Anirudh Rao
BE21B004

DA5400 – Foundations of Machine Learning

Assignment 3

1(i)

The MNIST dataset was downloaded from Hugging Face. For each digit, 100 images were randomly selected, creating a dataset of 1000 images. The images were converted to NumPy arrays using the Python Imaging Library (PIL). As each image had 28 × 28 pixels, each array had 784 elements.

Overall, the dataset being considered had 784 rows and 1000 columns when stored in matrix form. Let this be denoted by $X \in \mathbb{R}^{d \times n}$ with $d = 784$ and $n = 1000$.

Before performing principal component analysis, each row of X was shifted to have mean 0. Let the mean-shifted matrix be denoted by X'.

$$\mu_{X,i} = \frac{1}{n}\sum_{j=1}^{n} X(i,j) \ \forall\ i \in \{1, \dots, 784\}$$

$$X'(i,j) = X(i,j) - \mu_{X,i} \ \forall\ i \in \{1, \dots, 784\}, \forall\ j \in \{1, \dots, 1000\}$$

It can be verified that

$$\mu_{X',i} = \frac{1}{n}\sum_{j=1}^{n} X'(i,j) = \frac{1}{n}\sum_{j=1}^{n}\left(X(i,j) - \mu_{X,i}\right) = 0 \ \forall\ i \in \{1, \dots, 784\}$$

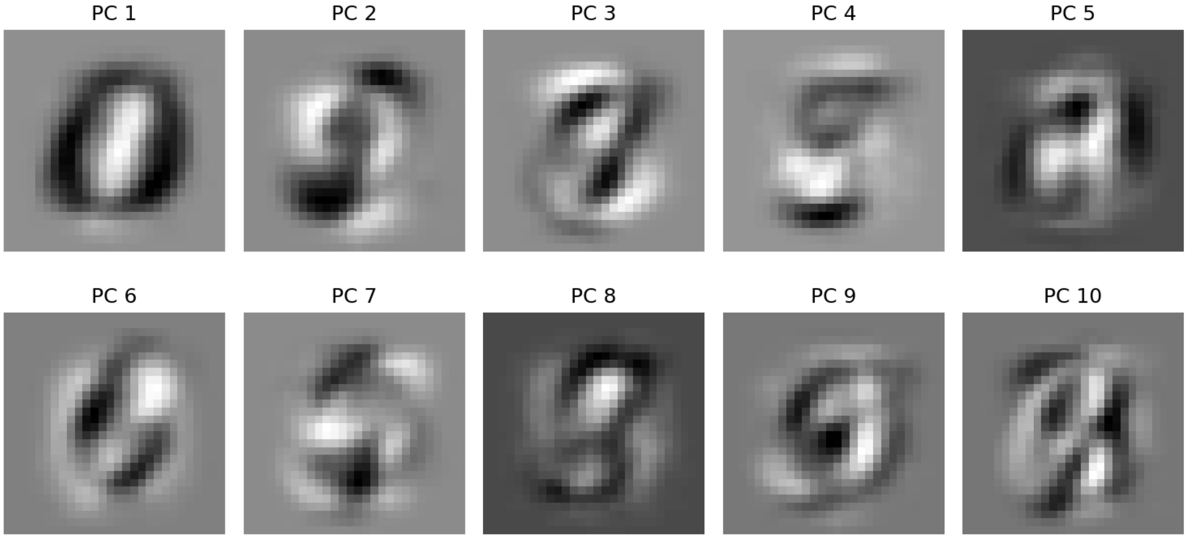
One image for each digit was visualised using the mean-shifted data.

The covariance matrix ($C \in \mathbb{R}^{d \times d}$ with $d = 784$) of the data was then computed

