## National University of Computer and Emerging Sciences



**Lab 13**

Department of Computer Science

FAST-NU, Lahore, Pakistan

*Unsupervised Learning*

Unsupervised learning is a type of machine learning where we use a model to discover patterns or structures in a dataset **without labeled outputs**. The goal is to organize the data or gain insights without predefined categories.

**No Labels**:

The data has no target variable or labels (e.g., no "spam" or "not spam" tags).

**Focus on Patterns**:

The model identifies similarities or structures within the dataset.

**Main Goal**:

Group similar data points together (clustering).

Reduce the dataset's complexity (dimensionality reduction).

# Clustering

Clustering is a type of **unsupervised learning** that involves dividing a dataset into groups (or clusters). Each cluster contains data points that are more similar to one another than to data points in other clusters. Clustering is widely used for pattern recognition, segmentation, and exploratory data analysis.

**Similarity-Based Grouping**:

Clusters are formed based on how similar data points are, often measured using metrics like Euclidean distance or cosine similarity.

**Unsupervised**:

No predefined labels; the algorithm determines groupings based on data characteristics.

**Applications of Clustering**

**Customer Segmentation**:

Group customers based on their behavior (e.g., online purchases).

**Image Compression**:

Group similar colors in an image to reduce file size.

**Anomaly Detection**:

Detect outliers in financial transactions for fraud prevention.

**Grouping Documents**:

Group similar articles or research papers.

**K-Means Clustering**

**What It Does**:

Divides data into k clusters.

Each cluster is represented by its center (called a centroid).

**Steps**:

Start with k randomly chosen centroids.

Assign each data point to the closest centroid.

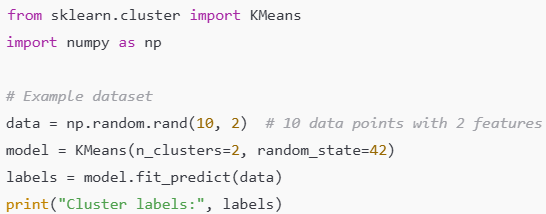
Update the centroids by averaging the points in each cluster.

Repeat until centroids stabilize.

**Key Points**:

You need to choose k (the number of clusters) in advance.

Works best when clusters are circular (spherical).



**K-Medoids**

K-Medoids is a clustering algorithm similar to **K-Means**, but instead of using the mean of the data points as the cluster center (centroid), it uses **actual data points** (called medoids) as the center of each cluster.

This approach is more robust than K-Means because it minimizes the impact of outliers or extreme values, which can heavily influence the mean in K-Means clustering.

**Medoid**:

A medoid is the most centrally located data point in a cluster, minimizing the total distance to all other points in the cluster.

**Objective**:

To minimize the sum of the distances between data points and their nearest medoid.

**How K-Medoids Works**

The algorithm works iteratively to find the optimal medoids:

**Initialization**:

Randomly select k data points as the initial medoids.

**Assignment**:

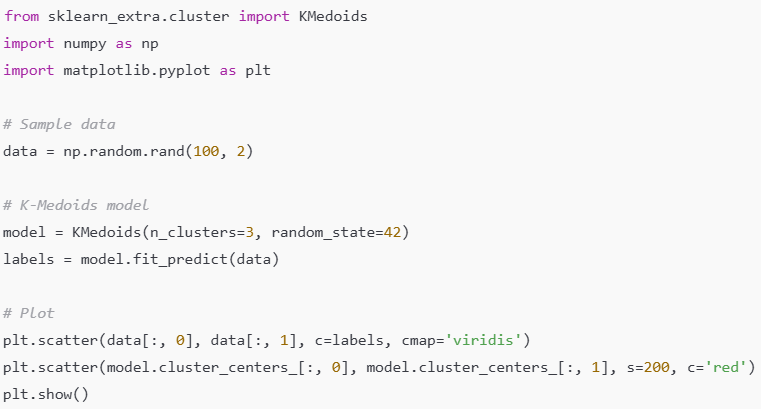
Assign each data point to the nearest medoid.

**Update**:

For each cluster, compute the total cost (sum of distances) and replace the medoid with another point if it reduces the cost.

**Repeat**:

Repeat the assignment and update steps until the medoids stabilize or the cost stops decreasing.



## Hierarchical Clustering

**What It Does**:

Groups data in a hierarchical tree structure.

Two approaches:

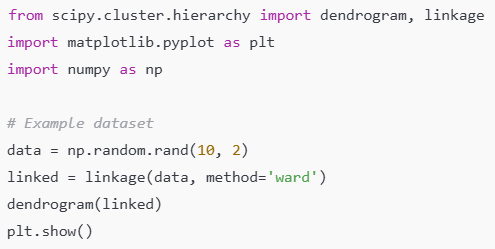
**Agglomerative**: Start with each data point as its own cluster, then merge clusters iteratively.

