Data Mining - Project 4
Cluster Analysis

- Vivek Sanghvi Jain

Under Guidance of
Prof. Michael Hahsler

Table of Contents

1. Executive Summary and Business Understanding

This objective of this project is to cluster a dataset of images of handwritten digits using one or more than one clustering algorithm into several clusters based on the digit itself and the pattern in which the digit is written. i.e. The digits need not be grouped according to the actual values, but should definitely be grouped according to the way they are written.

As we know, everyone has a different style of writing the same text, we have several different types of the same digit, which we have to cluster in common groups. To aid in clustering the numbers into groups, several features have to be extracted from the handwritten images and then, clustering is performed to cluster the data into homogeneous groups.

Features can be of different types, for example: The pixel density of an image. They have to be chosen intelligently. A good feature helps in clustering similar images together. In this project, I have used 7 features.

After, the features are created and then scaled accordingly without weights in this project. The reason for choosing, equal weights is that the results were good enough and not much different with different weights to different features. After performing clustering on the image feature matrix.

2. Data Preparation

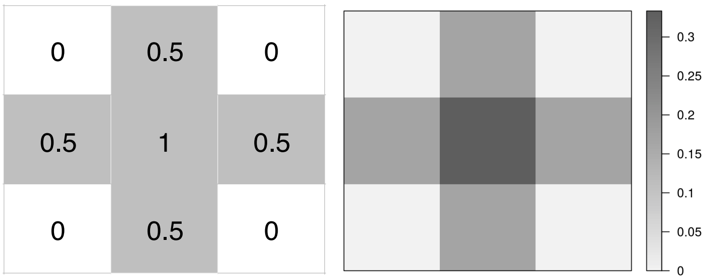
Numbers[1] data set has been used for this project. In the dataset, each row stores the pixel grey values for the image. Each image is 28x28 or 784 pixels with 256 grey values (8bit). The numbers data set stores the pixel grey values in 784 columns which are represented by the pixel numbers.

Before, any kind of feature extraction, is performed, it is advisable to perform some kind of image processing. Although, everybody has a unique style of writing, there are some common patterns and features which can be observed linguistically. Since everybody writes differently, some kind of image preprocessing greatly aids in effective generation of features which aids in effective clustering of images.

In, this project, some image processing based on mathematical convolution is performed[2]. Edge detection, sharpening, blurring, and other image processing techniques can be used for desired result. In, this project, the images are passed through the blurring filter twice and then passed for further analysis. Blurring to us, humans, would result in reduced detection of numbers, but for a machine, it gives it an added domain space, so that digits written a little differently, crookedly, slantingly can also be detected and clustered in the same cluster.

For Example: We can see three different kinds of 3 written.
When blurred twice, they look more similar to each other. Also, thinning is performed, which helps us identify additional features like number of end points, which for three is 3.

Images in this project are thinned and blurred with the Gaussian Kernel and as shown:



0	0.5	0
0.5	1	0.5
0	0.5	0

Figure 2.1: Blurring

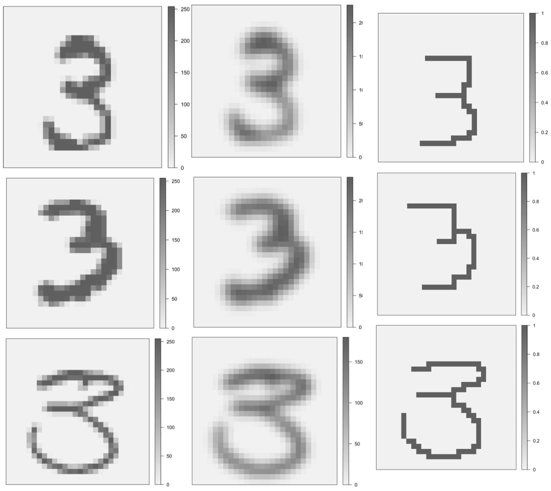


Figure 2.2 Effect of Blurring Mask and then Thinning on multiple images of a random digit

The blur mask as shown in Figure 2.1 is convoluted with the image matrix of 28x28 pixel. The same blurring action is repeated and then thinning is performed was demonstrated in Figure 2.2. Several other masks like the edge detection mask, the focus mask, tilt correction mask, negative mask, or a user defined mask can be used, to cater, to achieve the desired modification.

