CS5691: Pattern Recognition and Machine Learning Assignment 1

Anirud N CE21B014

1 Question 1

1.1 (i) Write a piece of code to run the PCA algorithm on this data-set. Show the images of the principal components that you obtain. How much of the variance in the data-set is explained by each of the principal components?

Figures (1),(2),(3), (4) and (5) denote the results of PCA for different percentages of reconstruction on a randomly chosen image.
Table 1 shows the number of eigenvectors required for each case.

1st eigen vector explain 9.978234970109387 percent variance
2st eigen vector explain 7.127842913226323 percent variance
3st eigen vector explain 6.292965387371177 percent variance
4st eigen vector explain 5.311458416034822 percent variance
5st eigen vector explain 4.897500164620657 percent variance
6st eigen vector explain 4.221722355458734 percent variance
7st eigen vector explain 3.231499350009128 percent variance
8st eigen vector explain 2.9643127391467012 percent variance
9st eigen vector explain 2.6892119673191357 percent variance
10st eigen vector explain 2.36284417951278 percent variance
11st eigen vector explain 2.11924308894583 percent variance
12st eigen vector explain 1.9916812727037083 percent variance
13st eigen vector explain 1.8286867878566415 percent variance
14st eigen vector explain 1.7741830953926805 percent variance
15st eigen vector explain 1.6658843620868355 percent variance
16st eigen vector explain 1.5552546403681444 percent variance
17st eigen vector explain 1.3360049182955291 percent variance
18st eigen vector explain 1.2465547134320452 percent variance
19st eigen vector explain 1.2118454540704182 percent variance
20st eigen vector explain 1.1674021170876354 percent variance
21st eigen vector explain 1.120761128475578 percent variance
22st eigen vector explain 1.0244839214657788 percent variance
23st eigen vector explain 0.9587625795340443 percent variance
24st eigen vector explain 0.9125259369099482 percent variance
Further values can be found in the code file : Question 1.ipynb

Percent of Reconstuction	Number of eigenvectors to consider
100	784
99	277
95	133
90	79
80	41

Table 1: Number of eigenvectors required for each case of reconstruction

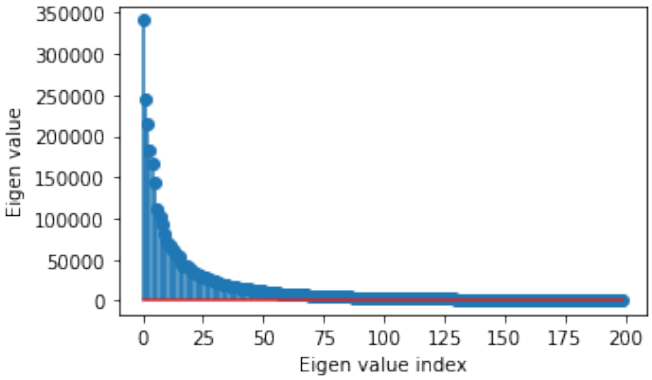


Figure 1: Eigen Values

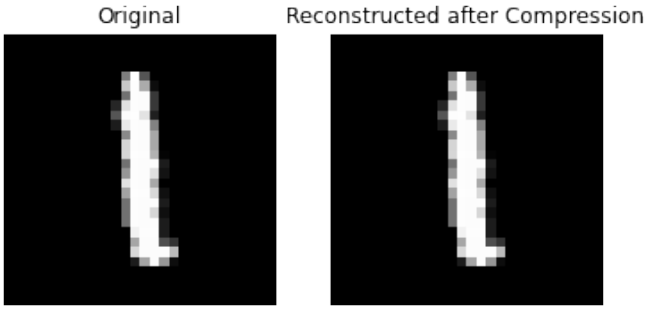


Figure 2: Fully reconstructed image

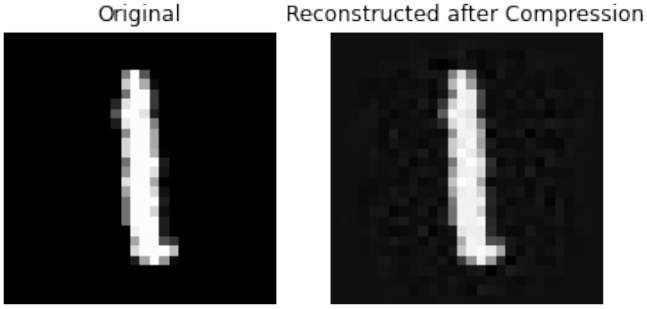


Figure 3: 99 percent reconstruction

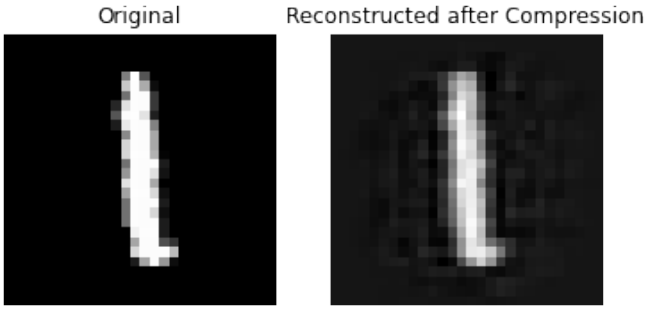


Figure 4: 95 percent reconstruction

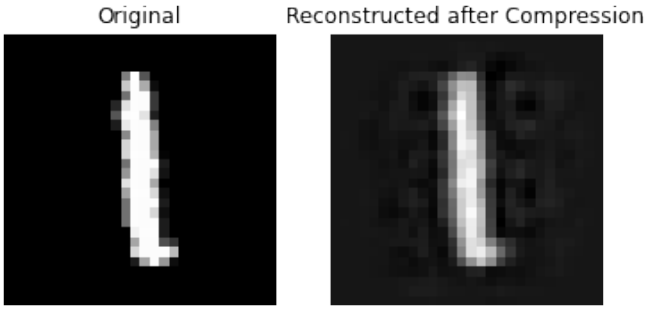


Figure 5: 90 percent reconstruction

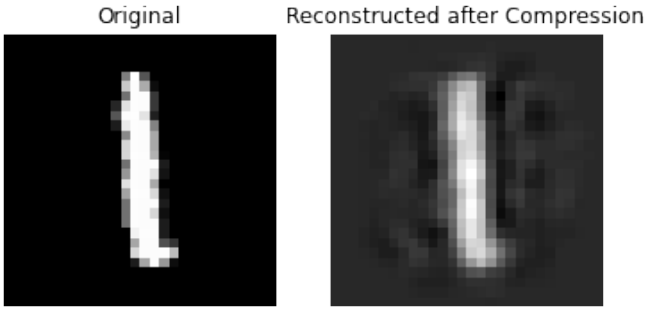
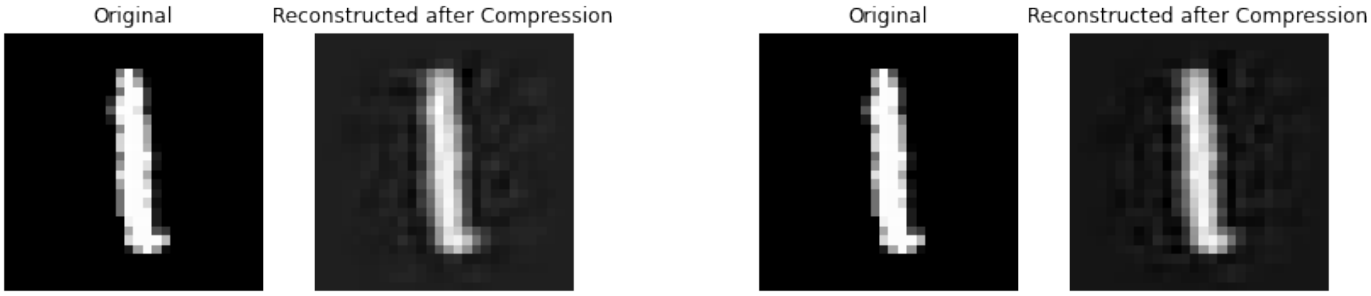


Figure 6: 80 percent reconstruction

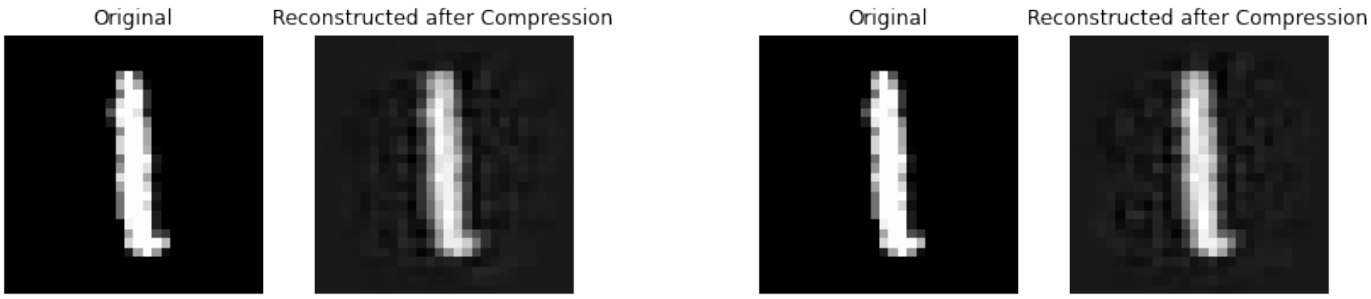
1.2 (ii) Reconstruct the dataset using different dimensional representations. How do these look like? If you had to pick a dimension d that can be used for a downstream task where you need to classify the digits correctly, what would you pick and why?

