CS5691: Pattern Recognition and Machine Learning
Assignment 1
Submitted by: Ganesh S (ME20B070)

1 Question 1

1.1 PCA

The MNIST dataset was downloaded from Hugging face. The "train" split of the dataset containing 60,000 images of handwritten digits was used. Each image was of resolution 28×28. Each image was flattened into an array of length 784.

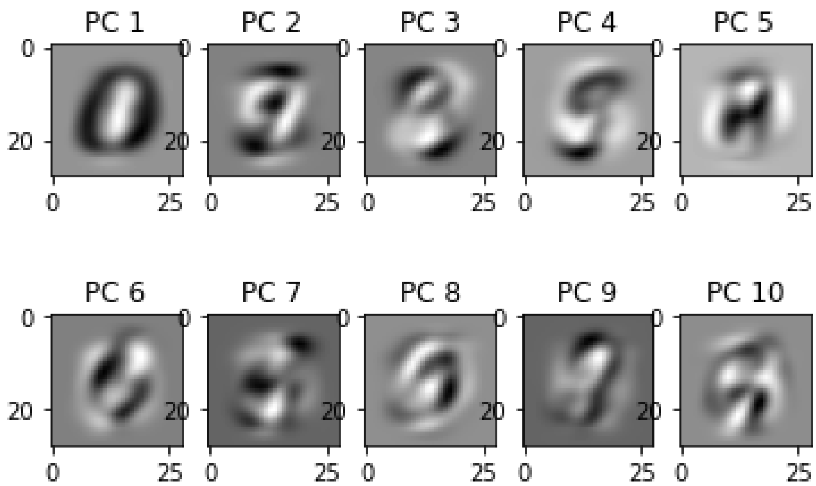


Figure 1.1 Images of the top 10 principal components

The eigenvalues and eigenvectors of the Covariance matrix are calculated. The ratio of the variance explained by each principal component is given by:

$$r_i = \frac{\lambda_i}{S}; \qquad S = \sum_{i=1}^{n} \lambda_i$$

where λ_i is the eigenvalue of the i^{th} most important principal component. Eigenvectors are sorted in descending order based on their eigenvalue which gives us the most important principal components in order.