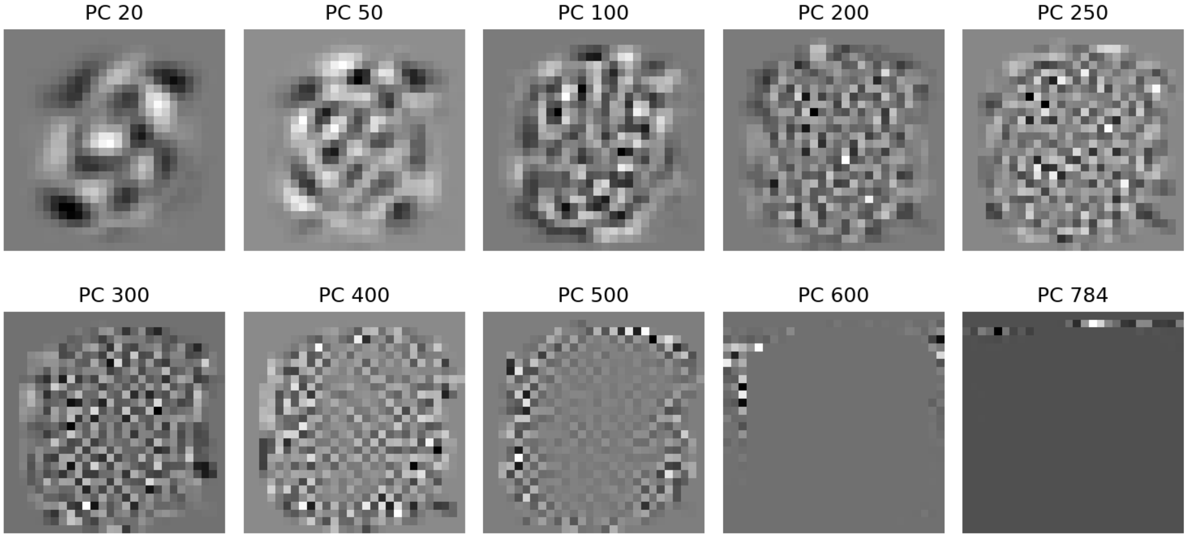
$$C = \frac{1}{n} X' X'^{T}$$

Next, the eigenvalues and eigenvectors of the covariance matrix were computed. The eigenvectors corresponding to the k largest eigenvalues form the first k principal components. There are 784 eigenvalues in total.

The first 10 principal components were visualised as images. It can be seen that different principal components loosely relate to different "strokes" in the digits.



The principal components with smaller eigenvalues were then visualised as images. These seem to carry lesser information about the dataset.

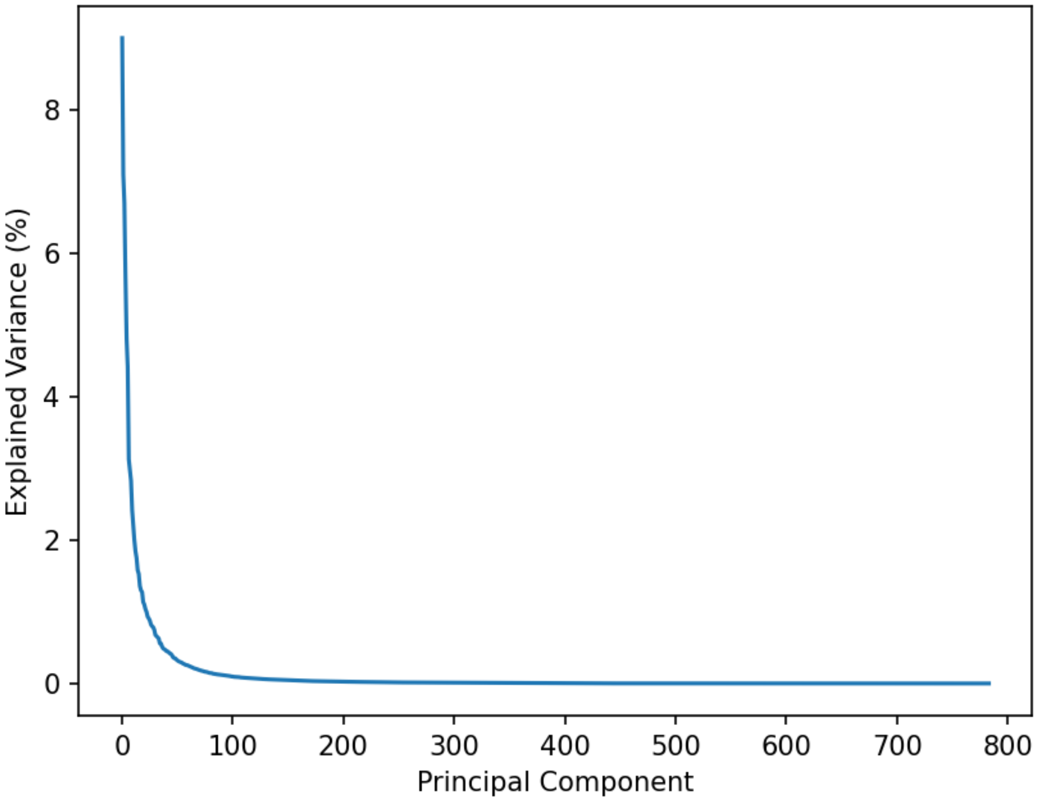


The variance explained by each principal component was then computed as follows

$$\text{Percentage of explained variance by PC } k = \frac{\lambda_k}{\sum_{i=1}^{d} \lambda_i} \times 100$$

λ_i denotes the i^{th} largest eigenvalue of C.

PC 1 explains about 9% of the variance in the dataset, PC 2 explains 7.1%, and PC 3 explains 6.7%. Subsequent PCs explain a lower percentage of the variance in the dataset. This was visualised as a plot.



