CS 512 Project Report

Topic: Implementing DBSCAN algorithm on clustered data using Python libraries Pandas, Numpy and showing implementation using Matplotlib.

Team Members:

- Advait Bhat (NetID: ab2253, Email: advait.bhat@rutgers.edu)
- Rohit Upadhyay (NetID: rru9, Email: rohit.upadhyay@rutgers.edu)
- Sahil Raut (NetID: srr149, Email: sahil.raut@rutgers.edu)
- Rasika Hasamnis (NetID: rmh229, Email: rmh229@scarletmail.rutgers.edu)

Project Description:

Deterministic Algorithm Animation and Algorithm Snippets:

- We implemented a Density-based spatial clustering of applications with a noise (DBSCAN) algorithm.
- Given a set of points in some space, it groups together points that are closely packed together; points that lie alone in low-density regions are marked as outliers. DBSCAN is one of the most common clustering algorithms.
- We have done the working of the algorithm with a sample data set of customers in a mall.
- We have used a novel approach to show the working of the algorithm using animation on a website front-end.

Data Description / Index and Data Transformation:

For both the datasets, we had to transform the data from whatever type it was, to numeric type because we have implemented it in NumPy which works best with numbers and only with one type of data (hence numeric). Spending score is a proxy metric here used on a scale of 1–100. Higher spend score means the customer has a higher affinity towards the purchase of items and thereby higher revenue for the shopping mall. The table below shows what the data represents and how we transformed it to fit our algorithm

Mall Customer Dataset.

Column Name / Number	**Data Meaning**	**Unique Column Values**	**Unique Transformed Column Values**
1 - CustomerID	Customer Identification Number	Numeric values representing Customer ID	No transformation done
2 - Gender	Gender of the customer	Male or Female	1-12, respectively
3 - Age	Age of the customer	Numeric values representing Customer Age	No transformation done
4- Annual Income (k$)	Annual Income of the customer in k$	Numeric values representing Customer Income in k$. Example – 10 is 10000$.	No transformation done
5-Spending Score (1-100)	The Spending score	Numeric values representing Customer Spending score	No transformation done

A snippet of Input Data:

Out[6]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Pseudocode:

Pseudocode For The DBSCAN Algorithm:

1. Assuming all of the points in the data set are outliers, iterate over them.
2. We now retrieve all points density-reachable from any arbitrary point p.
3. P is a core point and a new cluster is formed if the number of points in the Ɛ neighborhood is equal to or greater than the min Points.
4. Now, we go through each point in the newly formed cluster in Breadth-First Search fashion and find the points in their respective Ɛ neighborhood, ie. we are finding all indirectly density reachable points from p.
5. We move on to the next point still marked as an outlier and repeat steps 1 through 4, Once we have retrieved all the indirectly density reachable points, and have completed out the cluster.
6. The algorithms stop once all the points in the data sets are visited.

Pseudocode For K-Means Algorithm:

There are 3 main steps in the K-Means algorithm (known also as Lloyd's algorithm):

1. First, we split samples into initial groups by using seed points. The nearest samples to these seed points will be used to create initial clusters.
2. We then Calculate samples distances to groups' central points (centroids) and assign the nearest samples to their cluster.
3. The third step is to calculate newly created (updated) cluster centroids.

Then repeat steps 2 and 3 until the algorithm converges.

Time and Space Complexity

Time Complexity for DB Scan.

Our DBSCAN algorithm visits each point of the database multiple times. Time complexity in the algorithm is mostly dependent on the number of regionQuery invocations. The algorithm executes exactly one such query for each point, and if an indexing structure is used that executes a neighborhood query in O(n log n), then an overall average runtime complexity of O(n log n) is obtained.

When an index structure, or on degenerated data (e.g. all points within a distance less than ε) is not used, the worst-case run time complexity remains O(n^2).

Space Complexity for DB Scan.

At every point in the memory, we will be having the positions of the n points, their various labels, the neighbors of the current point, and the neighbors of a particular neighbor in the case that the point turns out to be a core point. Therefore this required N space. The Space complexity of the algorithm O(n)

Time Complexity for K-means

time complexity is O(n*k*i)

where (n is the total number of elements, k is the total number of clusters, i is the total number of iterations).

If k and d (the dimension) are fixed, the problem can be exactly solved in time O(ndk+1 log n), where n is the number of entities to be clustered.

