MTH514A:Multivariate Analysis

Project report

January 3, 2025

Overview of K-Means Clustering

Supervised by :
Dr. Minerva Mukhopadhyay

Manas Mishra(201340),
Koyel Pramanick(201333),
Shivani Yadav(201413),
Arijit Ghosh(201276)
Soumyadip Sarkar(201431)

Contents

K-means Clustering

1 Explanation of K-means

Clustering is to make group of similar kind of objects among larger group of available objects without using any prior label. A new element can also be assigned to particular cluster after cluster formation.

The two things we should keep in mind while clustering is: (a) Sum of Squares withing cluster should be small for all clusters i.e. points within each cluster should be closer and (b) Sum of Squares between clusters should be as large as possible for all clusters i.e. points between clusters should be far away.

Sum of average distances (Euclidean/ Manhattan) of data points within cluster from the cluster centroid is called **inertia**. To minimize this inertia is our main goal in clustering.

1.1 K-means

It is one popular method of clustering. Below let's see some details about it.

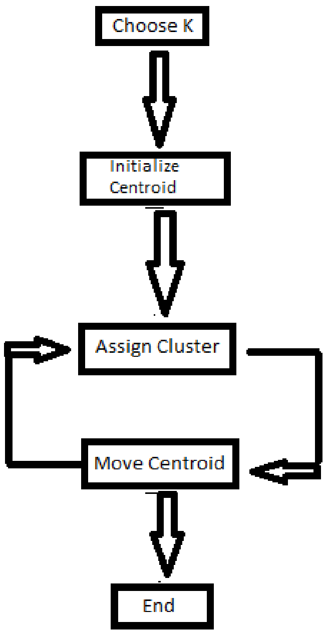


Figure 1: Diagram Showing Steps of Kmeans Clustering

1.1.1 Steps for Kmeans Clustering

1. Set the value of K where K = number of clusters we want.
2. Randomly initialize centroid for K clusters (these random centroids will be from available data points).
3. Calculate distance from each centroid to each point.
4. Assign a point to a cluster for which that point has minimum distance among distances from all other points.

5. Calculate mean of points assigned to cluster and take these means as new centroids of each cluster.

6. Perform step 3 to 5 (one round from step 3 to step 5 is called one iteration).

7. Keep running iteration till the convergence (no reassignment of point from one cluster to another).

1.1.2 Choice of K

(a) Elbow Method

One popular method for selecting K is by **Elbow Method** where we take a set of positive integers as value of K. Then from the graph of Inertia vs K, we consider that value of K as optimum K for which we get considerable change in inertia.

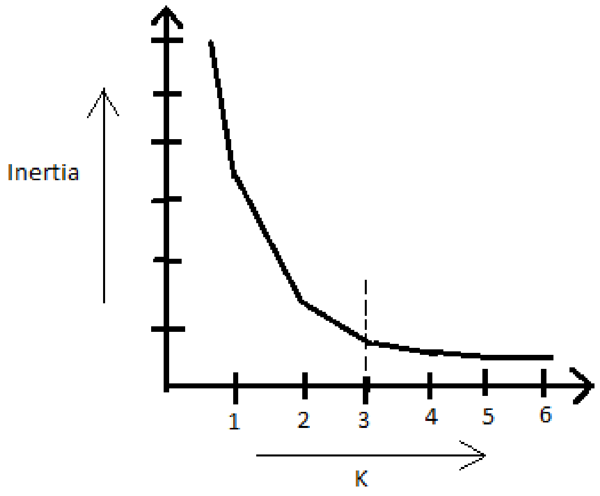


Figure 2: Elbow Method

In Fig: 2, clearly it can be assumed that it has not large amount of inertia reduction after K=3. So here we can take 3 as required cluster number.

But as this Elbow Method contains some amount of subjectivity, this does not work well in many cases specially when available data points are not clearly clustered i.e. a set of data points is not clearly separable from another set of another points, because in these cases whatever be the number of clusters, there is no much change in Sum of Squares within clusters (inertia).

There is another method called Silhouette Score (or Silhouette Coefficient) which is a better way to validate cluster number, checking reassignment of points to different cluster and overall cluster quality.

(b) Silhouette Score

Silhouette Score or Silhouette Coefficient is used to (a) Obtain optimum number of clusters, (b) Validate optimum number of clusters obtained by K-means clustering. This method is based on the idea of Cohension and Seperation. It ranges from -1 to +1 indicating worst quality of clustering to best quality of clustering. A 0 value of Silhouette Score indicates clusters are overlapping. Let's get a brief idea about these two terms:

1.1.3 Cohesion and Seperation

Cohension is measure of similarity of all data points from a particular data point which should be as much as possible for all points within a cluster. This can also be called 'Intra-Class Distance). This value should be as low as possible.

Seperation measures the similarity of all data points of another cluster from a particular data point of one cluster which should be as low as possible for all points of other clusters. Another term used for it is 'Inter-Class Distance). This value should be as large as possible.

Here similarity is computed (generaly) by Euclidean Distance, sometimes Manhattan Distance can also be used.

1.1.4 Steps for Calculating Silhouette Score

We will describe steps for calculating Silhouette Score using the diagram below:

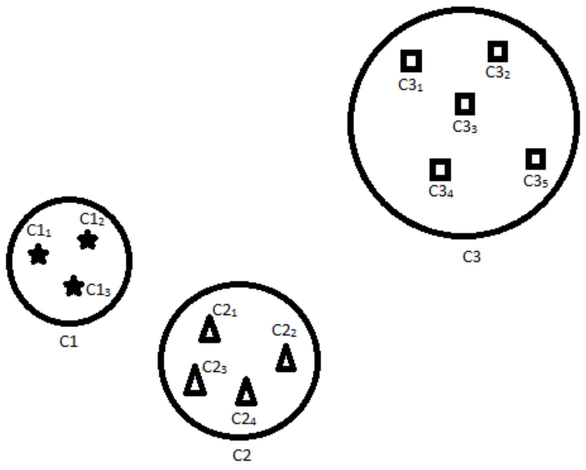


Figure 3: Diagram Showing Three Clusters of Data Points

Here as can be seen in Fig: 3 there are three clusters named C1, C2 and C3 containing 3, 4 and 5 points respectively. Let's take one point $C1_1$ as starting point.

