Foundations of Machine Learning Assignment 3

Principal Component Analysis, K Nearest Neighbours

DA24D402, Shuvrajeet Das

November 17, 2024

Contents

List of Tables

List of Figures

1 Question 1

Download the MNIST dataset from `https://huggingface.co/datasets/mnist`. Use a random set of 1000 images (100 from each class 0-9) as your dataset.

(i) Write a piece of code to run the PCA algorithm on this data-set. Visualize 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?

(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?

Solution

(i) As shown in the diagram above, we have plotted the first nine principal components derived from the MNIST dataset. These principal components are visual representations of the

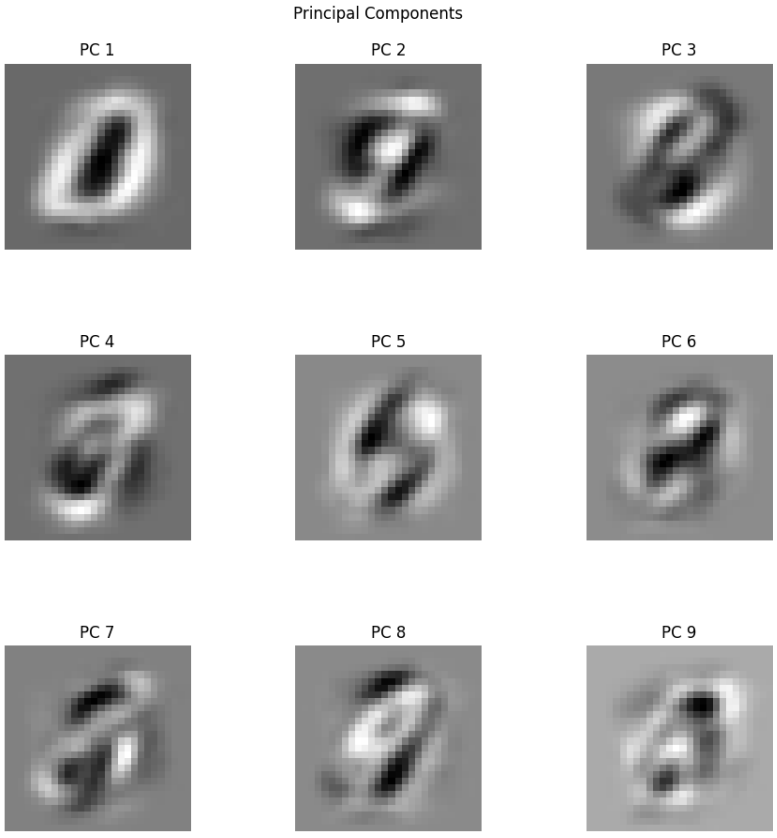

Figure 1: Visualisation of the principal components

dataset's directions that capture the data's maximum variance. The principal components are ordered based on the amount of variance they explain, starting from the component that explains the most variance (first principal component) to the subsequent ones that explain less.

Each of these components represents a weighted linear combination of the original pixel values in the MNIST images. By visualizing these components, we can interpret them as generalized patterns or features that the dataset uses to differentiate the digit classes. These patterns may correspond to specific regions of the images (e.g., strokes or curves) that play an essential role in characterizing different digits.

The percentage of variance explained by the first 10 principal components of the MNIST dataset is as follows:

- **Principal Component 1 (PC 1):** 9.69%
- **Principal Component 2 (PC 2):** 7.44%
- **Principal Component 3 (PC 3):** 6.92%
- **Principal Component 4 (PC 4):** 5.44%
- **Principal Component 5 (PC 5):** 4.88%
- **Principal Component 6 (PC 6):** 4.57%
- **Principal Component 7 (PC 7):** 3.49%
- **Principal Component 8 (PC 8):** 3.03%
- **Principal Component 9 (PC 9):** 2.80%

These values indicate the proportion of total variance in the dataset captured by each principal component. The first few components explain a significant amount of variance, demonstrating their importance in capturing the key features of the data.

(ii) The reconstruction is performed by projecting the original MNIST images onto a lower-dimensional space defined by the principal components and then mapping them back to the original space. This process demonstrates how much of the original image information is preserved when using a reduced number of dimensions.