Assuming that d here means the number of eigenvectors chosen to represent the data
Figures (7) to (14) show the reconstructed image results for different d values.
We find that as d increases,the reconstructed image is more similar toh the input or original image. Higher the value of d, lesser is the loss of information.
If I were to choose a particular d for downstream task or tasks like supervised learning etc, I would choose a d value of around 390-420, because this cuts down the required data storage by around 45-50 percentage, and also from Figure (14) it is clear that the reconstructed image looks very similar visually to the original image, meaning we have captured all the essential features.
Picking d to be 390 explains around 99.79 percent of the variance.



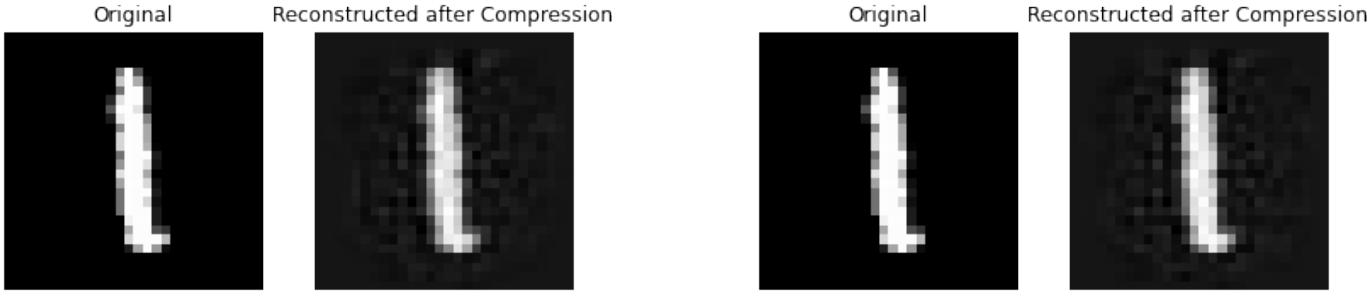
(a) (b)

Figure 7: d = 100 and d = 120



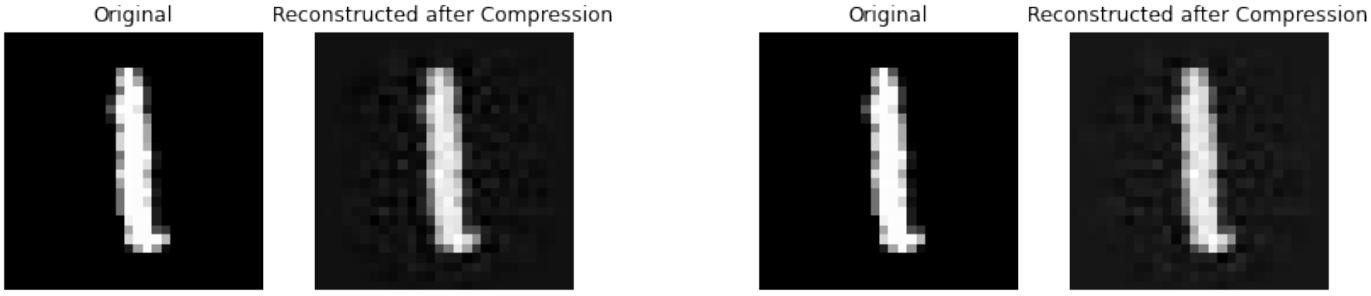
(a) (b)

Figure 8: d = 140 and d=160



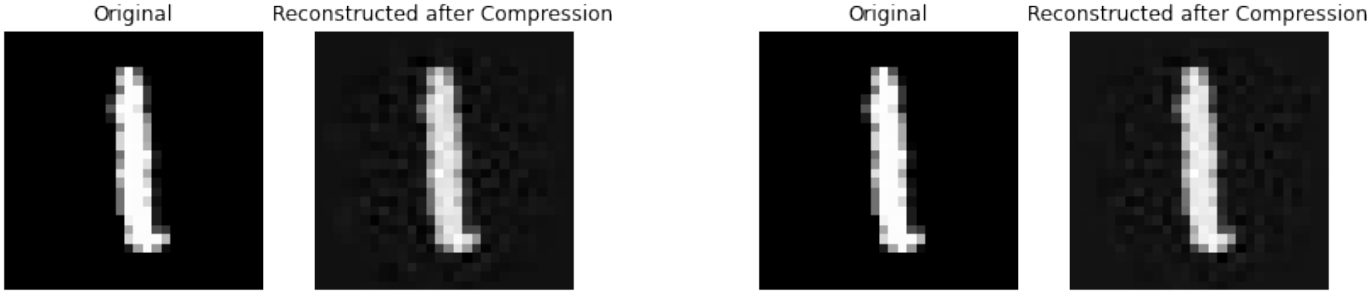
(a) (b)

Figure 9: d=180 and d=200



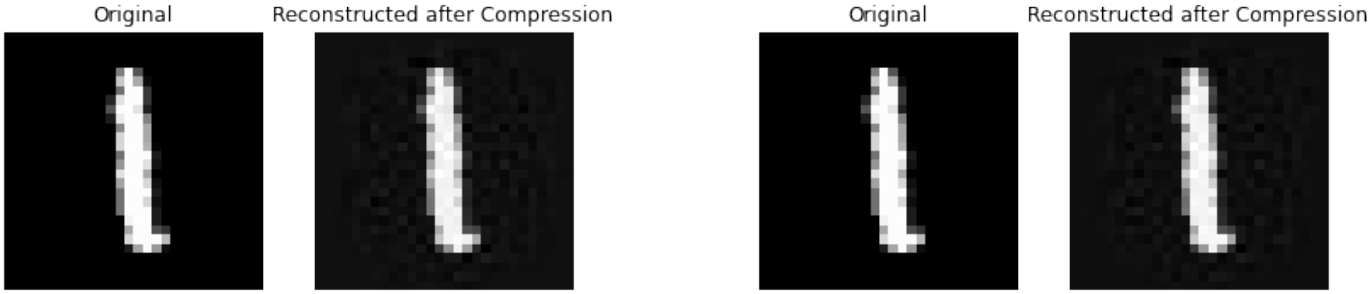
(a) (b)

Figure 10: d=210 and d=230



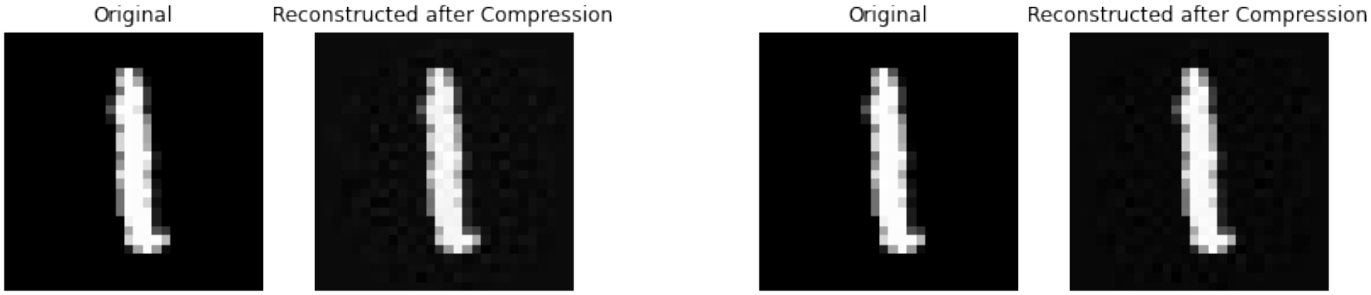
(a) (b)

