.2270000000723655e-07

So the mean is very close to zero. Therefore subtracting it form every data point does not have much effect.

Also, variance explained by first eigen vector: 54.17802452885223 and Variance explained by second eigen vector: 45.82197547114777

Which turns out to be exact same as when done without centering.

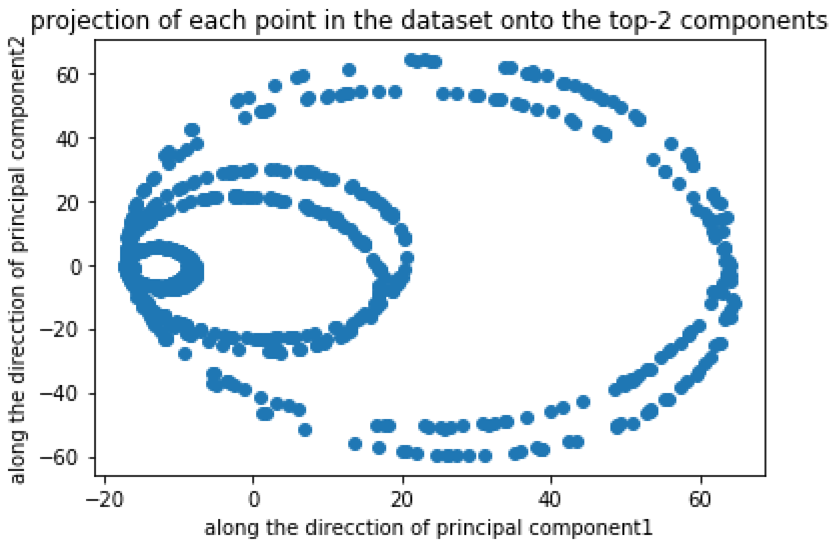
Therefore, the help obtained by centering is not much significant.

Code Reference: **Question1_(i,ii).ipynb / Question1_(i,ii).py**

iii. Kernel - PCA

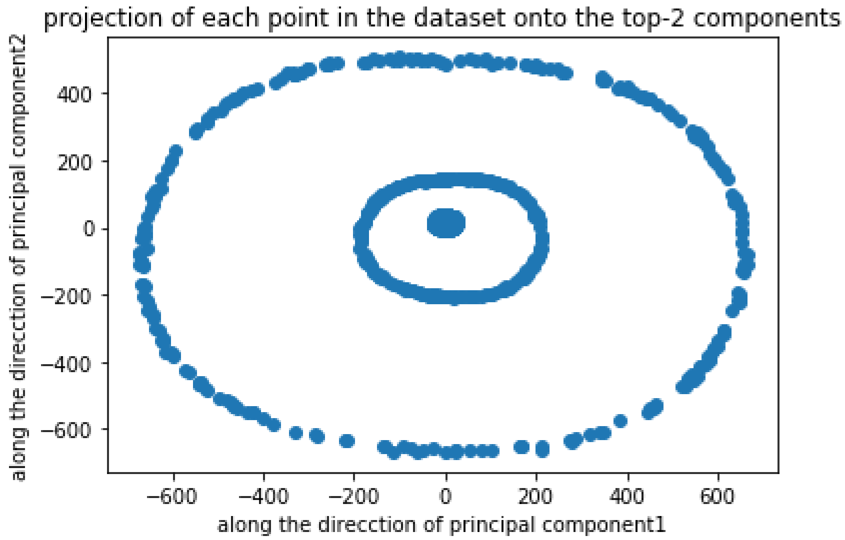
Part A: for d = 2

```
Kernel PCA for Polynomial Kernel Map function with degree 2
variance along maximum eigen vector
36.29556697731797
variance along second maximum eigen vector
32.23467671759193
```



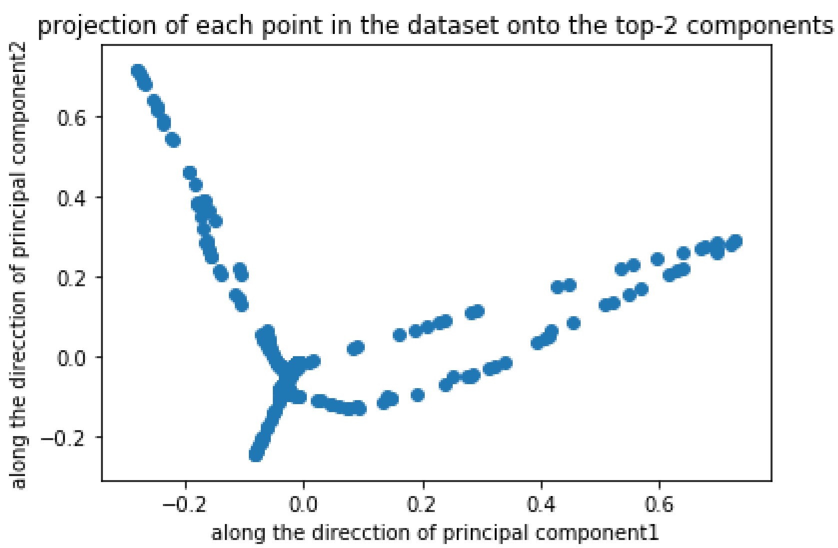
Part A: for d = 3

```
Kernel PCA for Polynomial Kernel Map function with degree 3
variance along maximum eigen vector
41.160512156500545
variance along second maximum eigen vector
32.252691529135184
```



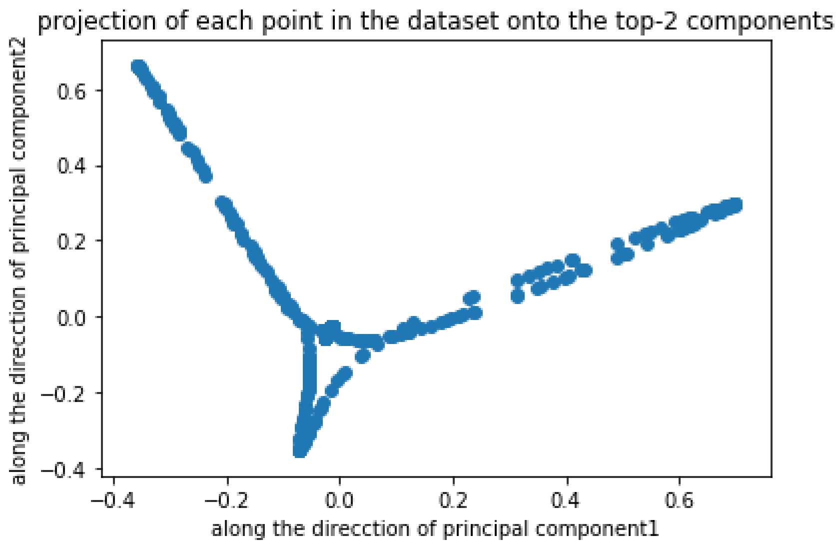
Part B: sigma = 0.1

```
Kernel PCA for Exponential Kernel Map function with sigma  0.1
variance along maximum eigen vector
1.2652688232908913
variance along second maximum eigen vector
1.1894409947940143
```