The quality of the reconstructed images varies based on the number of principal components used. With fewer components, the reconstructions are less detailed, losing finer features, but they still capture the overall structure and essence of the original images. As the number of components increases, the reconstructions become progressively more accurate, closely resembling the original images.

This comparison illustrates the trade-off between dimensionality reduction and reconstruction accuracy. It highlights the effectiveness of principal component analysis (PCA) in compressing high-dimensional data while retaining significant patterns and structures.

The reconstructed images, along with their corresponding original images, are displayed below.

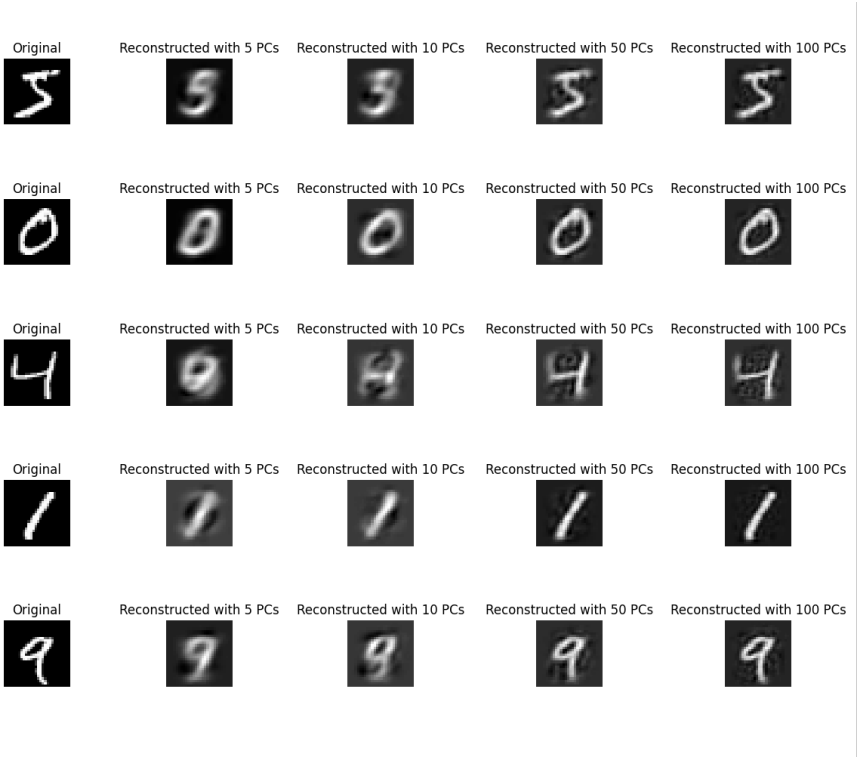

Figure 2: Reconstruction with different amount of principal components

If I had to pick a dimension d for a downstream task such as digit classification, I would select $d = 130$. This choice is based on the fact that using 130 principal components captures 95% of the variance in the dataset.

The rationale for selecting $d = 130$ is as follows:

- **Variance Explained:** Capturing 95% of the variance ensures that the majority of the information in the original dataset is retained, minimizing information loss that might negatively impact classification performance.
- **Dimensionality Reduction:** Reducing the dimensionality to 130 components significantly lowers the computational complexity compared to using the original feature space (e.g., 784 dimensions for MNIST). This makes training downstream models more efficient while still maintaining high accuracy.
- **Noise Reduction:** PCA inherently filters out noise by focusing on the components that contribute most to the variance. By retaining 130 components, the resulting dataset is likely to be less noisy and more suitable for classification tasks.
- **Empirical Trade-off:** While increasing the number of components beyond 130 would marginally increase the variance explained, it may introduce diminishing returns in terms of classification accuracy relative to the added computational cost.

Thus, $d = 130$ represents a practical balance between preserving the dataset's essential features and ensuring computational efficiency for downstream tasks such as digit classification.

2 Question 2

You are given a data-set with 1000 data points each in $\mathbb{R}^2$ (`cm_dataset_2.csv`).

(i) Write a piece of code to implement the Lloyd's algorithm 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.

