MACHINE LEARNING

CLUSTERING ALGORITHMS

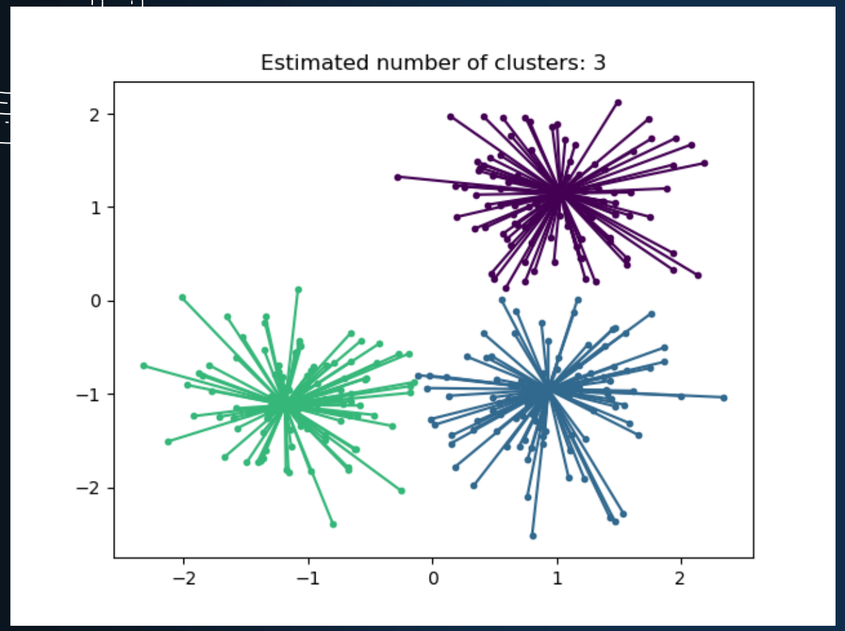
Affinity Propagation Clustering

MACHINE LEARNING CLUSTERING

Affinity Propagation Clustering

Introduction to Affinity Propagation

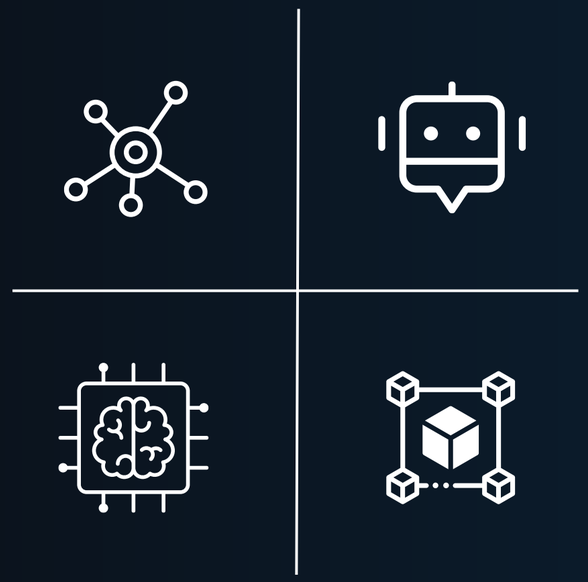
A clustering algorithm that identifies exemplars (representative data points) and forms clusters around them. Automatically determines the number of clusters based on data.



Key Concepts:

- **Exemplar:** A data point that represents a cluster.
- **Similarity Matrix (S):** Measures similarity between data points.
- **Responsibility (r(i, k)):** Suitability of point k as an exemplar for point i.
- **Availability (a(i, k)):** Appropriateness of choosing point k as an exemplar.

Mechanics of Affinity Propagation



1. Initialization:

Initialize responsibilities r(i, k) and availabilities a(i, k) to zero.

2. Update Responsibilities r(i, k).

3. Update Availabilities a(i, k).

4. Iteration:

Repeat responsibility and availability updates until convergence.

Application and Evaluation

Application:

- Compute similarity matrix from data.
- Run affinity propagation to identify exemplars and form clusters.

Advantages:

- Automatically determines the number of clusters.
- Can handle complex clustering structures.
- Suitable for large datasets.

Disadvantages:

- High memory and computational requirements.
- Sensitive to preference parameter.
- Potential convergence issues.
- Less scalable than simpler methods like K-means.
- Results can be difficult to interpret.