Figure 11: d=240 and d=260



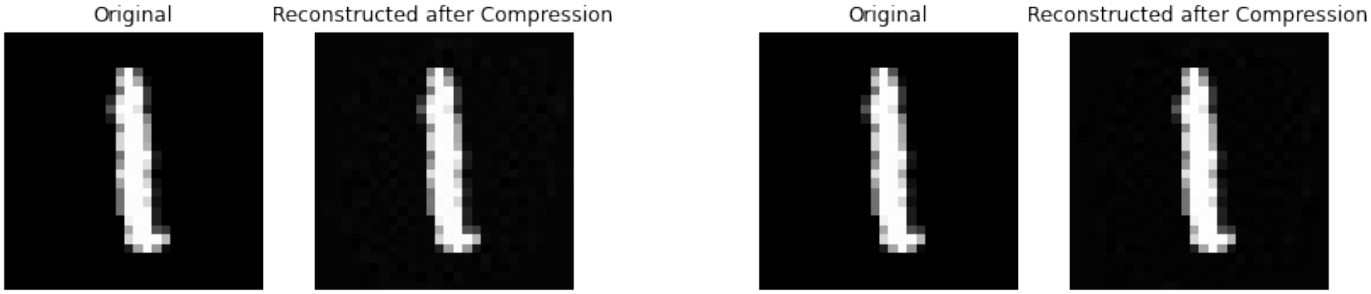
(a) (b)

Figure 12: d=280 and d=300



(a) (b)

Figure 13: d=330 and d=360



(a) (b)

Figure 14: d=390 and d=420

1.3 (iii) Write a piece of code to implement the Kernel PCA algorithm on this dataset. Plot the projection of each point in the dataset onto the top-2 components for each kernel

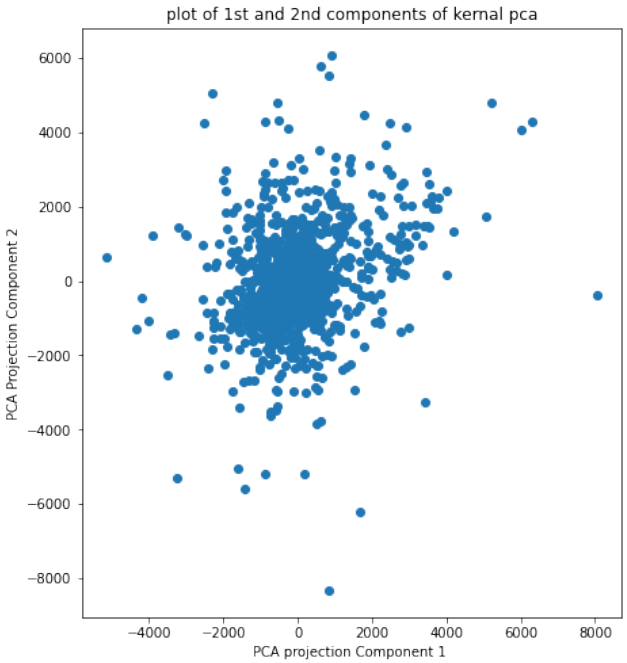
Kernal PCA was performed on polynomial and exponential kernels for different degrees and different variances, respectively. Figure (15) and Figure (16) shows the plots for polynomial kernel (part a of the question) and the exponential kernel respectively (part b of the question)

1.4 (iv) Which Kernel do you think is best suited for this dataset and why?

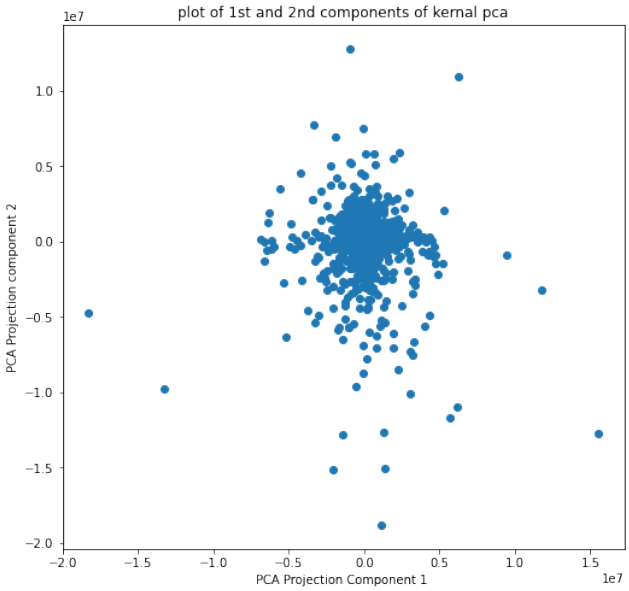
The kernel (b), i.e., the exponential kernel, is a better kernel because, The degree of decorrelation of the projection of the two principle components is seen more in the exponential kernel than in the polynomial kernel. For instance, in a polynomial kernel, the y values, i.e., the values of the second principal components, go as high as 10000 or more, which shows that the components are not well de-correlated, and the values of the second principal component projections for each data point are not relatively closer to y-axis than the x-axis.

The exponential Kernel plots, on the other hand, show good level of decorrelation between the two. The values of the second principal component projection for each data point are relatively closer to 0 than the first principal component. The features are better de-correlated.

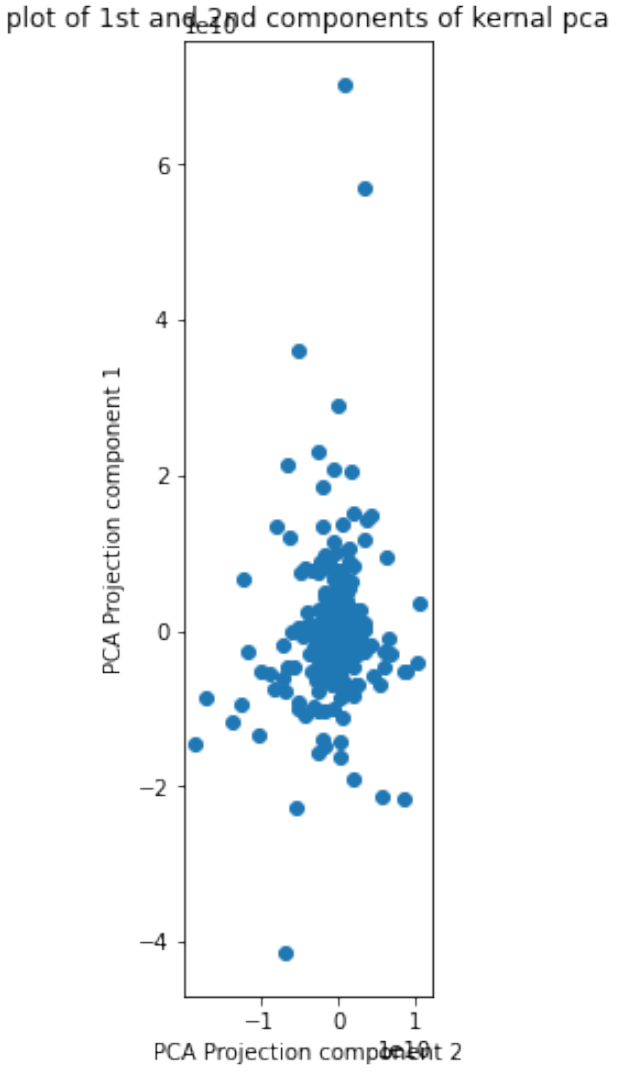
So the Exponential Kernel (b) is better suited for this data.