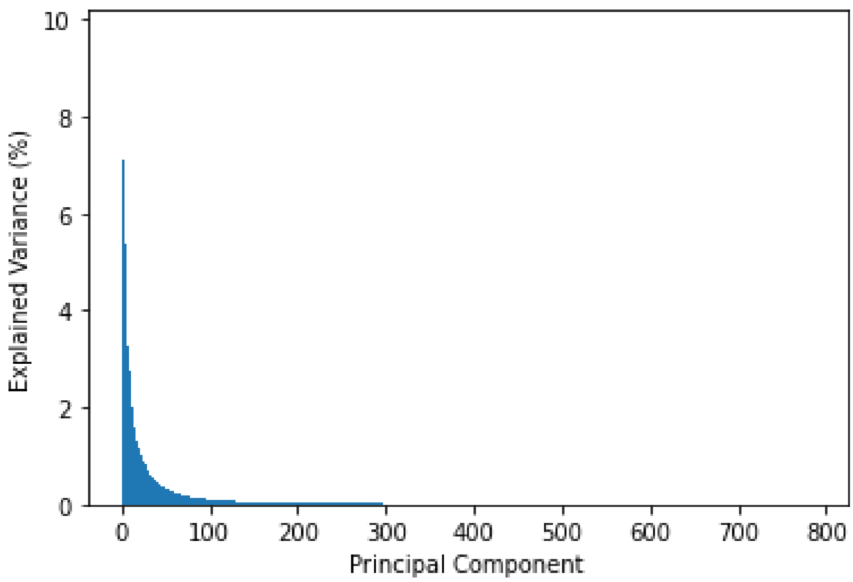


Figure 1.2 Explained variance (%) of each principal component

1.2 Reconstruction

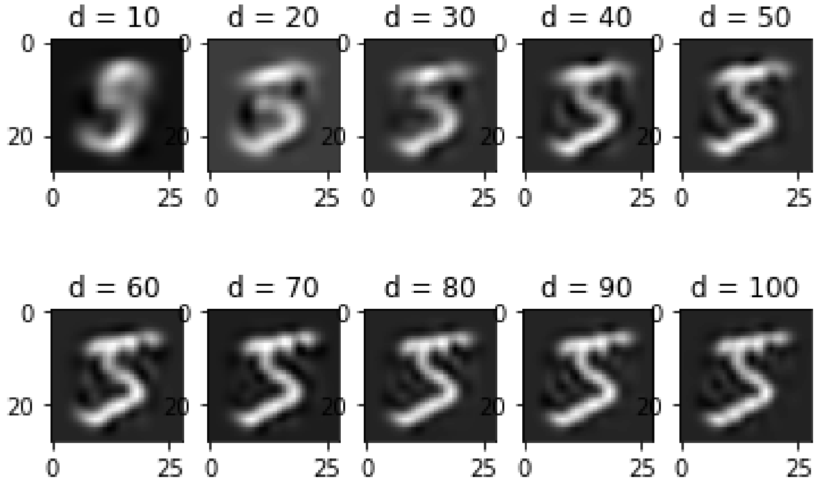


Figure 1.3 Reconstruction of image in different dimensions(d)

Reconstruction using a low-dimensional representation gives a blurry version of the original image. For downstream tasks where we need to classify images correctly, $d = 60$ or $d = 70$ is appropriate. This compresses our dataset significantly while retaining the uniqueness of each digit.

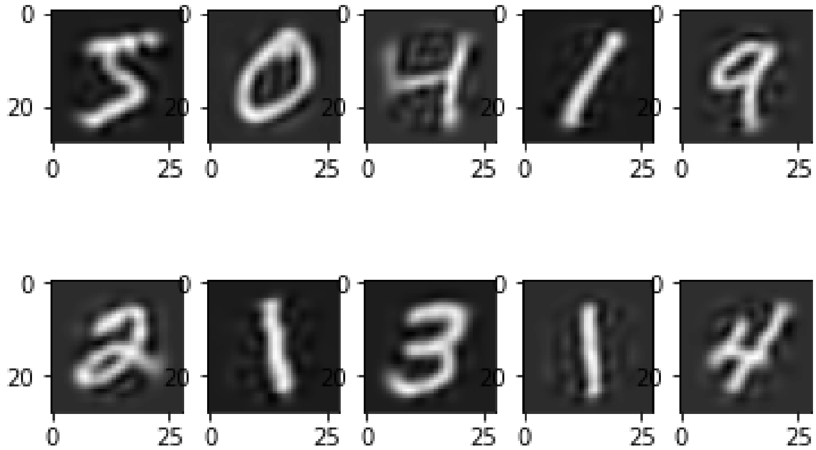


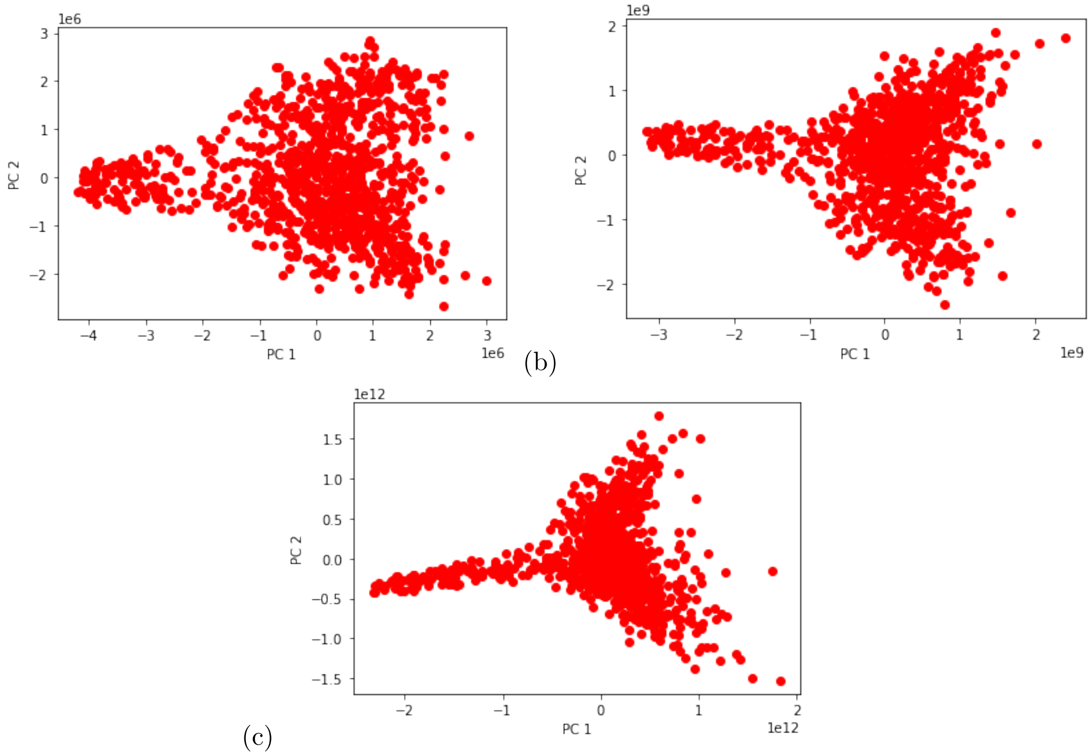
Figure 1.4 Reconstruction of different digits in 70-dimensions