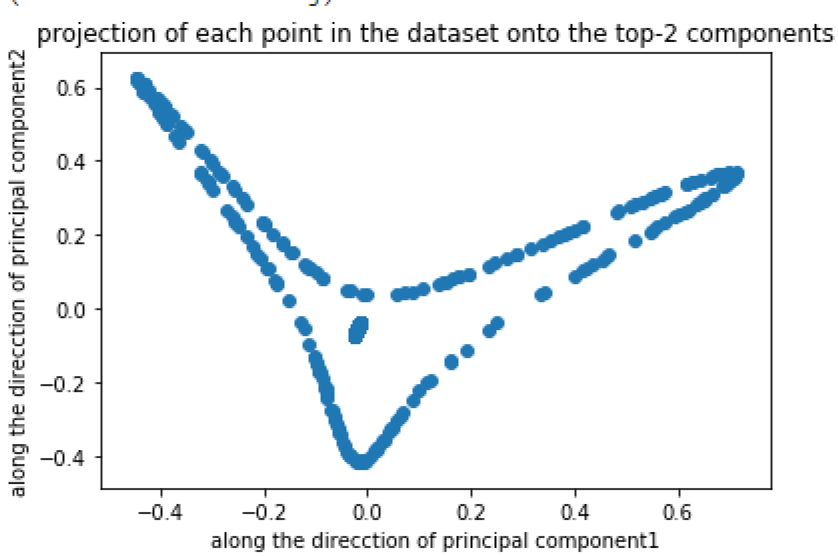
Sigma = 0.2

```
Kernel PCA for Exponential Kernel Map function with sigma  0.2
variance along maximum eigen vector
(2.3934580140210304+0j)
variance along second maximum eigen vector
(2.169895090350989+0j)
```



Sigma = 0.3

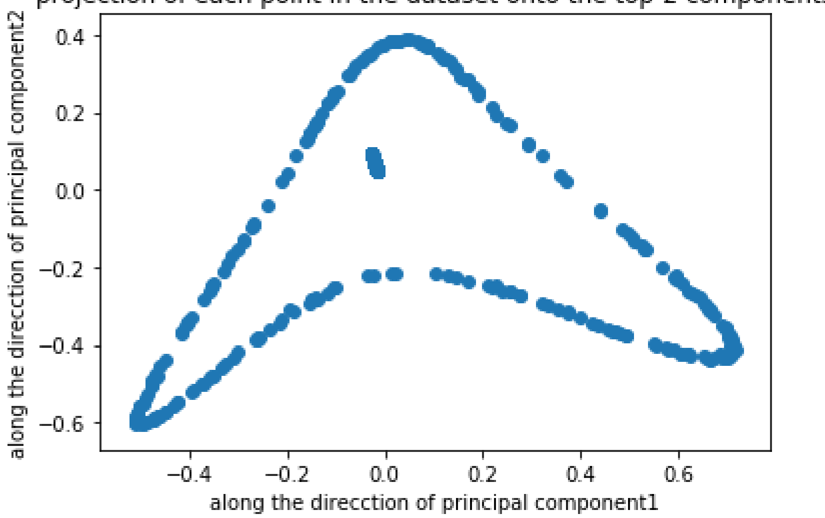
```
Kernel PCA for Exponential Kernel Map function with sigma  0.3
variance along maximum eigen vector
(3.4102060029658126+0j)
variance along second maximum eigen vector
(3.0314726532837932+0j)
```



Sigma = 0.4

```
Kernel PCA for Exponential Kernel Map function with sigma  0.4
variance along maximum eigen vector
(4.327999589589939+0j)
variance along second maximum eigen vector
(3.782429518155212+0j)
```

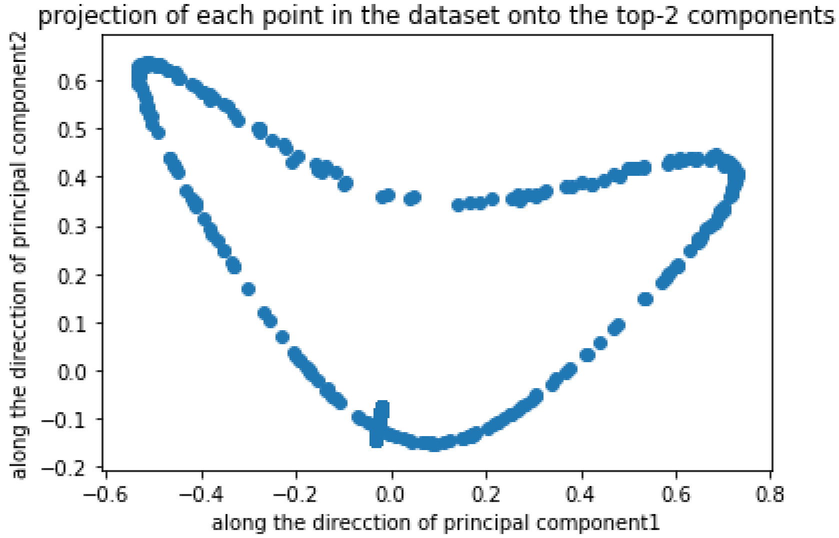
projection of each point in the dataset onto the top-2 components



