Report - Assignment 3

Nikshay Jain - MM21B044

November 2024

1 Introduction

The project demonstrating unsupervised learning contains:

- Code files: mm21b044.py and mm21b044.ipynb (with the same contents)
- Code PDF: `mm21b044_code.pdf` (to refer code conveniently)
- Personal info doc: Details.txt
- Report: Report.pdf

2 Q1: Dataset

the dataset used here is MNIST digits dataset which is sampled for 1000 images (100 for each number) using the code written initially. Each image is a grayscale one with dimentions 28*28 pixels,

Figure 1: MNIST Digits Dataset

3 Q1: Part 1(i)

We apply PCA on the sampled images to get the eigen values and eigenvectors by the function named **pca()**. It shows that the images have 784 non zero eigen values, i.e., no feature of the image array is linearly dependent on others.

These eigenvalues and eigenvectors are then passed to `analyse_pca()` to calculate the number of Principal components needed to represent 95% of the information in the images, which comes out to be 130 as shown in the picture below at the elbow of the graph.

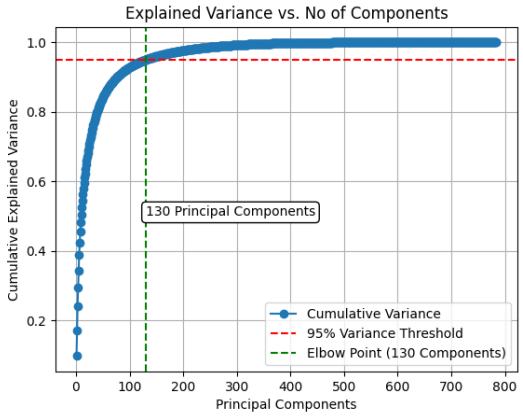


Figure 2: Variance vs No of eigenvalues

We then visualize the top 130 principal components individually as follows:

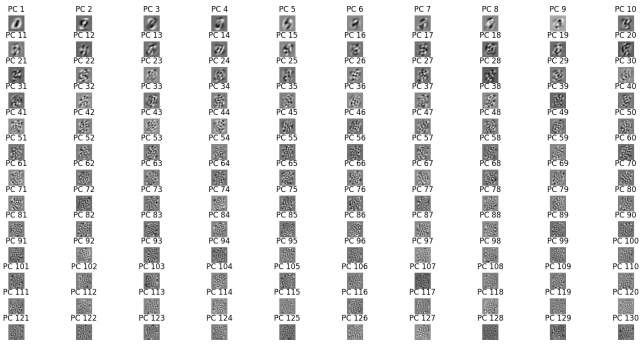


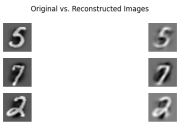
Figure 3: Visualisation of PCs

We calculate the variance covered by each Principal Components and print them on the console, which goes like (10.2% by PC1, 7.8% by PC2, and so on). Note that 99.99% of the total variance is covered by the top 130 PCs put together, so we are not losing information.

4 Q1: Part 1(ii)

To reconstruct the images, we take 10, 20, 50, 100, 130 top PCs to observe the difference in quality of pictures generated, which are as follows:

(a) Top 10 Principal Components



(b) Top 20 Principal Components

(c) Top 50 Principal Components

(d) Top 100 Principal Components

(e) Top 130 Principal Components

Figure 4: Principal Components Analysis

Dimensions	Reconstruction Error
10	46.0730
20	38.1439
50	26.0558
100	17.5086
130	14.5712

Table 1: Dimensions vs Reconstruction Error

As it is evident from above plots, we can clearly identify that the **image contains a number with just top 20 principal components** and the numbers become clearly **distinguishable with top 50** of them.

To reconstruct the images reasonably well, we may go ahead by 50 PCs, but the Reconstruction error table above suggests that going by 130 PCs reduces the error to half of that by 50 PCs. This is also shown as the **elbow** on the graph plotted above which contains **95% of the information** of the image and the rest components are just remaining 5%.

Therefore, it is safe to use **top 130 components** to get the images reconstructed.

5 Q2: Dataset

The dataset has 2 columns with 1000 rows. On plotting, it looks like the figure below. As the decision boundary is intuitively non linear, we know Lloyd's algorithm, without kernelisation would not wok well here.