1.3 Kernel PCA

The Kernel matrix is of shape : $n \times n$, where n is the number of datapoints. Since the number of datapoints is too large in this dataset (60,000), we restrict our dataset to 1000 points.

1.3.1 Polynomial kernel

$$\kappa(x, y) = (1 + x^T y)^d \text{ for d = 2,3,4}$$

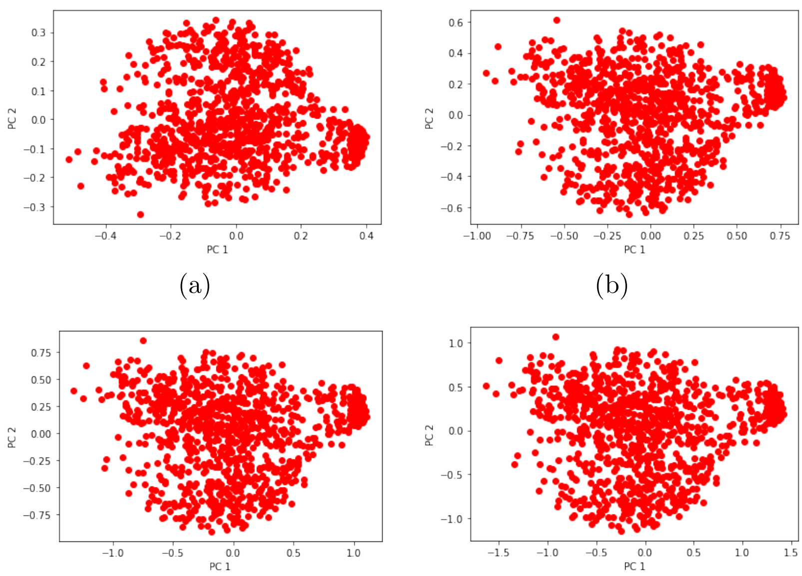


(a) (b) (c)

Figure 1: (a) PC for d = 2 (b) PC for d = 3 (c) PC for d = 4

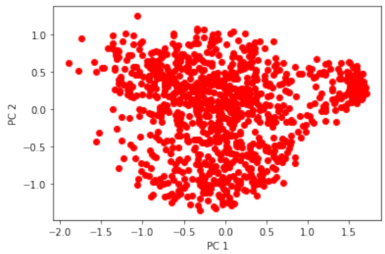
1.3.2 RBF Kernel

The RBF Kernel is given by: $\kappa(x, y) = exp\frac{-(x-y)^T(x-y)}{2\sigma^2}$ for $\sigma = 0.1, 0.2, ...1$

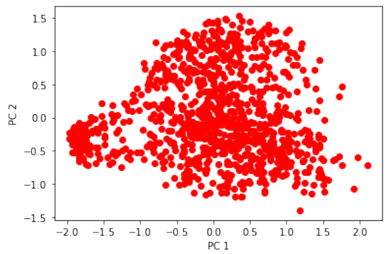


(a) (b)

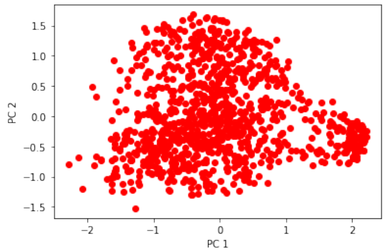
(c) (d)