Mean Shift Clustering

MACHINE LEARNING CLUSTERING

Mean Shift Clustering

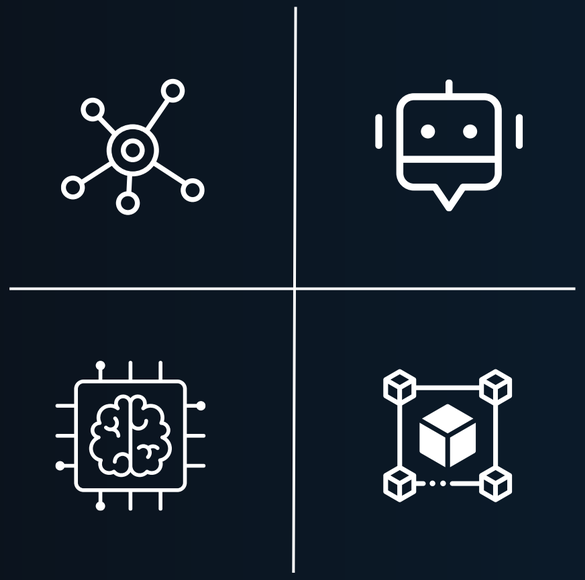
Introduction to Mean Shift Clustering

A non-parametric clustering technique that does not require prior knowledge of the number of clusters. Finds dense regions in the feature space by iteratively shifting data points towards the mode (peak) of the data distribution.

Key Concepts:

- **Bandwidth (h):** Controls the size of the window used to compute the mean shift.
- **Kernel Function:** Determines the weight of nearby points for the mean computation.
- **Convergence:** Points are shifted iteratively until they converge to the mode.

Mechanics of Mean Shift Clustering



1. **Initialization:**

   Select initial points (seeds) for the mean shift procedure.

2. **Mean Shift Vector Calculation**

3. **Update Points**

   Move each point towards the mean (mode) using the mean shift vector.

4. **Iteration:**

   Repeat the mean shift calculation and point updates until convergence (points no longer move significantly).

Application and Evaluation

Application:

- Image segmentation, object tracking, clustering in feature space.
- Steps:
  - ✓ Compute the mean shift vectors for all data points.
  - ✓ Shift data points iteratively until convergence.
  - ✓ Group points converging to the same mode into clusters.

Advantages:

- No need to specify the number of clusters.
- Can identify arbitrarily shaped clusters.
- Robust to noise and outliers.

Disadvantages:

- Computationally expensive for large datasets.
- Choice of bandwidth (h) is critical and non-trivial.
- May converge to local maxima, resulting in suboptimal clustering.