(a)

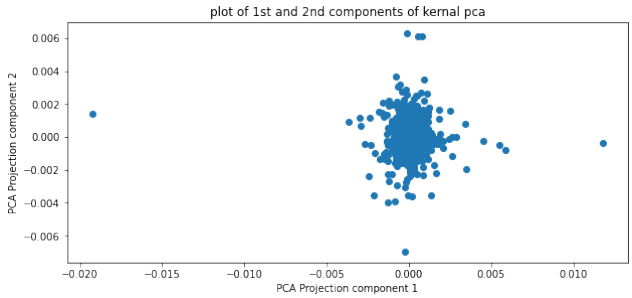


(b)

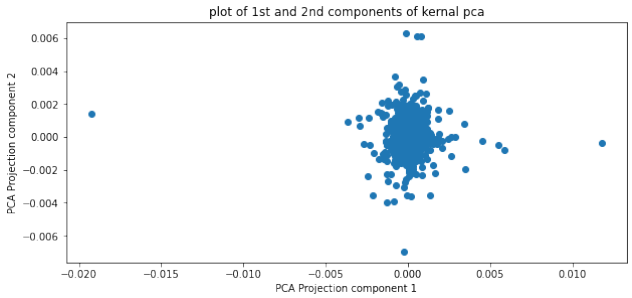


(c)

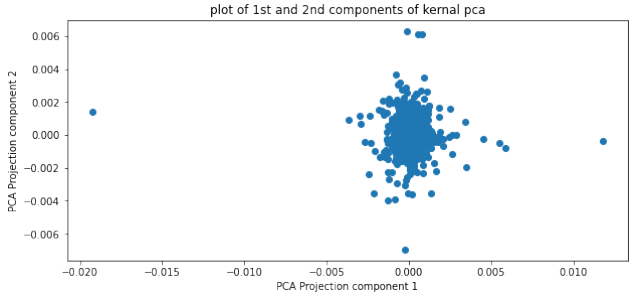
Figure 15: (a) d=2 (b) d = 3 (c) d=4 Polynomial kernel plots for 1st vs 2nd components of PCA



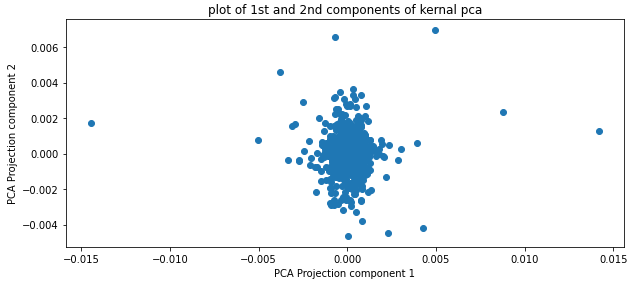
(a)



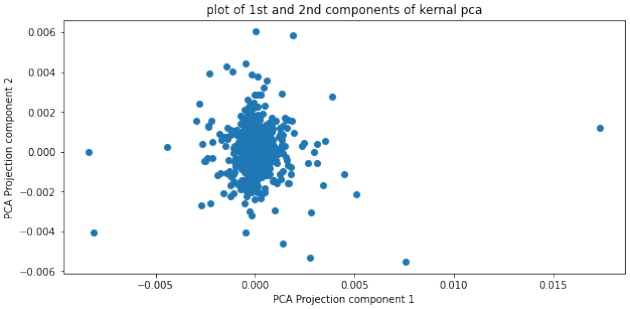
(b)



(c)



(d)



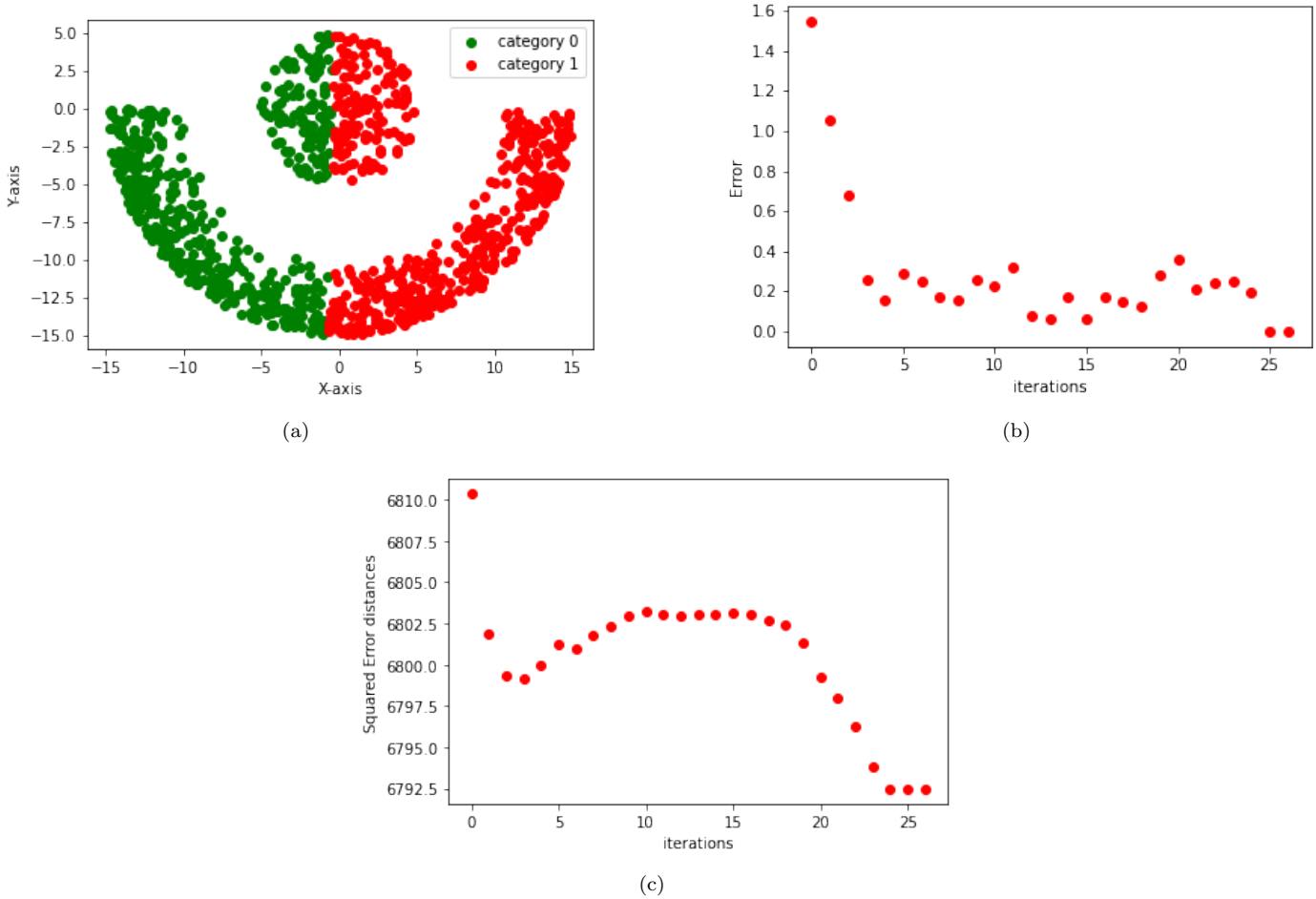
(e)

Figure 16: (a) $\sigma_2 = 0.1$ (b) $\sigma_2 = 1$ (c) $\sigma_2 = 10$ (d)$\sigma_2 = 50$ (e) $\sigma_2 = 100$ Exponential kernel plots for 1st vs 2nd components of PCA

2 Question 2

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

Let O be the list of means at iteration t, and O' be the list of means of the clusters at iteration t+1.
Then, the error is defined as the sum of absolute values of the differences of all the coordinates of all the cluster mean points between O and O'
The clustering converges when the error function remains 0 for more than 1 iteration.
Plots have been done for the above-defined error function and the standard Squared error function(this is the sum of squared distance values of each data point from the mean or center point of the cluster it belongs to). Figure (17),(18),(19),(20),(21) show the clustering and the error function plots for 5 random initialisations

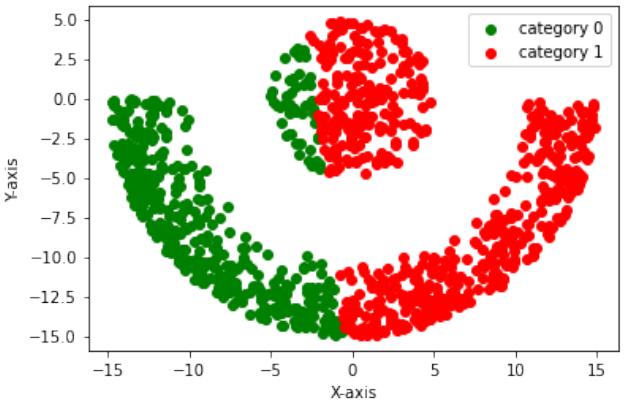


(a)

