SPPU

# MACHINE LEARNING

UNIT 5

## Unsupervised Learning

# Contents

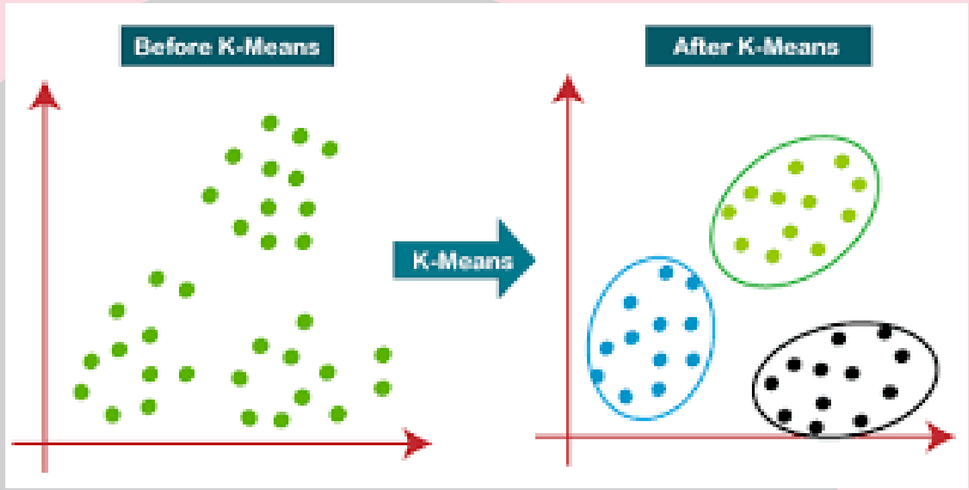
- K-Means
- K-medoids
- Hierarchical
- Density-based Clustering
- Spectral Clustering
- Outlier analysis: introduction of isolation factor, local outlier factor.
- Evaluation metrics and score: elbow method, extrinsic and intrinsic methods.

# K-Means Clustering

K-Means is an unsupervised machine learning algorithm used to group similar data points into clusters.

It works like this:

- You decide how many groups (K) you want.
- The algorithm places K points (centroids) randomly.
- It groups nearby data points around each centroid.
- It moves the centroid to the center of its group.
- Repeats until groups become stable.




**Advantages of K-Means**

✓ 1. Fast & Efficient :Works quickly even for huge datasets.

✓ 2. Easy to Understand & Implement :Just select K and it groups similar items.

✓ 3. Works Well When Clusters Are Clearly Separable :If data forms natural groups, K-Means finds them nicely.

✓ 4. Scalable :Can work with millions of data points.

**Disadvantages of K-Means**

✗ 1. Need to Choose K (Number of Clusters) :You must decide K beforehand (which isn't always obvious).

✗ 2. Fails on Non-Spherical / Odd-Shaped Data :K-Means assumes round-shaped clusters.

✗ 3. Sensitive to Outliers : One extreme data point can shift the centroid.

✗ 4. Different Initial Centroids → Different Results :Random starting points can affect final clusters.

Cluster the following eight points (with (x, y) representing locations) into three clusters: **[6]**

P1(1, 3), P2(2, 2), P3(5, 8), P4(8, 5), P5(3, 9), P6(10, 7), P7(3, 3), P8(9, 4), P9(3, 7)

Use K-Means Algorithm to find the three cluster

## Step 1: Initial Centroids (random)

- C1 = (1,3), C2 = (5,8), C3 = (8,5)

## Step 2: Assign Points

Cluster 1:
- P1(1,3), P2(2,2), P7(3,3)

Cluster 2:
- P3(5,8), P5(3,9), P9(3,7)

Cluster 3:
- P4(8,5), P6(10,7), P8(9,4)

## Step 3: Recalculate New Centroids

New Centroid C1'
Cluster 1 → { (1,3), (2,2), (3,3) }
- Mean X = (1+2+3)/3 = 2
- Mean Y = (3+2+3)/3 = 2.66

➡ C1' = (2, 2.66)
➡ C2' = (3.67, 8)
➡ C3' = (9, 5.33)

## Step 4: Reassign Points Based on New Centroids

After recomputing distances , the clusters remain the same.