1. Calculate distance (generally Euclidean Distance, Manhattan Distance can also be used according to situation) from $C1_1$ to $C1_2$ and $C1_3$. Suppose these two distances are $d1_1$ and $d1_2$ respectively.

2. Take average of these two distances as measure of cohension of data point $C1_1$, say a_1. So,

$$a(1) = \frac{d_1 + d_2}{2}$$

3. Calculate distance from $C1_1$ to points of 2^{nd} cluster (C2) i.e. to points $C2_1$, $C2_2$, $C2_3$ and $C2_4$. Suppose these distances are $d2_1$, $d2_2$, $d2_3$ and $d2_4$ respectively.

4. Take average of these distances, say B_1. So,

$$B_1 = \frac{d2_1 + d2_2 + d2_3 + d2_4}{4}$$

5. Similarly follow S3 and S4 for calculating distance from $C1_1$ to points of 3^{rd} cluster (C3) and let the average of these distances ($d3_1$, $d3_2$, $d3_3$, $d3_4$, $d3_5$) be B_2. So,

$$B_2 = \frac{d3_1 + d3_2 + d3_3 + d3_4 + d3_5}{5}$$

.

6. Take minimum of two distances from $C1_1$ to two clusters (C2 and C3) as measure of seperation of data point $C1_1$, say b(1). So,

$$b(1) = min(B_1, B_2)$$

7. Now, calculate Silhouette Coefficient for data point $C1_1$ of 1^{st} cluster (C1) as:

$$s = \frac{b(1) - a(1)}{max(b(1), a(1))}$$

.

(The denominator of above score scales it between -1 to +1.)

Now from here:

- If we want to cluster a set of data points, then we have to perform K-means clustering for different number of clusters first, and then to consider that number of clusters for which average silhouette score is maximum.
- It can be used as a tool to validate the result from elbow method so we take the number of clusters such that it has the lowest inertia among the possible choices.

1.2 Example of Elbow Method

Here we are showing application of K-means **using Elbow Method**. We have taken Country clusters.csv dataset.

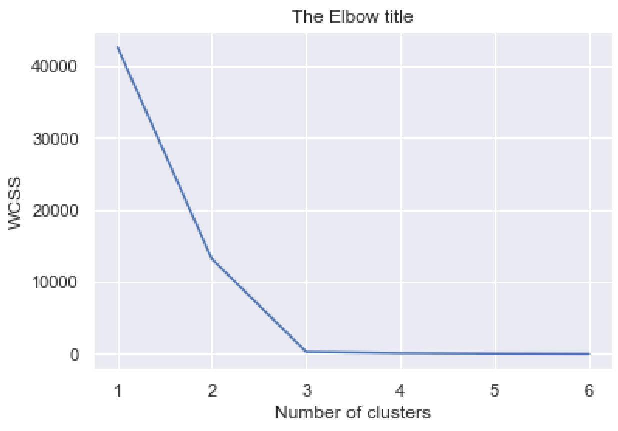
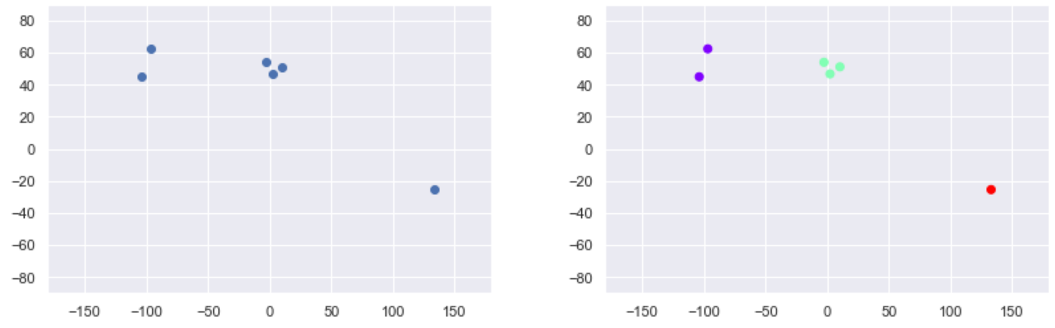


Figure 4: Diagram Showing Three Clusters of Data Points



(a) Plotting of Data

(b) Clusters by Kmeans

Figure 5: Kmeans Clustering by Elbow Method

As discussed earlier, Elbow method works better when data points are clustered with clarity. From Fig: 4, the number of clusters obtained is 3 (as clear elbow shape appearing at 3).

From Fig: 5 (a), it can be seen that data points are clearly seperately clustered and from Fig: 5 (b), the number of clusters obtained from Elbow method is suitable for this data.

