NAME HUZAIFA SALMAN

ROLL NO DT-34

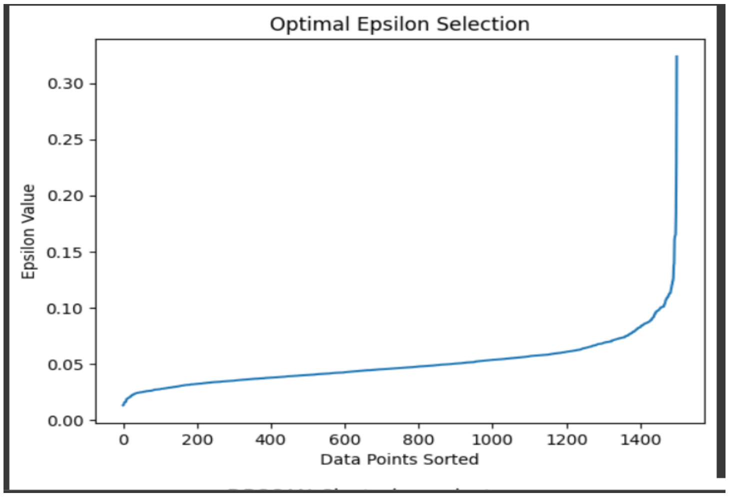
DM LAB 6

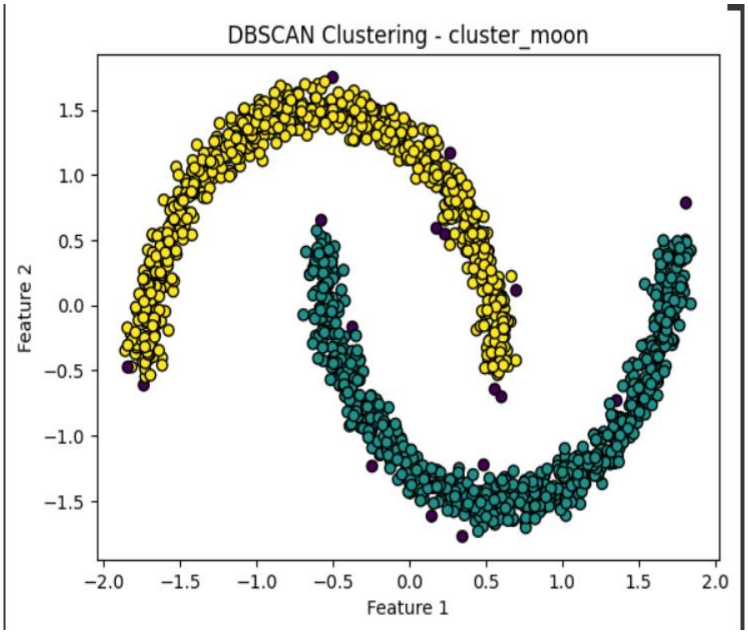
1. Apply the DBSCAN on the following dataset a. 1.1 cluster_blob.csv b. 1.2 cluster_moon.csv c. 1.3 cluster_circles.csv d. 1.4 cluster_two_blob_outliers.csv

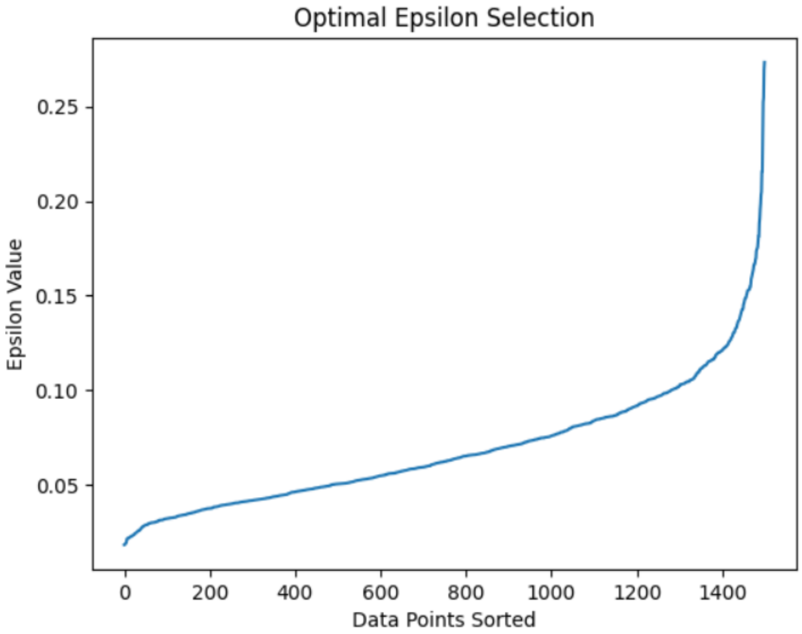
CODE :