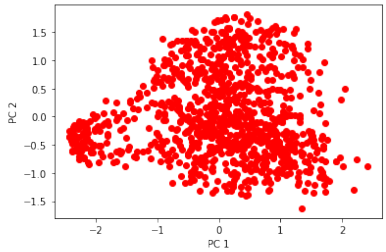
(e)



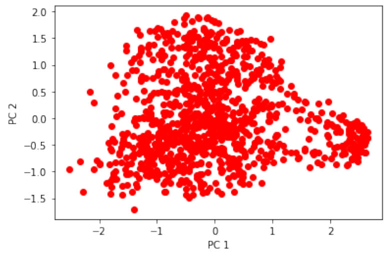
(f)



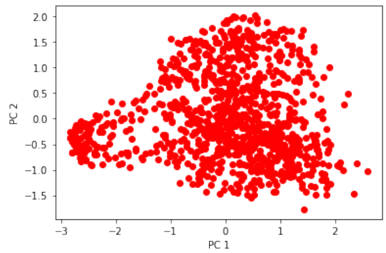
(g)



(h)



(i)



(j)

Figure 1.6 Projection for $\sigma =$ (a) 0.1 (b) 0.2 (c) 0.3 (d) 0.4 (e) 0.5 (f) 0.6 (g) 0.7 (h) 0.8 (i) 0.9 (j) 1.0

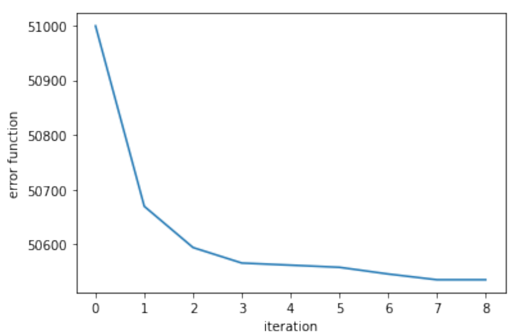
1.4 Choice of kernel

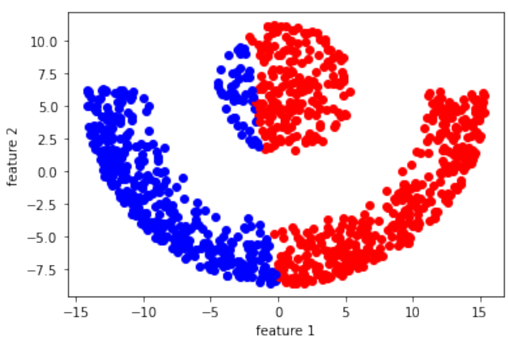
For image classification purpose, normal PCA is sufficient for dimensionality reduction. For this dataset, to capture the non-linearity of the images, RBF kernel would be appropriate, since it maps the datapoints to an infinite dimensional space.

2 Question 2

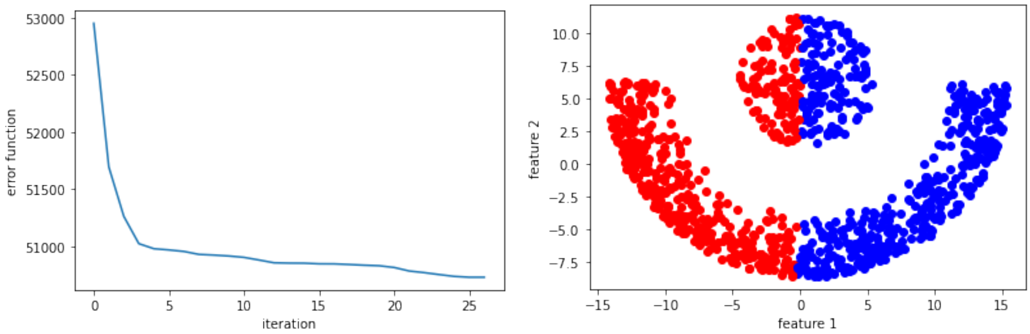
2.1 Clustering with k = 2

The following results were obtained after mean-centering the data
Means $\mu_1 = (-6, -2.4)\mu_2 = (-10.034, -2.131)$

