CSE 601 Data Mining and Bioinformatics

Clustering Algorithms

Gautam Othayoth Ganapathiyadan - 50208358
Moonis Javed - 50208261
Surabhi Singh Ludu - 50207139

K-MEANS

K-means is a clustering algorithm used in Data Mining Applications to group together similar feature values.

The following steps are followed to cluster -

- The Initial parameters are selected - the number of clusters, the initial centroid points, and the number of iterations to run the algorithm.
- Group the data into the various clusters based on smallest euclidean distance for the initial points provided.
- Find the centroid of all the clusters and the these centroids are the cluster centroid points used to run the next iteration.
- Run the algorithm on these centroids for as many times as mentioned in the number of iteration parameter provided
- Make an n*n matrix (where n is the number of records) of clusters which compares if the cluster of a record matches with other clusters (1 if it matches and 0 if it doesn't match).
- Make an n*n matrix with the ground truth which compares if the cluster of a record matches with other clusters.
- Find the variables that are required to find the Rand and Jaccard Index namely m00(an element is 0 in the result matrix and ground truth matrix), m01(an element is 0 in the result matrix and 1 in the ground truth matrix), m10(an element is 1 in the result matrix and 0 in the ground truth matrix) and m11(an element is 1 in the result matrix and ground truth matrix).
- Compute Rand (m11+m00)/(m11+m01+m10+m00) and Jaccard Index m11/(m11+m10+m01)

Advantages

- It is simple and easy to implement.
- It works well on clusters that are spherical.
- It is fast and efficient as the time complexity is linear with the number of records.
- It produces tight clusters.
- It is scalable.

Disadvantages

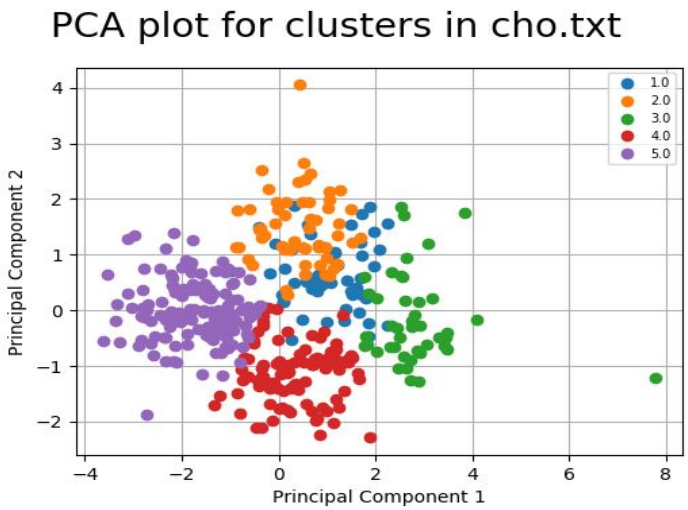
- Depends heavily on hyper parameters like number of clusters and initial centroids.

- Differing density can also cause aberrations, we may merge high density clusters and break low density clusters.
- K-means is a center based concept, It can't detect irregular shape clusters.

Result

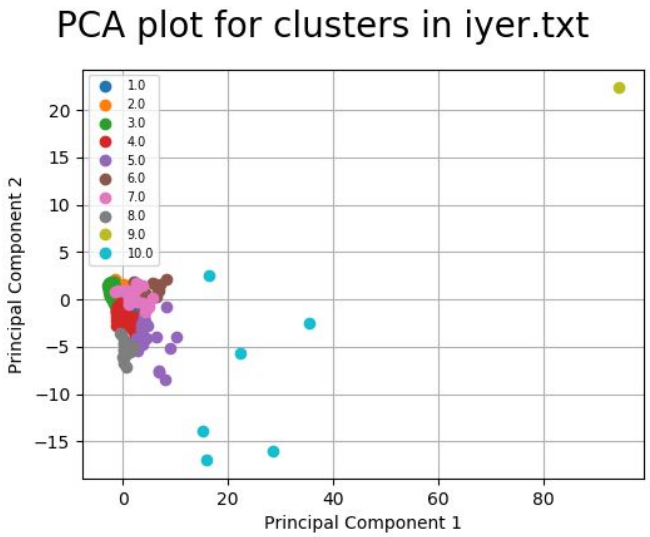
cho.txt

Clusters: 5, Number of Iterations: 30, Initial Centroid Gene IDs: 10,22,45,96,103
The Rand Index is 0.8007060592230664
The Jaccard Index is 0.4056920983107838



iyer.txt

Clusters: 10, Number of Iterations: 70, Initial Centroid Gene IDs: 10,22,45,96,103,224,278,312,378,456
The Rand Index is 0.7680488160754838
The Jaccard Index is 0.35296757428067504