**Divisive**: Start with one large cluster and split it iteratively.

**Key Points**:

No need to specify k initially.

Creates a **dendrogram** (tree-like diagram) to show how clusters are formed.



## Density-Based Spatial Clustering (DBSCAN)

**What It Does**:

Groups points that are closely packed together.

Points in sparse regions are marked as **noise** (outliers).

**Parameters**:

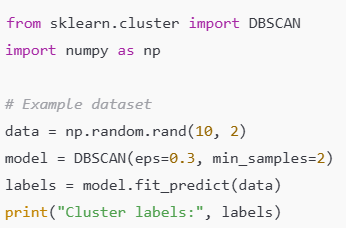
**Epsilon (ϵ\epsilonϵ)**: Maximum distance between two points to consider them neighbors.

**MinPts**: Minimum number of points required to form a dense cluster.

**Key Points**:

Handles noise and clusters of arbitrary shapes well.

You don’t need to specify the number of clusters (k).



## Mean-Shift Clustering

**What It Does**:

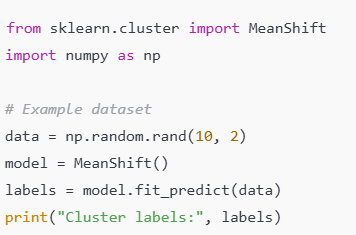
Iteratively moves data points toward the center of their neighborhood (mode of the data).

Automatically detects the number of clusters.

**Key Points**:

Works well for non-circular clusters.

Computationally expensive for large datasets.



|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithms** | **How to Import** | **Purpose** | **When to Use** |
| K-Means | from sklearn.cluster import KMeans | Groups data into k clusters by minimizing within-cluster distances. | Use when you know the number of clusters and data is well-separated. |
| K-Medoids | from sklearn\_extra.cluster import KMedoids | Clusters data into k clusters using actual data points as cluster centers. | Use for robust clustering, especially when outliers are present. |
| Hierarchical Clustering | from scipy.cluster.hierarchy import linkage, dendrogram | Creates a hierarchy of clusters using a dendrogram. | Use when a hierarchy of clusters or tree representation is needed. |
| DBSCAN | from sklearn.cluster import DBSCAN | Identifies dense regions in data and treats sparse regions as noise. | Use for arbitrary-shaped clusters and datasets with noise. |
| Mean-Shift | from sklearn.cluster import MeanShift | Clusters data by shifting points towards the densest region. | Use when the number of clusters is unknown and data density varies. |
| Affinity Propagation | from sklearn.cluster import AffinityPropagation | Clusters data by exchanging messages between data points. | Use for clustering data with varying densities and message passing models. |
| Gaussian Mixture | from sklearn.mixture import GaussianMixture | Models data as a mixture of multiple Gaussian distributions. | Use for probabilistic clustering with overlapping clusters. |
| Spectral Clustering | from sklearn.cluster import SpectralClustering | Clusters data based on graph connectivity and spectral embedding. | Use for clustering data that can be represented as a graph. |
| BIRCH (Balanced Iterative Reducing and Clustering) | from sklearn.cluster import Birch | Performs clustering on large datasets in an incremental manner. | Use for large-scale datasets requiring memory efficiency. |
| OPTICS (Ordering Points To Identify the Clustering Structure) | from sklearn.cluster import OPTICS | Finds clusters based on density and hierarchical ordering of points. | Use for datasets with varying densities and complex structures. |

# How to Evaluate Clustering?

Evaluating clustering results is challenging because there are no predefined labels or targets in unsupervised learning. The evaluation depends on the goal of the clustering and the nature of the dataset. Below are commonly used methods

# Intrinsic Metrics

These metrics evaluate the quality of clustering based on the input data alone (without external labels).

**Silhouette Score:**

**What It Measures**:

How similar a data point is to its own cluster compared to other clusters.

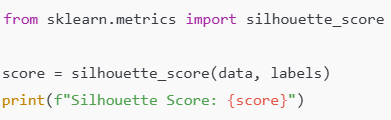
Combines cohesion (within-cluster similarity) and separation (between-cluster dissimilarity).

**Range**:

−1: Poor clustering.

0: Overlapping clusters.

1: Perfect clustering.



**Davies-Bouldin Index**

**What It Measures**:

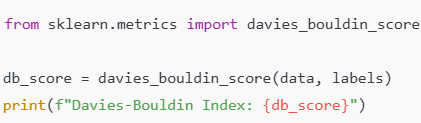
The average ratio of within-cluster distances to between-cluster distances.

Lower values indicate better-defined clusters.

**Range**:

0: Perfect clusters.