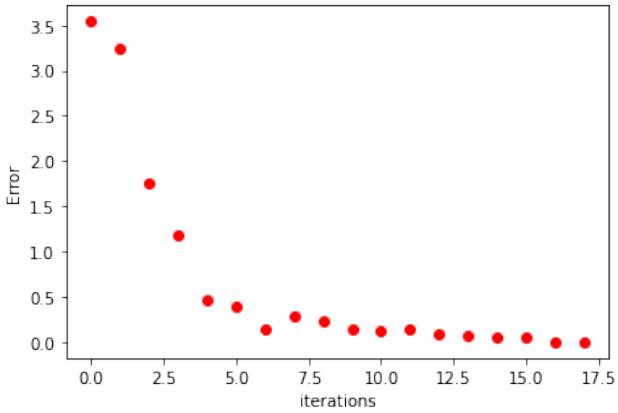
(b)

(c)

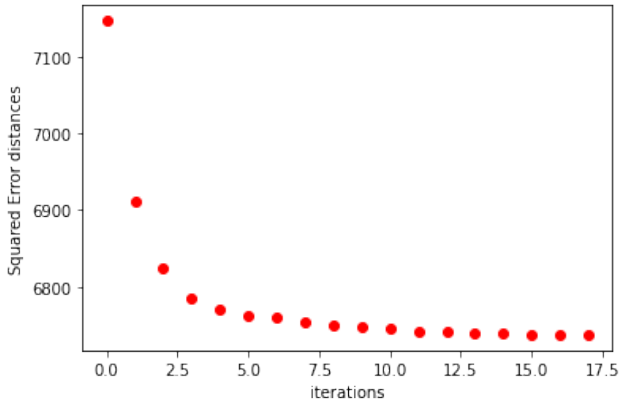
Figure 17: Random Initialisation 1 Clustering and PLot of error function vs iterations



(a)

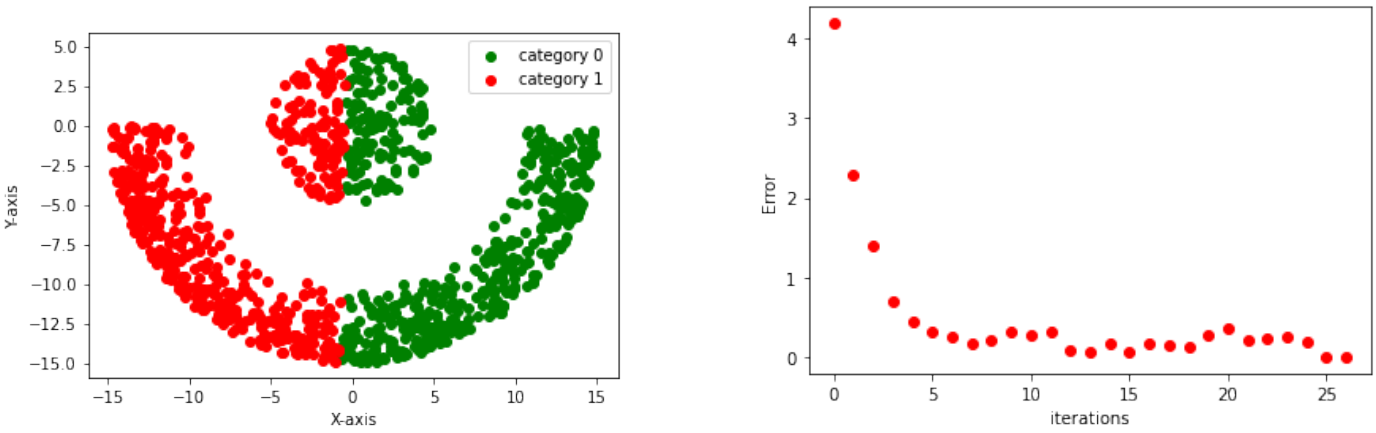


(b)



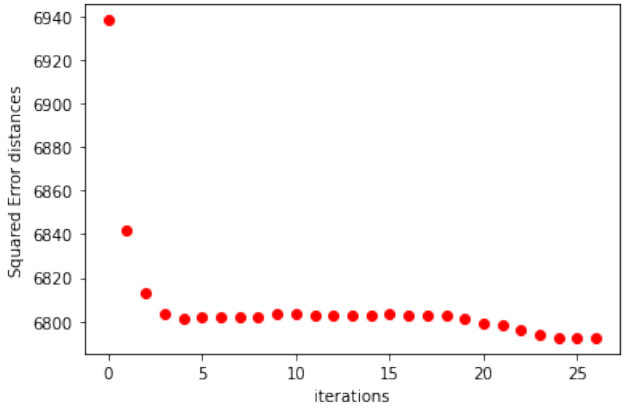
(c)

Figure 18: Random Initialisation 2 Clustering and PLot of error function vs iterations



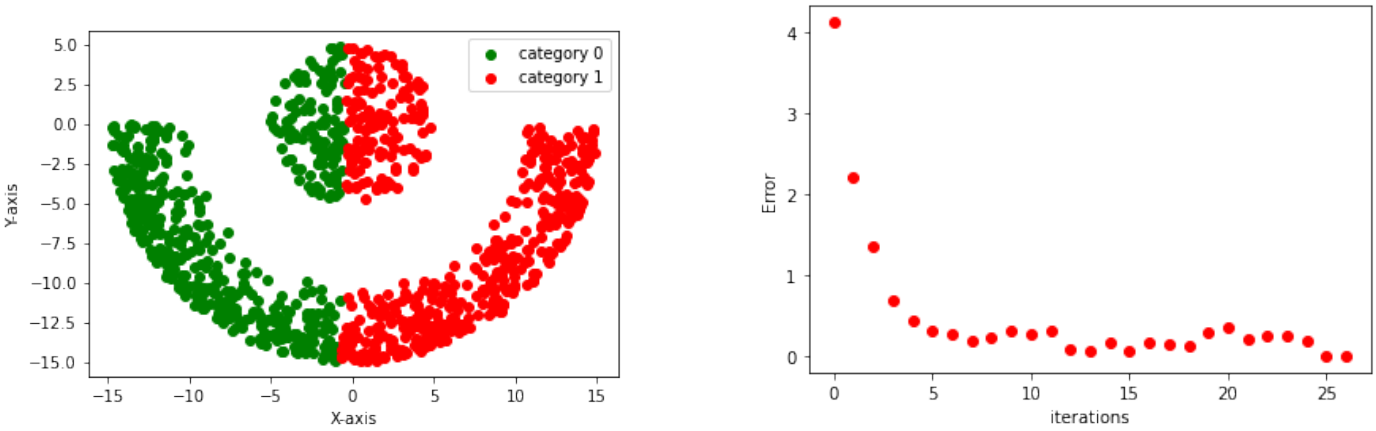
(a)

(b)



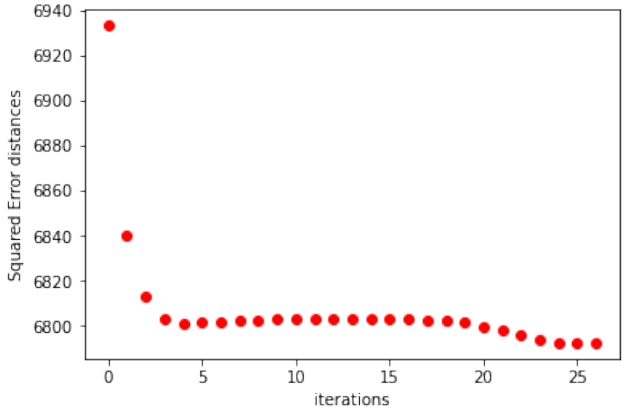
(c)

Figure 19: Random Initialisation 3 Clustering and PLot of error function vs iterations



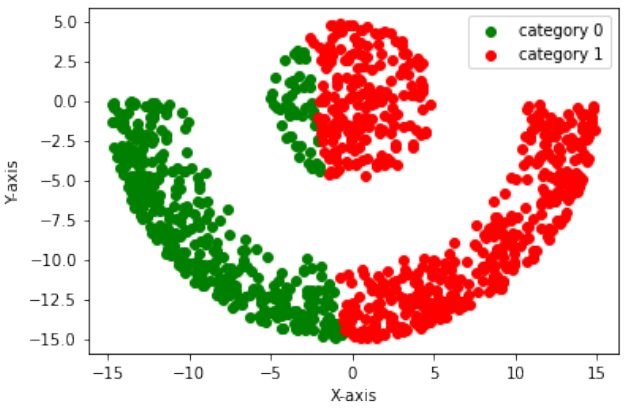
(a)