1.3 Example of Silhouette Method Used to Validate Result from Elbow Method

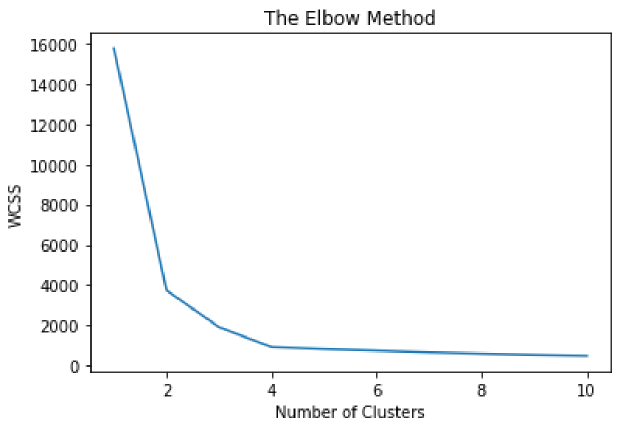


Figure 6: Elbow method

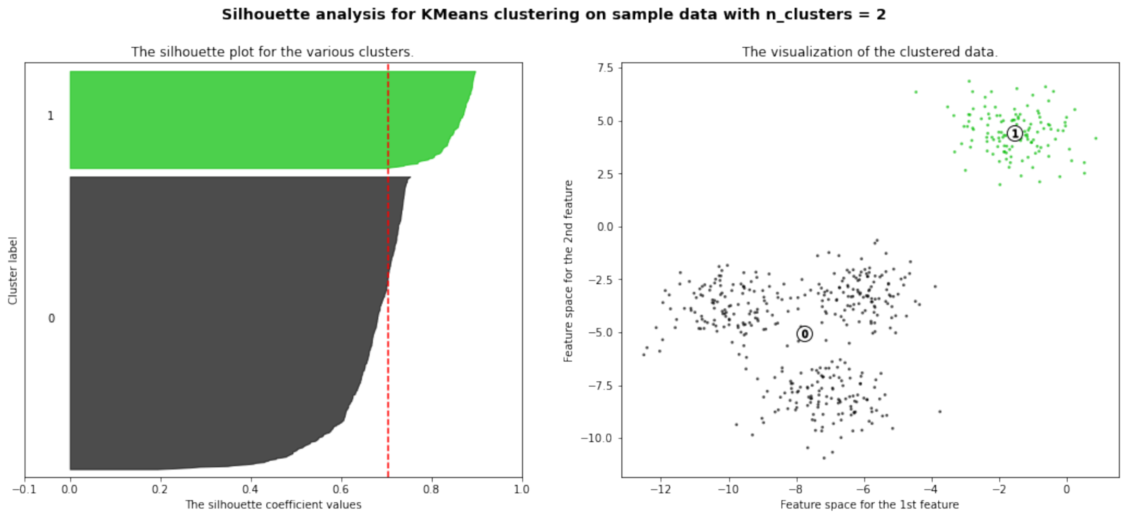


Figure 7: Silhouette Diagram with 2 Clusters of Data Points

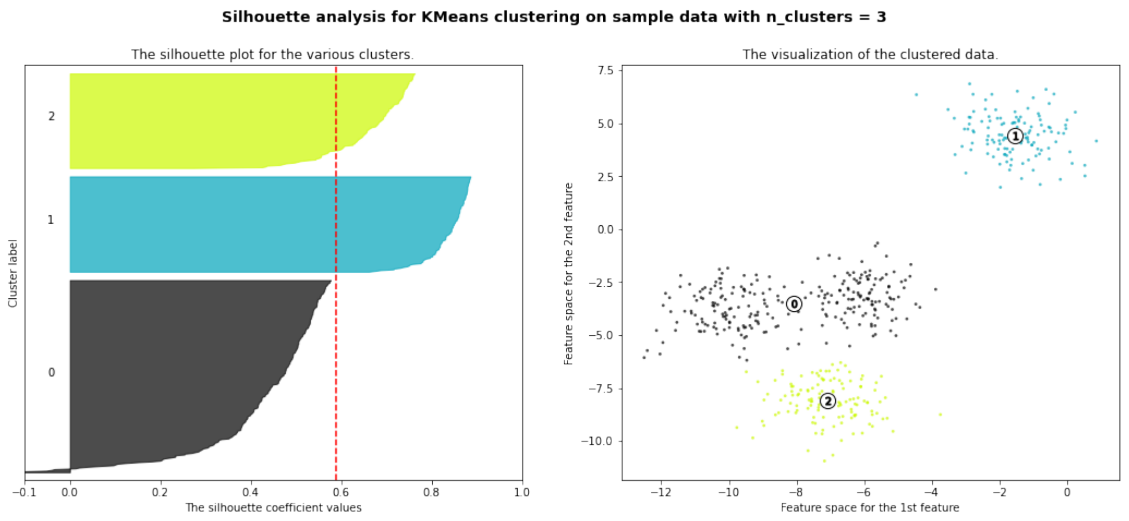


Figure 8: Silhouette Diagram with 3 Clusters of Data Points

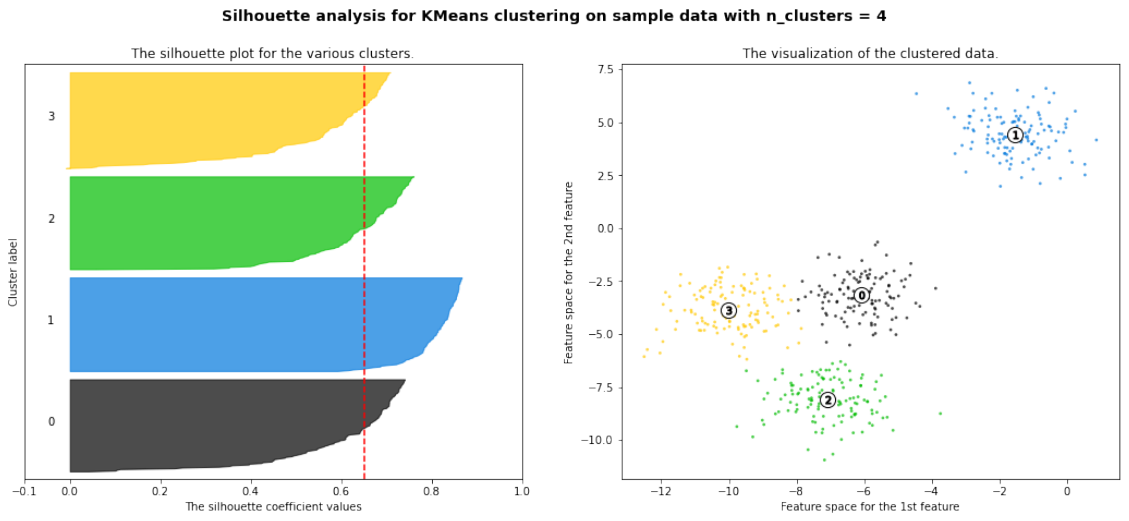


Figure 9: Silhouette Diagram with 4 Clusters of Data Points

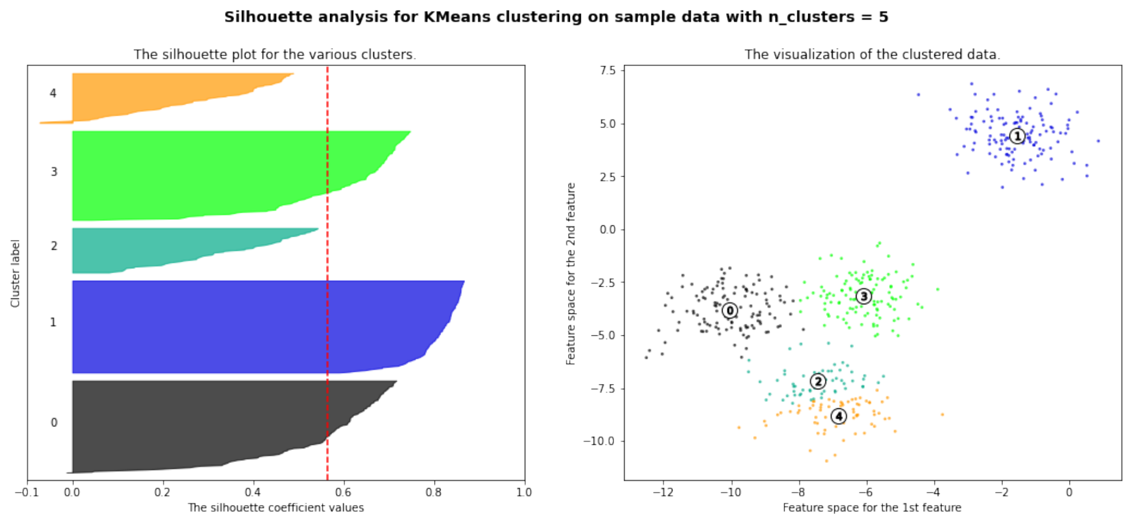


Figure 10: Silhouette Diagram with 5 Clusters of Data Points

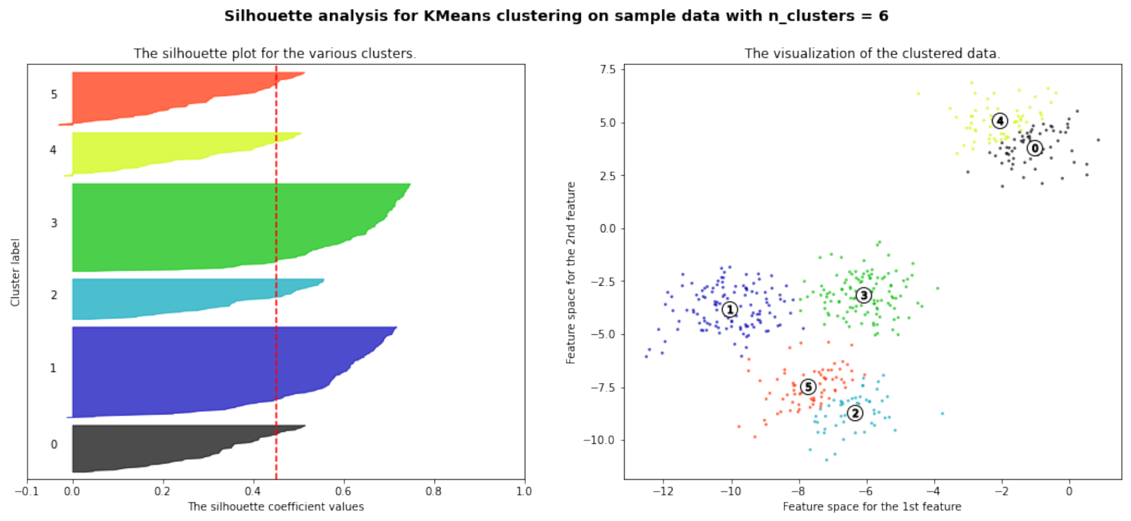


Figure 11: Silhouette Diagram with 6 Clusters of Data Points

From Figure 6 we can see that the elbow method gives optimal number of clusters=4.To validate this result silhouette score results are often used.We can see that silhouette score for two clusters is highest and for four clusters it is second highest but if we take number of clusters equal to 2 then inertia is quite high and hence taking into account inertia and silhouette score we take number of clusters=4

1.4 K-means for Categorical Variable

K-Means works using concept of numerical between two points. But, for dataset with all categorical columns, distance between two data points makes no sense. In these cases, there is **K-Modes** algorithm for clustering. This algorithm can be considered as K-Means with **Dissimilarity Score** *(if the number of mismatched features between two observations is l, then l will be dissimilarity score between that two observation)* in place of distance between two data points. Below we are describing, procedure in details:

1.4.1 Steps for K-modes

1. Choose K observations randomly (they will be called leaders of clusters, which are going to be formed, similar as centroid initialization in case of K-means).
2. Calculate dissimilarity scores between leader and other observations.
3. Assign each observation to its closest cluster (if for any observation dissimilarity score is same for all leaders, assign that observation to a particular cluster randomly).
4. Define new modes for the clusters (check for each cluster, among all possible options present for each feature which is most frequent among all observations present in that cluster).
5. Perform S2 and S3 again.
6. Stop here if there is no change in cluster assignment, otherwise repeat S4 and S5 again.

1.4.2 Elbow Curve in K-modes

The **cost** is sum of all dissimilarities present in the clusters from leaders to member observations here. For different values of K, plot of cost against K gives Elbow Curve. the value of K, which will show bend like Elbow and where cost will be low, can be treated as initial cluster number.