Sigma = 0.5

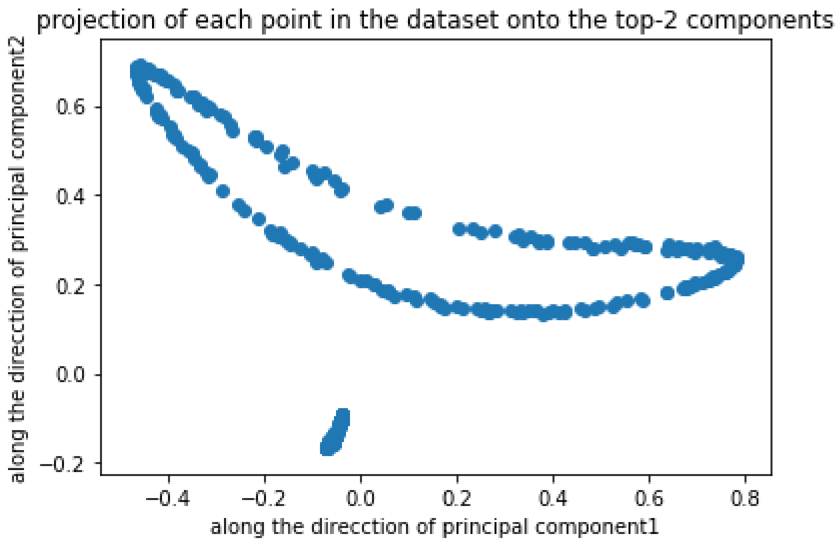
```
Kernel PCA for Exponential Kernel Map function with sigma  0.5
variance along maximum eigen vector
(5.129707836674097+0j)
variance along second maximum eigen vector
(4.4956777790828015+0j)
```

projection of each point in the dataset onto the top-2 components



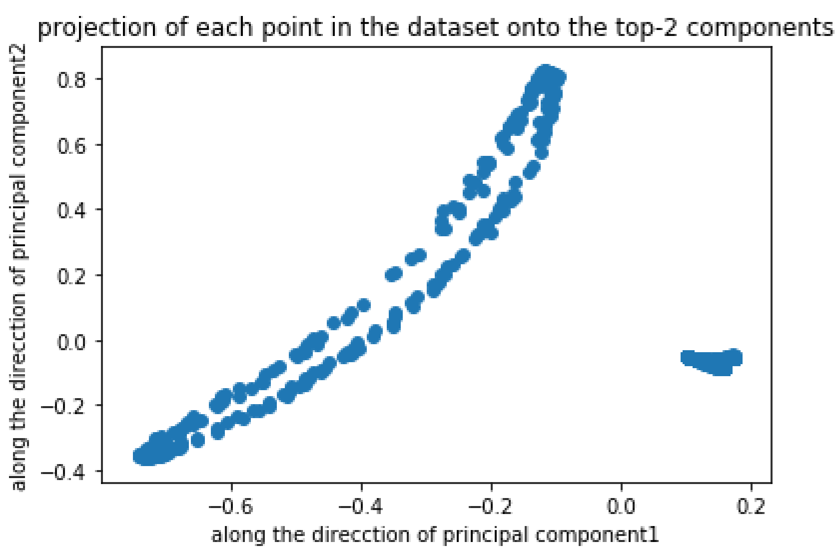
Sigma = 0.6

```
Kernel PCA for Exponential Kernel Map function with sigma  0.6
variance along maximum eigen vector
(5.775854081469021+0j)
variance along second maximum eigen vector
(5.318868526368117+0j)
```



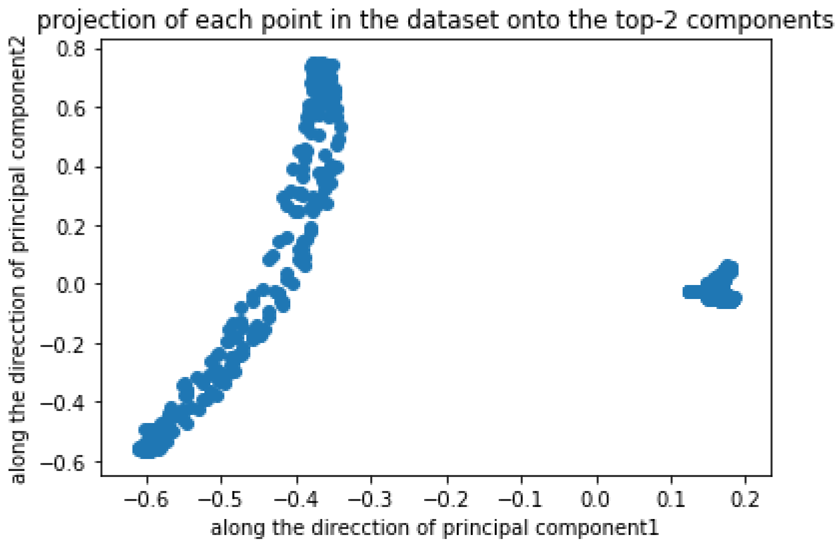
Sigma = 0.7

```
Kernel PCA for Exponential Kernel Map function with sigma  0.7
variance along maximum eigen vector
(6.435231684725467+0j)
variance along second maximum eigen vector
(6.025418760900352+0j)
```



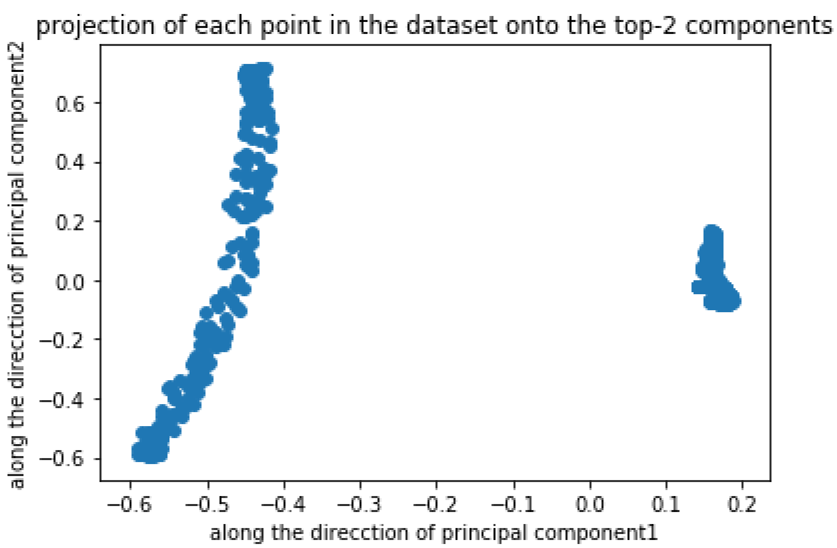
Sigma = 0.8

```
Kernel PCA for Exponential Kernel Map function with sigma  0.8
variance along maximum eigen vector
(7.405184255694549+0j)
variance along second maximum eigen vector
(6.243693976808587+0j)
```