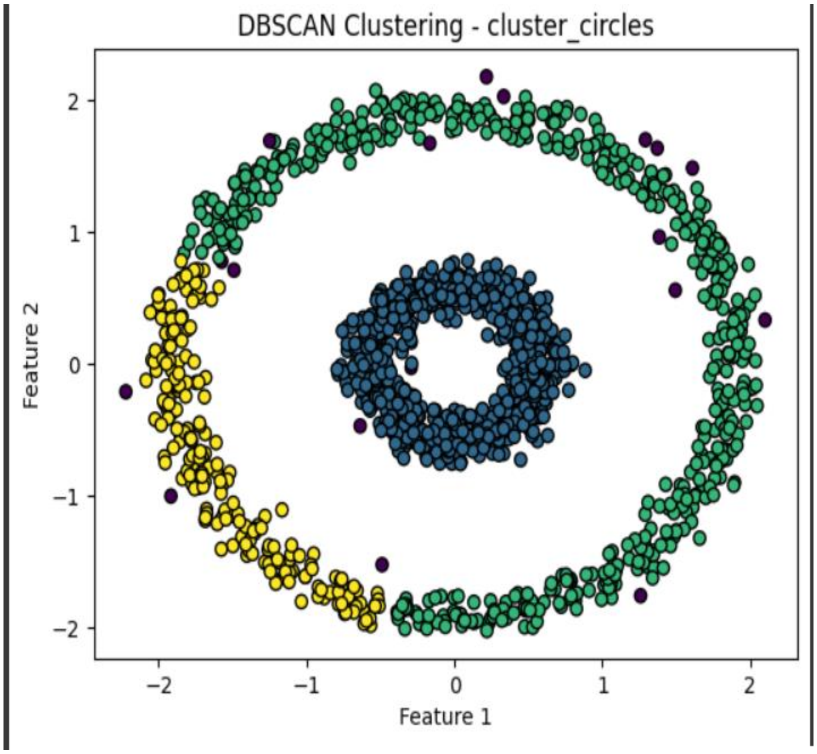
```
def apply_dbscan(data, eps, min_samples):
    """Apply DBSCAN clustering."""
    dbscan = DBSCAN(eps=eps, min_samples=min_samples)
    labels = dbscan.fit_predict(data)
    return labels
```

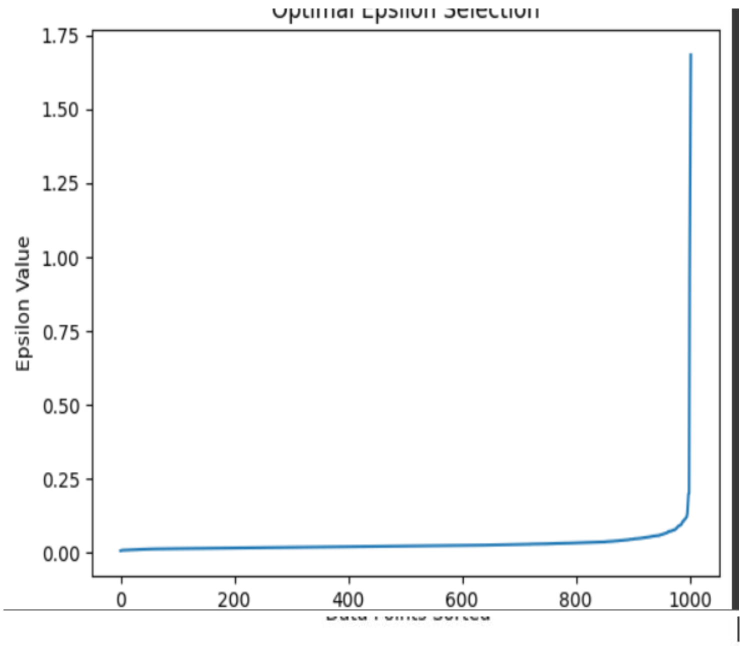
```
def plot_clusters(data, labels, title):
    """Plot clusters."""
    plt.figure()
    plt.scatter(data[:, 0], data[:, 1], c=labels, cmap='viridis', edgecolors='k')
    plt.xlabel("Feature 1")
    plt.ylabel("Feature 2")
    plt.title(title)
    plt.show()
```

