Deep Unsupervised Learning (Overview)

Naeemullah Khan
naeemullah.khan@kaust.edu.sa

KAUST Academy
King Abdullah University of Science and Technology

May 23, 2025

Table of Contents

أكاديمية كاوست
KAUST ACADEMY

1. Definition
2. Applications
3. Challenges
4. Types
   4.1 Clustering
   4.2 Dimensionality Reduction
   4.3 Anomaly Detection
5. Dimensionality Reduction
   5.1 PCA
   5.2 t-SNE
   5.3 UMAP

Unsupervised Learning - Definition

- We have a dataset without labels. Out goal is to learn something interesting about the underlying structure of the data:
  - Clusters hidden in the dataset.
  - Outliers: particularly unusual and/or interesting data points.
  - Useful signals hidden in the noise, e.g., human speech over a noisy background.

Components of Unsupervised Learning

- **Data**: Unlabeled data, e.g., images, text, or sensor readings.
- **Model**: A mathematical representation of the data, e.g., a mixture model or a neural network.
- **Objective function**: A measure of how well the model fits the data, e.g., likelihood or reconstruction error.
- **Optimization algorithm**: An algorithm to minimize the objective function, e.g., gradient descent or expectation-maximization.
- **Evaluation metrics**: Measures to assess the quality of the learned model, e.g., silhouette score or clustering accuracy.
- **Applications**: Use cases for unsupervised learning, e.g., clustering, dimensionality reduction, or anomaly detection.

Supervised vs Unsupervised Learning

Aspect	Supervised Learning	Unsupervised Learning
Objective	Learn a function f from labeled input–output pairs.	Discover structure or representations in unlabeled data.
Evaluation	Accuracy, precision/recall on held-out labels.	Clustering validity indices (e.g. silhouette), reconstruction error.
Cost	Methods range from $\mathcal{O}(n)$ to $\mathcal{O}(n^3)$ per fit.	k-means $\mathcal{O}(nkd)$, hierarchical $\mathcal{O}(n^2)$, PCA $\mathcal{O}(nd^2)$.
Labels/Clusters	Fixed, known set of classes.	Number of clusters unknown; must be chosen or inferred.
Output	Classifier or regressor for new inputs.	Cluster assignments, embeddings, density models, or generative samples.

Table 1: Key differences between Supervised and Unsupervised Learning

Unsupervised Learning - Applications

Unsupervised learning is used in various fields and applications, including:

- **Visualisation**: Identifying and making accessiblge useful hidden structures in the data.
- **Anomaly Detection**: Identifying factory components that are likely to break soon.
- **Signal denoising**: Extracting human speech from a noisy recording.
- **Generative Models**: Learning to generate new data points similar to the training data.
- **Feature Learning**: Automatically discovering useful representations of the data.
- **Data Preprocessing**: Cleaning and transforming data for better performance in supervised learning tasks.

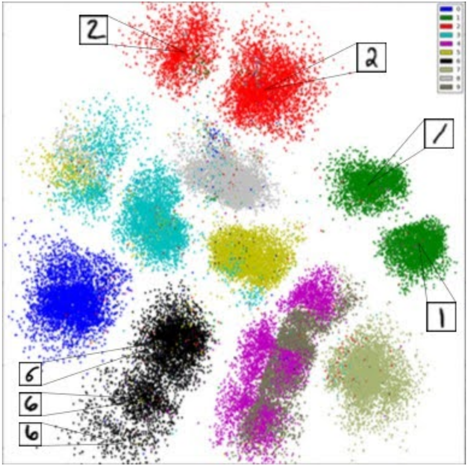


Figure 2: Unsupervised learning can discover structure in digits without any labels.