That means the algorithm has converged.

## Final K-Means Clusters

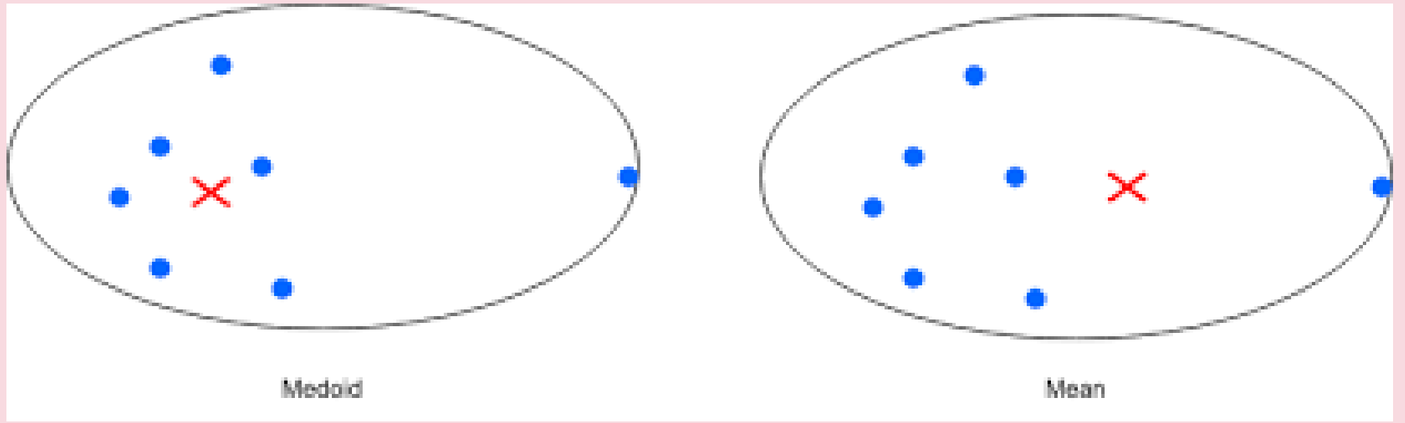
| Cluster 1 | Cluster 2 | Cluster 3 |
|---|---|---|
| P1 (1,3) | P3 (5,8) | P4 (8,5) |
| P2 (2,2) | P5 (3,9) | P6 (10,7) |
| P7 (3,3) | P9 (3,7) | P8 (9,4) |

# K-Medoids

- K-Medoids is a clustering algorithm similar to K-Means, but:
- Instead of average (centroid), it uses an actual data point (medoid) as center.
- A medoid = the most centrally located point in a cluster.
- This makes K-Medoids more stable, less sensitive to outliers, and better for real datasets.

**How K-Medoids Works**

- Step 1: Choose K random medoids : Example: K = 3 → choose 3 actual data points.
- Step 2: Assign each point to the nearest medoid : Using distance (Euclidean / Manhattan).
- Step 3: Try “swapping” medoids : For each medoid, replace it with another point in the cluster and check:
  Does total cost decrease?
  (cost = sum of distances between all points and their medoid)
  If YES → keep the new medoid.
- Step 4: Repeat until no more improvement : Final Result → Stable clusters + real medoids