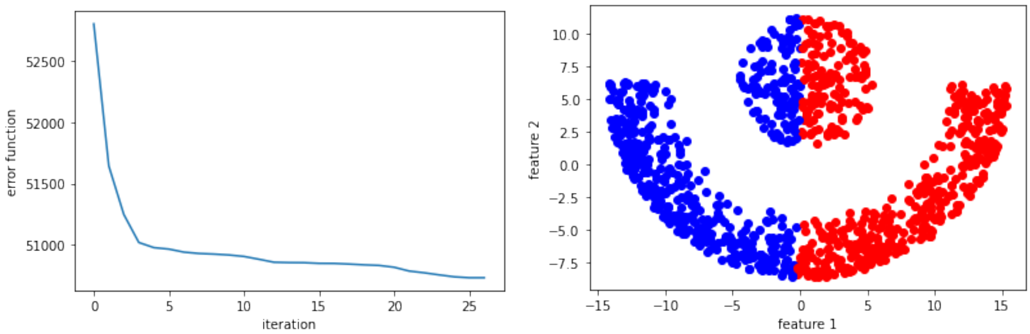




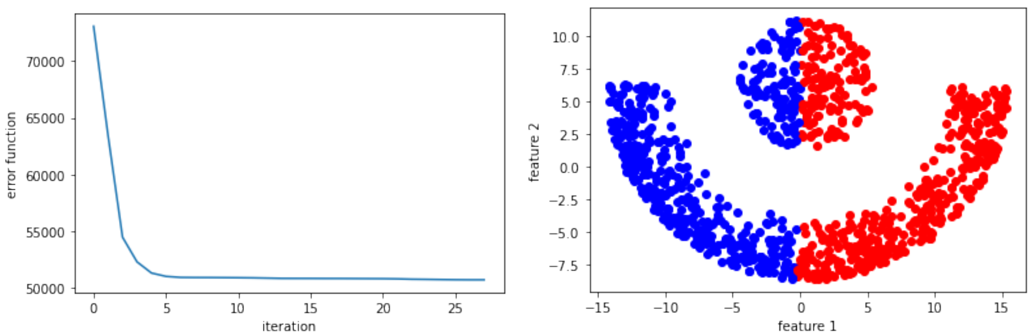
Means $\mu_1 = (0.206, 4.804)\mu_2 = (12.097, -0.1986)$



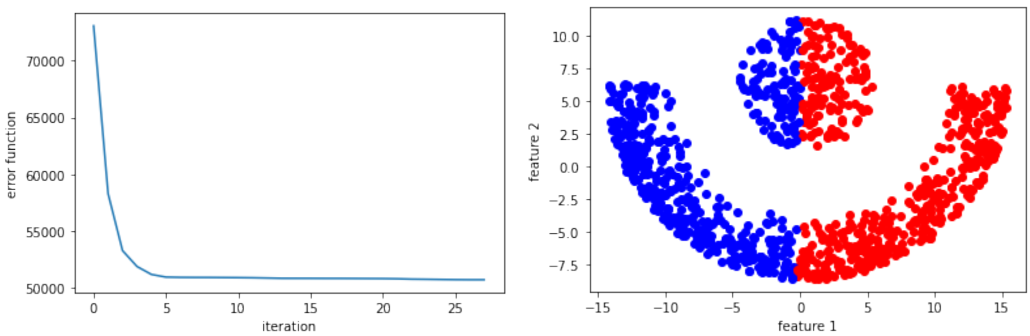
Means $\mu_1 = (15.004, 4.289)\mu_2 = (-2.847, 9.547)$



