### **1. Algorithm Overview (20%)**

### **Algorithm Overview: DBSCAN (Density-Based Spatial Clustering of Applications with Noise)**

#### **Cluster Identification**

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is an unsupervised clustering algorithm that groups points based on their density. Unlike k-Means and Hierarchical Clustering, which require predefined cluster numbers, DBSCAN determines clusters dynamically by identifying **dense regions** in the dataset. It classifies points into three categories:

* **Core Points**: Have at least min\_samples points (including itself) within a defined radius eps.
* **Border Points**: Are within eps of a core point but have fewer than min\_samples points in their neighborhood.
* **Noise Points**: Do not fall into the first two categories and are considered outliers.

#### **Key Parameters**

DBSCAN relies on two key parameters that influence cluster formation:

* **eps (epsilon)**: The radius within which points are considered part of a neighborhood. A larger eps value results in fewer, larger clusters, while a smaller eps can lead to more, fragmented clusters.
* **min\_samples**: The minimum number of points required to form a dense region. A higher value makes the algorithm stricter, reducing the number of clusters and classifying more points as noise.

#### **Strengths and Limitations**

| **Aspect** | **Strengths** | **Limitations** |
| --- | --- | --- |
| **Cluster Shape** | Detects arbitrarily shaped clusters, unlike k-Means, which prefers spherical clusters. | Struggles when clusters have varying densities. |
| **Noise Handling** | Effectively identifies and removes noise from data. | May misclassify meaningful points as noise if eps and min\_samples are not chosen correctly. |
| **Number of Clusters** | No need to predefine the number of clusters. | Results depend on eps and min\_samples, requiring tuning for different datasets. |
| **Computational Efficiency** | Efficient for small-to-moderate datasets. | Scales poorly with large datasets (O(n²) complexity in worst-case). |

### 

### **2. Algorithm Comparison (40%)**

### **Comparison of Performance**

| **Dataset** | **k-Means Performance** | **Hierarchical Performance** | **DBSCAN Performance** |
| --- | --- | --- | --- |
| **Moons Dataset** | Struggles because the clusters are non-spherical and overlapping. | Performs decently but requires careful tuning of linkage method. | ✅ Best performer—correctly detects two clusters due to density-based approach. |
| **Blobs Dataset** | ✅ Best performer—since blobs are well-separated and spherical. | ✅ Performs well, correctly detecting clusters. | ❌ Struggles due to **varying densities**, classifying some points as noise. |
| **Circles Dataset** | Fails—forces spherical clusters, incorrectly grouping points. | Performs better than k-Means but does not fully capture the structure. | ✅ Best performer—correctly detects concentric clusters due to density-based approach. |

### **Failure Cases: When Does DBSCAN Struggle?**

1. **Varying Density Clusters (Blobs Dataset)**
   * DBSCAN assumes clusters have similar densities.
   * If clusters have different densities, the algorithm may misclassify lower-density regions as noise.
2. **Parameter Sensitivity (eps and min\_samples)**
   * Choosing the right **eps (radius)** is critical; too small, and meaningful points are marked as noise; too large, and separate clusters merge.
   * **min\_samples** should be adjusted based on dataset size; incorrect values can lead to poor clustering.
3. **Computational Complexity**
   * DBSCAN is **O(n²) in worst case**, making it slow for large datasets.
   * k-Means and Hierarchical Clustering are often **faster** for large-scale data.

### **Trade-offs Between Clustering Methods**

| **Factor** | **k-Means** | **Hierarchical Clustering** | **DBSCAN** |
| --- | --- | --- | --- |
| **Speed** | ✅ Fast (O(nk)) | ❌ Slow for large data (O(n²)) | ❌ Can be slow (O(n²) worst case) |
| **Number of Clusters** | Must be pre-defined | Can be inferred from dendrogram | ✅ Automatically detects clusters |
| **Cluster Shape** | ❌ Prefers spherical clusters | ✅ Handles various shapes but depends on linkage | ✅ Handles arbitrary cluster shapes |
| **Noise Handling** | ❌ Poor (all points assigned to clusters) | ❌ Poor (assigns all points) | ✅ Best at identifying noise |
| **Scalability** | ✅ Efficient for large data | ❌ Not suitable for very large datasets | ❌ Can struggle with high-dimensional data |

### **Final Summary**

* **DBSCAN is best when clusters are arbitrary shapes and contain noise** (e.g., Moons and Circles datasets).
* **k-Means is ideal for well-separated, spherical clusters and scales well for large data** (e.g., Blobs dataset).
* **Hierarchical Clustering is useful when a hierarchical structure is needed**, but it is computationally expensive.

### **3. Table Update (20%)**

📌 **Table Content – Summarize the following characteristics for all three algorithms:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **k-Means** | **Hierarchical Clustering** | **DBSCAN** |
| **Definition** | Partitioning algorithm that assigns points to k clusters based on centroids | Builds a hierarchy of clusters using distance metrics | Density-based clustering that groups points if they have enough neighboring points. |
| **Approach** | Iteratively minimizes variance within k clusters | Agglomerative (bottom-up) or divisive (top-down) | Expands clusters based on eps and min\_samples thresholds. |
| **Number of Clusters** | Requires predefined k | Can be determined from dendrogram but subjective | Determines clusters automatically based on density. |
| **Cluster Shape** | Prefers spherical clusters | Works well with various shapes but can be unstable | Handles arbitrarily shaped clusters. |
| **Initialization** | Randomly selects k initial centroids | No initialization needed | No initialization needed. |
| **Result** | Hard assignments—each point belongs to a single cluster | Hierarchical structure (tree/dendrogram) | Soft assignments—some points classified as noise, others form dense clusters. |
| **Interpretability** | Moderate—cluster assignments but no hierarchy | High—dendrogram can be analyzed | Moderate—depends on eps and min\_samples. |
| **Strengths** | Simple, fast and efficient on large datasets | Can capture hierarchical relationships | Identifies noise, detects arbitrarily shaped clusters. |
| **Limitations** | Sensitive to initial centroids and k choice | Computationally expensive for large datasets | Struggles with varying densities, requires careful tuning of eps. |

### 

### 

### **4. Code Documentation & Submission Quality (20%)**

<https://github.com/Kaleem-QADR/BINF5507_ML-AI_in_Bioinfo/tree/main/Assignment_3>