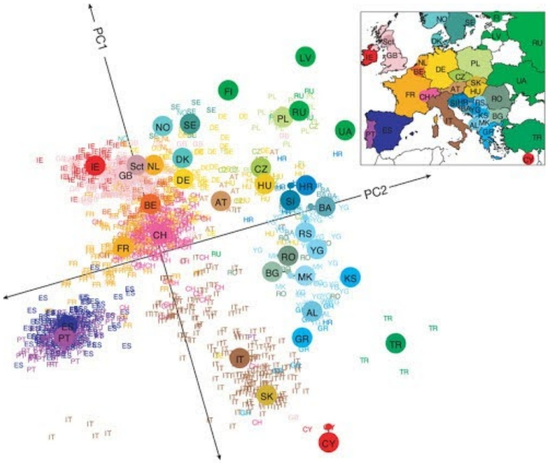


Figure 3: Dimensionality reduction applied to DNA reveal the geography of European countries.

What is Deep Unsupervised Learning?

What is Deep Unsupervised Learning? (cont.)

- Capturing rich patterns in raw data with deep networks in a label-free way.

What is Deep Unsupervised Learning? (cont.)

- Capturing rich patterns in raw data with deep networks in a label-free way.
  - **Generative Models**: Recreate raw data distribution.

Unsupervised Learning - Challenges

Why is unsupervised learning challenging?

- **Exploratory data analysis**: Unsupervised learning is often used for exploratory data analysis, where the goal is to discover patterns or structures in the data without any prior knowledge of the labels.
- **Difficult to assess performance**: Evaluating the performance of unsupervised learning algorithms can be challenging, as there are no ground truth labels to compare against ("right answer" unknown).
- **Sensitivity to noise**: Unsupervised learning algorithms can be sensitive to noise and outliers in the data, which can lead to misleading results.
- **Curse of dimensionality**: As the number of features increases, the data becomes sparse, making it difficult to find meaningful patterns.

Unsupervised Learning - Types

- **Cluster Analysis**:
  - For identifying homogenous subgroups of samples.
  - **Examples**: K-means, hierarchical clustering, DBSCAN.
- **Dimensionality Reduction**:
  - For finding a low-dimensional representation to characterize and visualize the data.
  - Reducing the number of features in a dataset while preserving important information.
  - **Examples**: PCA, t-SNE, UMAP.
- **Anomaly Detection**:
  - **Finding outliers in the dataset**: Identifying unusual (rare items, events, or observations) data points that do not conform to expected patterns.
  - **Examples**: Isolation Forest, One-Class SVM, Autoencoders.

Clustering

A set of methods for finding subgroups within the dataset.

- Observations should share common characteristics within the same group, but differ across groups.
- Groupings are determined from attributes of the data itself — differs from classification.

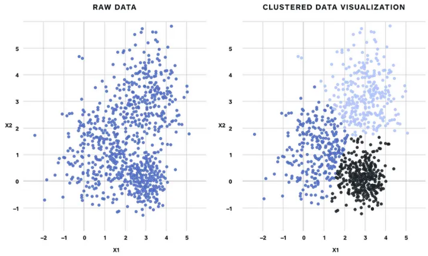


Figure 4: Taking a 2 dimensional dataset and separating it into 3 distinct clusters. [Source]

Clustering (cont.)

Input: Dataset $D = \{x_1, x_2, \ldots, x_n\}$, number of clusters k
Output: Cluster assignments for each data point

Initialization: Randomly initialize k cluster centroids or seeds;

repeat

   Assignment Step: Assign each data point x_i to the nearest cluster based on a distance metric;

   Update Step: Recompute cluster centroids using current assignments;

until *convergence or maximum iterations reached*;

return Final cluster assignments;

Algorithm: Generic Clustering Algorithm

Clustering Vs Classification

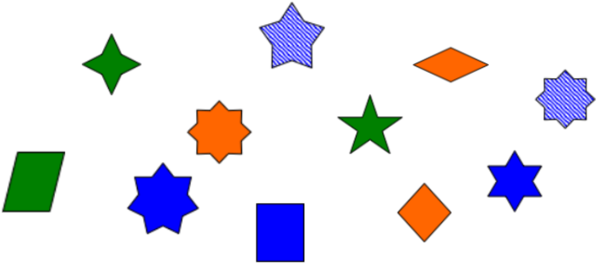


Figure 5: Sample data points.