(b)

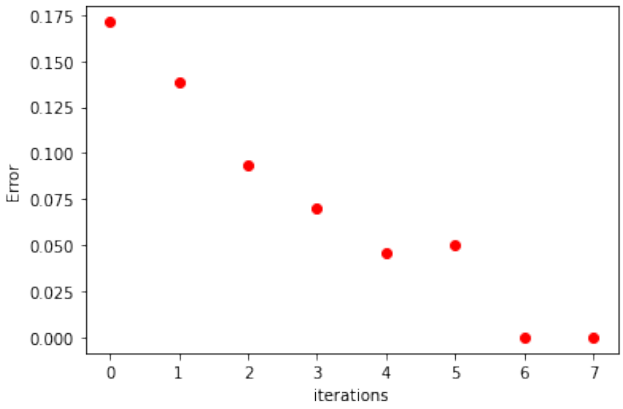


(c)

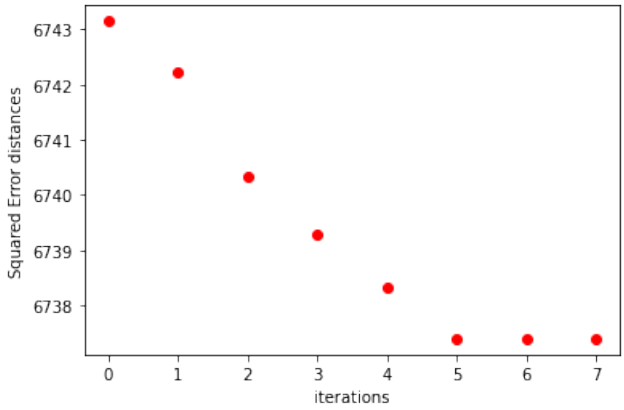
Figure 20: Random Initialisation 4 Clustering and PLot of error function vs iterations



(a)



(b)



(c)

Figure 21: Random Initialisation 5 Clustering and Plot of error function vs iterations

2.2 (ii) Fix a random initialization. For K = 2,3,4,5 , obtain cluster centers according to K-means algorithm using the xed 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).

Figure 22 shows the Voronoi regions for k=2,3,4,5.
Voronoi regions were split by finding the mean points of each cluster and then finding the perpendicular bisector for line joining mean points of any 2 clusters. This perpendicular bisector divides the regions

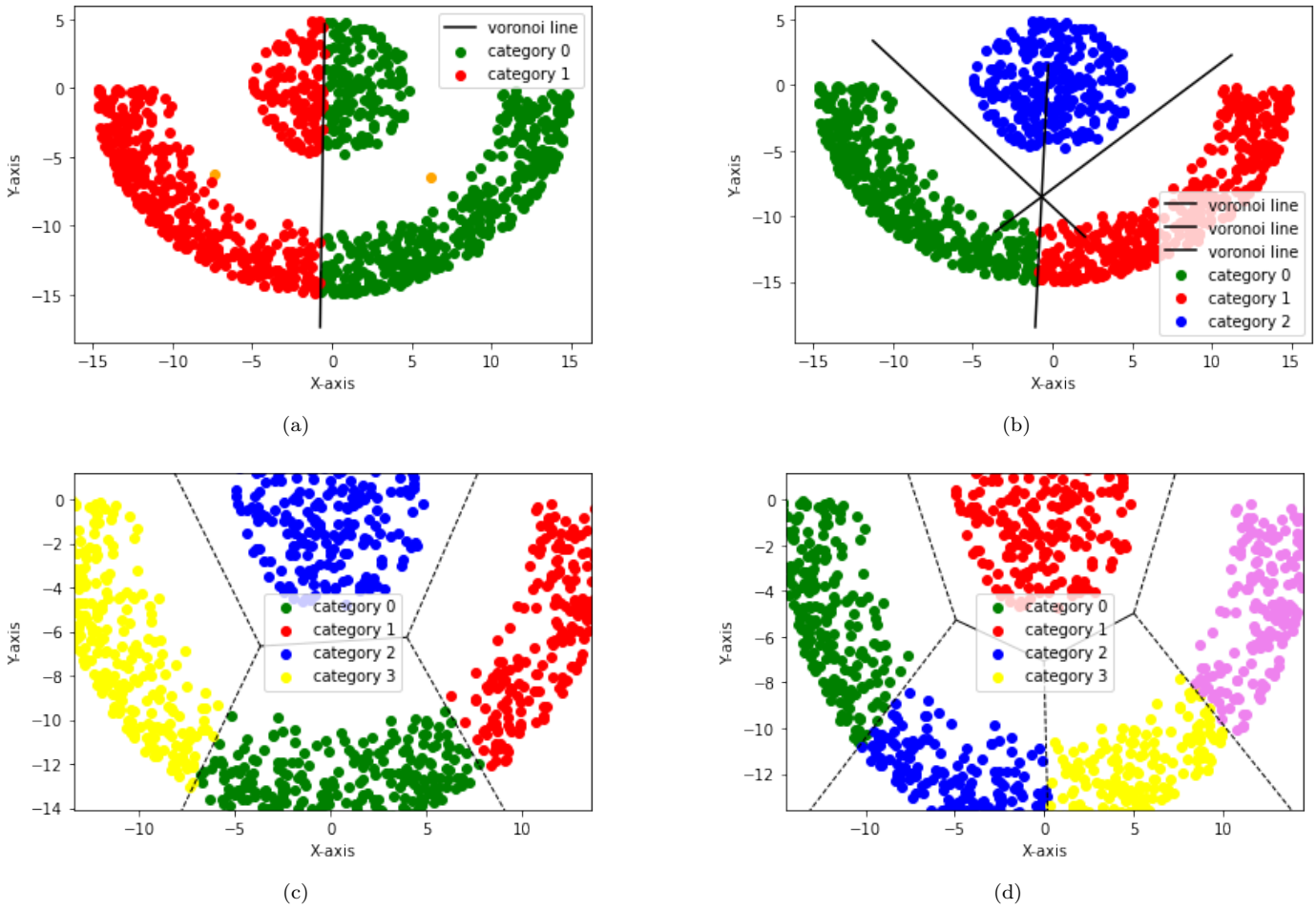
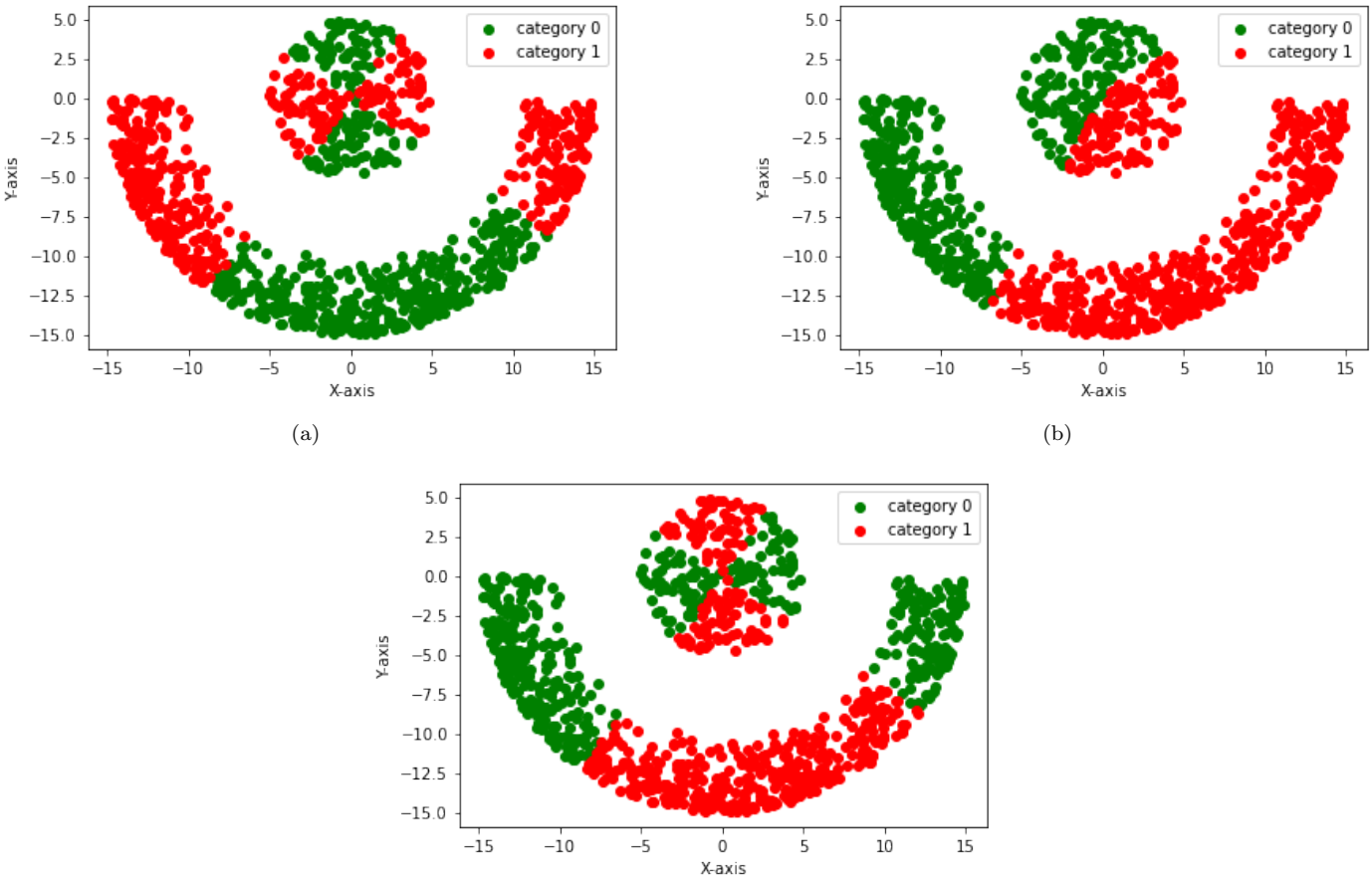