## Why K-Medoid is Used?

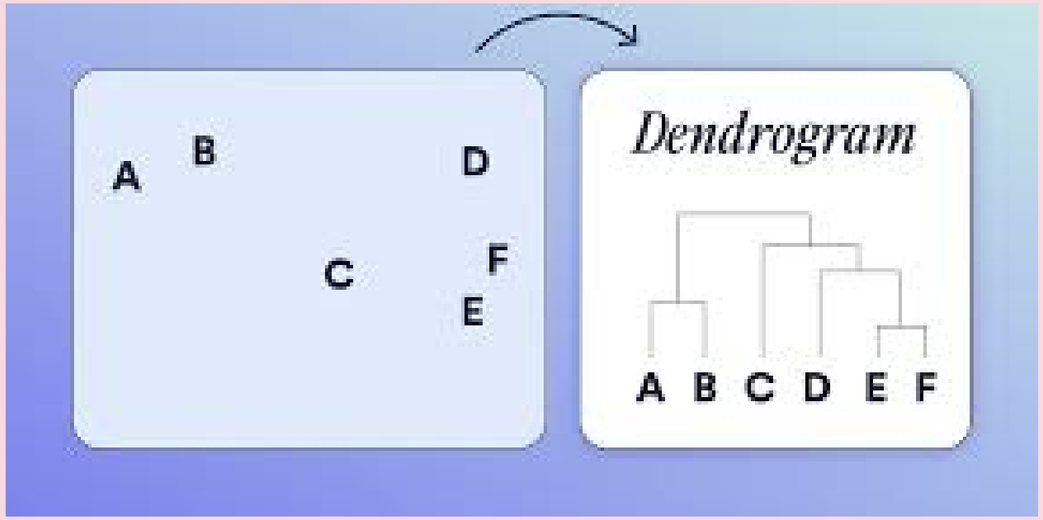
K-Medoid is used because it is:

1. **More Robust to Outliers**
   - Unlike K-Means (which uses average as the center), K-Medoid uses actual data points as centers.
   - ➡ So extreme or noisy data does not affect the cluster center.
2. **More Stable**
   - Small changes in data do not shift the medoid drastically.
3. **Better for Real-World Data**

   When the dataset has:
   - noise
   - irregular shapes
   - non-linear patterns
   - K-Medoid gives more reliable clusters.
4. **Works with Any Distance Metric**
   - Manhattan, Euclidean, Cosine — it supports flexible distance measures.

| Feature | K-Means | K-Medoids |
|---|---|---|
| **Center** | Average point (centroid) | Actual data point (medoid) |
| **Sensitive to Outliers** | ❌ Yes | ✅ No |
| **Stability** | Medium | High |
| **Best For** | Large numeric data | Small–medium noisy datasets |
| **Cost Function** | Squared distance | Total distance |

# Hierarchical Clustering

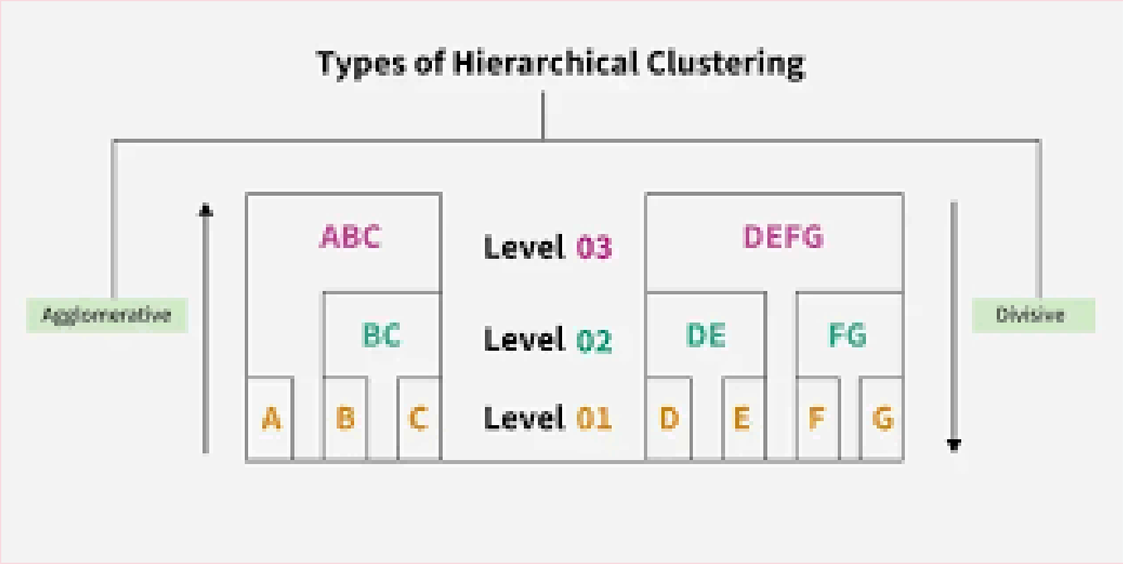



Hierarchical clustering creates a tree-like structure (dendrogram) that shows how data points are grouped step-by-step.
It builds clusters either:

🔻 **Bottom-up (Agglomerative)** → Start with individual points, keep merging

🔺 **Top-down (Divisive)** → Start with one big cluster, keep splitting

Most commonly used: Agglomerative.




| Feature | Agglomerative (Bottom-Up) | Divisive (Top-Down) |
|---|---|---|
| **Direction** | Starts from single points → merges upward | Starts from one big cluster → splits downward |
| **Initial State** | Each data point is its own cluster | All data points in one single cluster |
| **Most Common?** | ✓ Yes, widely used | ✗ Rarely used |
| **Complexity** | Lower computation | Higher computation |
| **Dendrogram Interpretation** | Tree grows from bottom to top | Tree grows from top to bottom |
| **Backtracking** | No backtracking after merging | No backtracking after splitting |
| **When to Use** | When merging similar groups is easier | When you want to start from a global perspective |
| **Example** | Build small groups first → form big groups | Start with all items → separate step-by-step |

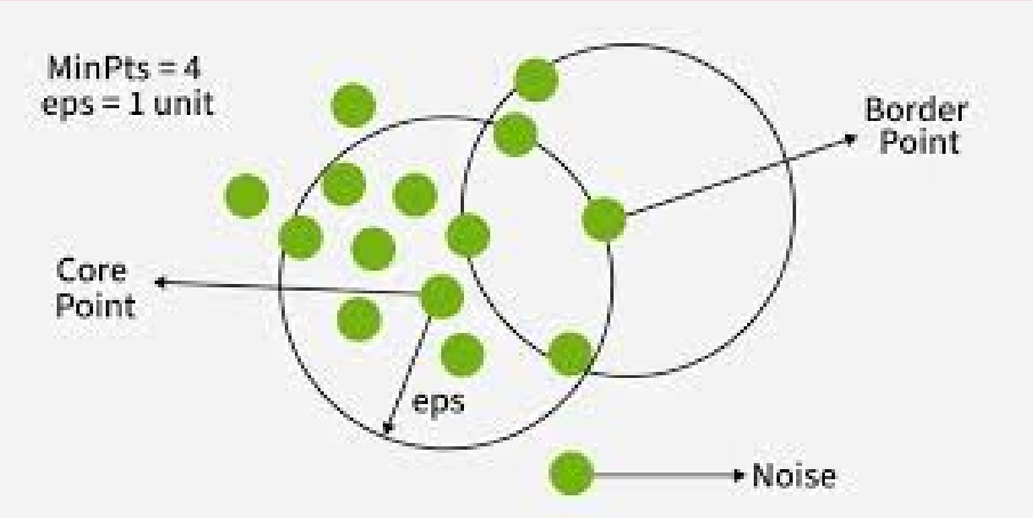
# Density-Based Clustering

Density-based clustering groups data based on how closely packed (dense) the points are.

Key idea:

- Clusters = regions where many points are close together
- Noise/Outliers = points in low-density areas

The most popular algorithm is DBSCAN.




**Main Concepts**

ε (Epsilon)

- Radius around a point → neighborhood distance.

MinPts

- Minimum number of points required inside ε to form a cluster.

**DBSCAN (Density-Based Spatial Clustering of Applications with Noise)**

The most famous and widely used algorithm.

How DBSCAN Works

1. Pick a point.
2. If it has ≥ MinPts inside radius ε → form a cluster.
3. Expand cluster by including all density-reachable points.
4. Points not reachable → marked as noise.

## OPTICS (Ordering Points To Identify Clustering Structure)

- Improved version of DBSCAN.
- Designed to solve DBSCAN's biggest problem: different density levels.