Clustering Vs Classification (cont.)

Classification
- Labels available
- Assigning to known classes
- Supervised

Clustering
- No labels
- Grouping based on similarity
- Unsupervised

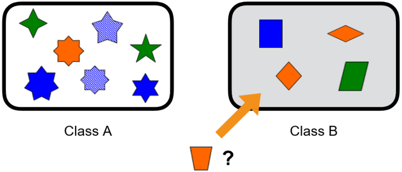


Figure 6: Classification result.

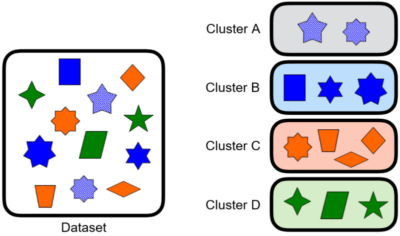


Figure 7: Clustering result.

Clustering: Types

- **Centroid-Based Clustering**: Groups data points based on their proximity to a central point, such as K-means or K-medoids.
- **Hierarchical Clustering**: Builds a hierarchy of clusters using either agglomerative (bottom-up) or divisive (top-down) approaches.
- **Model-Based Clustering**:
  - Each cluster is represented by a parametric distribution.
  - Dataset is a mixture of distributions.
  - Assumes a probabilistic model for the data and uses statistical methods to identify clusters, such as Gaussian Mixture Models (GMM).
- **Hard Clustering**:
  - Each data point is assigned exclusively to exactly one cluster.
  - **Example algorithms**: K-means, Hierarchical clustering.

Clustering: Types (cont.)

- **interpretation**: No ambiguity — clusters are crisp and non-overlapping.

▶ **Soft/Fuzzy Clustering**:

- Each data point can belong to multiple clusters simultaneously with varying degrees of membership (probabilities or weights).
- **Example algorithms**: Gaussian Mixture Models (GMM), Fuzzy C-means.
- **interpretation**: Reflects uncertainty or mixed membership — clusters can overlap.

Groups data into K clusters that satisfy two properties.

1. Each observation belongs to at least one of the K clusters.
2. Clusters are non-overlapping. No observation belongs to more than one cluster.

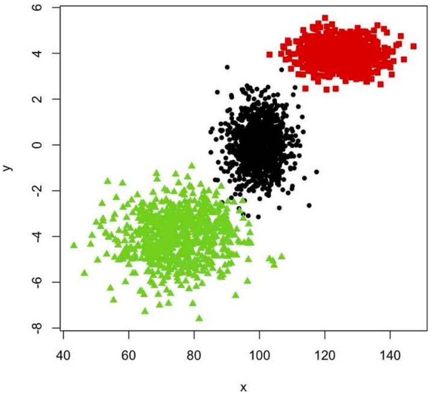


Figure 8: Clusters.

A good clustering is one for which the *within-cluster variation* is as small as possible.

Denote each cluster by C_k, and let $W(C_k)$ be a measure of the within-cluster variation.

K-means aims to solve the following optimization problem:

$$\underset{C_1,\ldots,C_k}{\text{minimise}} \left\{ \sum_{k=1}^{K} W(C_k) \right\} \quad (1)$$

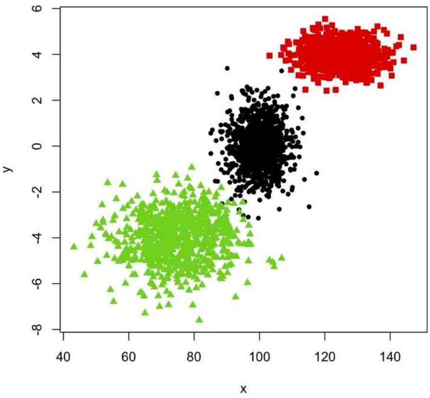


Figure 9: Clusters.

Clustering - K-means (cont.)

How to measure within-cluster variation?