The following features have been created to perform clustering. The values in brackets aside the features represent scaled values.

1. **Total Number of Pixels** (Min: -2.4291 Max: 6.1330):
   This was calculated by computing the sum of all pixels in the image. Image Pixel Density can also be derived from this feature.

2. **Pixel Intensity in Upper Half of the Image** (Min: -2.30699 Max: 6.24527):
   Since the image is a 28x28 matrix made of 784 pixels, the first 392 pixels give this. The ratio of pixels intensities in the upper half of the image to the total pixel intensities.

3. **Pixel Intensity in Lower Half of the Image** (Min: -2.26865 Max: 5.36660):
   Since the image is a 28x28 matrix made of 784 pixels, the last 392 pixels give this. The ratio of pixels intensities in the upper half of the image to the total pixel intensities.

4. **Amount of Ink Used** (Min: 2.4291 Max: 6.1330):
   Computed by calculating average of all pixel intensities throughout the image.

5. **Number of Pixels in Vertical Lines** (Min: 2.4291 Max: 6.1330):
   Firs, the vertical lines are identified after applying the vertical convolution mask[3] and then picking only 5% pixels of highest value. Then, the sum of these pixels is computed.

6. **Number of Pixels in Horizontal Lines** (Min: 2.4291 Max: 6.1330):
   The horizontal lines are identified after applying the transpose of vertical convolution mask and then picking only 5% pixels of highest value. Then sum of those pixels is computed

7. **Number of End Points** (Min: 2.4291 Max: 6.1330):
   Thinning mask[3] is applied on the image after passing it through the thinImage method from the thinning r script. And then the number of end points is computed as the sum of the remaining points after the thinning filter as shown in class.

Scaling and Normalisation is performed after the extraction of the above features, using a scaling function, which does: $x_{[i]}$-mean(x)/sd(x), where x represents the feature.

3. Modelling and Evaluation

Modelling deals with clustering, its analysis followed by its validation. Clustering can be performed in several ways.

It can be centroid based clustering like the K-Means or Hierarchical or DBScan which use different types of linkages like single, complete or average linkage.

3.1 K-Means Clustering

If, we perform K-Means Clustering, we need to determine, *how many clusters would be ideal for our features ?*

To calculate, this, we can go about several methods, using several properties, like: *Within Sum of Squares, Average Silhouette Width, Dunn Index or Gap Statistic*