DBSCAN ALGORITHM

Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm based on the idea that clusters are dense regions in the data space, separated by regions of lower object density. A cluster is defined as a maximal set of density-connected points. This algorithm can discover clusters of arbitrary shape.

The following steps are followed to cluster -

1. Two values ***Epsilon*** , i.e. the maximum distance between two points for them to be considered as in the same cluster, and ***MinPoints***, i.e the number of points in a cluster for a point to be considered as a core point, are passed as a parameter to the script.
2. We start by visiting one point P and getting all its neighbors within Epsilon distance
3. If the numbers of neighbors of P are less than MinPoints points, then we mark that point as noise(-1)
4. If the number of neighbors of P are more than MinPoints, then we start expanding the cluster
5. We mark the point P in cluster C and then, visit each of it's neighbor points P'
6. If P' was marked as noise before, we mark it in that cluster
7. If the point P' was unvisited, then we mark it in cluster C and again look for its neighbors and add them to neighbors of original point P
8. We repeat this steps 5 to 7 till all neighbors P' of P are marked
9. We repeat steps 2 to 8 till all the points in the dataset are visited
10. Once all points are labelled :
    - Make an n*n matrix (where n is the number of records) of clusters which compares if the cluster of a record matches with other clusters (1 if it matches and 0 if it doesn't match).
    - Make an n*n matrix with the ground truth which compares if the cluster of a record matches with other clusters.
    - Find the variables that are required to find the Rand and Jaccard Index namely m00(an element is 0 in the result matrix and ground truth matrix), m01(an element is 0 in the result matrix and 1 in the ground truth matrix), m10(an element is 1 in the result matrix and 0 in the ground truth matrix) and m11(an element is 1 in the result matrix and ground truth matrix).
    - Compute Rand (m11+m00)/(m11+m01+m10+m00) and Jaccard Index m11/(m11+m10+m01)

Advantages

- Resistant to Noise
- Can handle clusters of different shapes and sizes
- DBSCAN does not require you to know the number of clusters in the data a priori, as opposed to k-means.
- Mostly insensitive to the ordering of points in the database

Disadvantages

- Cannot handle varying densities
- Sensitive to parameters—hard to determine the correct set of parameters
- Not partitionable for multiprocessor systems unlike K-means with Hadoop

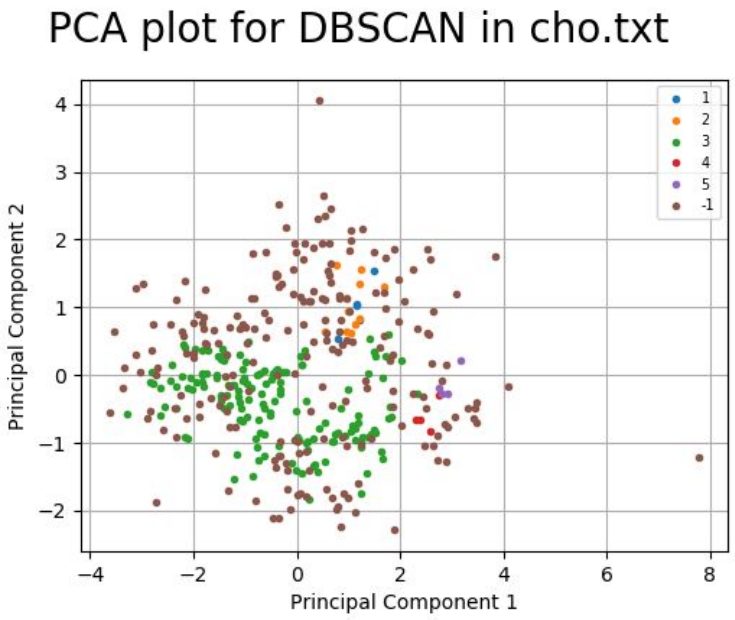
- Does not work well in case of high dimensional data