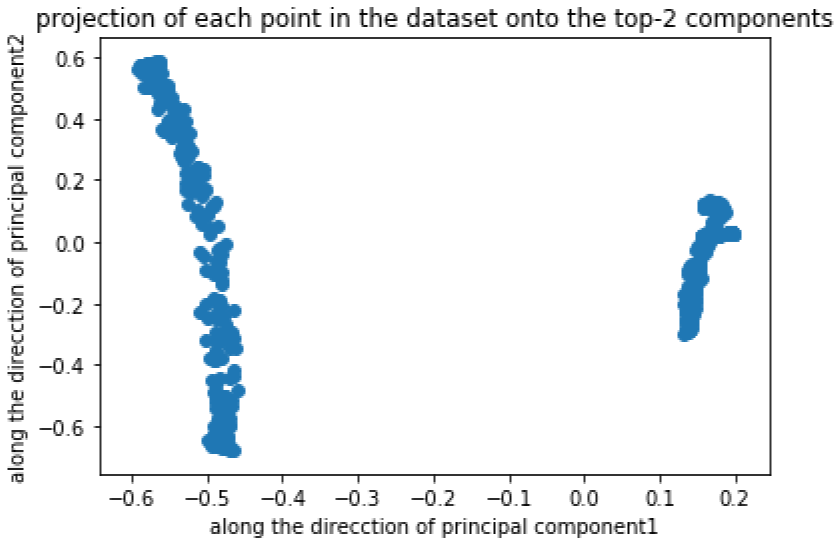
Sigma = 0.9

```
Kernel PCA for Exponential Kernel Map function with sigma  0.9
variance along maximum eigen vector
(8.40810314985004+0j)
variance along second maximum eigen vector
(6.273518566700462+0j)
```



Sigma = 1.0

```
Kernel PCA for Exponential Kernel Map function with sigma  1.0
variance along maximum eigen vector
(9.369748261244858+0j)
variance along second maximum eigen vector
(6.276936610656228+0j)
```



Code Reference: Question1-(iiiAB).ipynb Question1-(iiiAB).py

iv. The kernel we got by using Polynomial Mapping function of the following form

$$\kappa(x, y) = (1 + x^T y)^d$$

using d = 3 is best suited for the dataset.

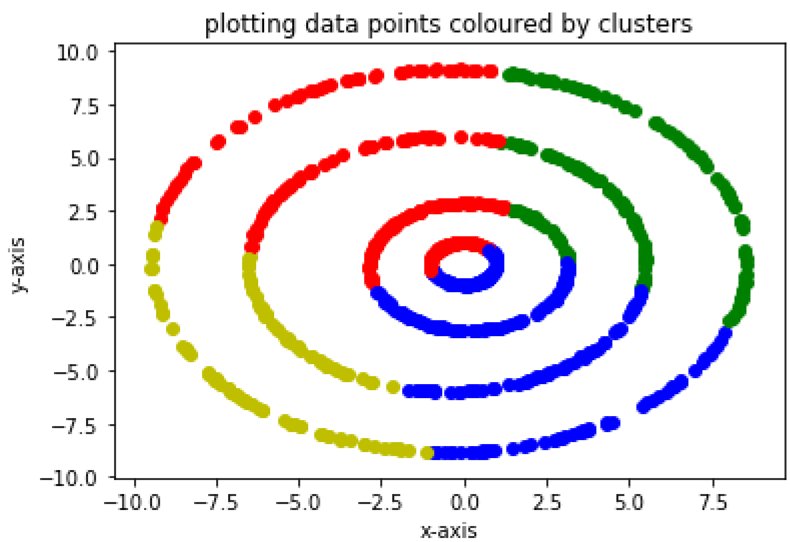
Since the variance explained by two principal components are as follows: 41.16%, 32.25%. Which is comparatively better than all the other approaches followed to obtain Kernel map.

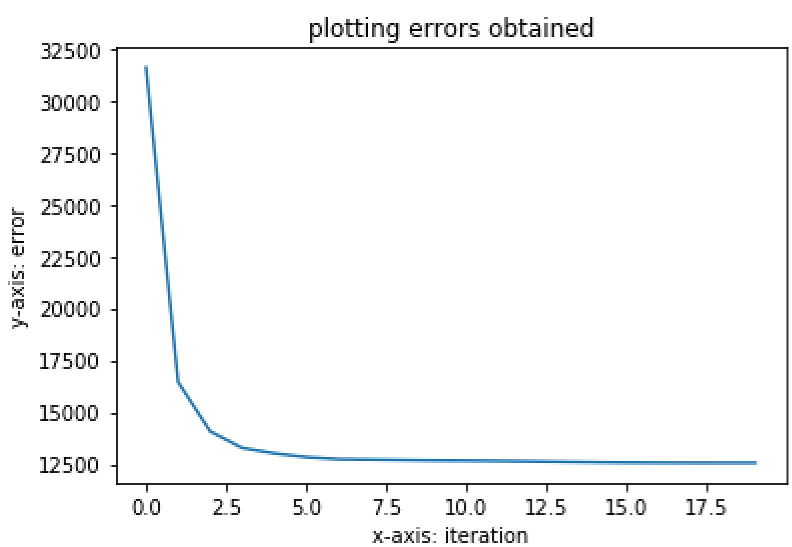
Q2:

Code Reference: Question2.ipynb Question2.py

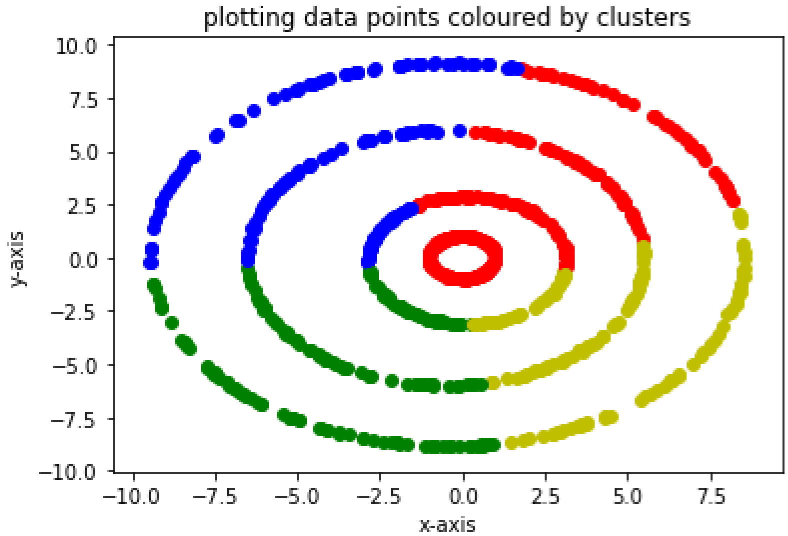
i. Results obtained after trying 5 different random initialization:

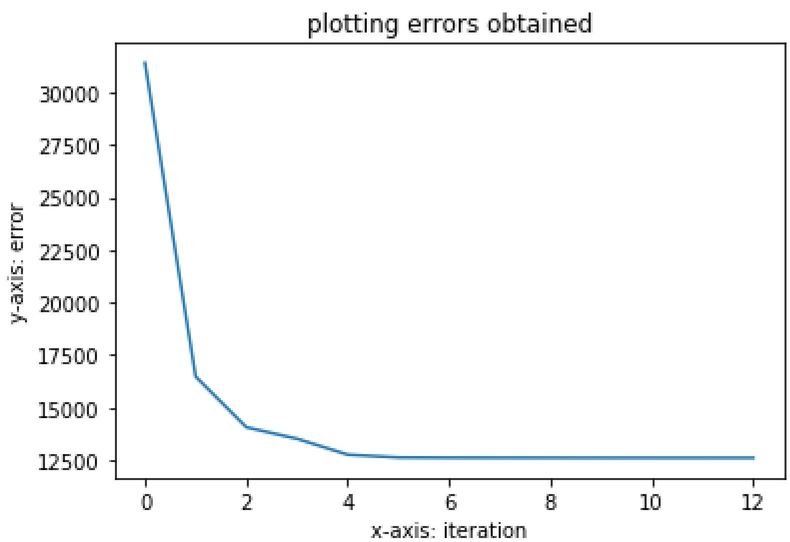
Try: 1



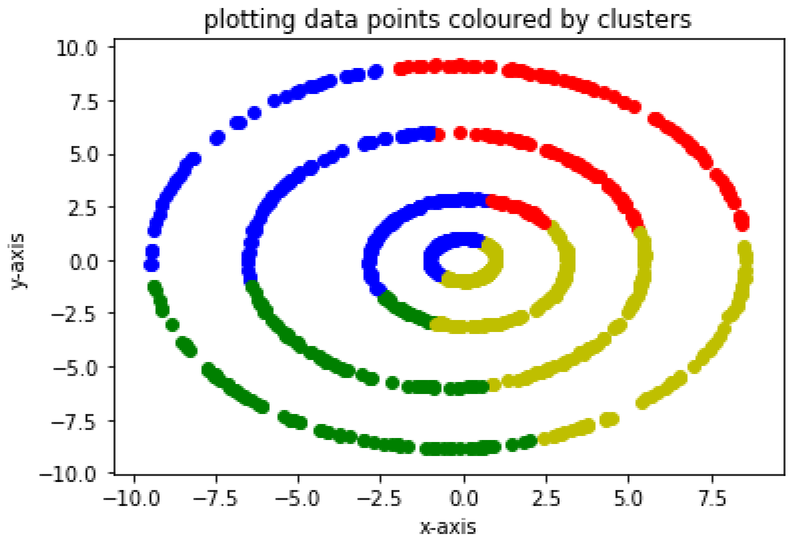


Try: 2

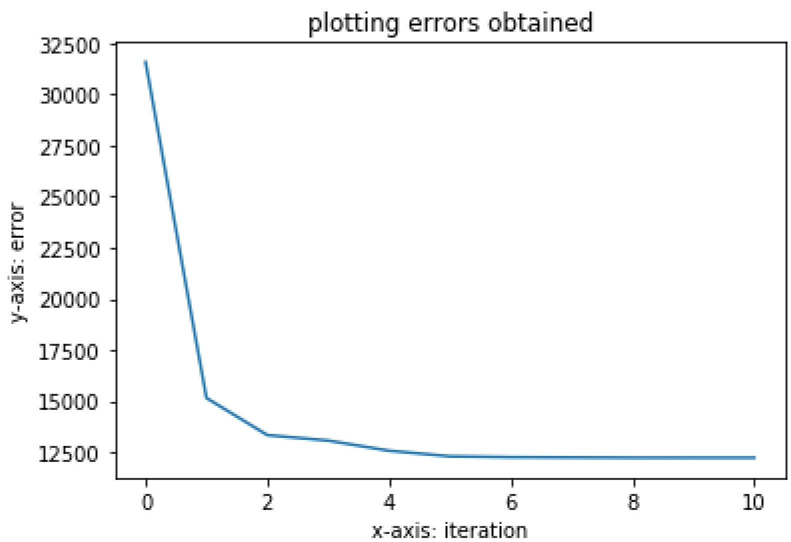

plotting data points coloured by clusters
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
x-axis
y-axis


plotting errors obtained
30000
27500
25000
22500
20000
17500
15000
12500
0
2
4
6
8
10
12
x-axis: iteration
y-axis: error