The most common choice is squared Euclidean distance:

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \tag{2}$$

where $|C_k|$ is the number of points in cluster C_k and x_{ij} is the j^{th} feature of the i^{th} point.

Which means overall we solve:

$$\underset{C_1,\dots,C_k}{\text{minimise}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\} \tag{3}$$

Clustering - K-means (cont.)

- It turns out that this optimization problem is difficult to solve, as it is discrete and there are nearly K^n ways to split n samples into K clusters.
- In practice, use an iterative algorithm that finds a local minimum to this optimization.

Input: Dataset $D = \{x_1, x_2, \ldots, x_n\}$, number of clusters k
Output: Cluster assignments for each data point

Initialization: Randomly initialize k cluster centroids or seeds;

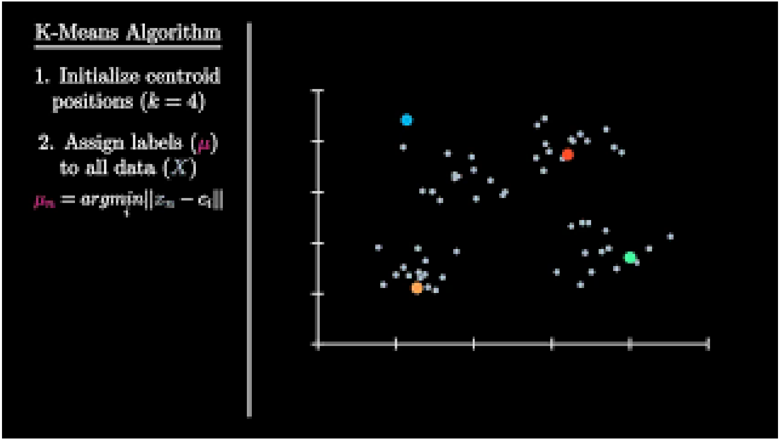
Repeat until convergence:

- **Assignment Step:** Assign each data point x_i to the nearest cluster based on a distance metric;
- **Update Step:** Recompute cluster centroids using current assignments;
- **Convergence Check:** Check if cluster assignments have changed or if centroids have stabilized;

Return: Final cluster assignments and centroids;

Algorithm: K-means Clustering Algorithm

Watch the K-means clustering algorithm in action:



1. It can be shown that the value of the objective function will never increase at each iteration of k-means.
2. Since the algorithm finds local minima, however, it will result in different clusters with different initializations.

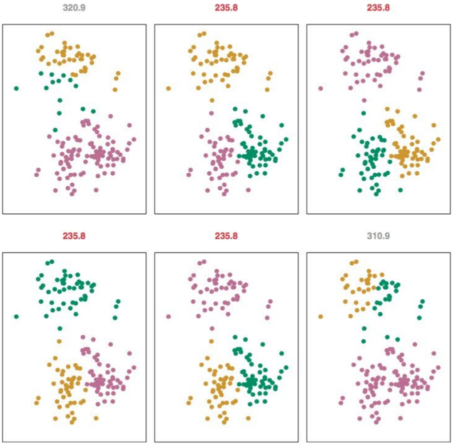


Figure 10: Different initializations of K-means.

K-means - Pros and Cons

Pros

- Simple and easy to implement
- Efficient for large datasets
- Works well with spherical clusters
- Scalable to large datasets

Cons

- Not robust to data perturbations and different initializations
- Sensitive to initial centroid placement
- Assumes spherical clusters
- Requires specifying the number 'K' of clusters in advance
- Sensitive to outliers
- May converge to local minima
- Not suitable for non-convex shapes

Cluster based on distances between observations.

Represented as a tree hierarchy (*dendrogram*) rather than a partition of data.

Does not require committing to a choice of K.

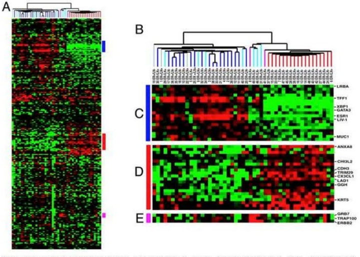


Figure 11: Sørlie, Therese, et al. (2003) "Repeated observation of breast tumor subtypes in independent gene expression data sets," PNAS.