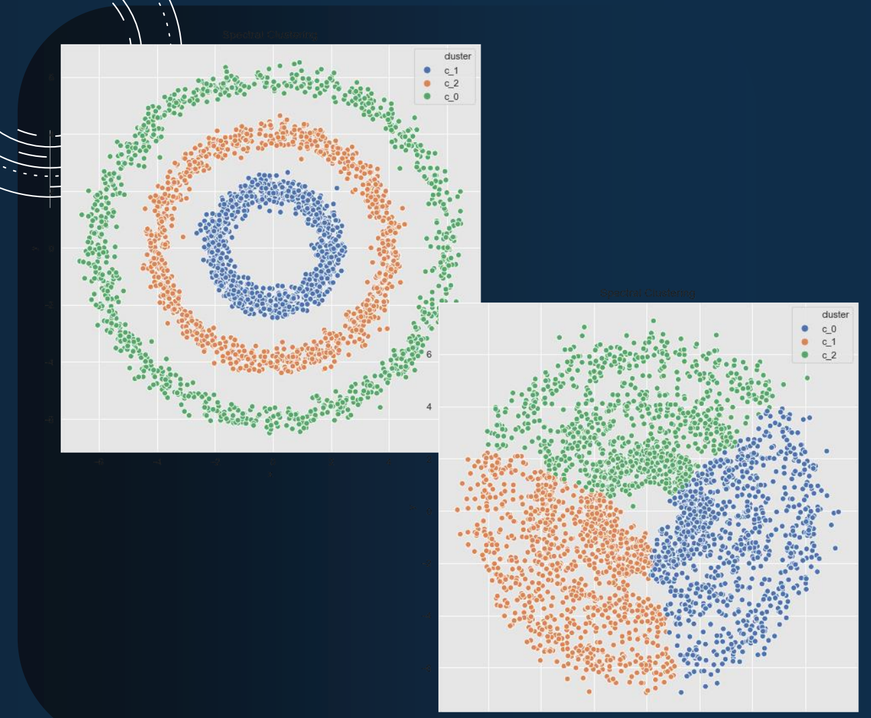
Spectral Clustering

MACHINE LEARNING CLUSTERING

Spectral Clustering

Introduction to Spectral Clustering

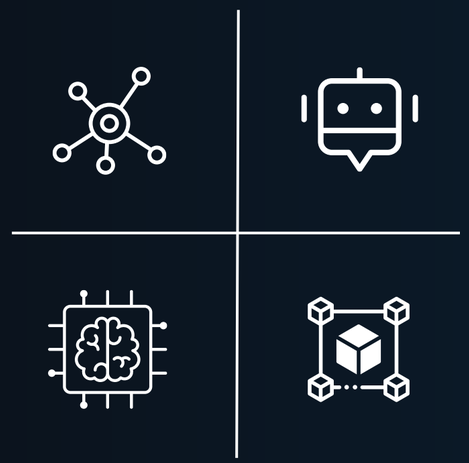
A clustering technique that uses the eigenvalues of a similarity matrix to perform dimensionality reduction before clustering. Captures the structure of data by considering the connectivity and proximity of points.



Key Concepts:

- **Similarity Matrix (W):** Represents the pairwise similarities between data points.
- **Laplacian Matrix (L):** Derived from the similarity matrix; crucial for spectral clustering.
- **Eigenvectors:** Used to map data points to a lower-dimensional space.

Mechanics of Spectral Clustering



1. **Construct Similarity Matrix:**
   - Compute the similarity matrix W from the data points.
2. **Compute Laplacian Matrix:**
   - $L=D-W$
   - Here, D is the diagonal degree matrix where $D(i,i) = \sum_j W(i,j)$.
3. **Compute Eigenvectors:**
   - Find the eigenvectors corresponding to the smallest k eigenvalues of L.
4. **Form Lower-dimensional Representation:**
   - Use the eigenvectors to create a new representation of the data.
5. **Cluster in Lower-dimensional Space:**
   - Apply a clustering algorithm (e.g., K-means) on the lower-dimensional data to form clusters.

Application and Evaluation

Application:

- Image segmentation, network analysis, document clustering.
- Steps:
  - ✓ Construct the similarity matrix.
  - ✓ Compute the Laplacian matrix and its eigenvectors.
  - ✓ Perform clustering in the lower-dimensional space.

Advantages:

- Can capture complex cluster structures.
- Works well with non-linearly separable data.
- No need to specify the number of clusters beforehand.

Disadvantages:

- Computationally intensive for large datasets due to eigenvector computation.
- Requires careful choice of the similarity function.
- Sensitive to the choice of the number of clusters (k).