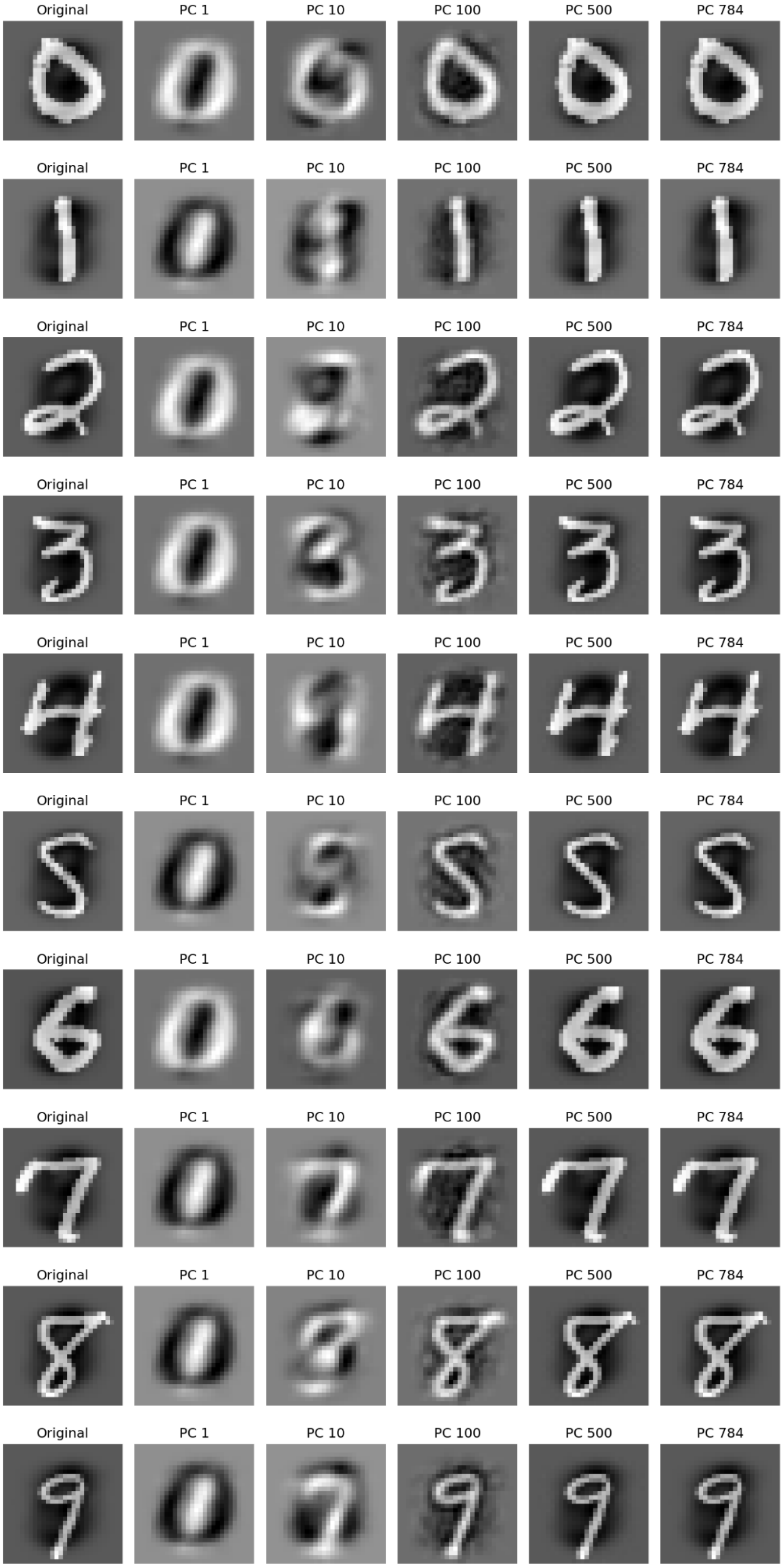
1(ii)

The principal components can be used to reconstruct the original images. To reconstruct the image x using the first k principal components, we compute

$$x' = \sum_{i=1}^{k} (x^T w_i) w_i$$

Here, w_i is the eigenvector corresponding to the i^{th} largest eigenvalue of C.