2 K-Means++ Clustering

One disadvantage of the K-means algorithm is that it is sensitive to the initialization of the centroids or the mean points. So, if a centroid is initialized to be a “far-off” point, it might just end up with no points associated with it, and at the same time, more than one cluster might end up linked with a single centroid. Similarly, more than one centroids might be initialized into the same cluster resulting in poor clustering. For example, consider Fig: 12 shown below:

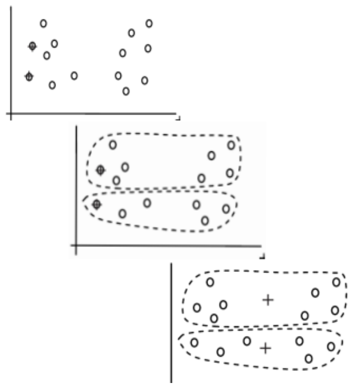
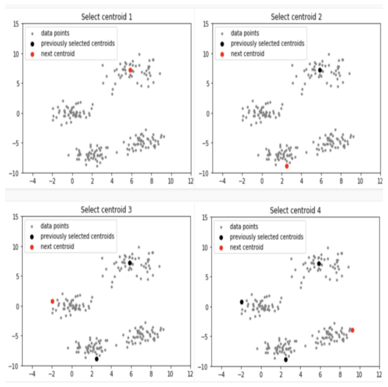


Figure 12: Diagram Showing Issue Associated with K-Means Algorithm

2.1 Algorithm of K-Means++ Clustering

As proposed by https://theory.stanford.edu/~sergei/papers/kMeansPP-soda.pdf, the algorithm is as follows:

1. Choose an initial center c_1 uniformly at random from χ.
2. Choose the next center c_i, selecting $c_i = x', x' \in \chi$ with probability $\frac{D(x')^2}{\sum_{x \in \chi} D(x)^2}$
3. Repeat Step 2 until we have chosen a total of k centers.
4. Proceed with the standard k-means algorithm. Here D(x) denote the shortest distance from a data point x to the closest center we have already chosen and χ is the set of data points.



3 Uses of K-means

3.1 Image Compression

The internet is filled with huge amounts of data in the form of images. People upload millions of pictures every day on social media sites such as Instagram, Facebook and cloud storage platforms such as google drive, etc. With such large amounts of data, image compression techniques become important to compress the images and reduce storage space.In this project, we will look at image compression using K-means clustering algorithm which is an unsupervised learning algorithm.An image is made up of several intensity values known as Pixels. In a colored image, each pixel is of 3 bytes containing RGB (Red-Blue-Green) values having Red intensity value, then Blue and then Green intensity value for each pixel

3.1.1 Approach

K-means clustering will group similar colors together into 'k' clusters (say k=64) of different colors (RGB values). Therefore, each cluster centroid is the representative of the color vector in RGB color space of its respective cluster. Now, these 'k' cluster centroids will replace all the color vectors in their respective clusters. Thus, we need to only store the label for each pixel which tells the cluster to which this pixel belongs. Additionally, we keep the record of color vectors of each cluster center.

Below here are two examples of image compression.

3.1.2 Examples of Image Compression

Example 1:

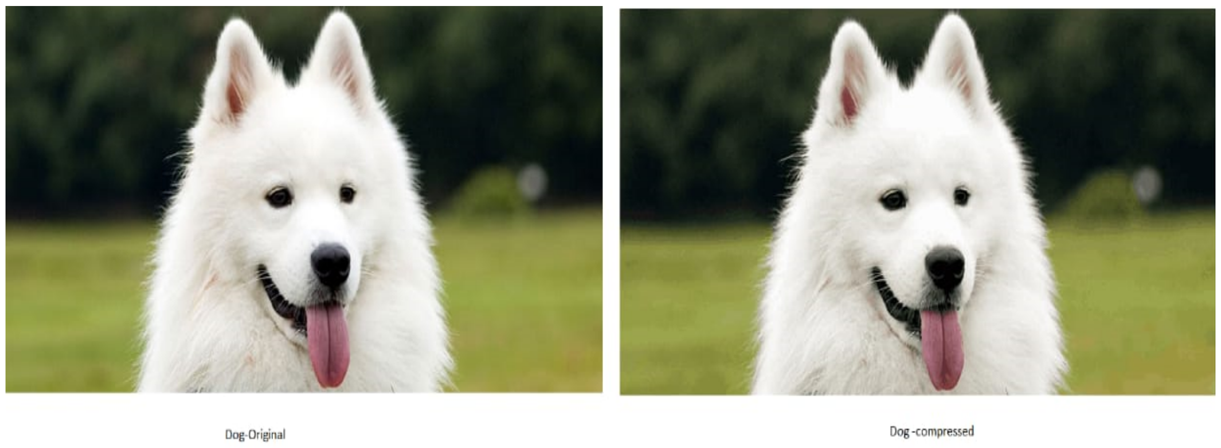
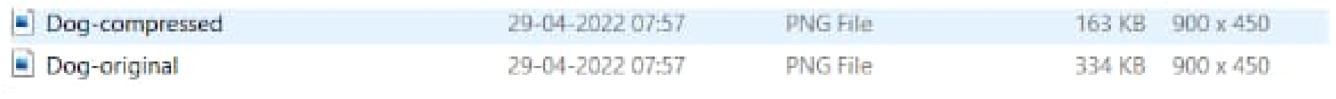


Figure 13: Dog Images (Original and Compressed)



Dog-compressed	29-04-2022 07:57	PNG File	163 KB	900 x 450
Dog-original	29-04-2022 07:57	PNG File	334 KB	900 x 450

Figure 14: Sizes of Dog Images

Here Fig:13 is showing original and compressed images of a dog. The effect of compression is not clearly visible from these images but Fig:14 is giving proof of the line stated before (334 KB and 164 KB of Original and Compressed respectively).

Example 2:

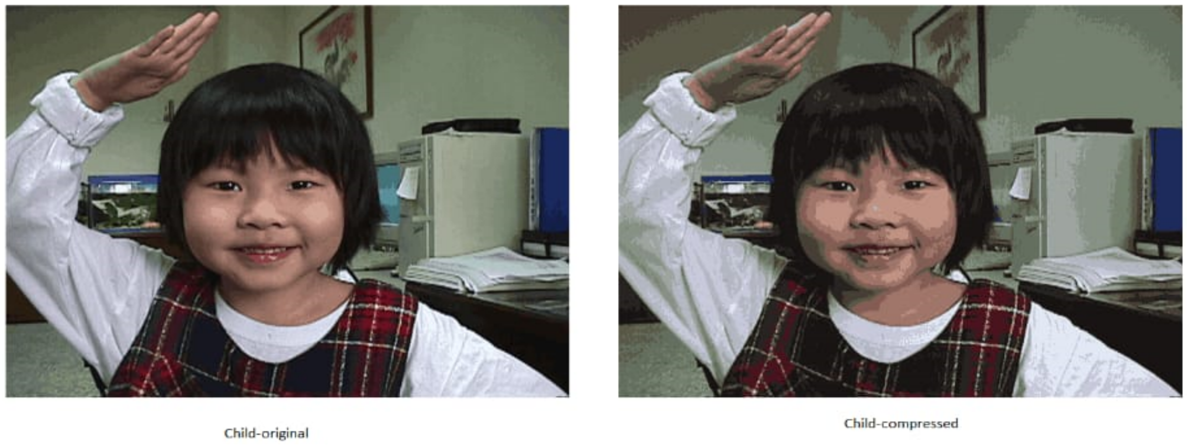
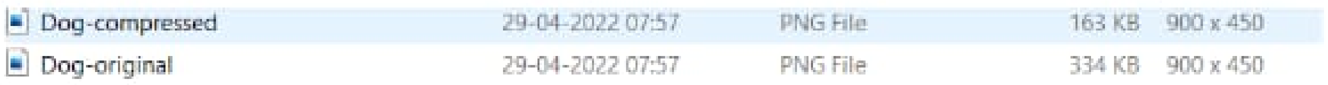