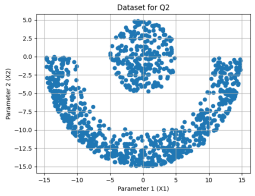


Figure 5: Dataset

6 Q2: Part(i): Lloyd's Algorithm with K = 2

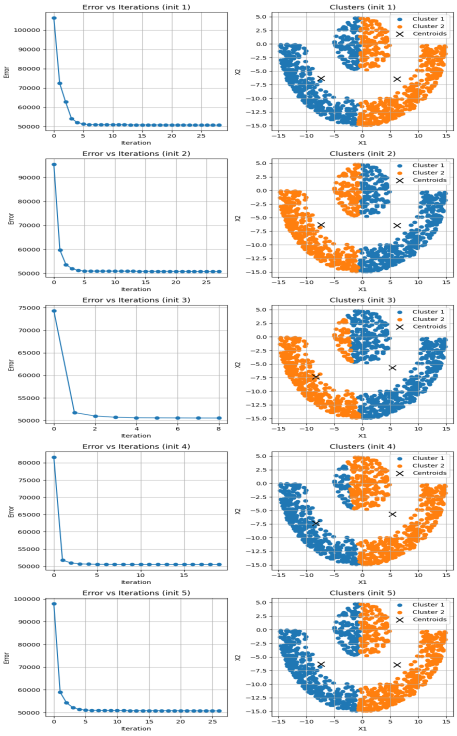


Figure 6: K-means algo runs with 5 random inits with K = 2.

The above plot shows the clustering is not at all useful for us.

7 Q2: Part(ii): Lloyd's algo with K=2,3,4,5

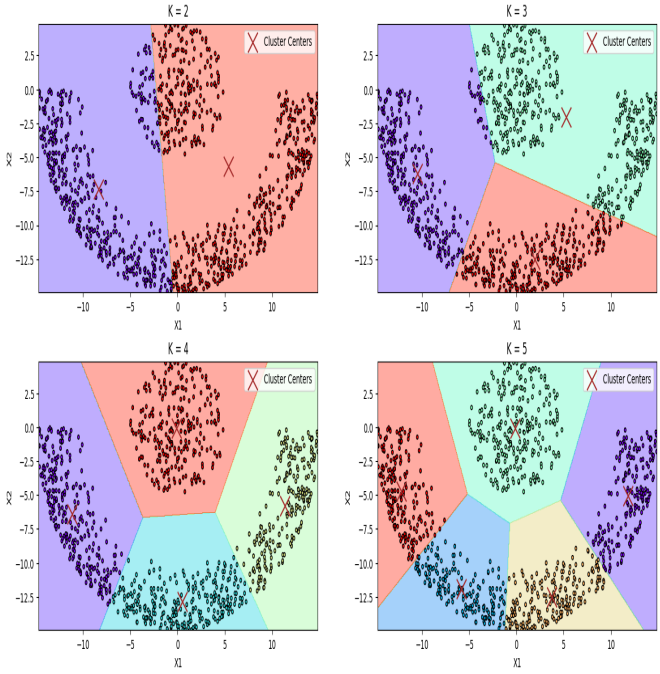


Figure 7: Enter Caption

Here, the clustering is better than the one in K=2 but still not useful as we need nonlinear decision boundaries here, which cannot be achieved without kernelization.

8 Q2: Part (iii)

The Lloyd's algo, which is the standard implementation of the K-means algorithm, is **not an ideal approach** for clustering this dataset due to its nonlinear decision boundary.

Some important points to note are:

- The lloyd's algo assigns clusters points by **2nd-norm** which creates **linear decision boundaries** for the voronoi region.
- This algo assumes clusters are **spherical** in shape which is not the case in this dataset.

Alternative methods for this dataset:

- **Kernelised K-Means:** we can extend the given data to higher dimentions using kernel function, where the data becomes linearly seperable (a polynomial kernel of degree 2 would be suitable for this case). Then the standard k means algo would be safely applicable on this data, while still being computationally feasible.
- **Spectral Clustering:** In spectral clustering, dimensionality reduction is carried out prior to clustering utilizing the eigenvalues of a similarity matrix that is created using data point distances or affinities. This works very well for non-convex or non-linear clusters as in this case.
- **Gaussian Mixture Models (GMMs):** these are a probabilistic clustering approach that models the data as a mixture of Gaussian distributions. This data can also be modeled well using GMMs. It can handle overlapping clusters as well as non linear decision boundaries between the clusters.

Summary: While Lloyd's algorithm is computationally simple and efficient, it is unsuitable for clustering this data due to its reliance on linear decision boundaries. Kernel-based approaches or Spectral Clustering should be preferred for such problems to effectively capture non-linear cluster structures.

****