Figure 22: Clustering and Voronoi regions for (a)K=2,(b)k=3,(c)k=4,(d)k=5 respectively

2.3 (iii) Run the spectral clustering algorithm (spectral relaxation of K-means using Kernel PCA) k =2. 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

From the plots in Figure (23) and Figure (24), it is clear that tuning of parameters is essential to get the best cluster out of any of the kernels used. This could be realised by seeing the difference in clustering between σ^2=0.7 and other σ^2 values for exponential kernel.For σ^2 values greater than 50, the clustering is similar to non kernalised cases.
Polynomial kernels do not perform well in clustering for this particular data, implying that the conversion of the data points into a higher dimension is better when we use RBF over Polynomial kernels
Exponential kernel with variance σ^2=0.7 clusters the best among all other combinations of hyperparameters and kernels. It perfectly clusters all the points in the larger semi-circle in a cluster and most of the points in the inner circle into another cluster, with a few errors or inaccuracies as compared to the expected clustering.

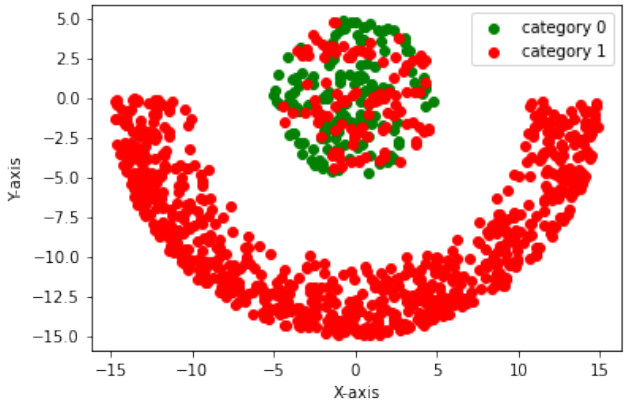


(a)

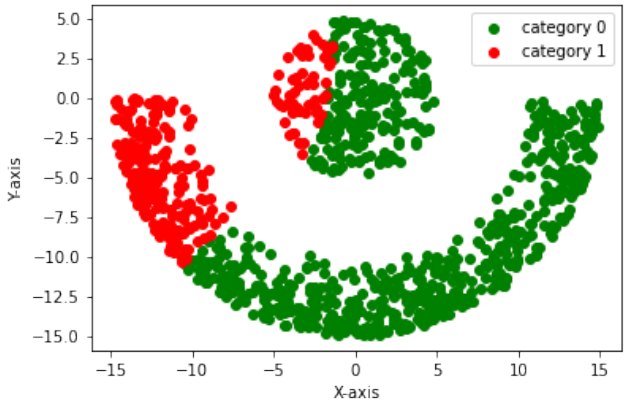
(b)

(c)

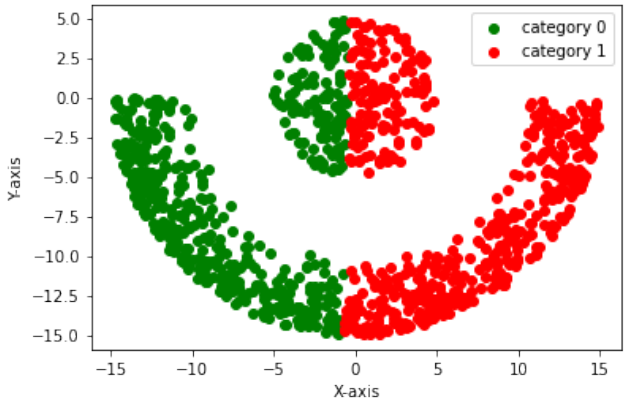
Figure 23: Spectral Clustering using polynomial kernel (a)d=2,(b)d=3,(c)d=4 respectively



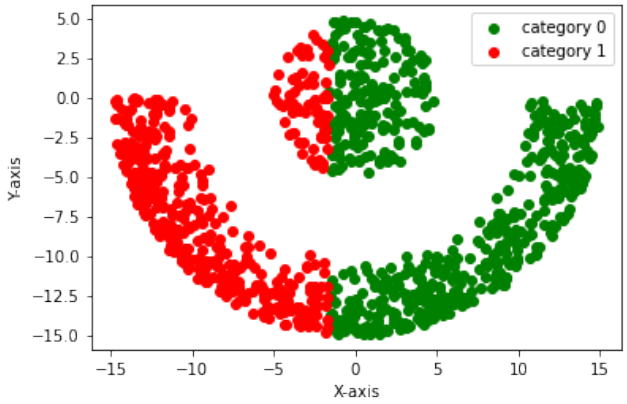
(a)



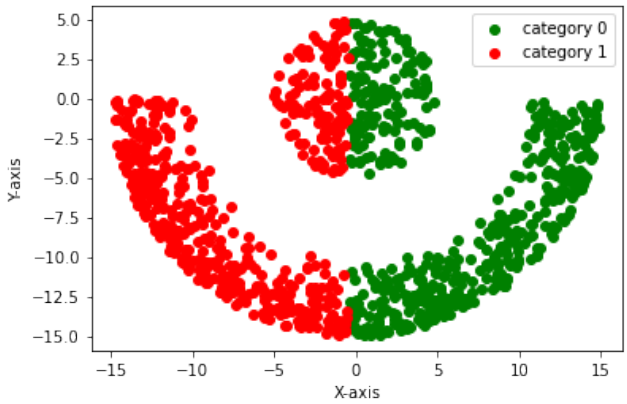
(b)



(c)



(d)