Means $\mu_1 = (-1.973, -6.982)\mu_2 = (-3.882, 5.869)$

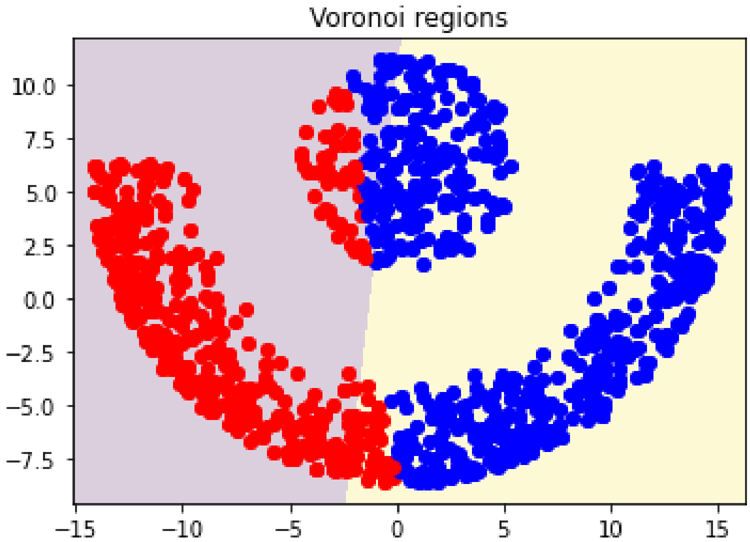


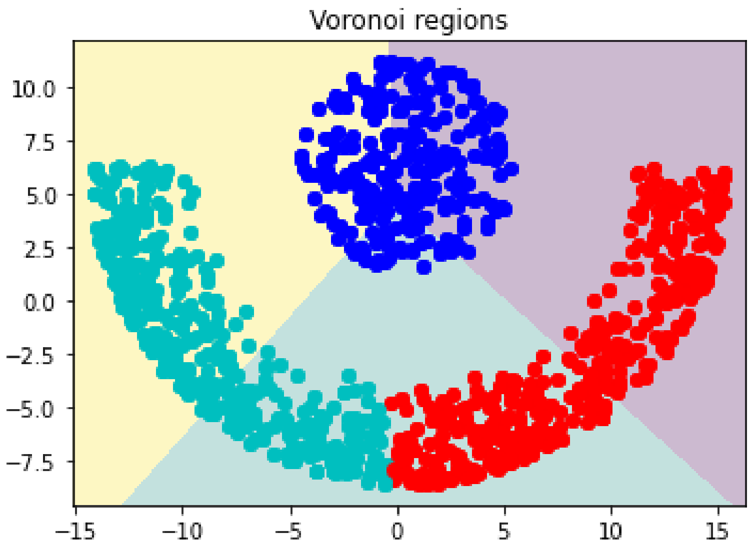
Means $\mu_1 = (-7.423, -6.192)\mu_2 = (-8.382, 2.222)$

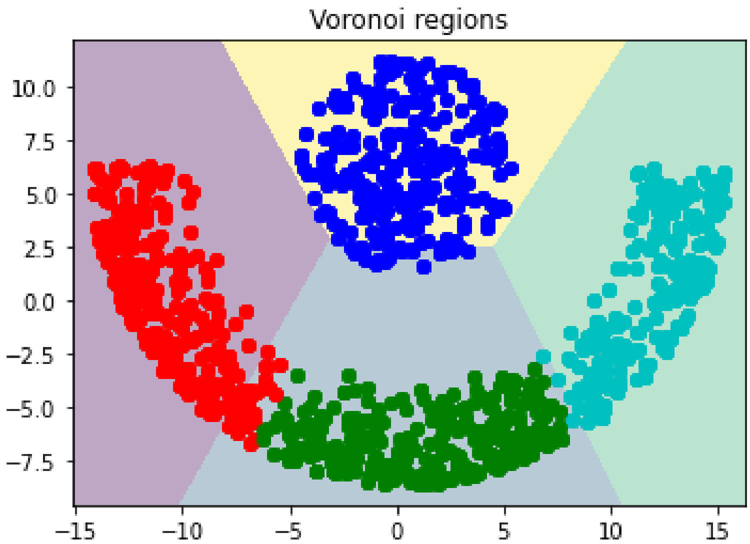


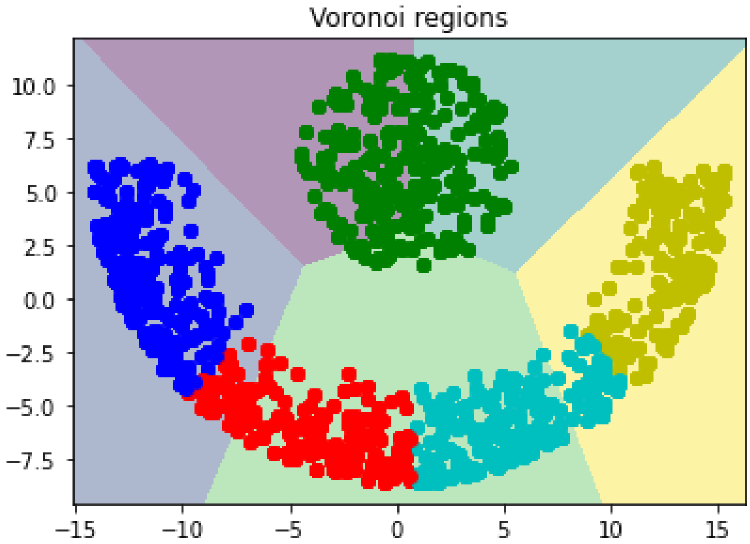
2.2 For K=2,3,4 and 5

The following are the voronoi regions for K = 2,3,4 and 5


Voronoi regions
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−15
−10
−5
0
5
10
15