Figure 15: Child Images (Original and Compressed)



Dog-compressed	29-04-2022 07:57	PNG File	163 KB	900 x 450
Dog-original	29-04-2022 07:57	PNG File	334 KB	900 x 450

Figure 16: Sizes of Child Images

Now similarly as of last example, here Fig:15 and Fig:16 is another example for image compression.

3.2 Semi Supervised Learning

It is a special form of classification. Traditional classifiers use only labeled data (feature / label pairs) to train. Labeled instances however are often difficult, expensive, or time consuming to obtain, as they require the efforts of experienced human annotators. Meanwhile unlabeled data may be relatively easy to collect, but there has been few ways to use them. Semi-supervised learning addresses this problem by using large amount of unlabeled data, together with the labeled data, to build better classifiers. Because semi-supervised learning requires less human effort and gives higher accuracy, it is of great interest both in theory and in practice.

3.2.1 Can we really learn anything from unlabeled data?

Yes we can – under certain assumptions. It's not magic, but good matching of problem structure with model assumption.

Many semi-supervised learning papers start with an introduction like: "labels are hard to obtain while unlabeled data are abundant, therefore semi-supervised learning is a good idea to reduce human labor and improve accuracy". Do not take it for granted. Even though you (or your domain expert) do not spend as much time in labeling the training data, you need to spend reasonable 4 amount of effort to design good models / features / kernels / similarity functions for semi-supervised

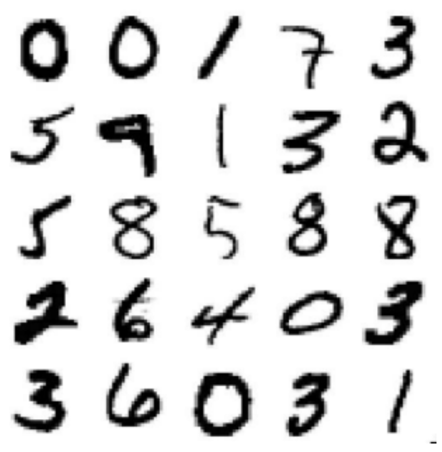


Figure 17: A Small Portion of MNIST Dataset

learning. In my opinion such effort is more critical than for supervised learning to make up for the lack of labeled training data.

3.2.2 Example application of semi-supervised learning

A common example of an application of semi-supervised learning is a text document classifier. This is the type of situation where semi-supervised learning is ideal because it would be nearly impossible to find a large amount of labeled text documents. This is simply because it is not time efficient to have a person read through entire text documents just to assign it a simple classification.

Semi-supervised learning is ideal for medical images, where a small amount of training data can lead to a significant improvement in accuracy. For example, a radiologist can label a small subset of CT scans for tumors or diseases so the machine can more accurately predict which patients might require more medical attention.

So, semi-supervised learning allows for the algorithm to learn from a small amount of labeled text documents while still classifying a large amount of unlabeled text documents in the training data.

3.2.3 Application on MNIST dataset

- Labelled training data is not easily available for supervised learning.Weare using MNIST DATASET containing 1797 images to demonstrate the application of semi-supervised learning.Standard logistic regression model for this has 96.9% accuracy.

- Suppose we take 50 random samples as training set then logistic regression gives 83.3%.

- Instead of 50 random samples,if we make 50 clusters using K means and use image nearest to each centroid as training set, Logistic regression accuracy jumps to 92.2

- Now if we propagate representative image label to entire cluster,Logistic regression improves to 93.3%

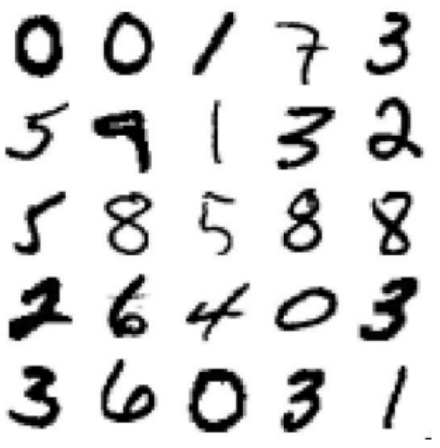


Figure 18: A Small Portion of Fashion MNIST Dataset

- If we propagate representive image label to only 20% items closest to centroid,Logistic regression improves to 94%

3.2.4 Application on Fashion MNIST dataset

- Now we apply semi-supervisd learning on Fashion MNIST dataset consisting of 60000 images where Standard Support vector classifier gives 84.36%.
- Suppose we take 2000 random sample then accuracy is 77.83%. If we propagate representive image label to only 40% items closest to centroid then Standard Support vector classifier jumps to 80.24%
- On propagating to entire cluster the classifier gives 80.52%

4 Disadvantages of K-means Clustering

Suppose we have a data which are the points drawn from 2 concentric circles with some noise added.Now we would like to an algorithm to be able to cluster these points into the 2 circles that generated them. The data looks like Fig: 19.