DBSCAN needs one ε value for entire dataset → fails if dataset has
both dense and sparse clusters.
OPTICS solves this by:

- Key Concepts
- Reachability Distance
- Core Distance
- Reachability Plot

### How OPTICS Works

1. Processes points in order of density.
2. Creates a reachability plot (graph).
3. You can choose clusters by cutting plot at different height levels.

## DENCLUE (DENsity CLUstEring)

A mathematically advanced density-based method.
Uses density functions instead of only ε & MinPts.

Main Idea
Each data point contributes to an overall density function (using kernel functions like Gaussian).
Clusters are found by the local maxima of this density function (density attractors).
Key Concepts

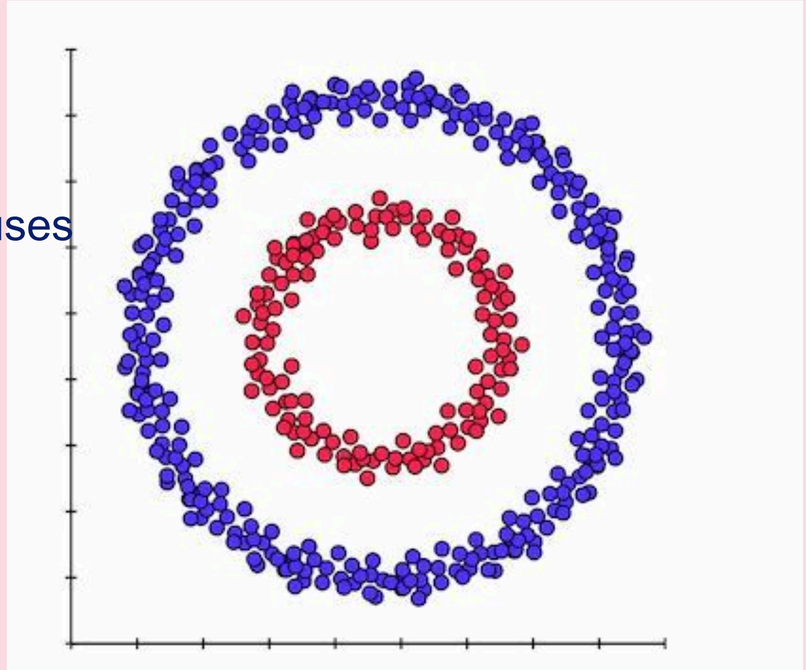
- Kernel Density Estimation (KDE)
- Density attractors (points where density is highest)
- Gradient ascent to move points towards density peaks

### How DENCLUE Works

1. Estimate density around every point using kernel functions.
2. Move points toward regions of maximum density.
3. Points converging to same density peak form a cluster.

# Spectral Clustering

- A clustering method that uses graph theory.
- It converts data into a graph, finds connections between points, and then uses eigenvalues/eigenvectors of the graph Laplacian to form clusters.



**How It Works (Simple Steps)**

- Create a similarity matrix : Shows how similar each pair of points is.
- Build a graph
  - Nodes = data points
  - Edges = strength of similarity
- Compute Graph Laplacian
- Find eigenvectors
  - These help separate data into meaningful groups.
- Apply a simple clustering method (like K-Means) on eigenvector space.