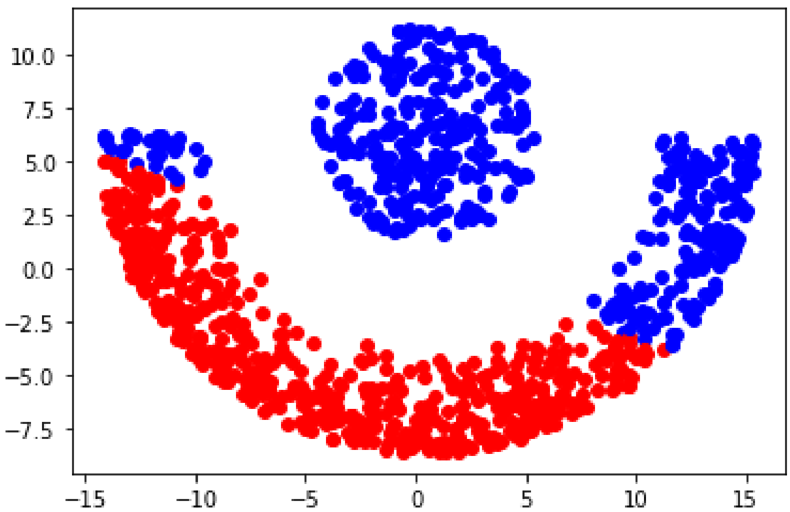

Voronoi regions
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−15
−10
−5
0
5
10
15


Voronoi regions
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−15
−10
−5
0
5
10
15



2.3 Spectral clustering

The RBF Kernel is chosen for this dataset and spectral clustering is run for k = 2. This is chosen due to it's ability to represent data in an infinite dimensional space, thus we would be able to cluster the datapoints efficently.

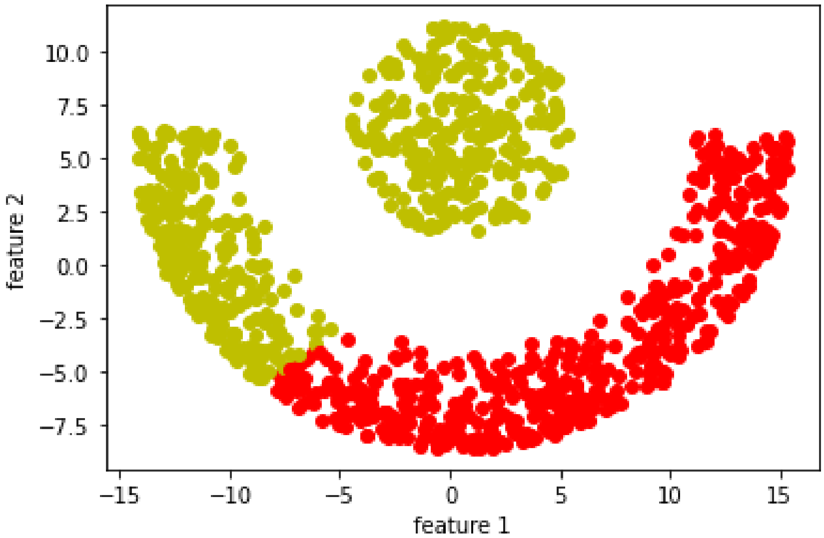


2.4 Alternate method

We assign a point i to cluster l when,

$$l = \arg\max_{j=1,..k} v_i^j, \tag{1}$$

where v_j is the eigenvector of Kernel matrix associated with the j^th largest eigenvalue. We get the following cluster.



This cluster assignment is very similar to running Lloyd's algorithm on the row-normalized H matrix, where columns of H are the top k eigenvectors of the Kernel matrix. This is because when $H = ZL^{\frac{1}{2}}$, where Z is the cluster assignment matrix, each row of H is of the form $H[i][j] = 1$ if datapoint i belongs to j^{th} cluster else 0. Now finding cluster from H would give a result equivalent to getting the index j where H[i][j] = 1. This is precisely what this alternate method does. Thus we get a result similar to Lloyd's algorithm.