CMPE 462 - Project 3
Implementing K-Means & PCA

Student ID1: 2015400069
Student ID2: 2015400177
Student ID3: 2019700087

June 2020

1 Introduction

In this project, we were asked to implement two unsupervised learning techniques. In the first technique, we implement k-means clustering algorithm using the provided data. Otherwise, in the second technique, we implement PCA and applied dimensionality reduction on the data provided for PCA. Implementation details and evaluations will be shown in the following sections.

2 K-Means Clustering

2.1 Plotting the Data

First of all, we started with visualizing the data provided for K-Means clustering algorithm. This visualization helped us to interpret data clearly and let us to find optimal iteration number for the algorithm by comparing our results with ground truth. While plotting, we have assigned different colors to different class points. The visualization can be seen in the following figure.

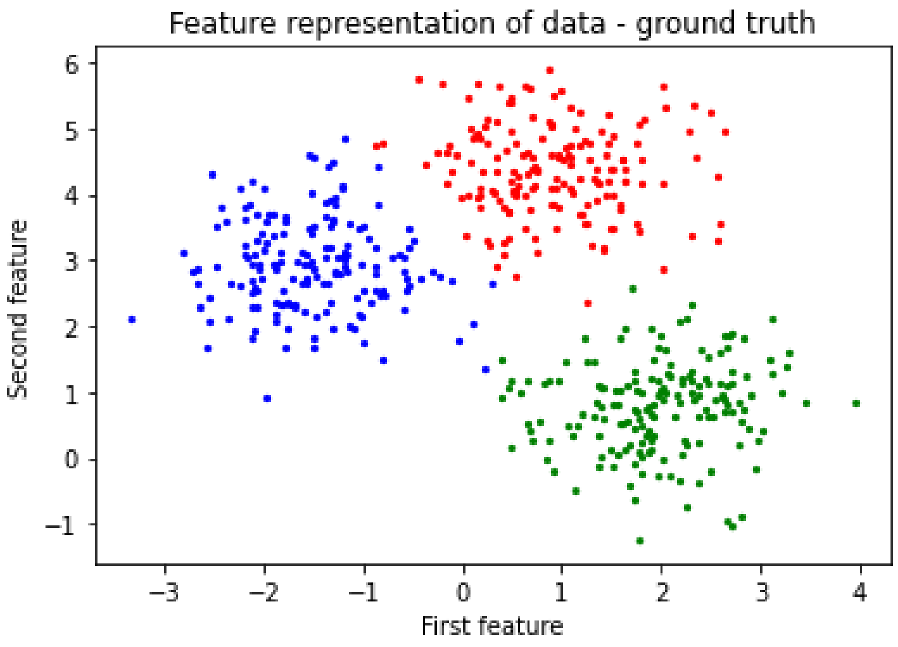

Figure 1: Plot of K-Means data with colored according to classes.

2.2 K-Means Implementation

In this section, we have implement known unsupervised K-Means clustering technique by using the number of iterations as the stopping condition. In this algorithm, firstly, we randomly select K cluster center points. Then, for each iteration, we calculate the sum of the squared distance between data points and all centroids(cluster center). After that, we assign data points to the closest centroid and update the centroids' values by taking the average of the all data points that belong to each cluster.
After iterations, similar data points are clustered and assigned to their most probable classes. Evaluations of algorithm for different number of iterations can be found in the following section.

2.3 K-Means Evaluation

For the evaluation purposes, we run our K-Means Clustering algorithm 9 times with number of iterations N = 1,2,...,9. For a fair comparison for different number of iterations, we start each run with the same initial cluster centers. For this purpose, we have seed the numpy random module which we use to determine starting points of centroids with 1. In order to visually investigate and determine convergence point of our algorithm on the data set by comparing resulting figures, we have plot the final clustering assignments for each run on 3x3 table. The resulting assignments with colored according to their classes can be found in the following figure.