Result

Cho.txt (eps = 1.03, m = 4, number of clusters = 5)

Rand Index: 0.5476120164299713

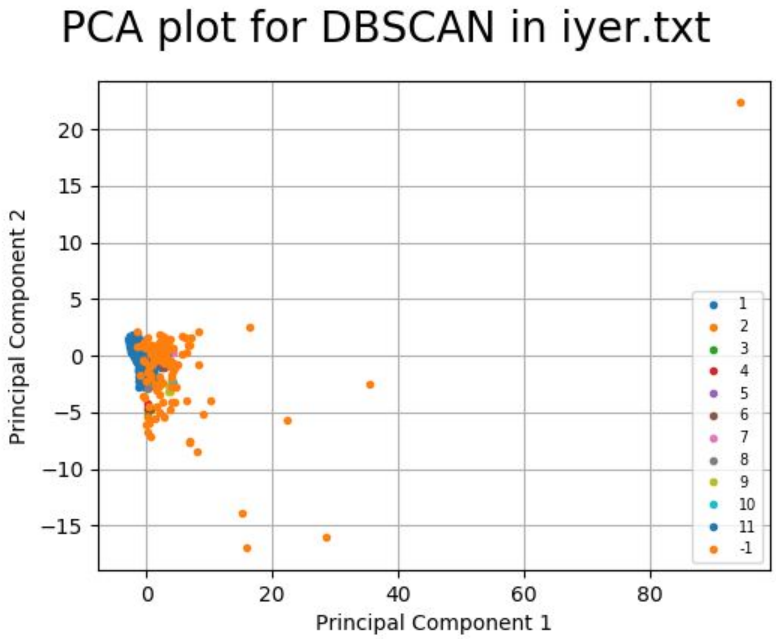
Jaccard Index: 0.20318706260639305



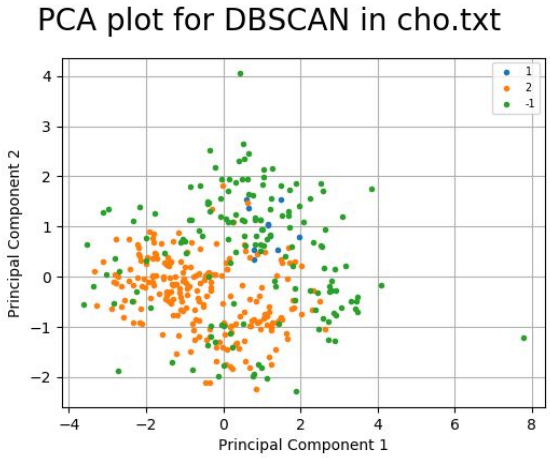
Iyer.txt (eps = 1.15, m = 2, number of clusters = 11)