# Outlier analysis

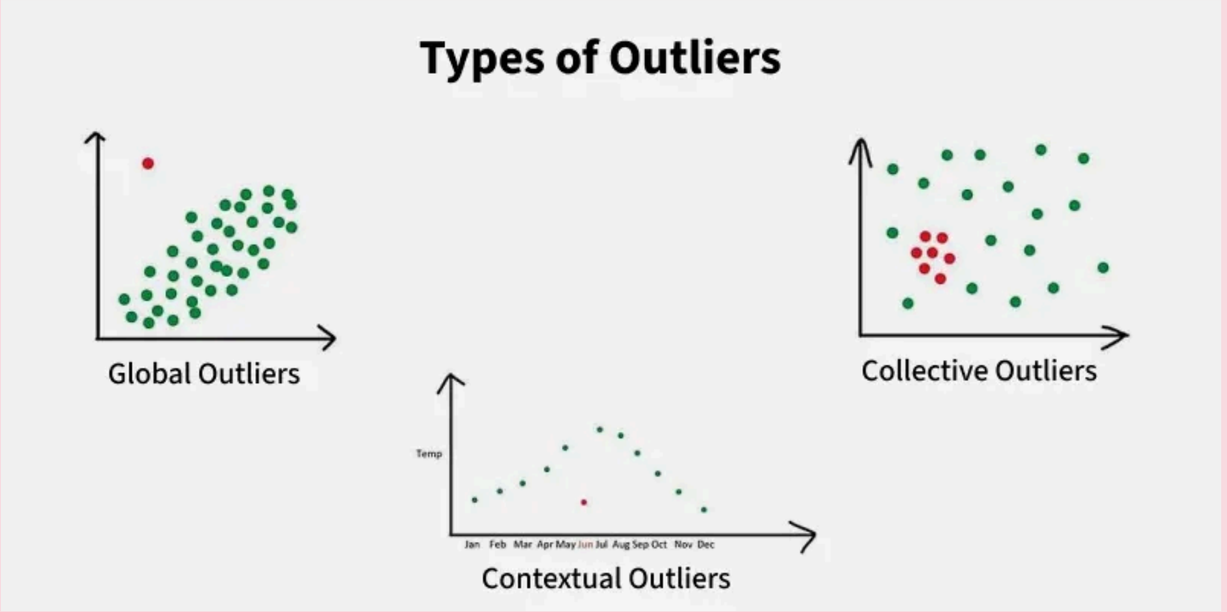
Outlier Analysis is the process of identifying data points that are very different from the majority of the data. These unusual points are called outliers or anomalies.

✓ Outliers can occur due to:

- Errors in data collection
- Fraudulent activities
- Unusual behavior
- Rare events

✓ Applications:

- Fraud detection (banking)
- Intrusion detection (cybersecurity)
- Fault detection (manufacturing)
- Healthcare (abnormal patient readings)



# Isolation Factor / Isolation Forest

Isolation Forest is a tree-based outlier detection algorithm.
It works on one simple idea:
Outliers are easier to isolate than normal points.

### ✓ How Isolation Forest Works

1. Randomly pick a feature (e.g., age, income).
2. Randomly pick a split value within that feature.
3. These random splits create a tree structure.
4. Outliers get isolated quickly, meaning they appear in:
    - ✓ Shallow nodes
    - ✓ Fewer splits needed
5. Normal points need more splits (deeper nodes) to be separated.

In short:

- Few splits → Outlier
- Many splits → Normal point

# Local Outlier Factor (LOF)

LOF is a density-based outlier detection method.
It focuses on the local density of points.

Key Idea:
A point is an outlier if it has much lower density than its neighbors.
So, LOF compares:

- How close a point is to its neighbors
- How dense the neighborhood is

### ✓ How LOF Works

1. For each point, find its k-nearest neighbors.
2. Calculate density around the point.
3. Compare this density with the density of its neighbors.
4. If the point is in a low-density area while its neighbors are in a high-density area → it is an outlier.