(ii) For each $K = \{2, 3, 4, 5\}$, fix an arbitrary initialization and obtain cluster centers according to K-means algorithm using the fixed 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 $\mathbb{R}^2$).

(iii) Is the Lloyd's algorithm a *good* way to cluster this dataset? If yes, justify your answer. If not, give your thoughts on what other procedure would you recommend to cluster this dataset?

Solution

(i) The figures below illustrate the results of running Lloyd's algorithm for the K-means clustering problem, using random initialization points for the cluster centers. In each case, the algorithm starts with a different set of randomly selected initial cluster centers. As Lloyd's algorithm iteratively updates these centers, the data points are reassigned to the nearest cluster, and the objective function—the sum of squared distances from each data point to its assigned cluster center—gradually minimizes.

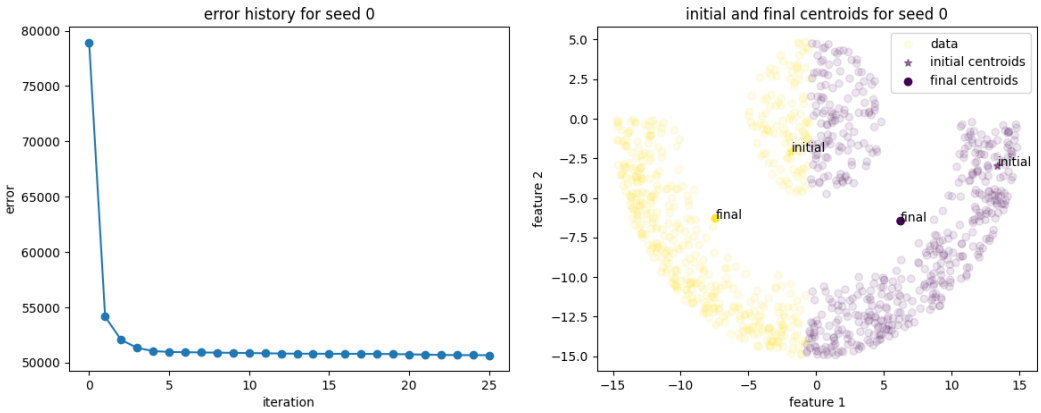

Figure 3: Error vs iteration and Feature plot

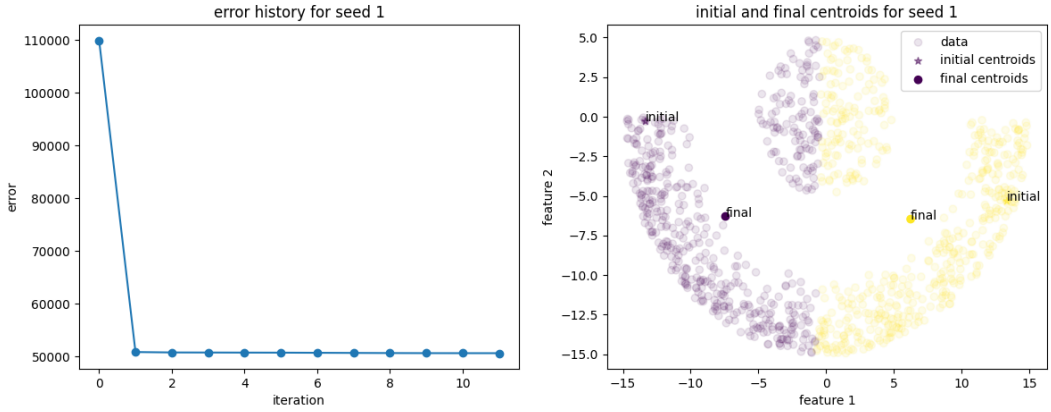

Figure 4: Error vs iteration and Feature plot

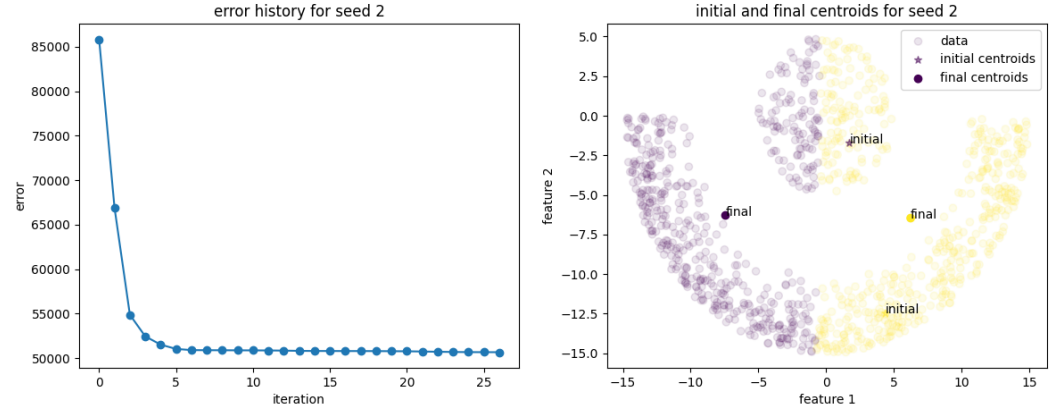

Figure 5: Error vs iteration and Feature plot

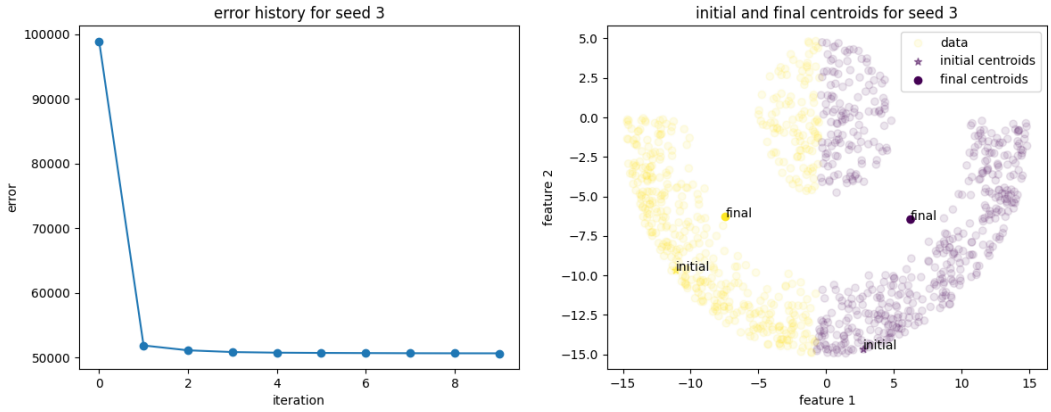

Figure 6: Error vs iteration and Feature plot

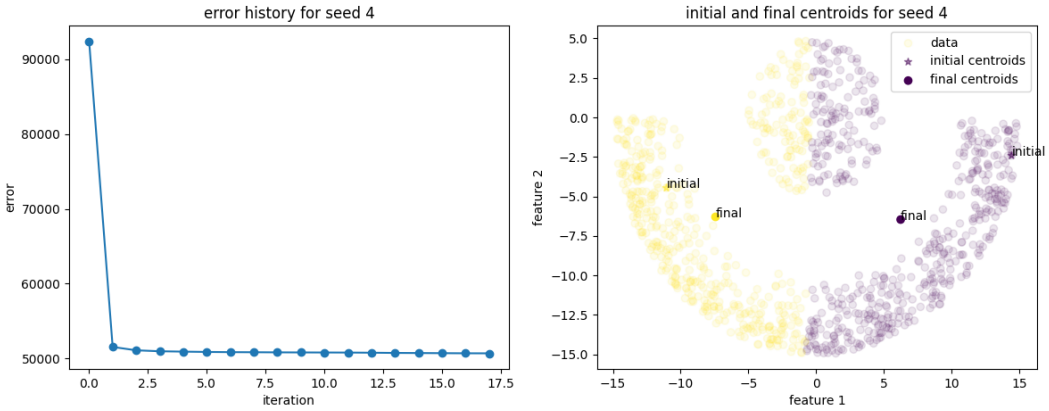

Figure 7: Error vs iteration and Feature plot

(ii) The figures below illustrate a Voronoi diagram that represents the classification regions for a dataset based on the K-means clustering algorithm. The plot provides a comprehensive understanding of how the classification evolves from the initial randomly selected mean points to the final optimized mean points.