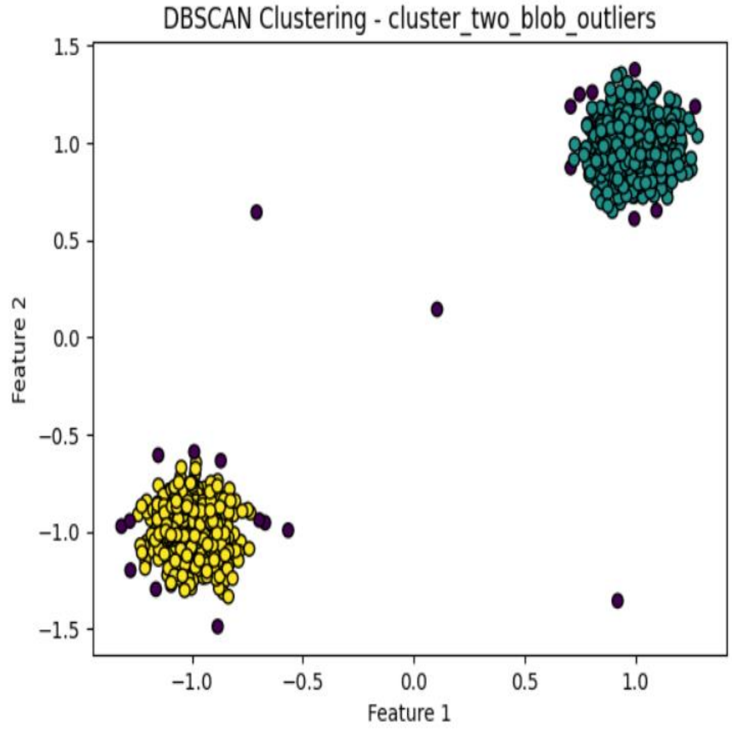

Optimal Epsilon Selection
Epsilon Value
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
200
400
600
800
1000
1200
1400
Data Points Sorted


DBSCAN Clustering - cluster_moon
Feature 2
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Feature 1


Optimal Epsilon Selection
Epsilon Value
0.25
0.20
0.15
0.10
0.05
0
200
400
600
800
1000
1200
1400
Data Points Sorted


DBSCAN Clustering - cluster_circles
Feature 2
2
1
0
−1
−2
−2
−1
0
1
2
Feature 1





2. Using the above datasets apply the hyperparameter optimization technique to get the optimal values of epsilon and min_sample parameters.

```
# Process each dataset
optimal_params = {}
for name, path in datasets.items():
    df = load_dataset(path)
    data = df.values
    data = StandardScaler().fit_transform(data)  # Normalize data

    eps = find_optimal_params(data)
    min_samples = 5  # Common default, can be tuned
    labels = apply_dbscan(data, eps, min_samples)

    optimal_params[name] = {"epsilon": eps, "min_samples": min_samples}
    plot_clusters(data, labels, f"DBSCAN Clustering - {name}")

print("Optimal Parameters:")
for key, value in optimal_params.items():
    print(f"Dataset: {key}, Epsilon: {value['epsilon']}, Min_samples:
{value['min_samples']}")
```

Dataset	Epsilon	Min_samples
cluster_moon	0.08731447935470686	5
cluster_circles	0.12864982749520645	5
cluster_two_blob_outliers	0.06128303573608501	5

3. Write down the at least three use cases of DBSCAN with real-life scenario
Real-Life Use Cases of DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Anomaly Detection in Fraudulent Transactions

Scenario: In banking and financial institutions, fraudulent transactions often appear as anomalies in transaction datasets. DBSCAN is useful for identifying clusters of normal transactions while marking unusual activities (e.g., sudden large withdrawals, transactions from different countries) as noise.

Example: Detecting credit card fraud by clustering normal spending behaviors and flagging outlier transactions as potential fraud.

Geospatial Data Analysis (Urban Planning & Traffic Management)

Scenario: Urban planners and traffic engineers use DBSCAN to analyze population density, detect traffic congestion zones, and identify underutilized road networks.

Example: In smart city development, DBSCAN can identify high-density pedestrian areas or clusters of traffic jams, helping city planners optimize road networks and improve public transport routes.

Image Segmentation in Medical Imaging

Scenario: DBSCAN is widely used in healthcare for segmenting medical images to identify tumors, anomalies, or patterns in MRI and CT scans.

Example: In cancer detection, DBSCAN can help isolate tumor regions in MRI scans by clustering high-density regions of abnormal cell growth while treating normal tissues as noise.