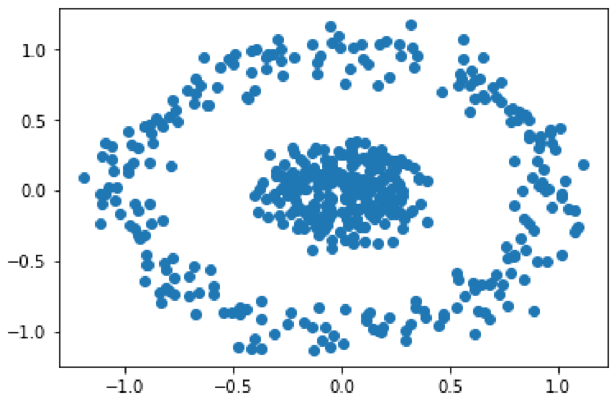


Figure 19: Raw Data Used

We should first try to cluster them out using K-Means algorithm.

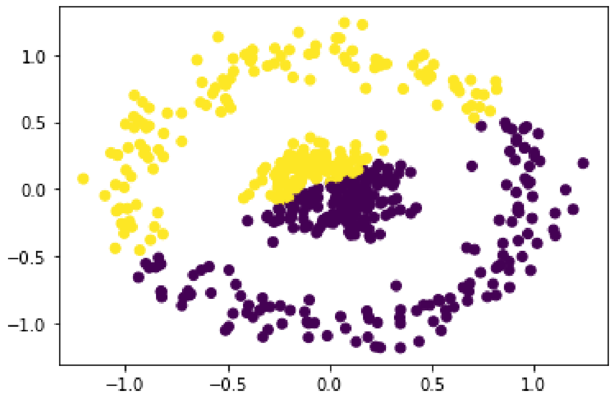


Figure 20: Clustering of Above Data by K-Means Clustering

Obviously K-Means is not going to work here as shown in Fig: 20. K-Means operates on Euclidean distance and it assumes that the clusters are roughly spherical. This data breaks these assumptions. In this case we should use spectral clustering.

4.1 Spectral Clustering

4.1.1 What is done in spectral clustering

This algorithm takes a similarity matrix which is also known as **adjacency matrix** between the instances and creates a low-dimensional embedding from it(i.e. reduces its dimensions) and then it uses another clustering algorithm in this lower dimensional space to obtain the desired clusters.

4.1.2 Steps to be followed

1. Transform the data into a **Distance Matrix**.
2. Convert the Distance Matrix into **Adjacency Matrix** A.
3. Compute the **Degree Matrix** D from A.

4. Using A and D compute the **Laplacian Matrix** L.
5. Find the eigen values and eigen vectors of L.
6. For a k-dimensional space, using the eigen vectors of k largest eigen values computed from the previous step, form a matrix.
7. Normalize the vectors
8. Cluster the data points in k-dimensional space.

4.1.3 Nearest neighbors graph

The illustration we have used to demonstrate the usage of spectral clustering is a random plot of concentric circles. But there are some ways to treat our data as graph and the easiest way is to construct a k-nearest neighbors graph. A k-nearest neighbors graph treats every data point as a node . An edge is then drawn from each node to k of its nearest neighbors in the original space. For k = 5 or 10 usually work pretty well.

4.1.4 Adjacency Matrix

- We can represent a graph as an adjacency matrix, where the row and column indices represent the nodes, and the entries represent the absence or presence of an edge between the nodes.
- Presence and absence of edges between ith and jth nodes are represented as 1 and 0 respectively. If there is no edge between ith and jth nodes then $A_{ij} = 0$ and else $A_{ij} = 1$.
- If the edges were weighted then instead of 1s and 0s the elements of the matrix would have been the weights of edges.
- If the graph is undirected then the adjacency matrix is symmetric. If it is directed then it may happen that $A_{ij} \neq A_{ji}$

4.1.5 Degree Matrix

- The degree matrix D is a diagonal matrix where D_{ii} is the degree of node i.
- The degree of ith node is computed as

$$D_{ii} = \sum_{j=1}^{n} A_{ij}$$

- If the graph is not weighted then D_{ii} represents the number of edges connected to ith node.
- In a directed graph there are notions of in-degree and out-degree.

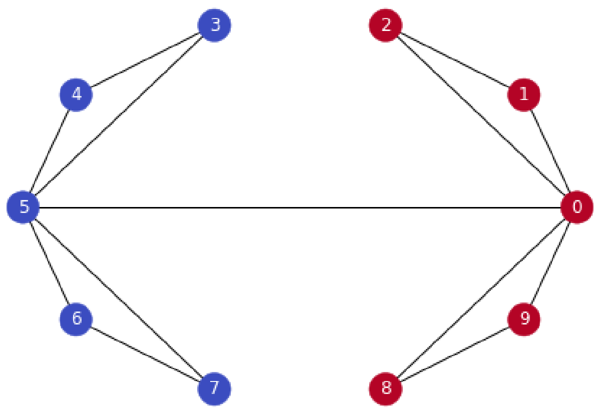
4.1.6 Graph Laplacian

- The laplacian is just another matrix representation of graph.
- To calculate normal laplacian we just substract the adjacency matrix from degree matrix.

$$L = D - A$$

4.1.7 Interpretations of Eigenvalues of Graph Laplacian

- When the graph is completely disconnected, all ten of our eigenvalues are 0. As we add edges, some of our eigenvalues increase.
- If the graph is completely connected then the number of 0 eigenvalue is 1
- In fact, the number of 0 eigenvalues corresponds to the number of connected components in our graph.
- The first nonzero eigenvalue is called the **spectral gap**. The spectral gap gives us some notion of the density of the graph. If this graph was densely connected (all pairs of the n nodes had an edge), then the spectral gap would be n.
- The second eigenvalue is called the **Fiedler value**, and the corresponding vector is the **Fiedler vector**. The Fiedler value approximates the minimum graph cut needed to separate the graph into two connected components. Recall, that if our graph was already two connected components, then the Fiedler value would be 0. Each value in the Fiedler vector gives us information about which side of the cut that node belongs.
  illustration:

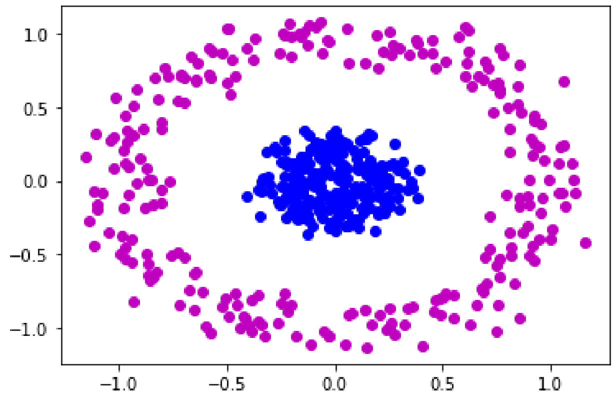
  Remember that zero eigenvalues represent connected components. Eigenvalues near zero are telling us that there is almost a separation of two components. Here we have a single edge, that if it didn't exist, we'd have two separate components. So the second eigenvalue is small.