Space Complexity for K-means

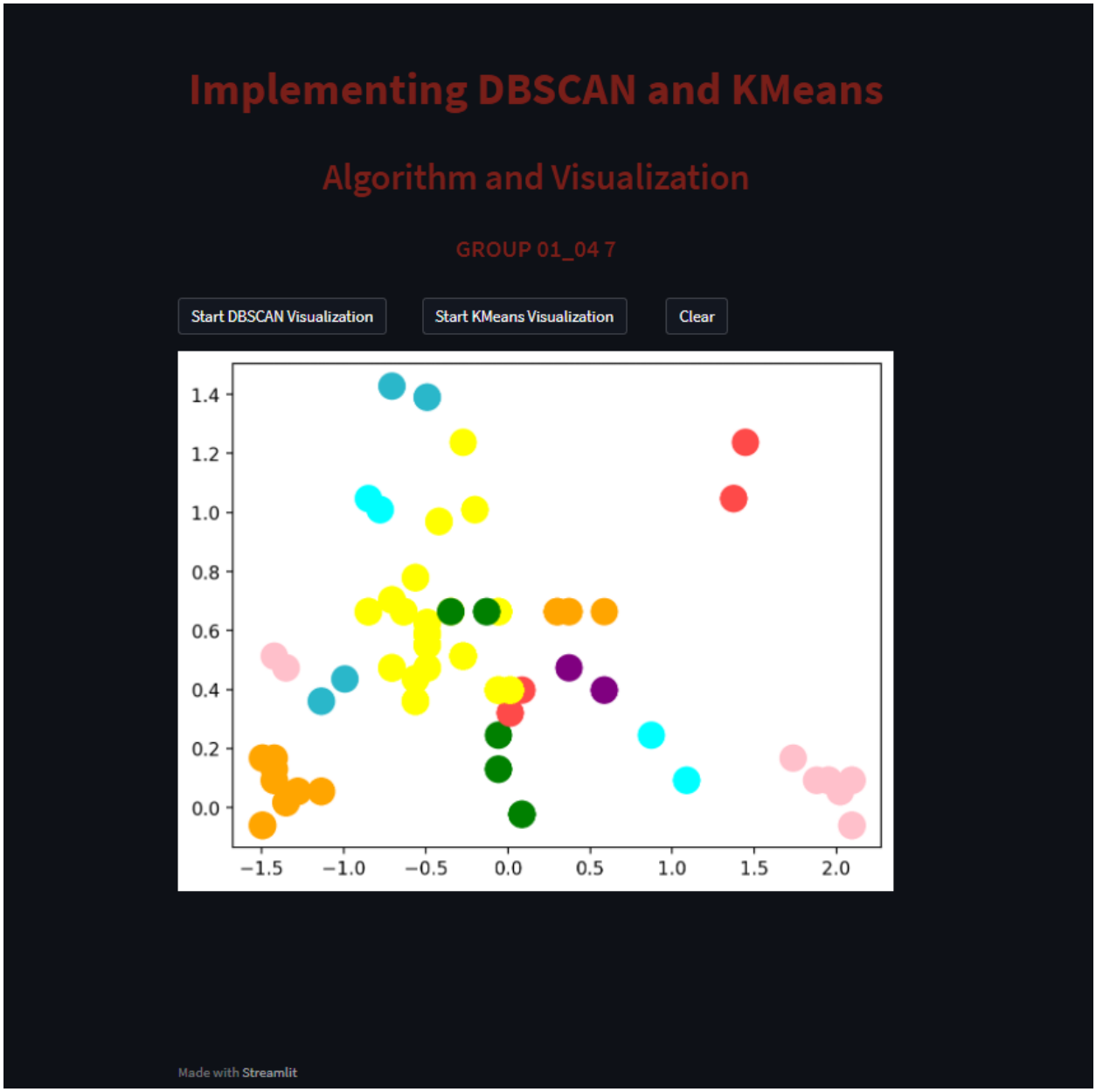
O((n+k)*d)

where(n is the total number of elements, k is the total number of clusters, d is the total number of attributes)

Working Application and how it answers questions proposed:

Link: https://share.streamlit.io/urhit45/dbscan/main.py

The application that is deployed on the link above is used to visualize and form clusters to do customer segmentation on the dataset. We tested the performance of our DB Scan clustering algorithm with K means algorithm to perform customer segmentation on the mall customer dataset as seen below.



Mall_Customers Dataset:

1. Which attributes predict the more purchases made by the customer (given by the spending score)?
    - While performing Exploratory analysis on the Mall dataset, we discovered that Age and Spending scores have a negative correlation, as age increases the spending score decreases. Alternatively, we also observed a very small positive correlation between Annual Income and Spending Score. However, the score correlation was far too low to be of any significance.
2. What other correlations or trends can we find in the data?
    - We found there was no particular strong correlation between the attributes. Two attributes that were somewhat important are spending score and age. We see as the age increases the spending score decreases.
3. Can the data be successfully clustered in a meaningful way using our implementation of the algorithm?
    - We divided the data into different clusters depending upon the 3 categories: Age, Annual Income, and Spending Score.
    - We found that the data was divided into clusters depending upon the implementation of the algorithm.

Conclusion:

We ran both the clustering algorithms on the dataset and here is what we observed:

DBSCAN failed to generate reasonable clusters. It is due to its problems in recognizing clusters of various densities.

In turn, K-Means and Affinity Propagation algorithms created reasonable 6 clusters.

No single algorithm is the best for all purposes. This means that there are situations where DBSCAN is very performant, while sometimes its performance is very bad. Density clustering seems to correspond more to human intuitions of clustering, rather than the distance from a central clustering point.

When you don’t know the number of clusters hidden in the dataset and there’s no way to visualize your dataset, it’s a good decision to use DBScan. DBSCAN produces a varying number of clusters, based on the input data.