- Each leaf in a dendrogram is a sample/ observation.
- As we move up the dendrogram, observations that are similar to each other begin to fuse into branches.
- Branches then fuse into bigger branches.
- Observations that fuse later (near the top of the tree, or root) are more different than observations that fuse earlier (near the leaves).

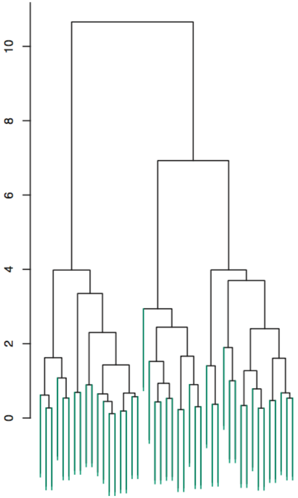


Figure 12: ISL (8th printing 2017)

Note that the horizontal distance between observations on a dendrogram is not the appropriate assessment of observation similarity. Instead, look at vertical axis where branches are first fused.

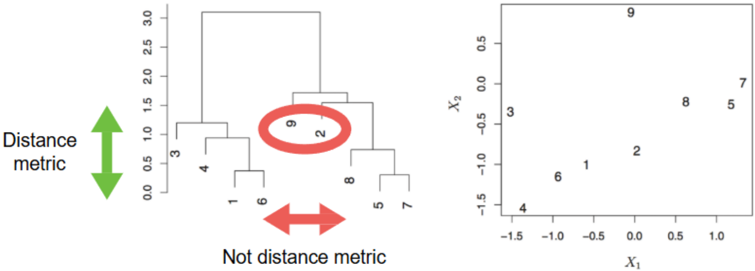


Figure 13: ISL (8th printing 2017)

Clusters are created by making a horizontal cut across the dendrogram. Clusters are the separate trees below the cut.

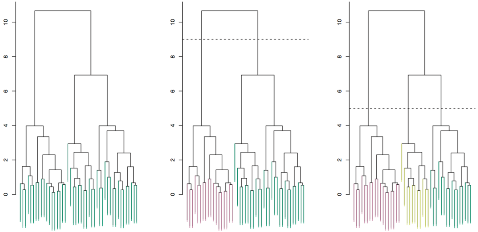


Figure 14: ISL (8th printing 2017)

Building a Dendrogram

A dendrogram is most commonly built using a bottom-up or agglomerative algorithm.

We start at the leaves and group observations until we reach the root containing the entire dataset.

Like in k-means, we need a measure of similarity. Again, the most common is Euclidean distance.

- Compute the distance between each pair of observations.
- Merge the two closest observations into a cluster.
- Compute the distance between the new cluster and all other observations.
- Repeat until all observations are in one cluster.
- The distance between clusters is computed using a linkage method.

Input: Dataset $D = \{x_1, x_2, \ldots, x_n\}$
Output: Dendrogram representing the hierarchical structure of clusters
Initialization: Treat each data point as a separate cluster;
Compute distance matrix: Calculate pairwise distances between all clusters;
Repeat until only one cluster remains:

- Find the two closest clusters based on the distance matrix;
- Merge the two clusters into a new cluster;
- Update the distance matrix to reflect the new cluster;
- Recompute distances between the new cluster and all other clusters using a linkage method;

Return: Dendrogram representing the hierarchical structure of clusters;

Algorithm: Hierarchical Clustering Algorithm

w3schools: Codes and Playground

Clustering - Hierarchical: Dendrograms (cont.)