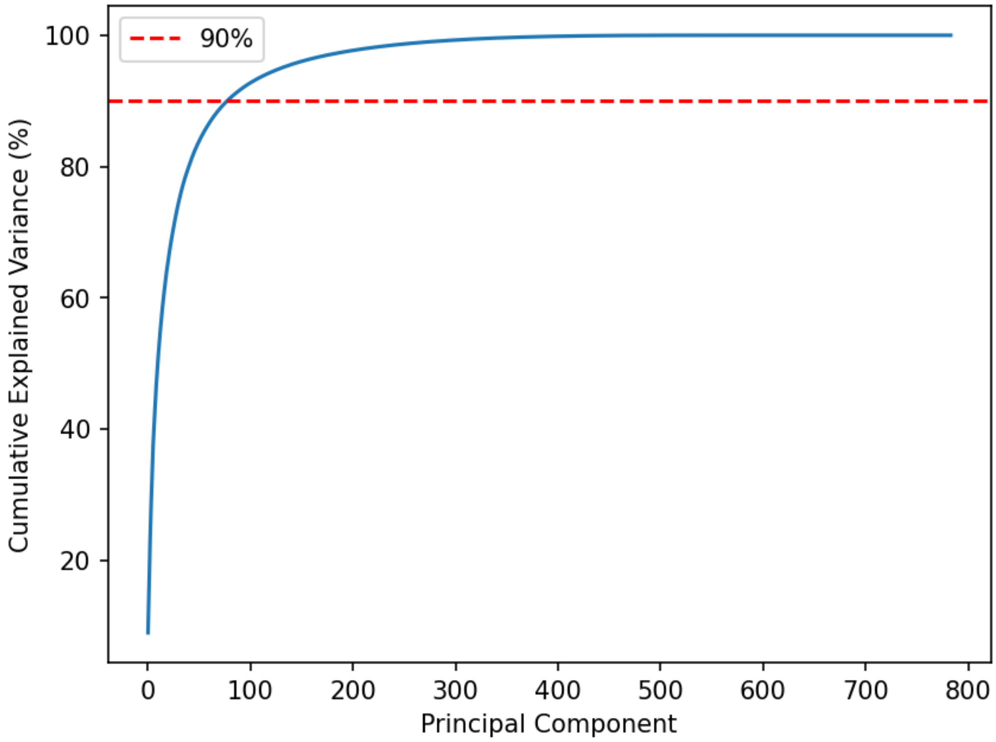
The digits were reconstructed using different values of k. As k increased, the resemblance of the reconstructed image to the original image also increased.



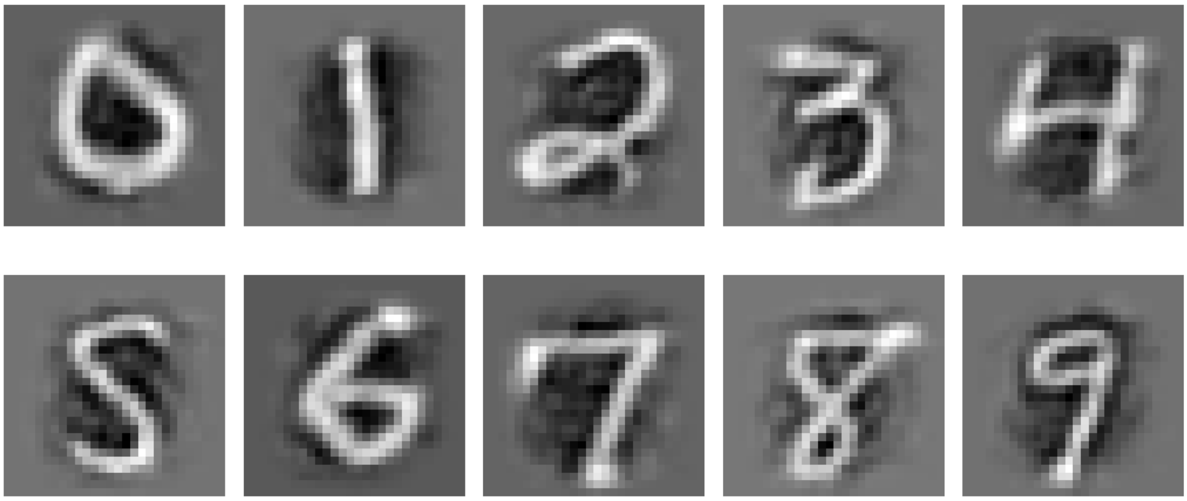
To determine the ideal number of PCs to consider, we can look at the cumulative explained variance.

$$\text{Cumulative explained variance upto } PC\ k = \sum_{i=1}^{k} \frac{\lambda_i}{\sum_{j=1}^{d} \lambda_j} \times 100$$

We can set some threshold for this, say 90% of the total variance, to determine up to which PC to consider while reconstructing the images. For this dataset, to capture 90% of the variance, it was found that considering the first 78 PCs was necessary.



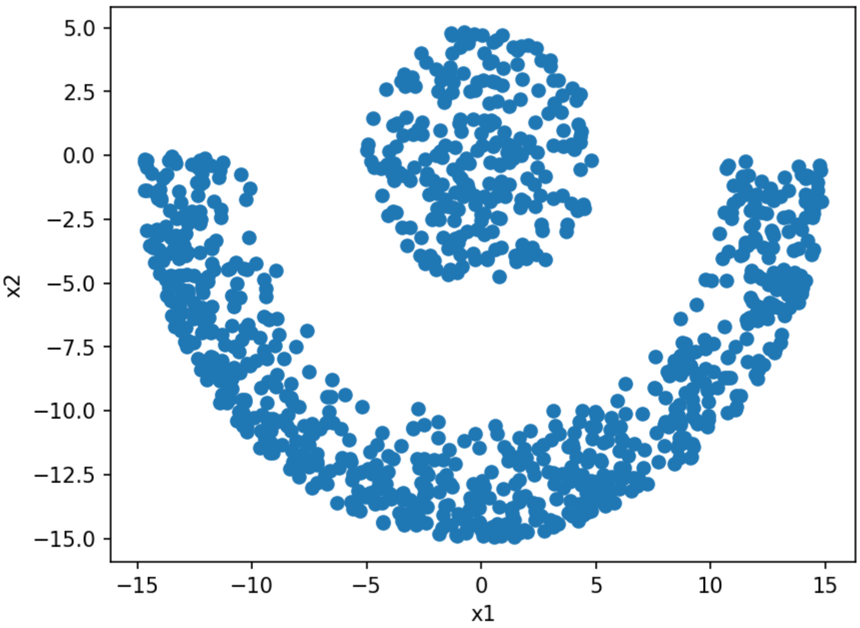
The digits were visualised using the first 78 PCs for reconstruction.