DBSCAN Clustering

MACHINE LEARNING CLUSTERING

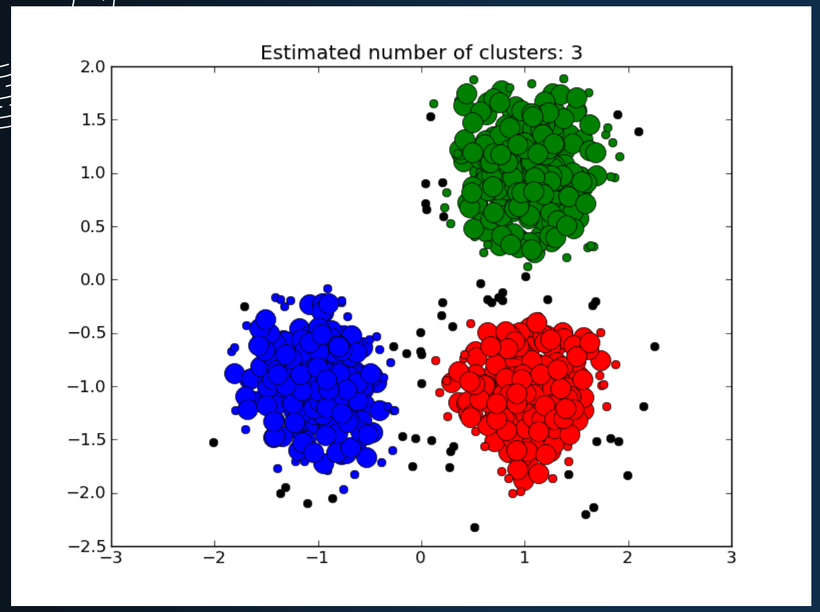
DBSCAN Clustering

Introduction to DBSCAN Clustering

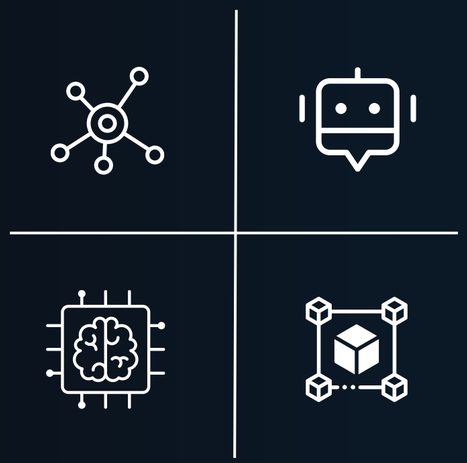
Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a clustering algorithm that identifies clusters based on the density of data points in the feature space. Clusters are formed by grouping together points that are closely packed, with outliers being identified as noise.

Key Concepts:

- **Epsilon (ε):** Maximum radius of the neighborhood around a point.
- **MinPts:** Minimum number of points required to form a dense region.
- **Core Point:** A point with at least MinPts within ε-neighborhood.
- **Border Point:** A point within the ε-neighborhood of a core point but with fewer than MinPts in its own neighborhood.
- **Noise Point:** A point that is not a core or border point.



Mechanics of DBSCAN Clustering



1. **Identify Core Points:**
   - For each point, count the number of points within its ε-neighborhood.
   - If the count is greater than or equal to MinPts, mark the point as a core point.
2. **Form Clusters:**
   - For each core point, form a cluster by including all points (core and border) within its ε-neighborhood.
   - Expand the cluster by recursively including points within the ε-neighborhoods of core points.
3. **Mark Noise Points:**
   - Points that are neither core points nor within the ε-neighborhood of any core points are marked as noise.

Application and Evaluation

Application:

- Suitable for spatial data, image analysis, and anomaly detection.
- Steps:
  - ✓ Determine appropriate values for ε and MinPts.
  - ✓ Run DBSCAN to identify core points and form clusters.
  - ✓ Analyze clusters and noise points for insights.

Advantages:

- Does not require the number of clusters to be specified.
- Can find arbitrarily shaped clusters.
- Robust to noise and outliers.

Disadvantages:

- Sensitive to the choice of ε and MinPts.
- Can struggle with varying densities.
- Computationally intensive for large datasets due to neighborhood searches.