Try: 3

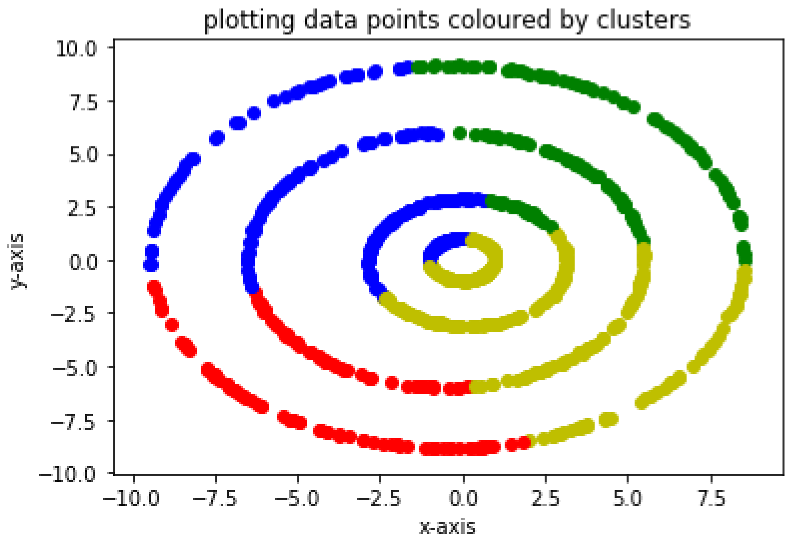


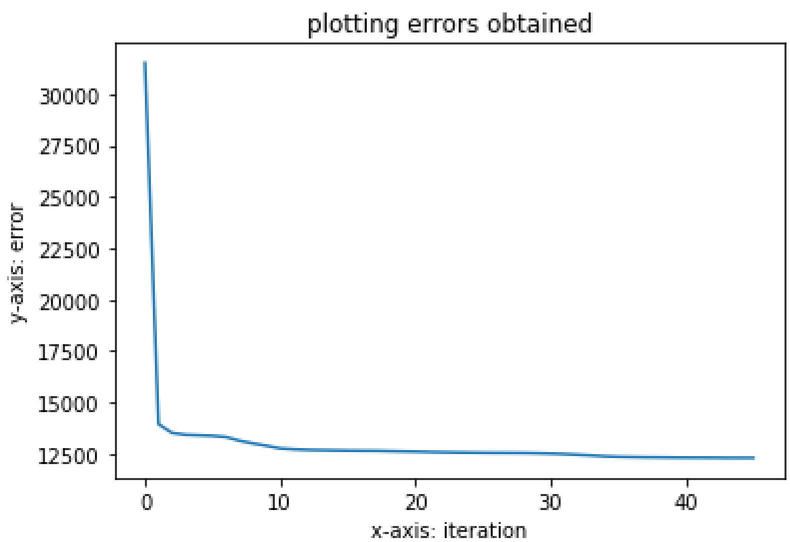
plotting data points coloured by clusters
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
x-axis
y-axis



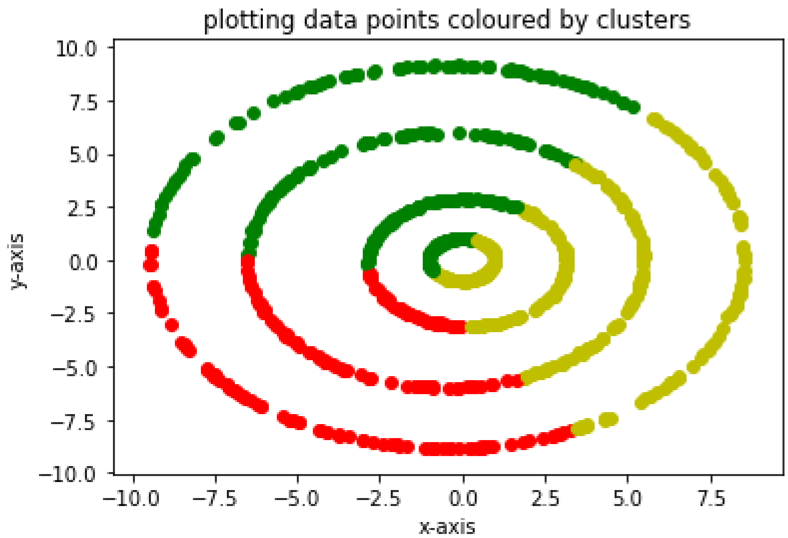
plotting errors obtained
32500
30000
27500
25000
22500
20000
17500
15000
12500
0
2
4
6
8
10
x-axis: iteration
y-axis: error

Try: 4

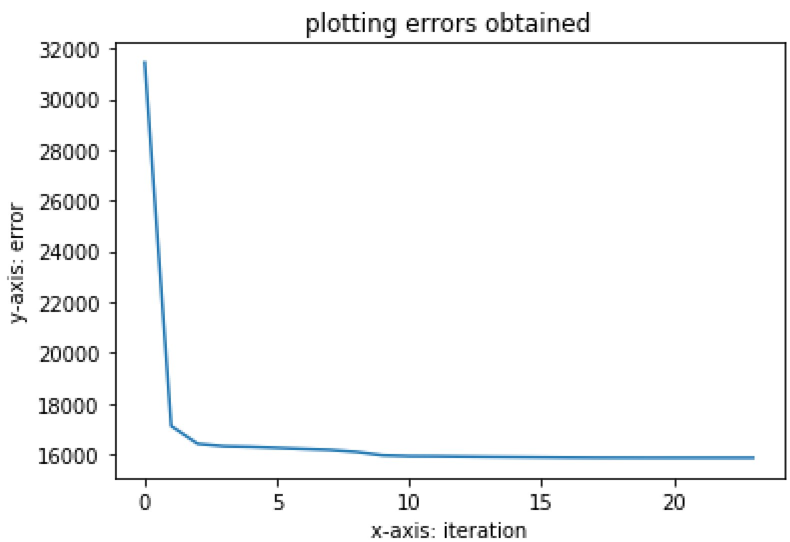

plotting data points coloured by clusters
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y-axis
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
x-axis


plotting errors obtained
30000
27500
25000
22500
20000
17500
15000
12500
y-axis: error
0
10
20
30
40
x-axis: iteration

Try: 5



plotting data points coloured by clusters
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
x-axis
y-axis