This could adequately capture the features of the various digits. Thus, using the first 78 PCs is sufficient for reconstruction and for use in downstream classification tasks.

2(i)

The given data, consisting of 1000 points in $\mathbb{R}^2$, was plotted.



Lloyd's algorithm was used to cluster the given data into k clusters. Each of the n points, $\{x_1, \ldots, x_n\}$ was assigned to a cluster, denoted by $\{z_1, \ldots, z_n\}$ with $z_i \in \{1, \ldots, k\}$. The algorithm can be stated as follows:

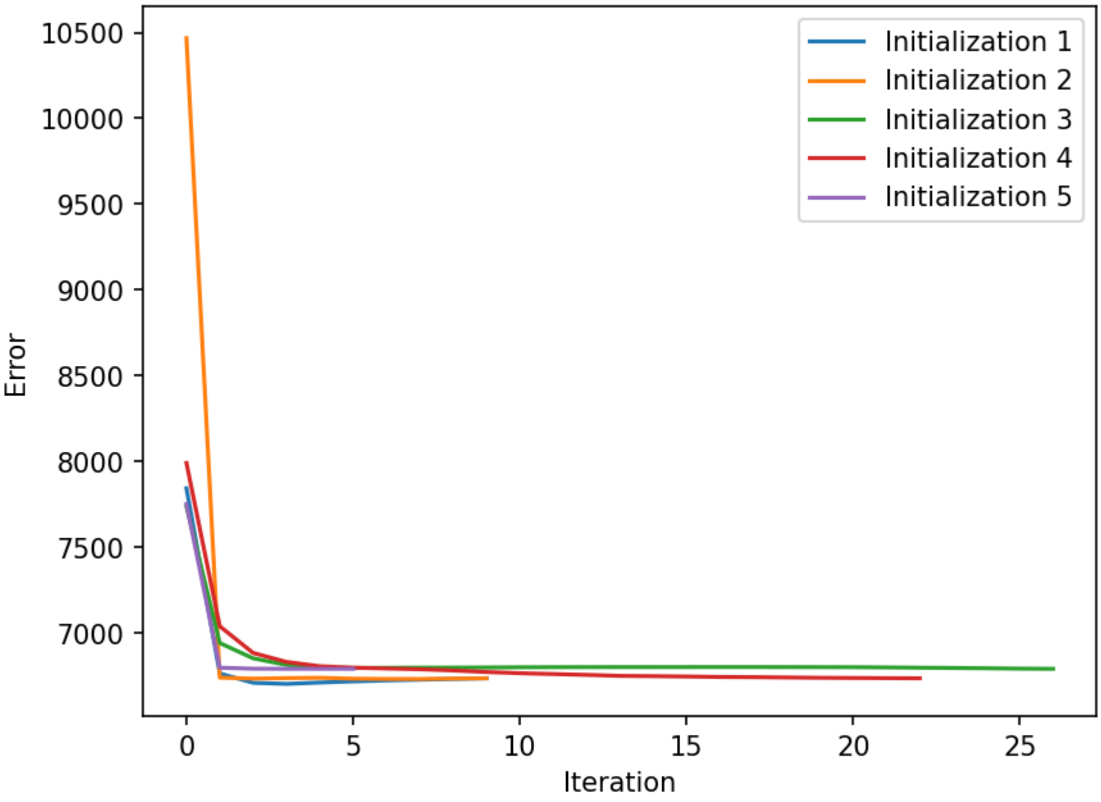
```
Initialize $z_1^0, \ldots, z_n^0 \in \{1, \ldots, k\}$

Until convergence

    Compute new cluster centroids $\mu_k^t = \frac{\sum_{i=1}^{n} x_i \mathbb{1}(z_i^t = k)}{\sum_{i=1}^{n} \mathbb{1}(z_i^t = k)}$

    Reassign $z_i^{t+1} = \underset{k \in \{1,\ldots,k\}}{\operatorname{argmin}} \|x_i - \mu_k^t\|^2$
```

The initialization was performed by randomly selecting k points from $\{x_1, \ldots, x_n\}$ to be the initial cluster centroids. The remaining points were assigned to a cluster based on which centroid they were closest to.