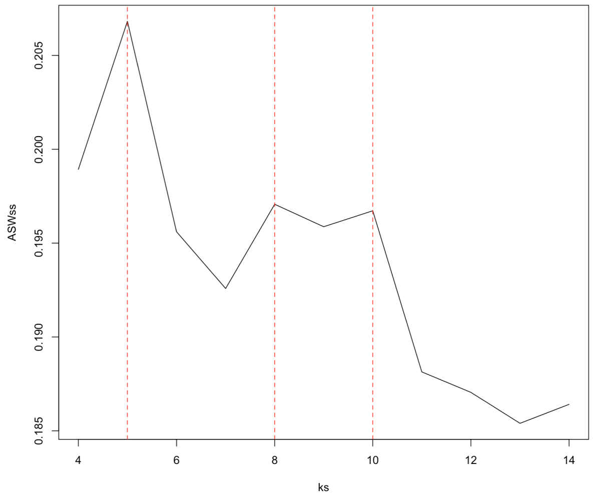


Figure 3.1: Average Silhouette Width for K-Means Clustering

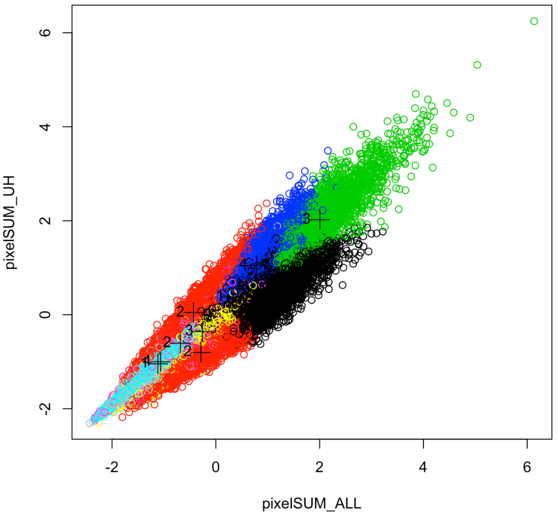


Figure 3.2: Plot of Clustes Using K-Means

The metric Average Silhouette Width is used here to determine the number of clusters. Dunn index also leads to the same conclusion as mentioned below.

Here, we can see, that K-Means would work best with 5 clusters for the current feature matrix, followed by 8 and 10 clusters as next best clustering. This means, that although, the features produce good results with 8 and 10 clusters, it provides best results with only 5 clusters, which implies 2 things. There is a lot of similarity between how different numbers are written, and the clustering algorithm is grouping them together, like 1 and 7 and 4 and 9. But, since, we get fairly good results in 10 as well, number of clusters used is 10.

After running KMeans, algorithm for 10 clusters, we get the (between_SS / total_SS = 64.1 %) km$size: 3725 2961 4644 3796 5402 6127 4621 5071 2556 3097