4.1.8 Computation

- Taking a look at the graph and an adjacency matrix is built.
- Then the Graph Laplacian is created by subtracting the adjacency matrix from the degree matrix.
- The eigenvalues of the Laplacian indicates the number of clusters.
- The vectors associated with those eigenvalues contain information on how to segment the nodes.
- Finally, K-Means is performed on those vectors in order to get the labels for the nodes.

4.1.9 Example

The example concentric circle like data which is shown where k-means fails to form clusters can be clustered successfully using spectral analysis. The following figure proves this claim.



4.2 Density Based Clustering

- K-means does not work well when data has outliers. For example: In Fig:21 given below, we can see that the clusters would have been different if the outliers were taken care of. To rectify this we will use DBSCAN clustering.

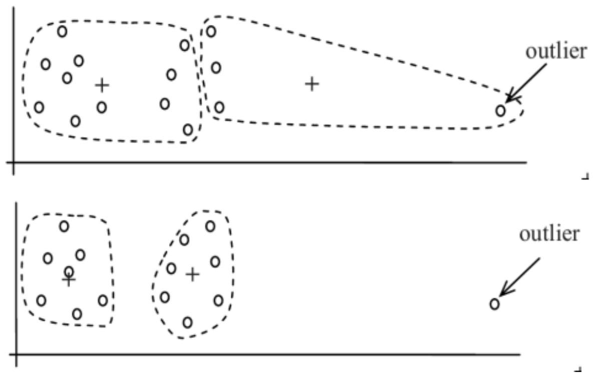


Figure 21: Example Used in DBSCAN

Fundamentally, all clustering methods use the same approach i.e. first we calculate similarities and then we use it to cluster the data points into groups or batches. Here we will focus on Density-based spatial clustering of applications with noise (DBSCAN) clustering method. Clusters are dense regions in the data space, separated by regions of the lower density of points. The **DBSCAN algorithm is based on this intuitive notion of "clusters" and "noise"**. The key idea is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.

4.2.1 Steps in DBSCAN Algorithm

DBSCAN algorithm can be abstracted in the following steps :

1. Find all the neighbor points within eps and identify the core points or visited with more than MinPts neighbours

2. For each core point if it is not already assigned to a cluster, create a new cluster

3. Find recursively all its density connected points and assign them to the same cluster as the core point. A point a and b are said to be density connected if there exist a point c which has a sufficient number of points in its neighbors and both the points a and b are within the eps distance. This is a chaining process. So, if b is neighbor of c, c is neighbor of d, d is neighbor of e, which in turn is neighbor of a implies that b is neighbor of a

4. Iterate through the remaining unvisited points in the dataset. Those points that do not belong to any cluster are noise.

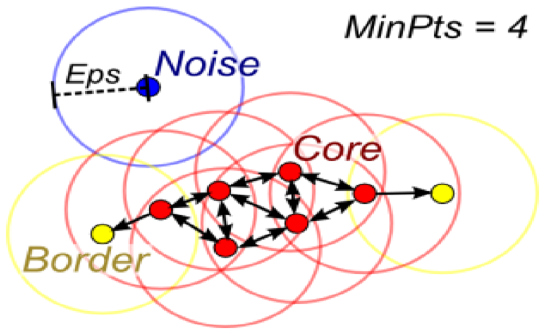


Figure 22: Pictorial Description of DBSCAN

The Fig: 22 gives some visualization over DBSCAN algorithm.

4.2.2 Example

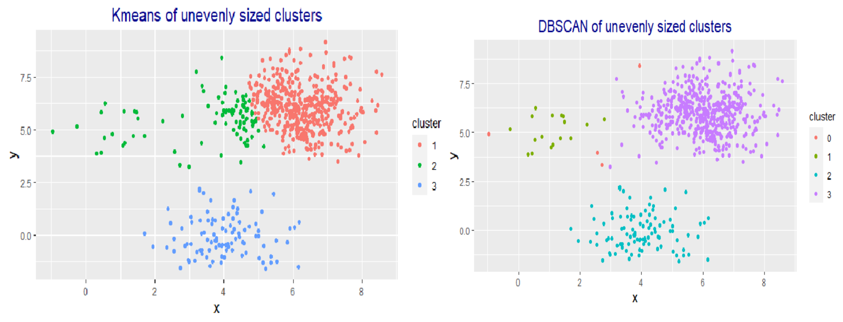


Figure 23: Kmeans Clustering vs DBSCAN on Same Dataset

Fig: 23 is clearly showing that to cluster the data points Kmeans is not working properly but DBSCAN is.

Acknowledgement

We express our gratitude towards Dr.Minerva Mukhopadhyay for her valuable feedback and constant guidance on this project.

References

1) Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition by Aurélien Géron
2) https://towardsdatascience.com/spectral-clustering-aba2640c0d5b#:~:text=Spectral%20clustering%20is%20a%20technique,non%20graph%20data%20as%20well.
3) Arthur, D. and Vassilvitskii, S. "k-means++: the advantages of careful seeding". ACM-SIAM symposium on Discrete algorithms. 2007
4) Datasets:
a)MNIST-https://data.deepai.org/mnist.zip
b)Fashion MNIST:https://www.kaggle.com/datasets/zalando-research/fashionmnist/download