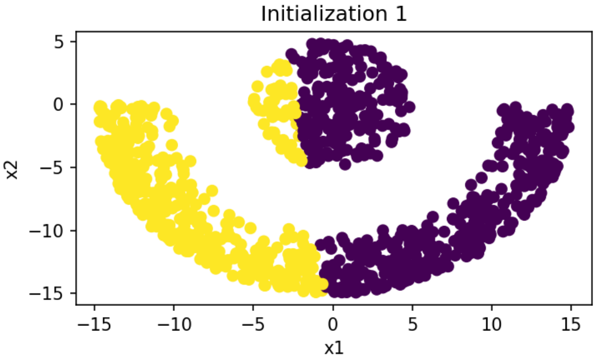
The error associated with a cluster assignment can be computed as

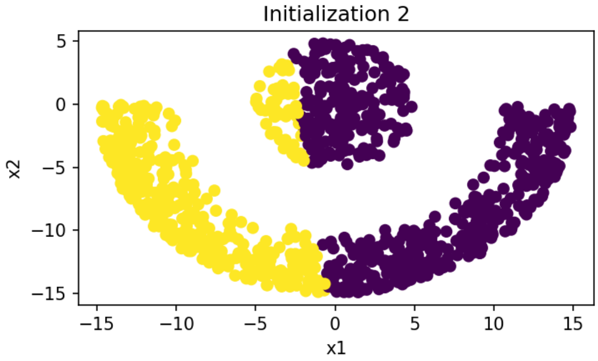
$$\text{Error} = \sum_{i=1}^{n} \left\| x_i - \mu_{z_i} \right\|^2$$

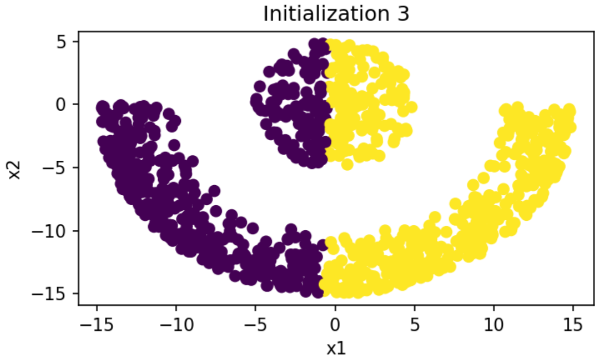
When $k = 2$ was used with 5 different random initializations, the error for each initialization was plotted for every iteration of Lloyd's algorithm. Some initializations take longer to converge and some converge to lower errors.

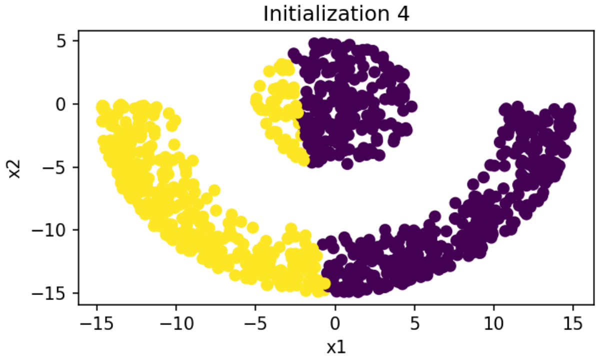


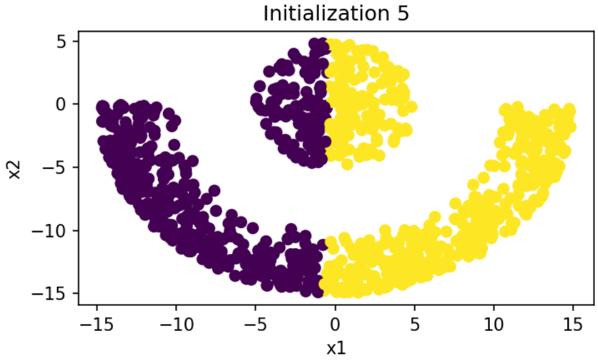
The cluster assignment (after convergence) for each random initialization was also plotted.