Key elements of the plot include:

- **Initial and Final Mean Points:** The plot highlights the initial mean points, which are randomly selected at the start of the algorithm, and the final mean points, which are the optimized cluster centers after the algorithm has converged. The progression from the initial points to the final points is depicted to show how the means move during iterations to better fit the data distribution.

- **Voronoi Diagram:** The Voronoi diagram partitions the space into regions, where each region corresponds to a specific cluster. Every point within a region is closer to the cluster center (mean) associated with that region than to any other center. This visually represents the classification boundaries created by the algorithm.

- **Class Labels:** The data points in the plot are labeled with their respective classes, making it easier to evaluate the effectiveness of the clustering algorithm in separating different classes. The labels provide context on how well the Voronoi regions align with the actual class structure of the data.

- **Feature Representation:** The plot is based on a one-feature representation of the data, demonstrating how the algorithm operates in a simplified space. This makes it easier to understand the mechanics of K-means clustering and the formation of classification regions.

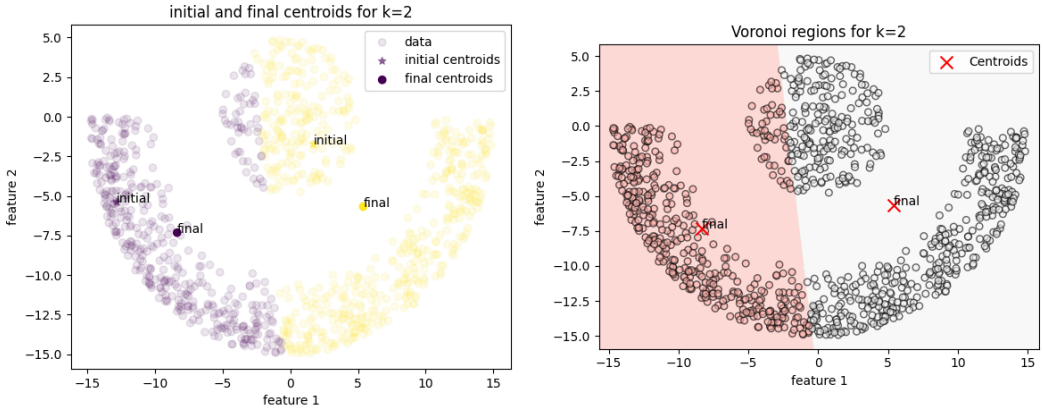

Figure 8: Error vs iteration and Feature plot for k=2