# Elbow Method

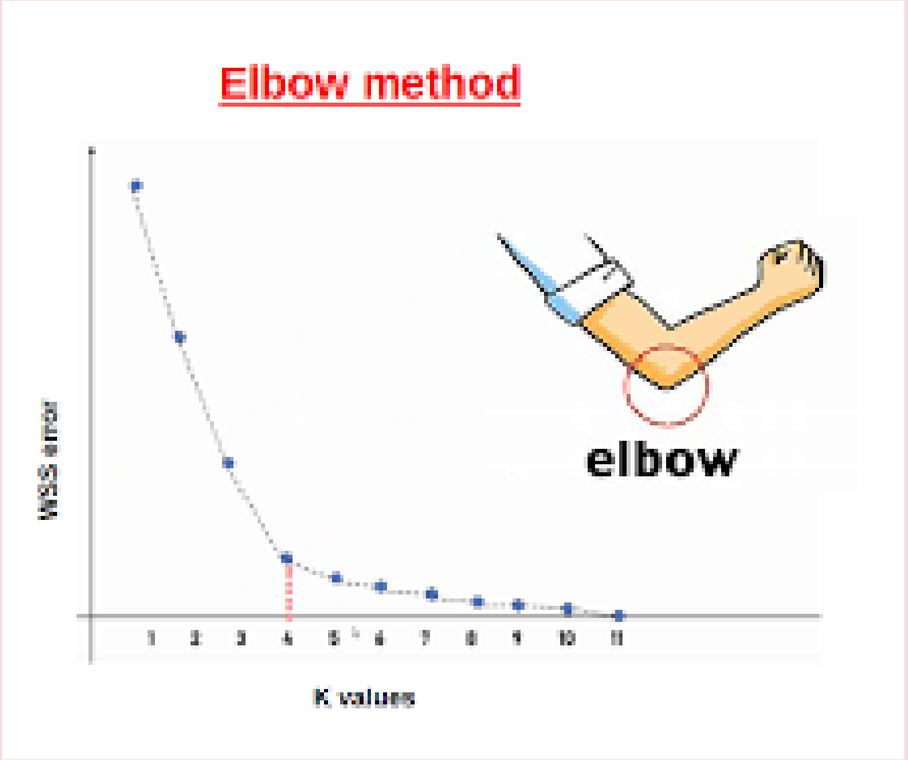
The Elbow Method is used to find the optimal number of clusters (K).

✓ **How It Works:**

1. Run K-Means for several values of K (e.g., K = 1 to 10)
2. For each K, compute WCSS (Within-Cluster Sum of Squares)

   WCSS = how compact each cluster is

1. Plot K vs WCSS
2. The curve decreases rapidly first, then slope becomes small
3. The point where it starts to bend like an “elbow” → Best K




✓ **Why Elbow Works?**

- Small K → Clusters are large → High WCSS
- Increasing K → Clusters become smaller → WCSS decreases
- After a point, improvement is minimal → That point is elbow

# Intrinsic Evaluation Methods

Definition

Intrinsic methods evaluate a model internally — focusing on the quality of the model itself, without using any external task.

Key Points

- Measures how well the model learns patterns from the given data.
- Evaluation happens within the model/output, not on real-world tasks.
- Faster & cheaper, because no external system or human feedback needed.

Examples

- Clustering Quality Metrics
  - Silhouette Score – how well points fit in their clusters.
  - Dunn Index – separation between clusters.
- In NLP / Embeddings
  - Word similarity, coherence measures.

When Used

- When testing unsupervised models (e.g., clustering, dimensionality reduction).
- When no external ground truth is available.

# Extrinsic Evaluation Methods

Definition

Extrinsic methods evaluate a model based on how well it performs in a real-world application or downstream task.

Key Points

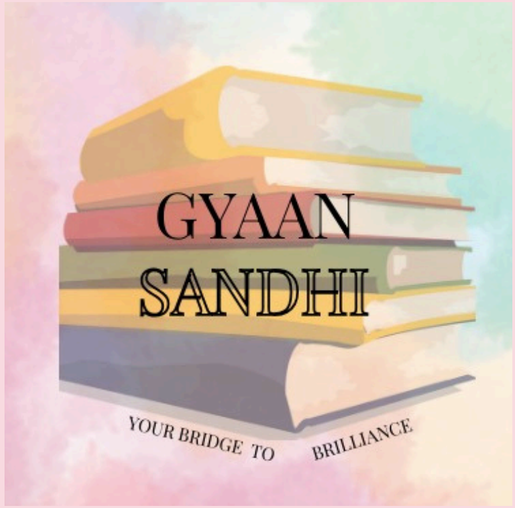
- Measures the real usefulness of the model.
- Uses external tasks, datasets, or objective results.
- More reliable, but time-consuming & costly.

Examples

- Using Clustering for Recommendation System
- Evaluate recommendation accuracy, engagement, conversions.
- Using Embeddings for Classification
- Measure precision, recall, F1-score.
- NLP tasks
- Machine translation accuracy, sentiment classification results.

When Used

- When you want to check practical performance.
- When the model is part of a larger application.




# THANK YOU


Share