Rand Index: 0.5392028852665093

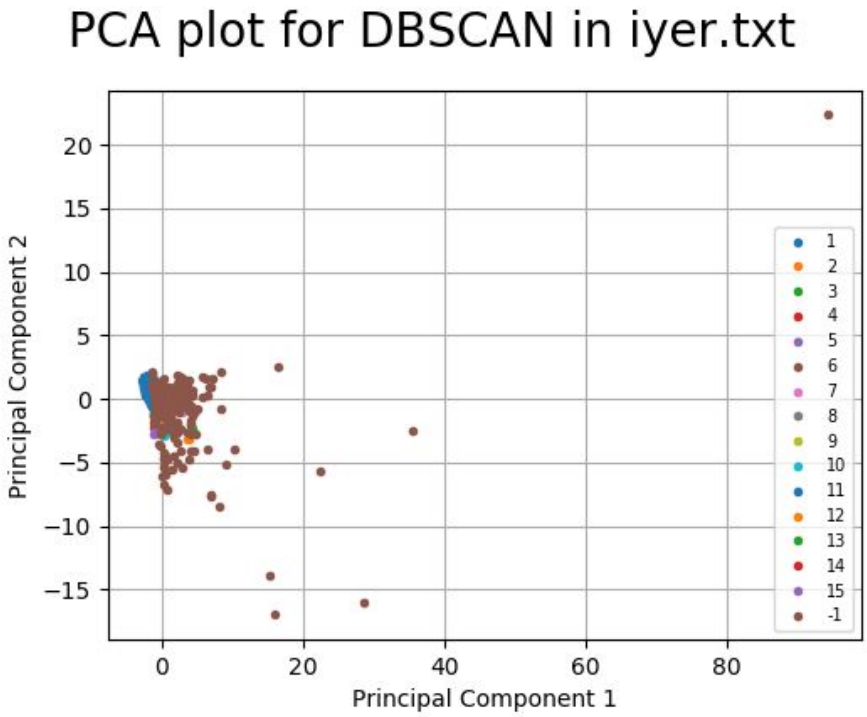
Jaccard Index: 0.22855872276192088



We tried varying the eps and m values to get the best jaccard coefficient possible.
For cho.txt, the best Rand Index and jaccard Index we got were for eps = 1.26, minPts = 10,
But the clusters gotten were 2
Rand Index: 0.5631694810598942
Jaccard Index: 0.24583439549489003



For iyer.txt, the best Rand Index and jaccard Index we got were for eps = 0.89 minPts = 2,
But the clusters gotten were 15
Rand Index: 0.667865119776721
Jaccard Index: 0.2906148867313916



K-MEANS HADOOP

K-means is a clustering algorithm used in Data Mining Applications to group together similar objects. It can be scaled for larger datasets by using mapreduce architecture.

The following steps are followed to cluster -

- Mapper
    - The Initial parameters are selected - the number of clusters, the initial centroid points, and the number of iterations to run the algorithm.
    - For each data point find the nearest centroid using euclidean distance
    - Emit the cluster id with the point

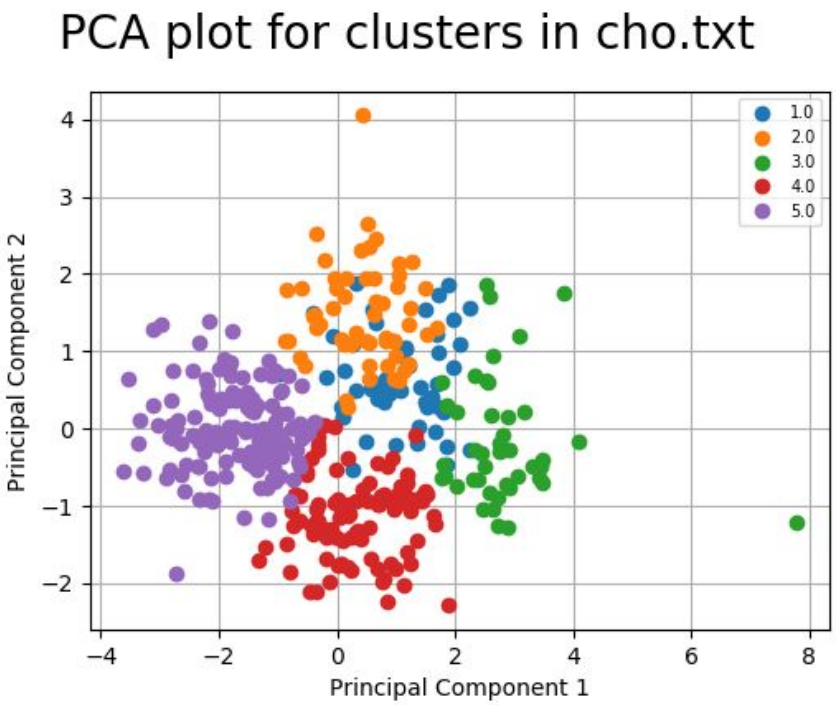
- Reducer
    - For each point belonging to the same cluster id compute the mean
    - Emit the new centroid

- Repeat mapreduce till the centroids don't change
- Map all the data points to the new centroids
- Make an n*n matrix (where n is the number of records) of clusters which compares if the cluster of a record matches with other clusters (1 if it matches and 0 if it doesn't match).
- Make an n*n matrix with the ground truth which compares if the cluster of a record matches with other clusters.
- Find the variables that are required to find the Rand and Jaccard Index namely m00(an element is 0 in the result matrix and ground truth matrix), m01(an element is 0 in the result matrix and 1 in the ground truth matrix), m10(an element is 1 in the result matrix and 0 in the ground truth matrix) and m11(an element is 1 in the result matrix and ground truth matrix).
- Compute Rand (m11+m00)/(m11+m01+m10+m00) and Jaccard Index m11/(m11+m10+m01)