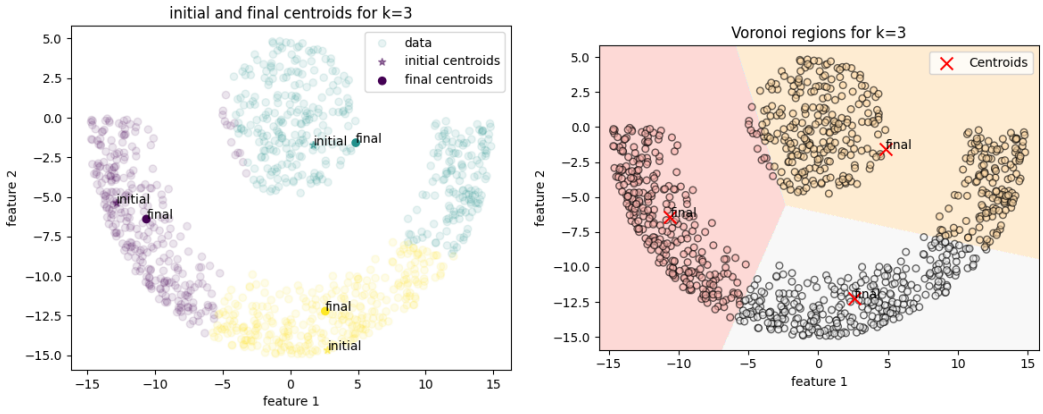

Figure 9: Error vs iteration and Feature plot for k=3

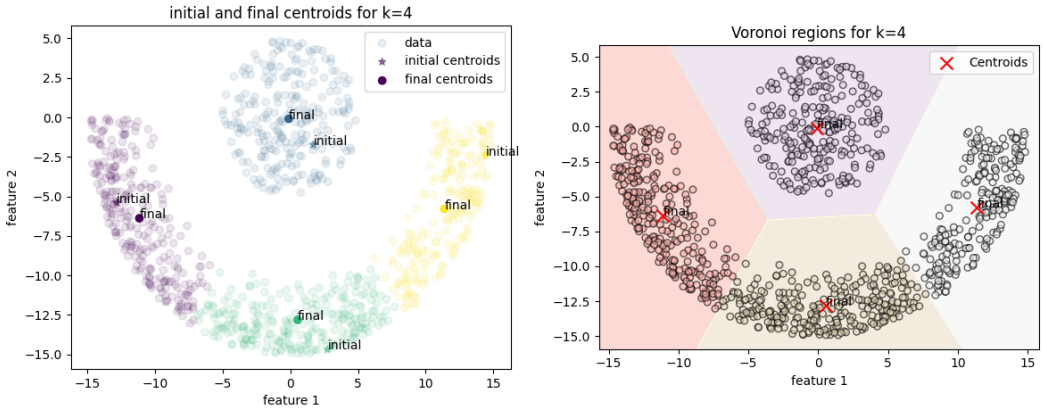

Figure 10: Error vs iteration and Feature plot for k=4

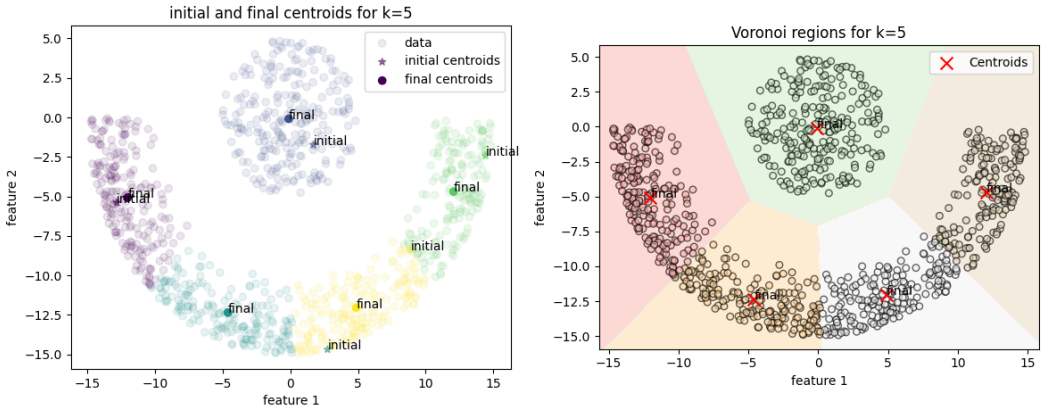

Figure 11: Error vs iteration and Feature plot for k=5

(iii) The suitability of Lloyd's algorithm for clustering this dataset depends on the dataset's structure and the algorithm's inherent characteristics. Below is an evaluation:

Limitations of Lloyd's Algorithm

- **Sensitivity to Initialization:** The algorithm's performance heavily depends on the initial choice of cluster centers. Poor initialization can lead to suboptimal solutions or convergence to local minima.
- **Cluster Shape Assumption:** Lloyd's algorithm assumes clusters are convex and isotropic (spherical). It struggles with datasets containing clusters that are non-spherical, overlapping, or vary significantly in size and density.
- **Fixed Number of Clusters:** The algorithm requires the number of clusters (k) to be predefined, which may not always align with the dataset's natural structure.
- **Outlier Sensitivity:** The algorithm is sensitive to outliers, which can distort cluster boundaries and lead to incorrect results.

Recommendation for This Dataset

Based on these limitations, if the dataset contains clusters that are non-spherical, imbalanced, or influenced by noise or outliers, Lloyd's algorithm may not be the best approach.

3 Conclusion

References