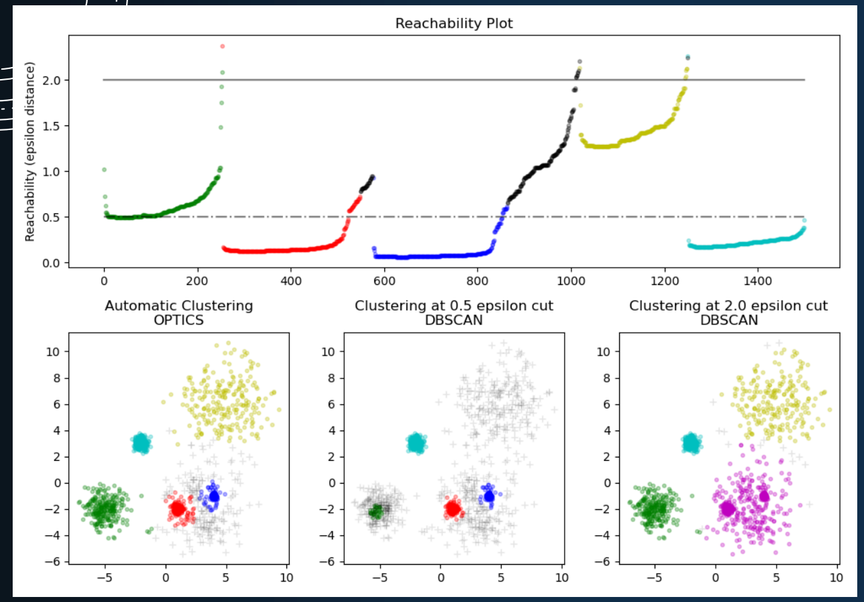
OPTICS Clustering

MACHINE LEARNING CLUSTERING

OPTICS Clustering

Introduction to OPTICS Clustering

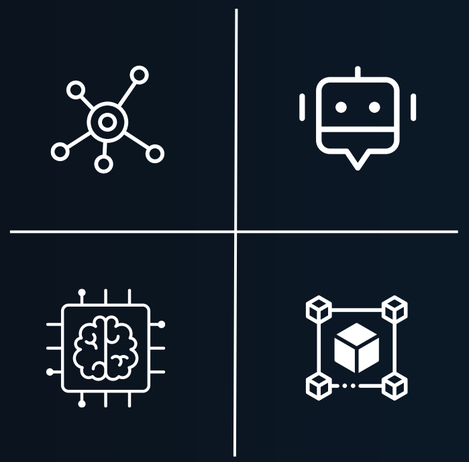
Ordering Points To Identify the Clustering Structure (OPTICS) is an algorithm that identifies clusters of varying density. Extends DBSCAN by creating an ordering of points based on density, which can be used to extract clusters at different density levels.



Key Concepts:

- **Epsilon (ε):** Maximum radius of the neighborhood around a point.
- **MinPts:** Minimum number of points required to form a dense region.
- **Core Distance:** Minimum distance ε' at which a point is a core point.
- **Reachability Distance:** Minimum distance a point needs to be reachable from a core point.

Mechanics of OPTICS Clustering



1. **Initialize:**
   - Start with an arbitrary point and mark it as visited.
2. **Core Distance Calculation:**
   - For each point, calculate the core distance if it has at least MinPts within ε.
3. **Expand Cluster Order:**
   - Use the core distance to determine the reachability distance of neighboring points.
   - Sort and add points to the ordering based on the reachability distance.
   - Expand the process to the next unvisited point with the smallest reachability distance.
4. **Extract Clusters:**
   - Use the reachability plot to identify clusters by identifying valleys in the plot, which represent dense regions.

Application and Evaluation

Application:

- Useful for data with varying densities, such as spatial data analysis and market segmentation.
- Steps:
  - ✓ Run OPTICS to generate an ordering of points and reachability distances.
  - ✓ Analyze the reachability plot to identify clusters.

Advantages:

- Identifies clusters of varying densities.
- Does not require the number of clusters to be specified.
- More robust to noise compared to DBSCAN.

Disadvantages:

- Computationally expensive, especially for large datasets.
- Sensitive to the choice of ε and MinPts.
- Interpretation of reachability plot can be complex.

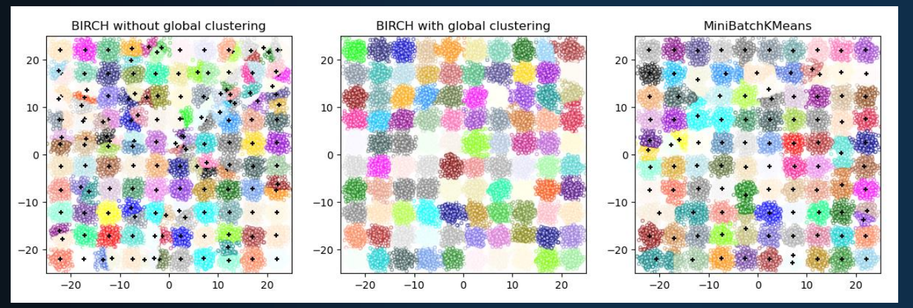
BIRCH Clustering

MACHINE LEARNING CLUSTERING

BIRCH Clustering

Introduction to BIRCH Clustering

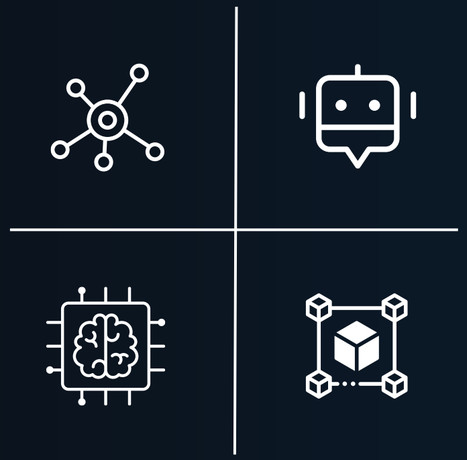
Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) is a hierarchical clustering algorithm designed for large datasets. Constructs a Clustering Feature (CF) tree to summarize the dataset efficiently and perform clustering in a single scan.