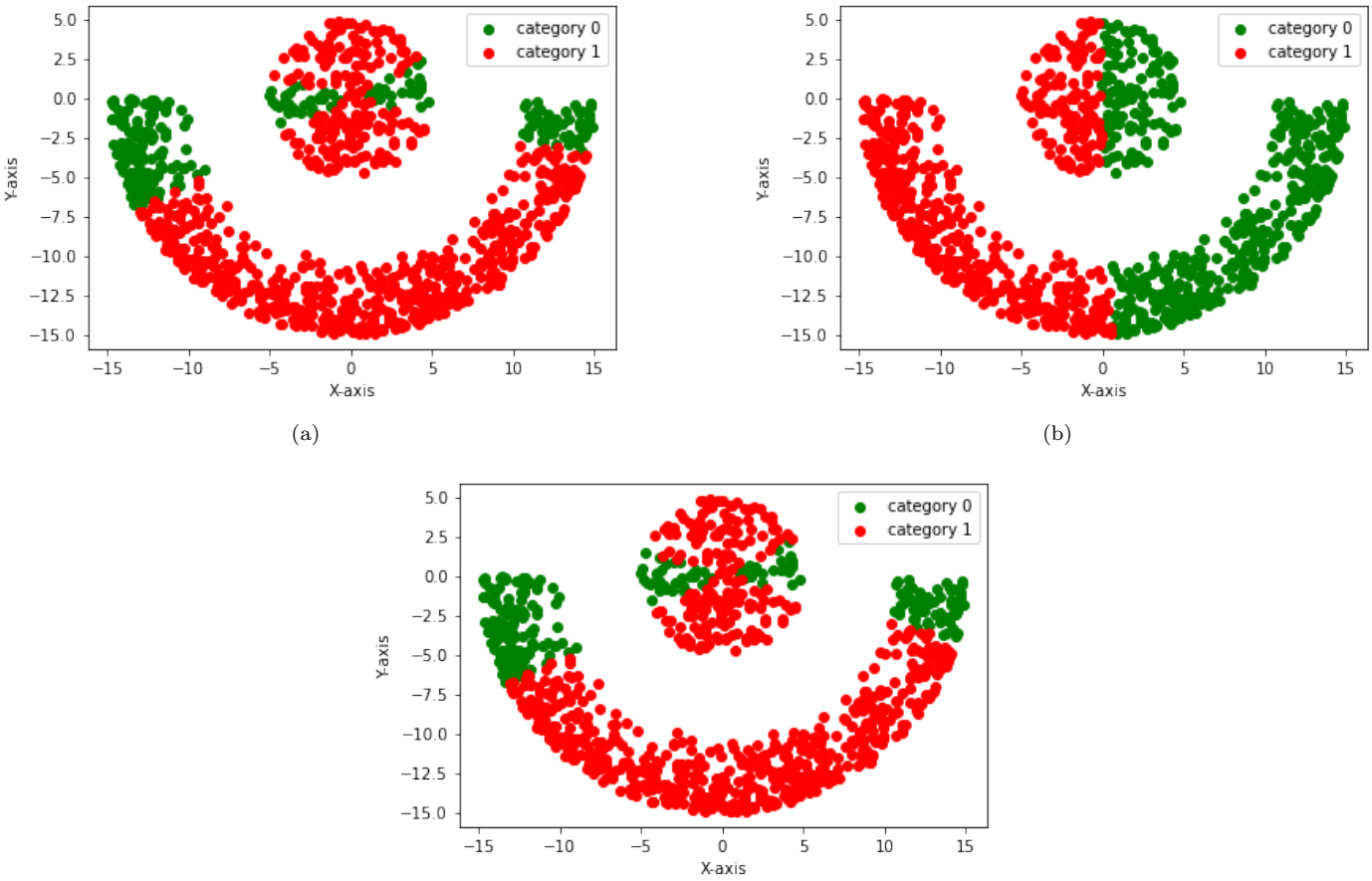


(e)

Figure 24: Spectral Clustering using exponential kernel (a)σ^2=0.7,(b)σ^2=1,(c)σ^2=10,(d)σ^2=64,(e)σ^2=100 respectively

2.4 (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

From Figures (25) and (26), this clustering algorithm performs as poorly as spectral clustering in the case of polynomial kernel. But this is more of a coincidence. Such clustering means we are splitting the eigenmatrix H along the line y=x.
This should be understood and realized as just another method to retrieve or obtain the Z matrix(assignment matrix) from the original H matrix .
The accuracy of the clustering approach is comparable to that of spectral clustering, which requires running k-means for each row.
However, the current method achieved similarly results with a straightforward, direct assignment without the need for additional k-means iterations for each row.
This can be attributed to the resemblance to the $ZL^{0.5}$ matrix.
In the exponential case, it is very clear that the spectral clustering performs better than the algorithm followed in this question
On the whole, such an arbitrary method to extract the Z matrix from the H matrix is not as good as spectral clustering.

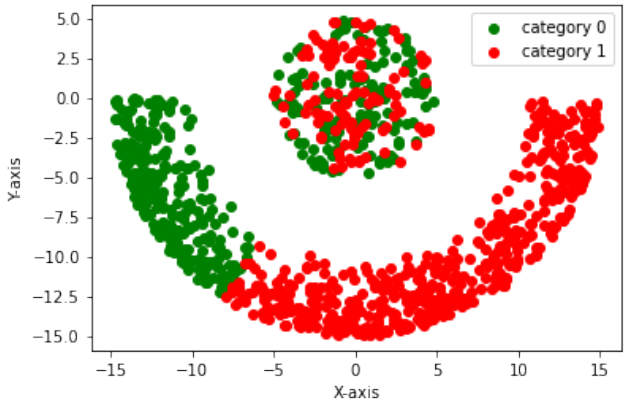


(a)

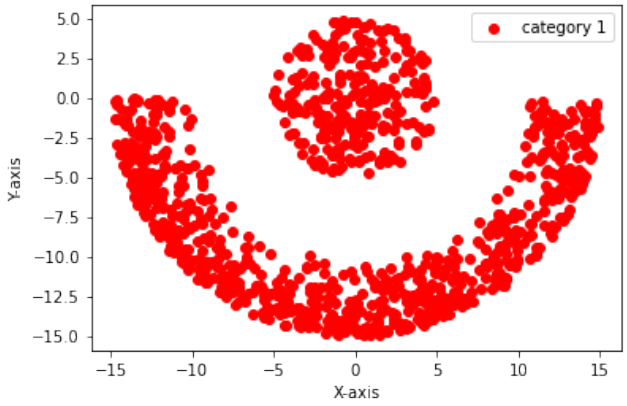
(b)

(c)

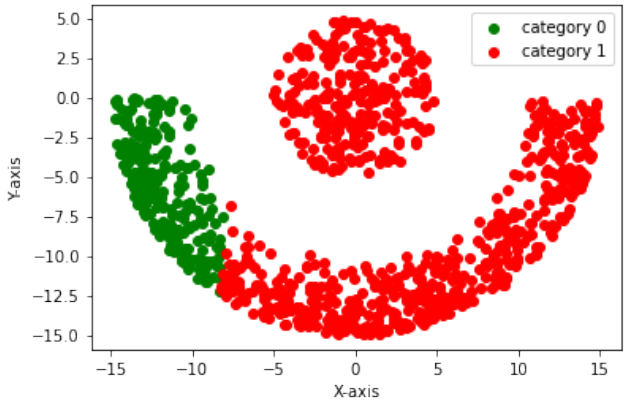
Figure 25: Question 4 - Clustering using polynomial kernel (a)d=2,(b)d=3,(c)d=4 respectively



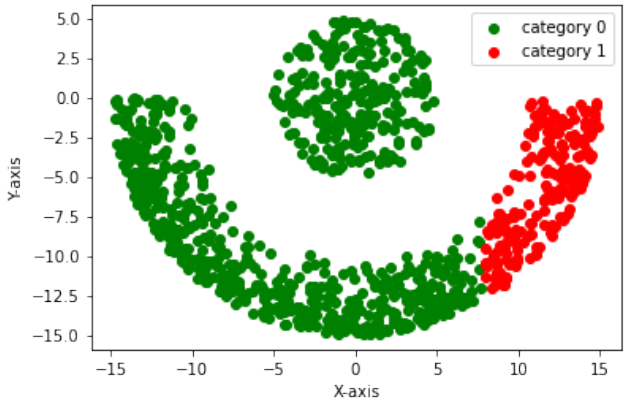
(a)



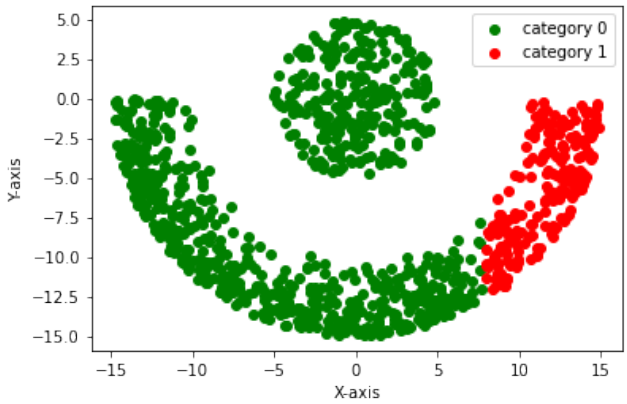
(b)



(c)



(d)



(e)

Figure 26: Question 4 - using exponential kernel (a)σ^2=0.7,(b)σ^2=1,(c)σ^2=10,(d)σ^2=64,(e)σ^2=100 respectively