**Student Name: Joseph Ayo Ojediran**

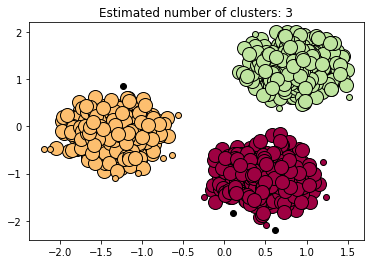
**Student No: 21078637**

**Course: Data Mining and Discovery**

**Github Link:** [**https://bit.ly/3GM2kFr**](https://bit.ly/3GM2kFr)

**Group 20**

**Density-Based Clustering**



The code uses the make\_blobs function from sklearn to generate sample data in the form of three clusters. The make\_blobs function generates a specified number of samples and clusters, with each cluster having a specified center and standard deviation. In this case, the function is generating 1500 samples and 3 clusters with centers at [2, 1], [-4, -2], and [1, -4], and a standard deviation of 0.7.

The data is then standardized by removing the mean and scaling to unit variance using the StandardScaler class from sklearn.preprocessing. This is done to ensure that all the features are in the same scale and to avoid any bias in the algorithm due to the scale of the features.

The function named "display" is defined that takes three input parameters: epsilon, minimum\_samples, and data. These are the three key parameters that are used to control the DBSCAN algorithm. The epsilon parameter defines the radius of the neighborhood around a data point and the minimum\_samples parameter defines the minimum number of points that need to be in the neighborhood of a point for it to be considered a core point. The data parameter is the input dataset on which the algorithm needs to be run.

The DBSCAN class is imported from sklearn.cluster and an instance of DBSCAN is created by passing the eps and min\_samples parameters to the class constructor. The fit() method is then used to fit the model to the input data. The core\_samples\_mask attribute of the DBSCAN object is used to identify which points are core points, and the labels\_ attribute is used to obtain the labels assigned to each data point by the algorithm.

The number of clusters in the labels is calculated by removing the -1 label, which represents noise and taking the length of the set of unique labels. Next, the unique labels are used to create colors for the clusters. The function then plots the data points, with the points in each cluster colored differently. Points that are not part of any cluster are plotted in black.

The function is called with the input parameters epsilon = 0.25, minimum\_samples = 6, and data = X. The resulting plot shows the three clusters generated by the DBSCAN algorithm and the points that are considered as noise.

In summary, the code uses the DBSCAN algorithm to cluster the input data into different groups and visualizes the result using a scatter plot. The key parameters of the algorithm epsilon and minimum\_samples can be changed to change the clustering results. The function makes it easy to change the parameters and visualize the results, making it easy to experiment and find the optimal parameters for a given dataset.