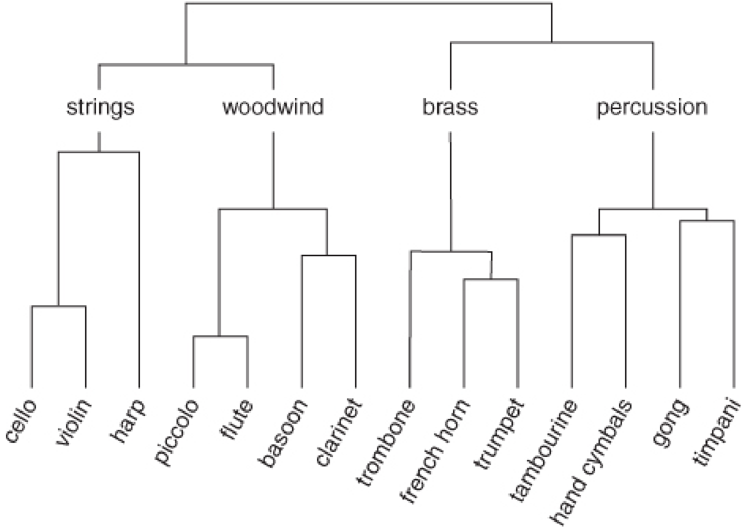


Figure 15: Dendrogram interpretation.

Distance between groups

It's easy to compute Euclidean distance between two observations. What is the distance or similarity between two groups or clusters of observations?

Linkage: defines the dissimilarity between two groups of observations. Most common types are *complete*, *average*, *single*, and *centroid*.

Clustering - Hierarchical: Distance Metrics (cont.)

- **Single Linkage**: Distance between two clusters is the minimum distance between any two points in the clusters.
- **Complete Linkage**: Distance between two clusters is the maximum distance between any two points in the clusters.
- **Average Linkage**: Distance between two clusters is the average distance between all pairs of points in the clusters.
- **Centroid Linkage**: Distance between two clusters is the distance between their centroids.
- **Ward's Linkage**: Distance between two clusters is the increase in variance when the two clusters are merged.

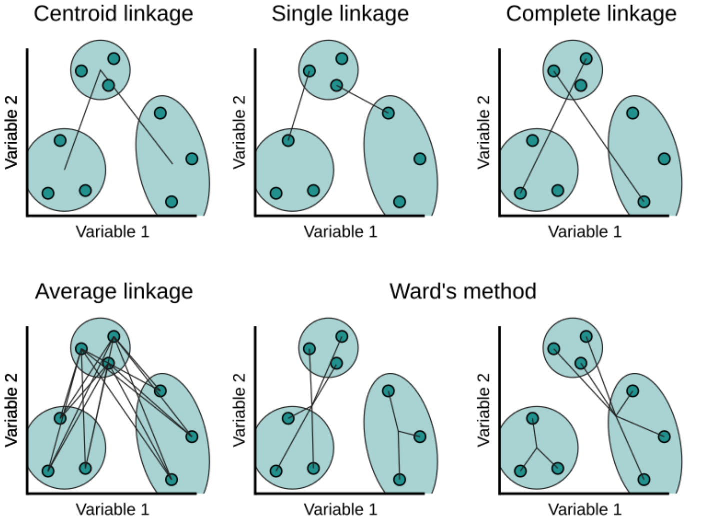


Figure 16: Linkage methods.

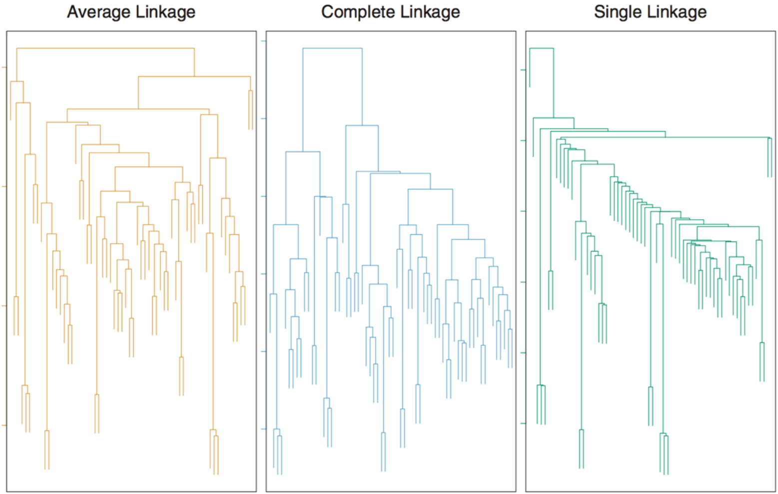


Figure 17: Dendrogram with different linkage types.