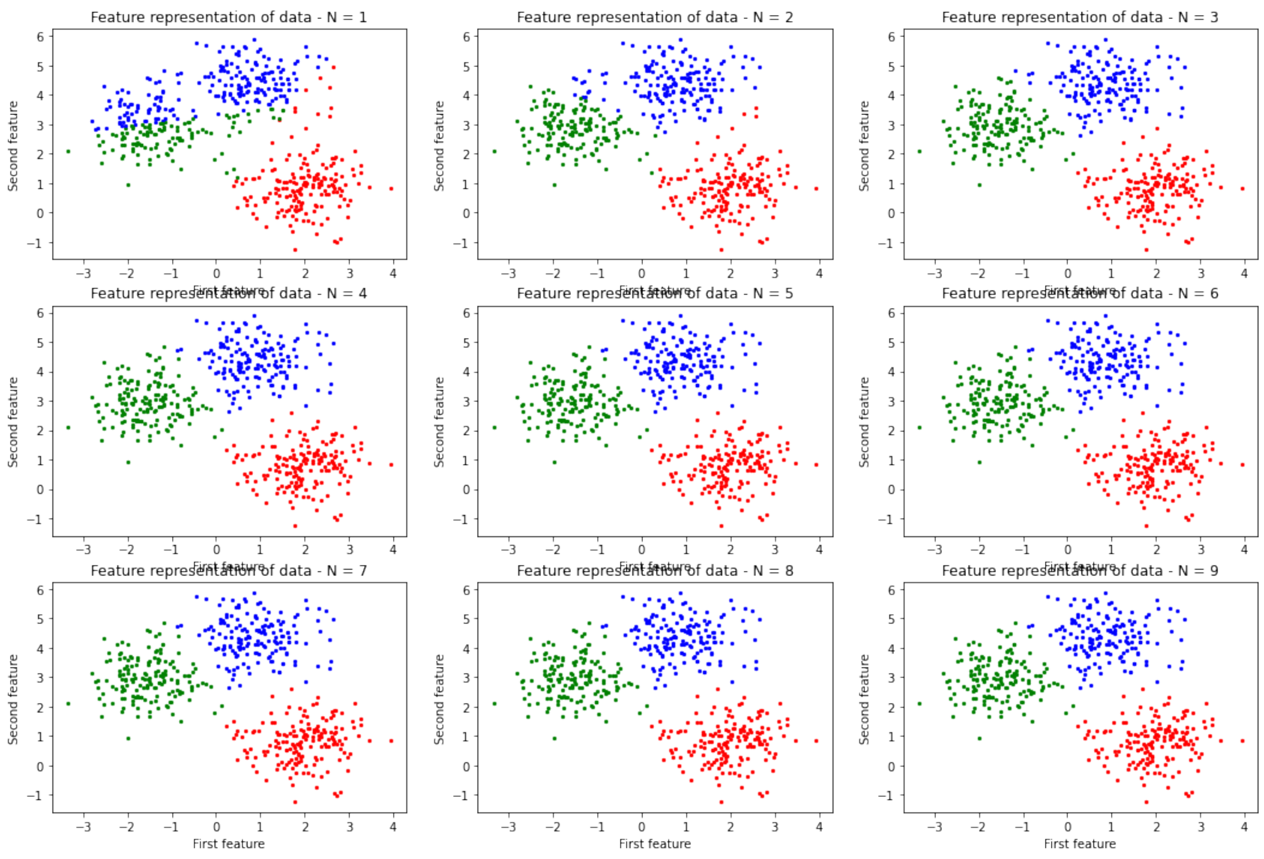

Figure 2: Plot of K-Means data with colored according to assigned classes for different number of iterations

As can be seen in the figure above, the resulting assignments start to converge to the optimal clustering and seem to be not changing after iteration number 4. This situation can be interpreted as the unknown data points are assigned to their ultimate classes after the end of iteration number

4 and further iterations are unnecessary since there is no change after 4.iteration. If we decided to run the algorithm until there will be no change in terms of assigned classes(as stopping condition) instead of certain number of iterations, than the algorithm would stop after 4.th iteration. However, if we investigate the ground truth class assignments and found class assignments in more detail, we can still see some points that are assigned incorrectly even after 9.th iteration. That's because we have some points that can be classified as outlier or a point that is very close to both two centroids as always happens.

3 Principal Component Analysis (PCA)

3.1 PCA Implementation

Nowadays most of the datasets have high complexity which cannot be easily interpreted by humans or machines. In these cases, we might want to reduce the number of dimensions which exists in the datasets to make them easily interpretable. Dimensionality reduction is a way to reduce the complexity of a model and avoid overfitting. In this project, in order to reduce the complexity, we applied PCA to our model.

First, we standardize the dataset with the following formula to balance the weights of the attributes.

$$\hat{X} = \frac{X - \mu}{\sigma} \tag{1}$$

After standardization process, we constructed the covariance matrix from the centered matrix which was calculated by subtracting the mean of the matrix from the original matrix and calculated the eigenvalues and eigenvectors of that covariance matrix.