Initialization 2


Initialization 3

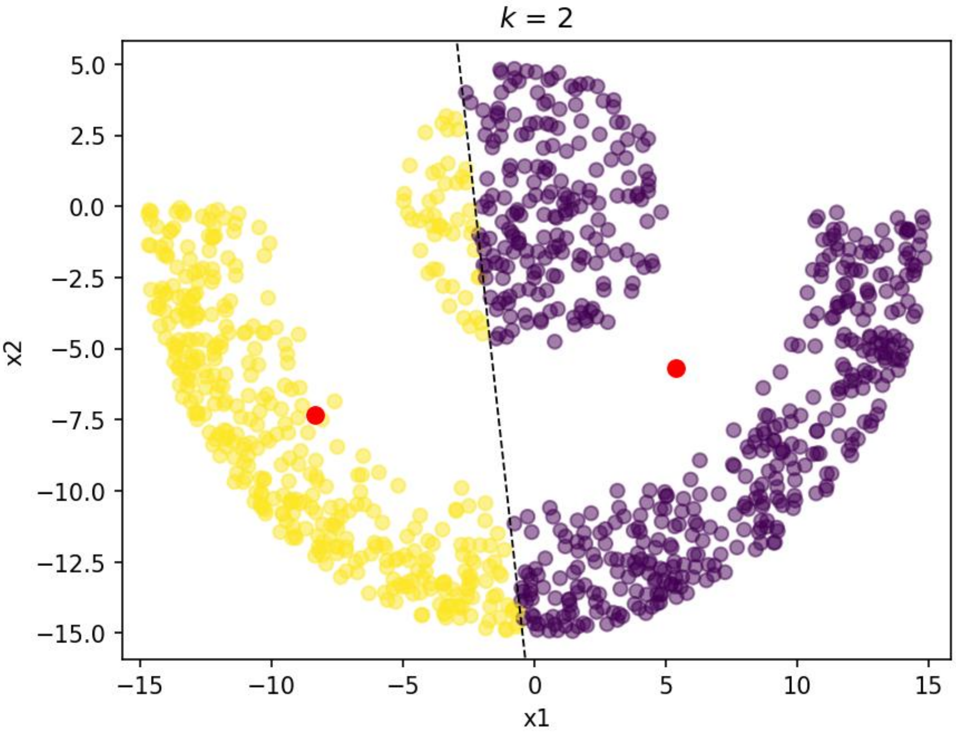

Initialization 4

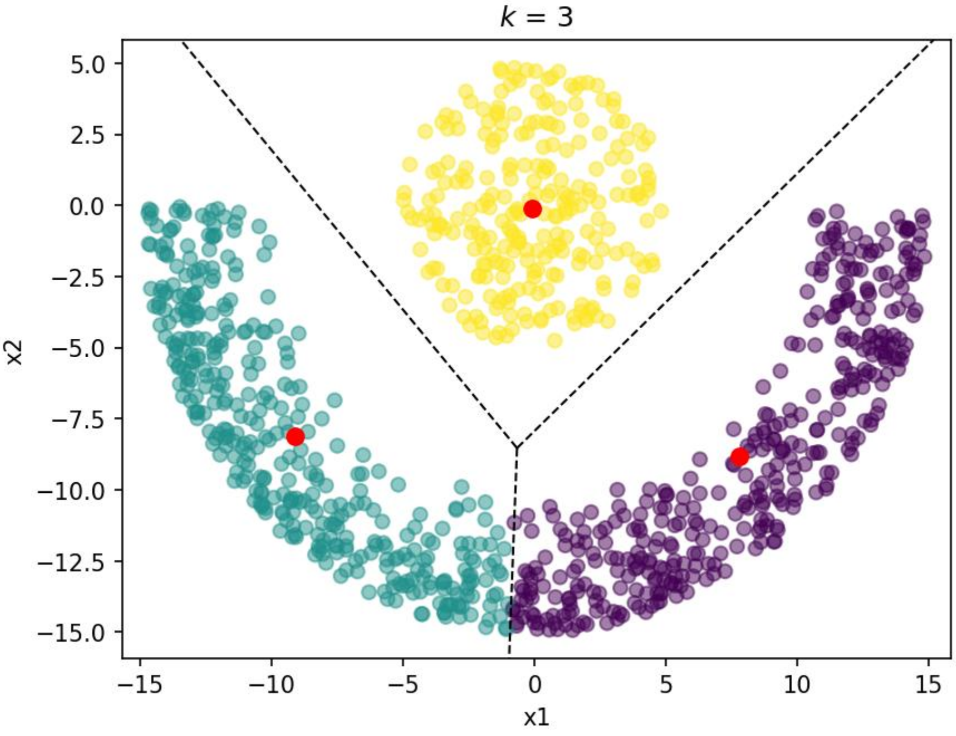

Initialization 5

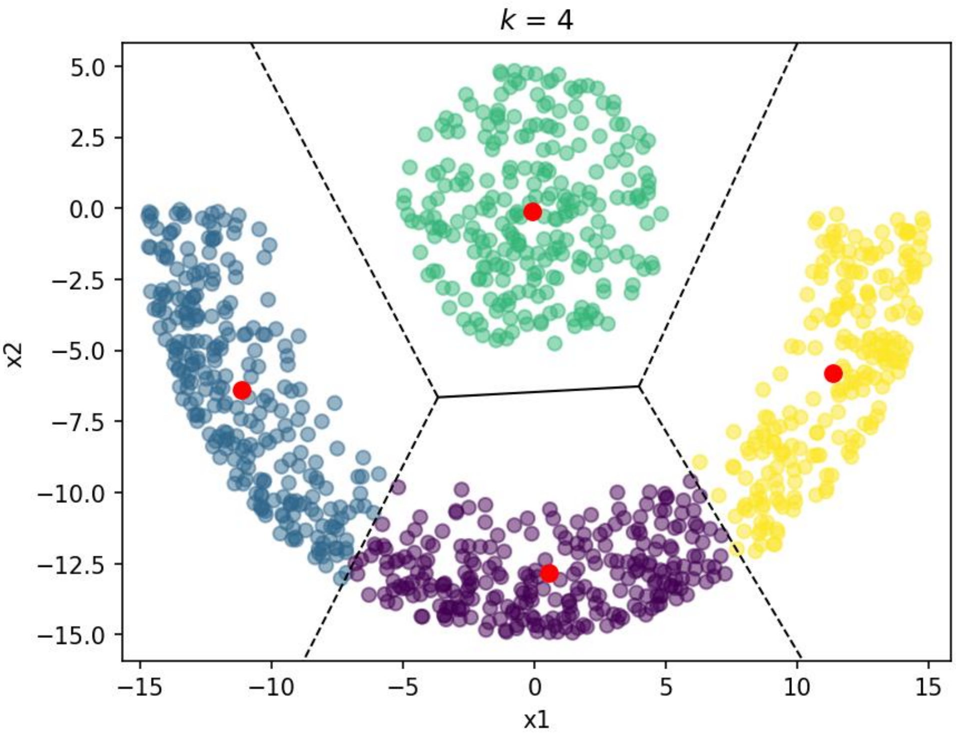
Clearly, the cluster assignments are very sensitive to the initialization.

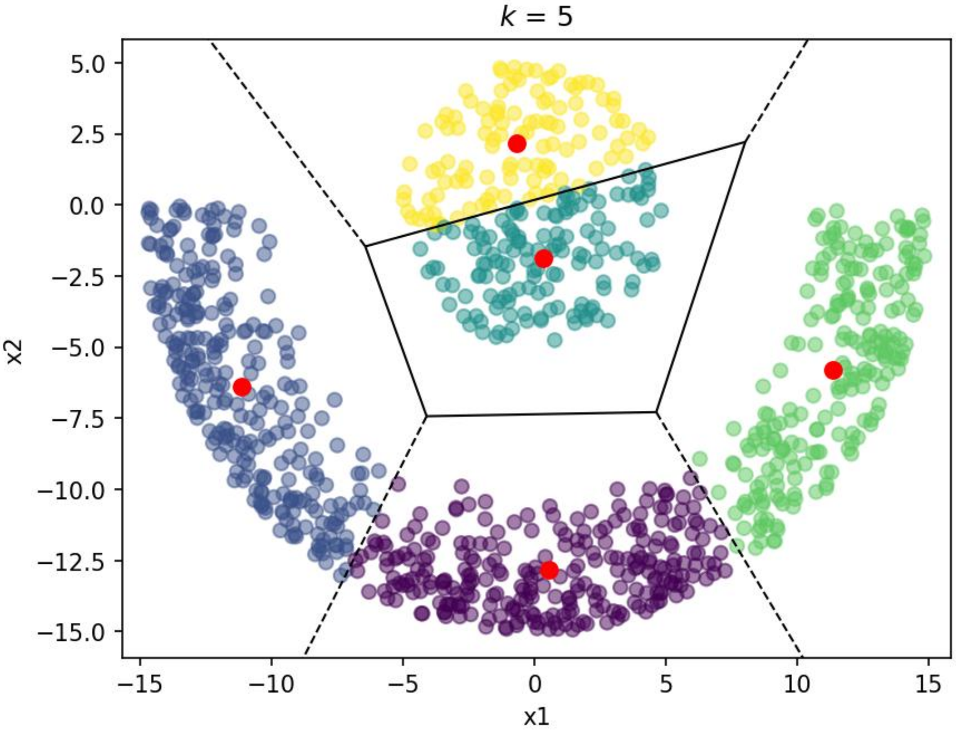
2(ii)

For $k = 2, 3, 4, 5$, the Voronoi regions associated with the final cluster centroids were plotted after running Lloyd's algorithm with a random initialization.









The centroids corresponding to the clusters created are marked in red.

2(iii)

From the results obtained, it is clear that Lloyd's algorithm is NOT a good way to cluster the dataset. Due to the inherent non-linearity in the data, it would be more useful to use a kernel before performing the clustering. For instance, the radial basis kernel could be used to compute the kernel matrix for the data. Spectral clustering can be applied on the top k eigenvectors of the kernel matrix. The algorithm can be stated as follows:

> Given data $\{x_1, \dots, x_n\} \in \mathbb{R}^d$, compute the kernel matrix $K \in \mathbb{R}^{n \times n}$.
>
> Using the k largest eigenvectors of K, compute H^*.
>
> Normalize the rows of H^* and run Lloyd's algorithm assuming each row of the normalized H^* is the data.