Clustering - Hierarchical: Pros and Cons

Pros

- No need to specify the number of clusters K in advance.
- Dendrograms provide a visual representation of the clustering process.
- Can capture complex cluster shapes and relationships.

Cons

- Computationally expensive for large datasets.
- Do have to pick where to cut the dendrogram to obtain clusters
- Sensitive to similarity measure and type of linkage used.
- Sensitive to noise and outliers.
- Difficult to interpret and choose the optimal number of clusters.

Clustering - Density-Based Methods

- Clustering based on density (local cluster criterion), such as density-connected points or based on an explicitly constructed density function.
- **Major features**:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters

- **Major algorithms**:
  - DBSCAN (Density-Based Spatial Clustering of Applications with Noise): Ester, et al. (KDD'96)
  - OPTICS (Ordering Points to Identify the Clustering Structure): Ankerst, et al. (SIGMOD'99)
  - HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise): Campello, et al. (ACM TIST'15)
  - DENCLUE (DENsity-based CLUstEring): Hinneburg and Gabriel (KDD'97)
  - CLIQUE (CLustering In QUEst): Karypis, Han, and Kumar (SIGMOD'98)

- **DBSCAN** (Density-Based Spatial Clustering of Applications with Noise):
  - Density = number of points within a specified radius ϵ.
  - A point is a core point if it has more than a specified number of points (MinPts) within ϵ. These are points that are at the interior of a cluster.
  - A border point has fewer than MinPts within ϵ, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point.
  - Groups together points that are closely packed together, marking as outliers points that lie alone in low-density regions.
  - Parameters:
    - ϵ: Maximum distance between two points for them to be considered as in the same neighborhood.
    - MinPts: Minimum number of points required to form a dense region.

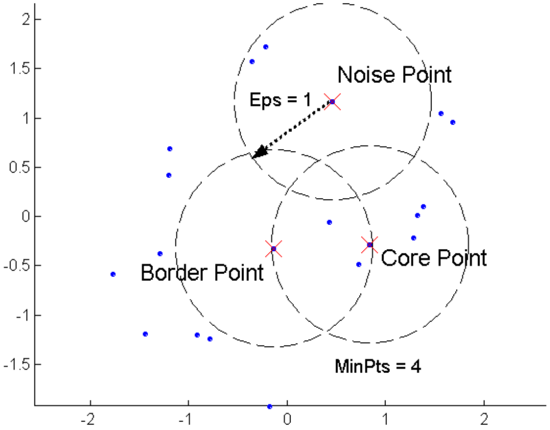


Figure 18: DBSCAN features: Core, Border, and Noise Points

Input: Set of points P, distance threshold ε, minimum number of points *minPts*

Output: A set of clusters

Construct a directed graph $G = (V, E)$ where each node in V corresponds to a point in P;

foreach *point* $c \in P$ **do**

  if *c is a core point (i.e., $|\mathcal{N}_\varepsilon(c)| \geq minPts$)* **then**

    foreach *point* $p \in \mathcal{N}_\varepsilon(c)$ **do**

      Add a directed edge $(c \to p)$ to E;

    end

  end

end

$N \leftarrow V$;

while *there exists a core point* $c \in N$ **do**

  Let X be the set of nodes reachable from c via directed edges in G;

  Form a cluster $C = X \cup \{c\}$;

  Remove all nodes in C from N;

end

Algorithm: Graph-Based DBSCAN Clustering

Credits

Dr. Prashant Aparajeya

Computer Vision Scientist — Director(AISimply Ltd)

p.aparajeya@aisimply.uk

This project benefited from external collaboration, and we acknowledge their contribution with gratitude.