Higher values indicate worse clustering.



# Extrinsic Metrics

These metrics require ground truth labels (if available) to compare the clustering results with the actual groupings.

**Adjusted Rand Index (ARI)**

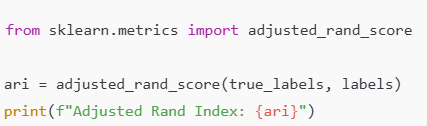
**What It Measures**:

The similarity between the predicted clusters and the ground truth labels.

**Range**:

−1: Completely different.

1: Perfect match.



**Normalized Mutual Information (NMI)**

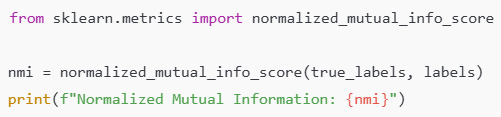
**What It Measures**:

The amount of information shared between the predicted clusters and the ground truth labels.

**Range**:

0: No mutual information.

1: Perfect match.



**Homogeneity, Completeness, and V-Measure**

**Homogeneity**:

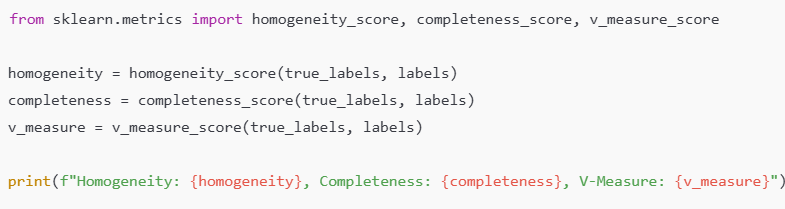
All clusters contain only points belonging to a single class.

**Completeness**:

All points of a single class are in the same cluster.

**V-Measure**:

Harmonic mean of homogeneity and completeness.

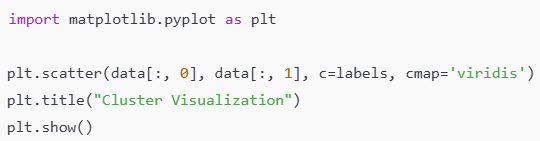


# Visual Evaluation

Visual methods are often used for smaller datasets to understand clustering performance intuitively.

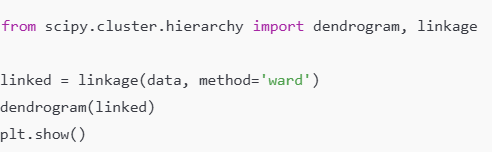
**Scatter Plot**

Visualize clusters in 2D or 3D space.



**Dendrogram**

Used for hierarchical clustering to visualize the tree of clusters.



**When to Use Which Metric?**

* **Intrinsic Metrics**:
  + Use when no ground truth is available (e.g., silhouette score, Davies-Bouldin index).
* **Extrinsic Metrics**:
  + Use when ground truth labels are available (e.g., ARI, NMI).
* **Visual Methods**:
  + Use for small datasets to get an intuitive understanding of clustering.

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Type** | **Range** | **Purpose** |
| Silhouette Score | Intrinsic | -1 to 1 | Measures cohesion and separation of clusters. |
| Davies-Bouldin Index | Intrinsic | 0 to âˆž | Evaluates compactness and separation (lower is better). |
| Calinski-Harabasz Index | Intrinsic | 0 to âˆž | Measures ratio of between-cluster to within-cluster dispersion (higher is better). |
| Adjusted Rand Index (ARI) | Extrinsic | -1 to 1 | Compares predicted clusters to ground truth labels. |
| Normalized Mutual Info (NMI) | Extrinsic | 0 to 1 | Measures information overlap between predicted clusters and ground truth. |
| Homogeneity | Extrinsic | 0 to 1 | Evaluates how well clusters contain only members of a single class. |
| Completeness | Extrinsic | 0 to 1 | Measures if all members of a single class are in the same cluster. |
| Scatter Plot | Visual | N/A | Provides a 2D/3D visual representation of clusters. |
| Dendrogram | Visual | N/A | Visualizes hierarchical clustering as a tree. |

# Activity:

1: Use **K-Means Clustering** to cluster the provided dataset into 3 clusters. Visualize the resulting clusters and their centroids using a scatter plot. Evaluate the clustering quality using the **Silhouette Score**.

2: Use **K-Medoids Clustering** to cluster the dataset into 4 clusters. Visualize the medoids and clusters using a scatter plot. Evaluate the clustering quality using the **Davies-Bouldin Index**.

3: Perform clustering on the same dataset using both **K-Means** and **K-Medoids** for 3 clusters. Visualize the results side-by-side using scatter plots for each algorithm. Compare their clustering quality using the **Calinski-Harabasz Index**.