Key Concepts:

- **Clustering Feature (CF):** A compact representation of a cluster, including the number of points, linear sum, and square sum.
- **CF Tree:** A height-balanced tree structure that stores the CFs.
- **Threshold (T):** Maximum diameter of a sub-cluster to control the growth of the CF tree.

Mechanics of BIRCH Clustering



1. **Initial Scan:**
   - Insert data points into the CF tree, updating CFs and splitting nodes as necessary based on the threshold (T).
2. **CF Tree Construction:**
   - Build the CF tree dynamically as data points are added, ensuring the tree remains balanced and within the specified threshold.
3. **Clustering:**
   - After the CF tree is built, perform an optional global clustering algorithm (e.g., K-means) on the leaf entries of the CF tree to refine clusters.
4. **Refinement:**
   - Optionally, refine the clusters by rebuilding the CF tree with a smaller threshold and reapplying the clustering algorithm.

Application and Evaluation

Application:

- Suitable for large datasets, such as customer segmentation and image analysis.
- Steps:
  - ✓ Build the CF tree from the dataset.
  - ✓ Apply a global clustering algorithm to the leaf entries.

Advantages:

- Efficient for large datasets due to its single scan and incremental nature.
- Handles noise effectively by summarizing data compactly.
- Can be applied to dynamic data, allowing incremental updates.

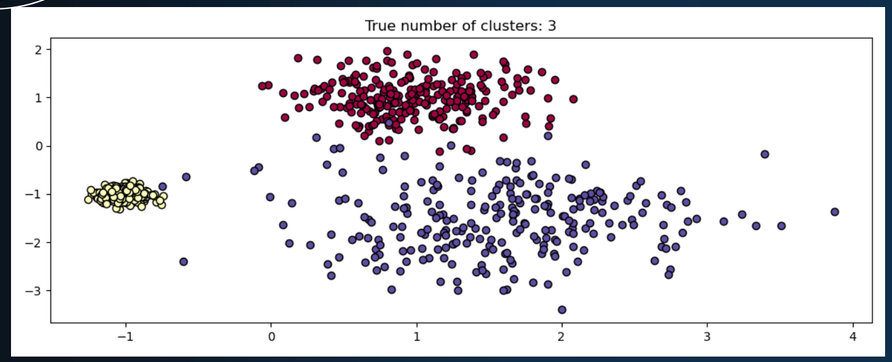
Disadvantages:

- Sensitive to the choice of threshold (T).
- The initial clustering quality depends on the CF tree construction.
- May not perform well with non-spherical clusters.

HDBSCAN Clustering

MACHINE LEARNING CLUSTERING

HDBSCAN Clustering



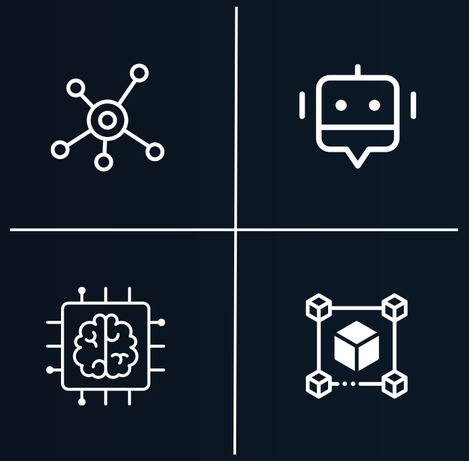
Introduction to HDBSCAN Clustering

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) extends DBSCAN by building a hierarchy of clusters and extracting the most stable clusters from this hierarchy. It is designed to handle varying densities and identify clusters of different shapes and sizes.

Key Concepts:

- **Epsilon (ε):** Maximum radius of the neighborhood around a point (used in DBSCAN).
- **MinPts:** Minimum number of points required to form a dense region.
- **Core Distance:** Minimum distance at which a point is a core point.
- **Mutual Reachability Distance:** Modified distance measure used to build the hierarchy.

Mechanics of HDBSCAN Clustering



1. **Compute Core Distances:**
   - Calculate the core distance for each point, which is the distance to its MinPts-th nearest neighbor.
2. **Build Mutual Reachability Graph:**
   - Construct a graph where edges represent the mutual reachability distance between points.
3. **Construct Hierarchy:**
   - Use a minimum spanning tree to build a hierarchical cluster structure from the mutual reachability graph.
4. **Extract Clusters:**
   - Condense the hierarchy and extract the most stable clusters based on persistence, removing the need to specify epsilon.