Before constructing the principal components (PC), eigenvalues of the covariance matrix are sorted by descending order to rank the corresponding eigenvectors.

Finally, we calculated the dot product between the centered matrix and the selected eigenvectors. The number of eigenvectors are given in the project description. We have used [50, 100, 200, 256] as number of eigenvectors.

3.2 Image Reconstruction

To reconstruct the reduced images, we calculated the dot product between the reduced matrix and transpose of the eigenvector matrix and sum that dot product with the mean of the original matrix. The reason behind summing with the mean of the original matrix comes from the PCA implementation. We have subtracted the mean of the original matrix from the original matrix to shift the data to the origin.

After all of the calculations, we have successfully reconstructed the initial 16x16 images.

3.3 PCA Evaluation

After implementing PCA and image reconstructing function, we ran that code for several times for different images with the indexes of [0, 500, 1000, 2000] with different numbers of [50, 100, 200, 256] principal components.

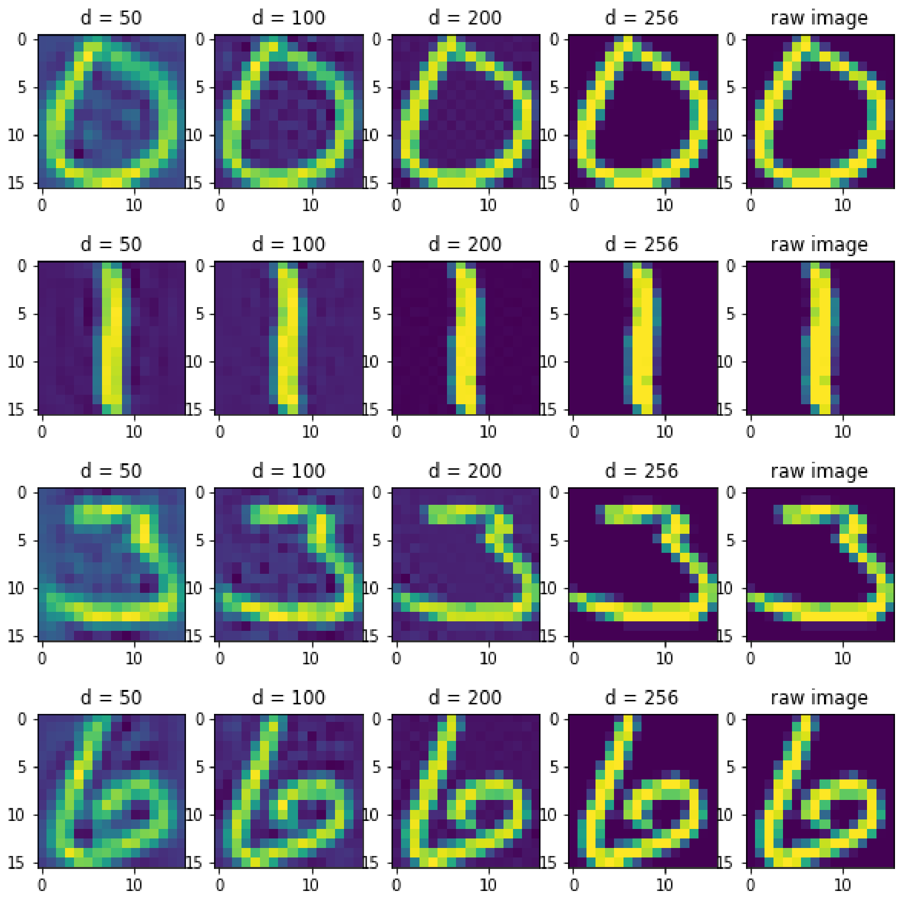

Figure 3: Visualization of digits

If we investigate the figure above deeply, we can easily say that the performance of the PCA algorithm heavily depends on the image (matrix) complexity. For example, reconstructed image of number 1 with d=50 has almost the same quality with the reconstructed image of 1 with d=256 because of it's linear shape. But, if we look at the reconstructed image of number 6 with d=50, it seems much blurry than the reconstructed image of 6 with d=256 because of the complex shape of the number 6.

To conclude, sharpness of the images were increased positively with respect to the increasing number of principal components. With d=256, we have managed to achieve the same qualities with the raw images.