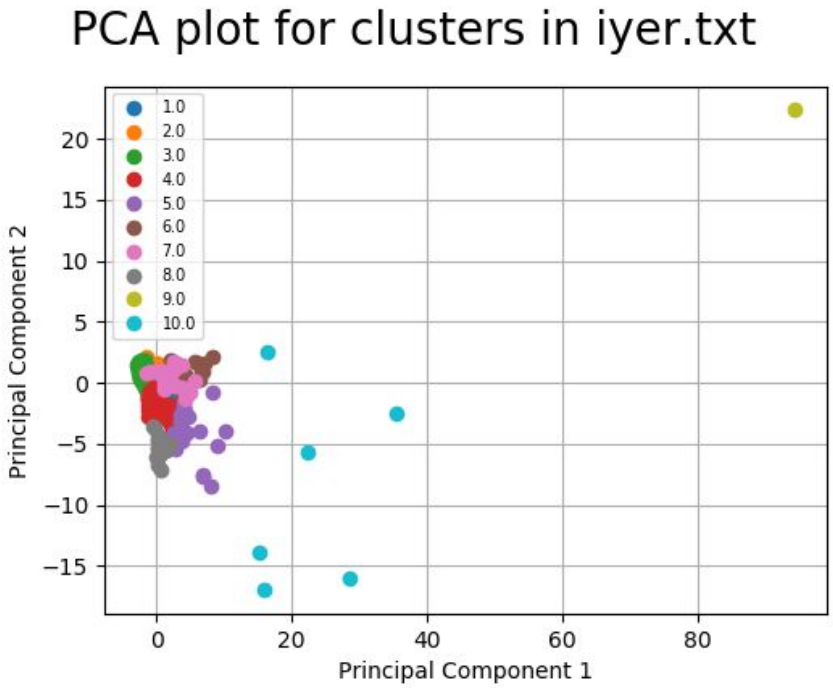
Comparison with Serial K-Means

- **Pros**
    - Can be scaled to multiple node clusters thereby decreasing runtime
    - Can be scaled for bigger datasets
- **Cons**
    - SInce our dataset is small, it adds a large overhead for each iteration thereby increasing runtime

Result
cho.txt
Rand Index: 0.800706059223
Jaccard Index: 0.405692098311



iyer.txt
Rand Index: 0.768048816075
Jaccard Index: 0.352967574281



Hierarchical Agglomerative Clustering

HAC is a clustering algorithm used in Data Mining Applications to group together similar objects.

The following steps are followed to cluster -

1. A euclidean matrix is created for data points where arr[i][j] signifies the euclidean distance between ith and jth element. The diagonal of this matrix is 0. It is of size n*n where n is the number of data points
2. Each point represents a cluster of size 1 containing that point
3. Steps 3 - 7 are repeated till the number of rows in cluster is equal to the number of required clusters
4. Find the smallest non zero element in matrix arr[i][j] = min euclidean
5. The two points i and j are merged to form a cluster containing points of cluster i and cluster j
6. A new row and column is added and smallest distance is calculated for every other point and the points of cluster [distance = min { distance(k,l) where k ∈ cluster points and l ∈ every other cluster points }
7. The rows and columns of clusters i and j are removed from the matrix
8. Make an n*n matrix (where n is the number of records) of clusters which compares if the cluster of a record matches with other clusters (1 if it matches and 0 if it doesn't match).
9. Make an n*n matrix with the ground truth which compares if the cluster of a record matches with other clusters.
10. Find the variables that are required to find the Rand and Jaccard Index namely m00(an element is 0 in the result matrix and ground truth matrix), m01(an element is 0 in the result matrix and 1 in the ground truth matrix), m10(an element is 1 in the result matrix and 0 in the ground truth matrix) and m11(an element is 1 in the result matrix and ground truth matrix).
11. Compute Rand (m11+m00)/(m11+m01+m10+m00) and Jaccard Index m11/(m11+m10+m01)