Application and Evaluation

Application:

- Suitable for data with varying densities, such as geographical data analysis and bioinformatics.
- Steps:
  - ✓ Compute core distances and build the mutual reachability graph.
  - ✓ Construct the cluster hierarchy and extract stable clusters.

Advantages:

- Identifies clusters of varying densities and shapes.
- Automatically determines the number of clusters.
- Robust to noise and outliers.
- Provides a hierarchical view of the data.

Disadvantages:

- Computationally intensive for very large datasets.
- Sensitive to the choice of MinPts.
- Interpretation of hierarchical results can be complex.

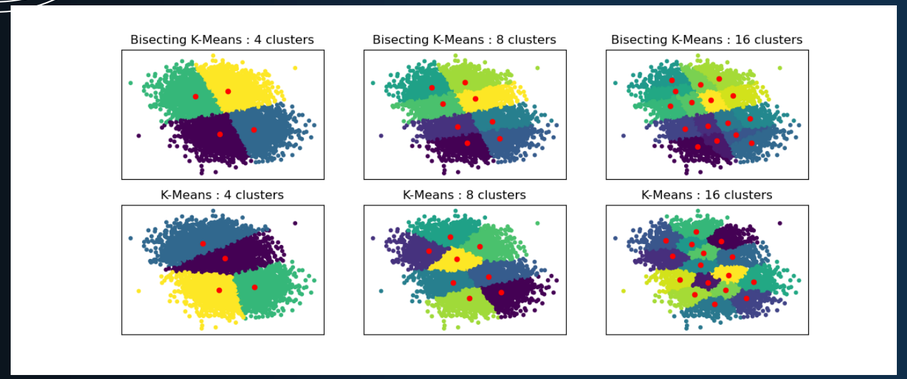
BISECTING K-MEANS Clustering

MACHINE LEARNING CLUSTERING

Bisecting K-Means Clustering

Introduction to Bisecting K-Means Clustering

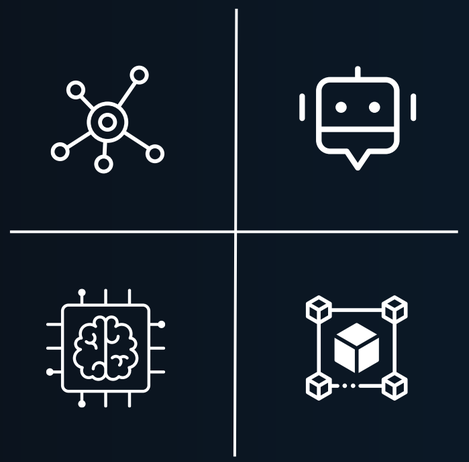
Bisecting K-Means is a hierarchical clustering algorithm that recursively splits clusters into two until the desired number of clusters is reached. Combines the strengths of K-Means and hierarchical clustering to improve clustering quality and efficiency.



Key Concepts:

- **Bisecting Approach:** Splitting one cluster at a time.
- **K-Means Algorithm:** A method to partition data into K clusters by minimizing variance within each cluster.
- **Hierarchical Clustering:** Builds a tree-like structure by recursively dividing clusters.

Mechanics of Bisecting K-Means Clustering



1. **Initialize:**
   - Start with all data points in a single cluster.
2. **Bisecting Step:**
   - Select the largest cluster to split.
   - Apply standard K-Means with K=2 to bisect this cluster into two sub-clusters.
3. **Iterate:**
   - Repeat the bisecting step for the cluster that maximizes the criteria (e.g., the largest cluster or cluster with the highest variance).
4. **Terminate:**
   - Continue until the desired number of clusters is reached.

Application and Evaluation

Application:

- Suitable for large datasets and applications requiring hierarchical clustering.
- Steps:
  - ✓ Start with a single cluster.
  - ✓ Repeatedly apply K-Means to split clusters until the desired number of clusters is achieved.

Advantages:

- More efficient than hierarchical clustering for large datasets.
- Produces more accurate clusters compared to regular K-Means.
- Provides a hierarchical structure of clusters.

Disadvantages:

- Still sensitive to the choice of initial centroids.
- May produce different results with different initializations.
- Requires the number of clusters to be specified beforehand.

Thank You