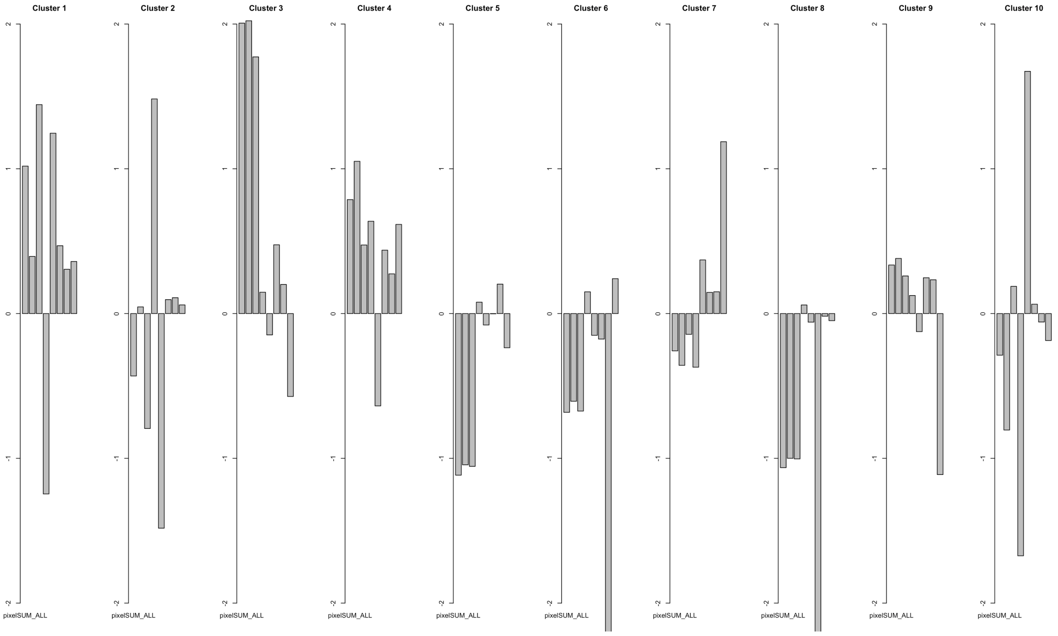


Figure 3.3: Bar Plot Describing all K-Means Clusters

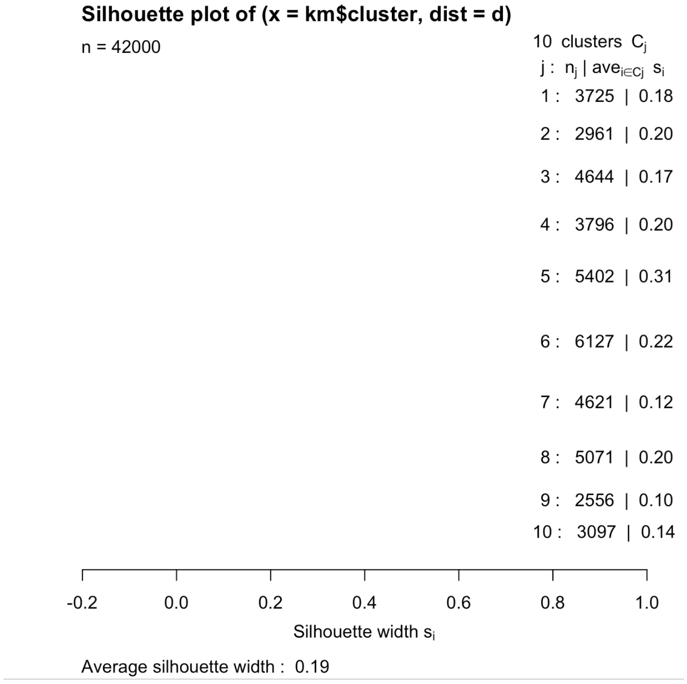


Figure 3.4: Silhouette Plot for K-Means

The plot is not easily visible, as there are 42000 images and 10 clusters. And the average silhouette width is 0.19, which is also evident from Figure 3.1

3.2 Hierarchical Clustering

Hierarchical Clustering helps in solving the centroid problem which arises in K-Means algorithm. Here, clusters follow a tree structure, which is called a Dendogram, as shown below in Figure 3.5

Similar to K-Means Clustering, Hierarchical Clustering is also performed for 10 Clusters. Diagram 3.5 shows Complete Hierarchical Clustering. The terminal end of each branch of a hierarchical cluster is called a leaf, which in this case is not visible, as there are over 42000 images. The depth of the branch points from the root node signifies the similarity between them. Nodes on the same depth are grouped in the same cluster. In case of complete linkage clustering, it we measure maximum distance between elements.

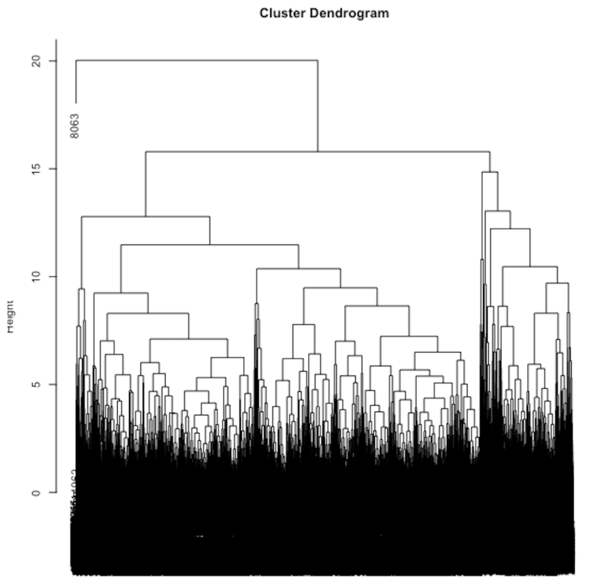


Figure 3.5: Dendrogram of Hierarchical Clustering

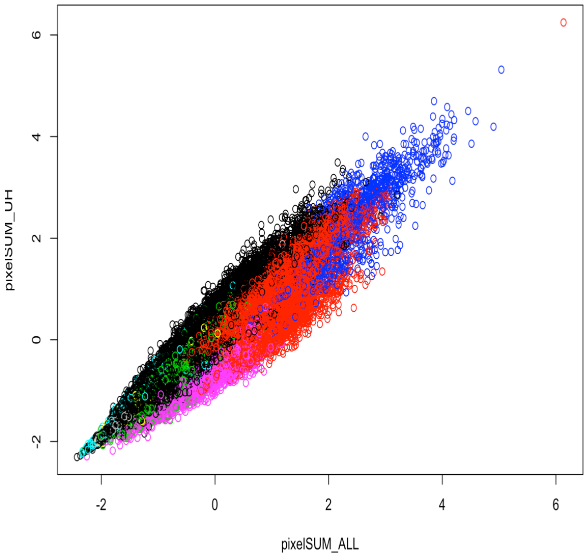


Figure 3.6: Plot of Clusters Using Hierarchical

From the above plot 3.6, we see that the clusters are much better defined then KMeans, and the results of external validation support this theory, as there are less incorrect clustering. Furthermore, the average silhouette height of Hierarchical clustering is also higher than K-Means clustering.