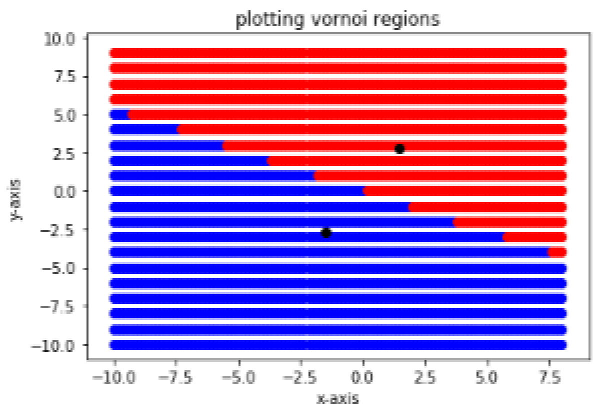


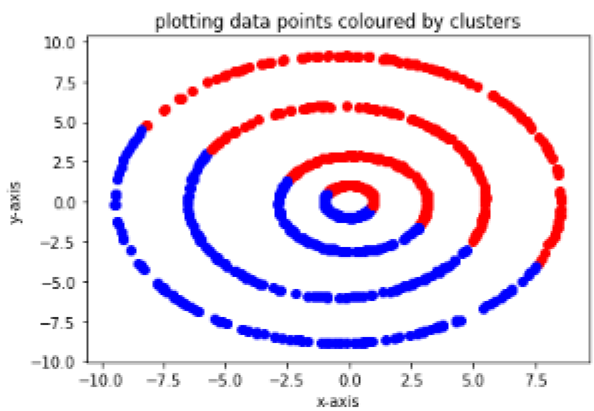
plotting errors obtained
32000
30000
28000
26000
24000
22000
20000
18000
16000
0
5
10
15
20
x-axis: iteration
y-axis: error

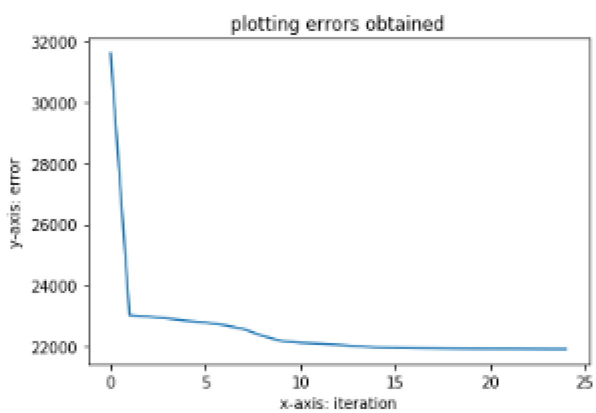
ii. For K = {2,3,4,5}

Obtained cluster centres according to K-means algorithm using fixed initialization. Results found are presented as follows:

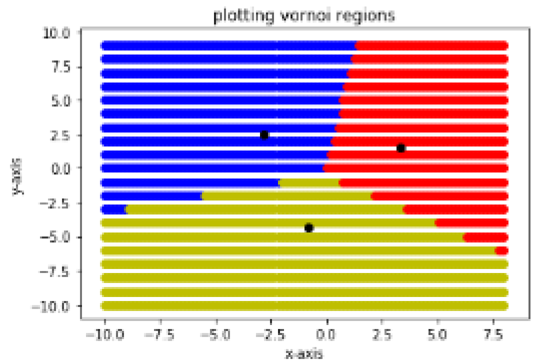
for K = 2

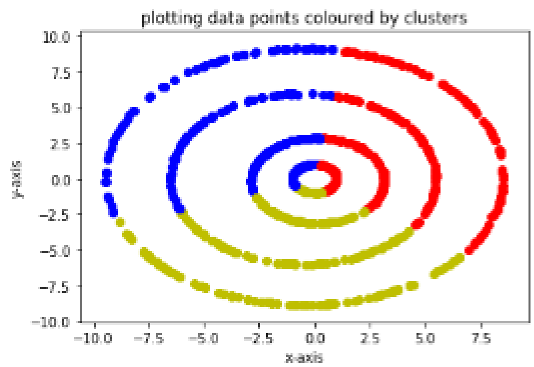


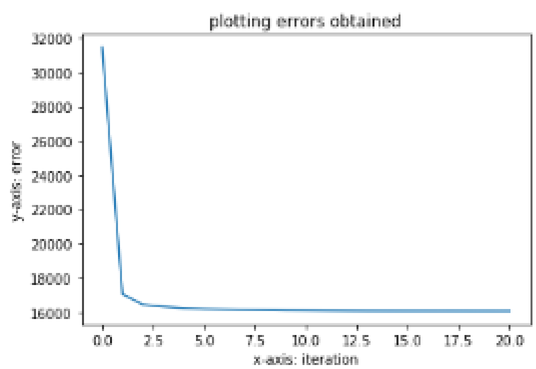




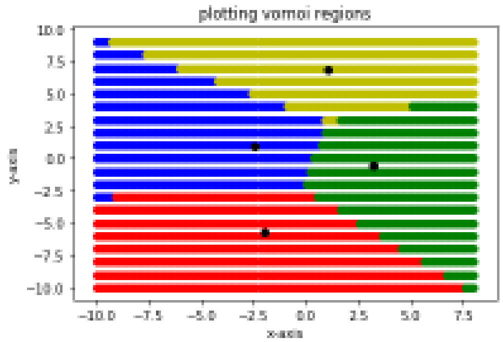
For K = 3

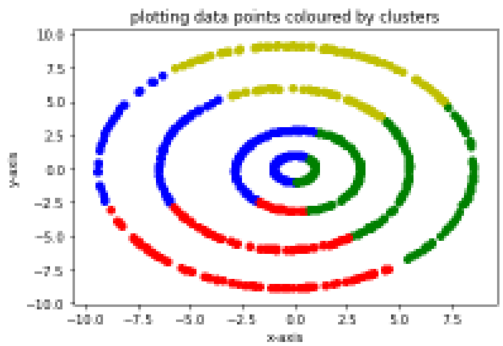

plotting vornoi regions
y-axis
x-axis

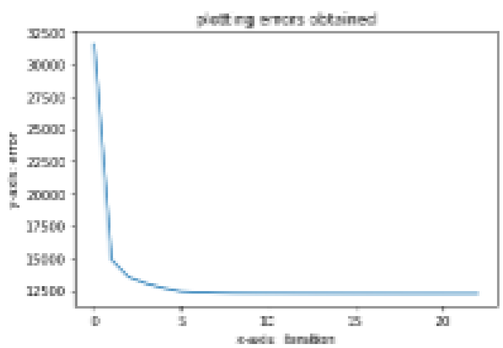

plotting data points coloured by clusters
y-axis
x-axis


plotting errors obtained
y-axis: error
x-axis: iteration

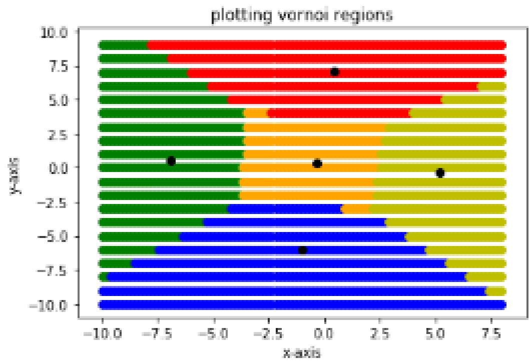
For K = 4

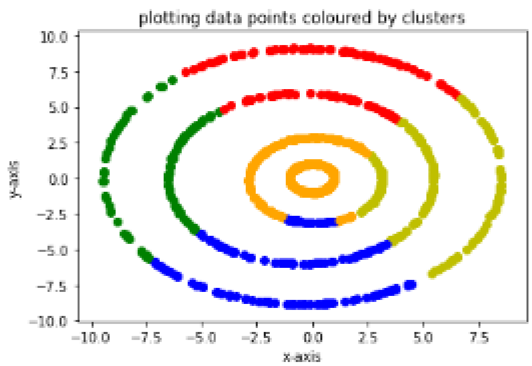

plotting vornoi regions
y-axis
x-axis

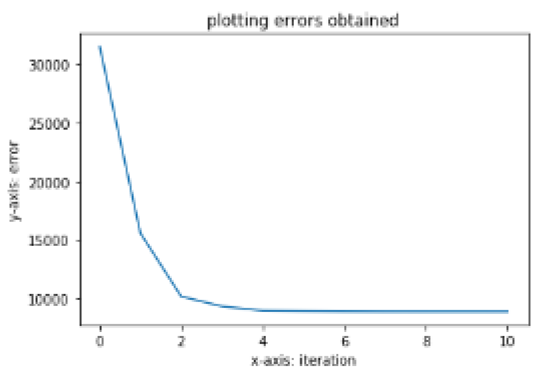

plotting data points coloured by clusters
y-axis
x-axis


plotting errors obtained

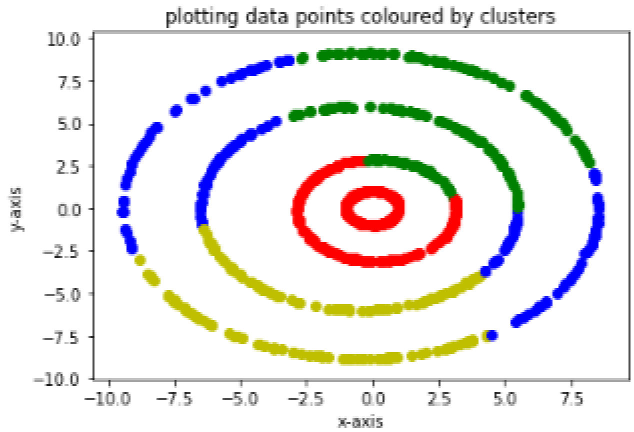
For K = 5

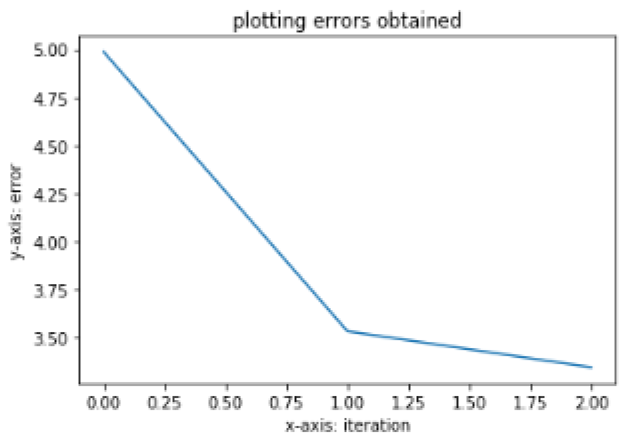

plotting vornoi regions
y-axis
x-axis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5


plotting data points coloured by clusters
y-axis
x-axis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5


plotting errors obtained
y-axis: error
x-axis: iteration
30000
25000
20000
15000
10000
0
2
4
6
8
10

iii. Spectral Clustering





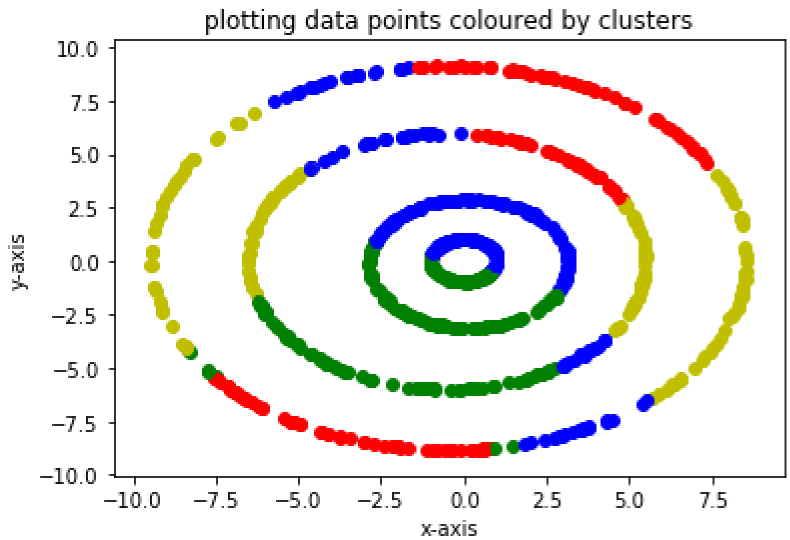

```
18284.83388272092
```

Choice of Kernel function:

Polynomial Kernel Mapping with degree: 2

Reasoning: This Kernel was explaining maximum variance over its top eigen-vectors amogst others for this dataset. Also the error function goes to a better minimum as compared to other kernel functions. Computed the sum of distances from the mean corresponding to each points and this result is also appears to be minimum for this choice of kernel.

iv. Result obtained by following the suggested method of cluster assignment:




```
28785.360966743778
```

Observation: this strategy of cluster assignment does not work as well as other strategies, as seen in the output plot. Also the sum of distances of the data points with their corresponding cluster mean is coming out to be much larger compared to other approaches.