Pros

- Conceptually simple
- Good for small datasets

Cons

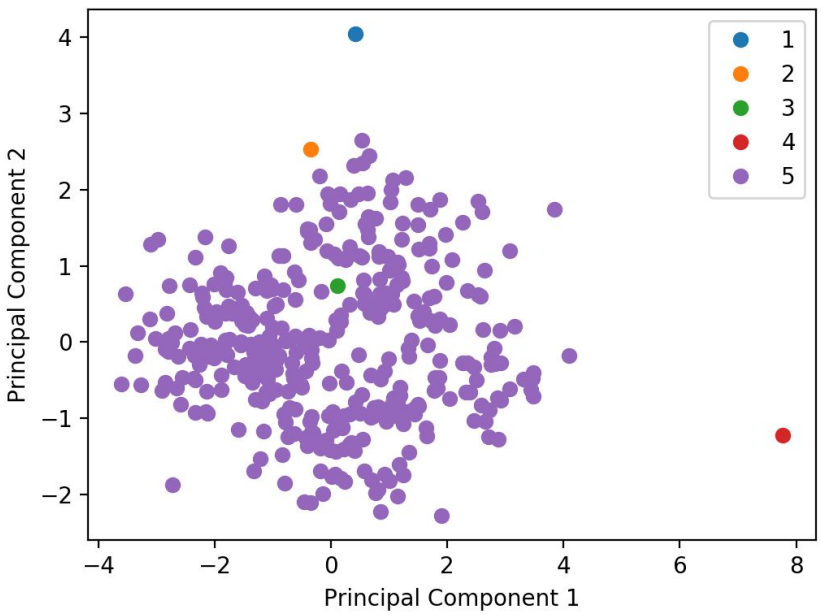
- Cluster merging/splitting is permanent and the error in this is impossible to fix
- Min linkage is sensitive to noise and outliers

Result

cho.txt

Rand Index: 0.240274906709

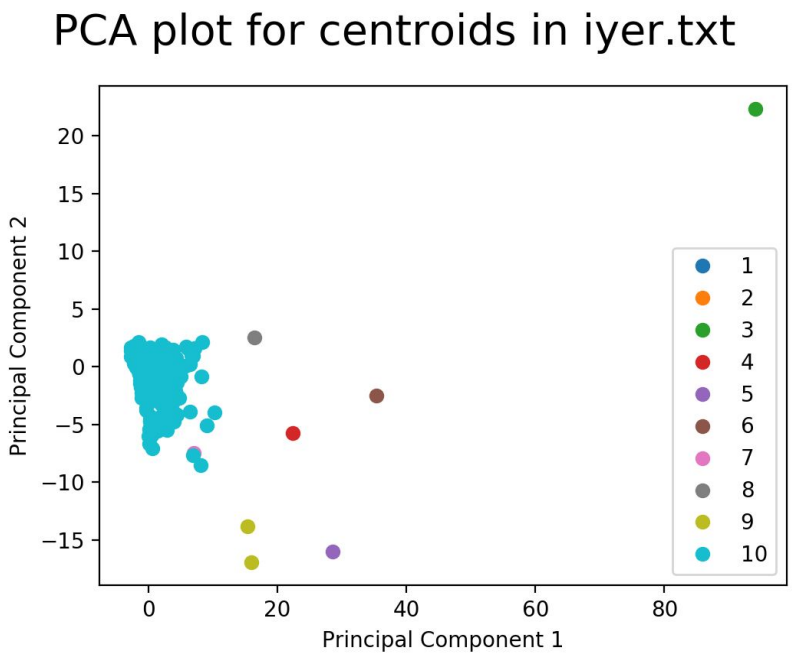
Jaccard Index: 0.228394977574



iyer.txt

Rand Index: 0.188286835597

Jaccard Index: 0.158243096966



Comparison of all four Clustering approaches

Measure	Serial K-means	HCA	DBSCAN	K-means HADOOP
Run time(average, cho.xtx)	11.82 sec	12.83 sec	1.42 sec	5 min 15 sec
Outliers	No	No	Yes	No
Rand Index	0.800706	0.240275	0.547612	0.800706
Jaccard Index	0.405692	0.228395	0.203187	0.405692

- From the above comparison we can see that, Serial K-means out-performs other clustering methods that we experimented with for the given datasets, cho.txt and iyer.txt
- Moreover, if we increase the nodes in K-means Hadoop, we can get good results in small runtime for large datasets, which makes it more scalable.
- If we were to use clustering to identify outliers in any given dataset, DBSCAN would be the algorithm of choice as it identifies outliers easily.