3.3 Density based clustering with DBSCAN

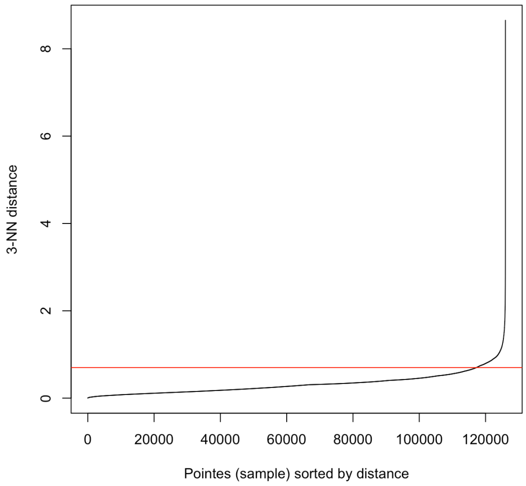


Figure 3.7:KNN plot to get epsilon

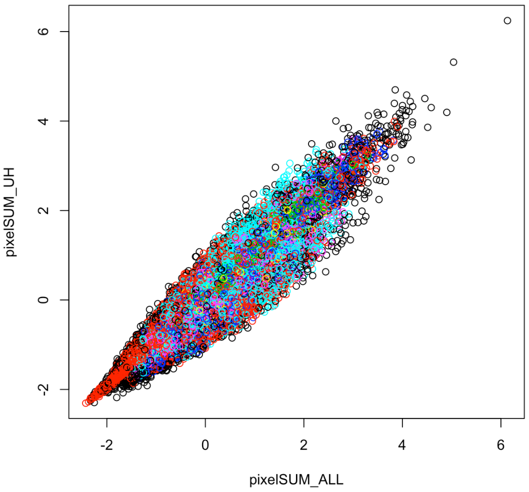


Figure 3.8:Plot of Clustes using

The Density based scan performed on the feature matrix yields 13 cluster(s) and identifies 4949 noise points with epsilon equal to 0.7. This is a very goof clustering mechanism, as it effectively eliminates noise points.

After performing external validation, with the help of the numbers_labels.csv, which has the correct image digit and since, we are talking about numbers, the number of cluster would be ideally 10. It is observed, there are several cases, in which clusters identified, constantly yield incorrect results. Particularly high, in the case of 1 and 7 and then 4 and 9 and some also observed in case of 3 and 8.

4. Evaluation

Hierarchical clustering was better and yielded better results then K-Means Clustering. But, DBScan Clustering was able to even identify better clusters, in all 13 clusters.

The following observations were made while doing the project.

Identifying 1s and 0s was the easiest and most succefull. But, often 7 was misidentified as 1 and in some cases, where 1 was very slant, it was misidentified as 7. 4 and 9 were often misclustered and so were 3 and 8.

There were several images which were at an angle. Common rotational transformation was attempted to remove images at an angle, but it resulted in pixel loss. A more effective way to do the same is required

This is my further observation. Some better feature selection could have resulted in creation of better clusters each, having, its own unique style of writing a digit and could also prevent mis clustering of images. Furthermore, Clustering should not be performed for optical character recognition. It used to be a very effective technique, when computational resources were limited, but with advancement, in technologies, now we have Neural Networks, in, which we can not only train the images, but also very easily identify them, in finite amount of time.

5. References:

[1] Numbers Data Set
URL: http://michael.hahsler.net/SMU/EMIS7332/data/numbers/numbers.csv

[2] Convolution
URL: https://en.wikipedia.org/wiki/Kernel_%28image_processing%29

[3] http://michael.hahsler.net/SMU/EMIS7332/data/numbers/simple_image_processing.html