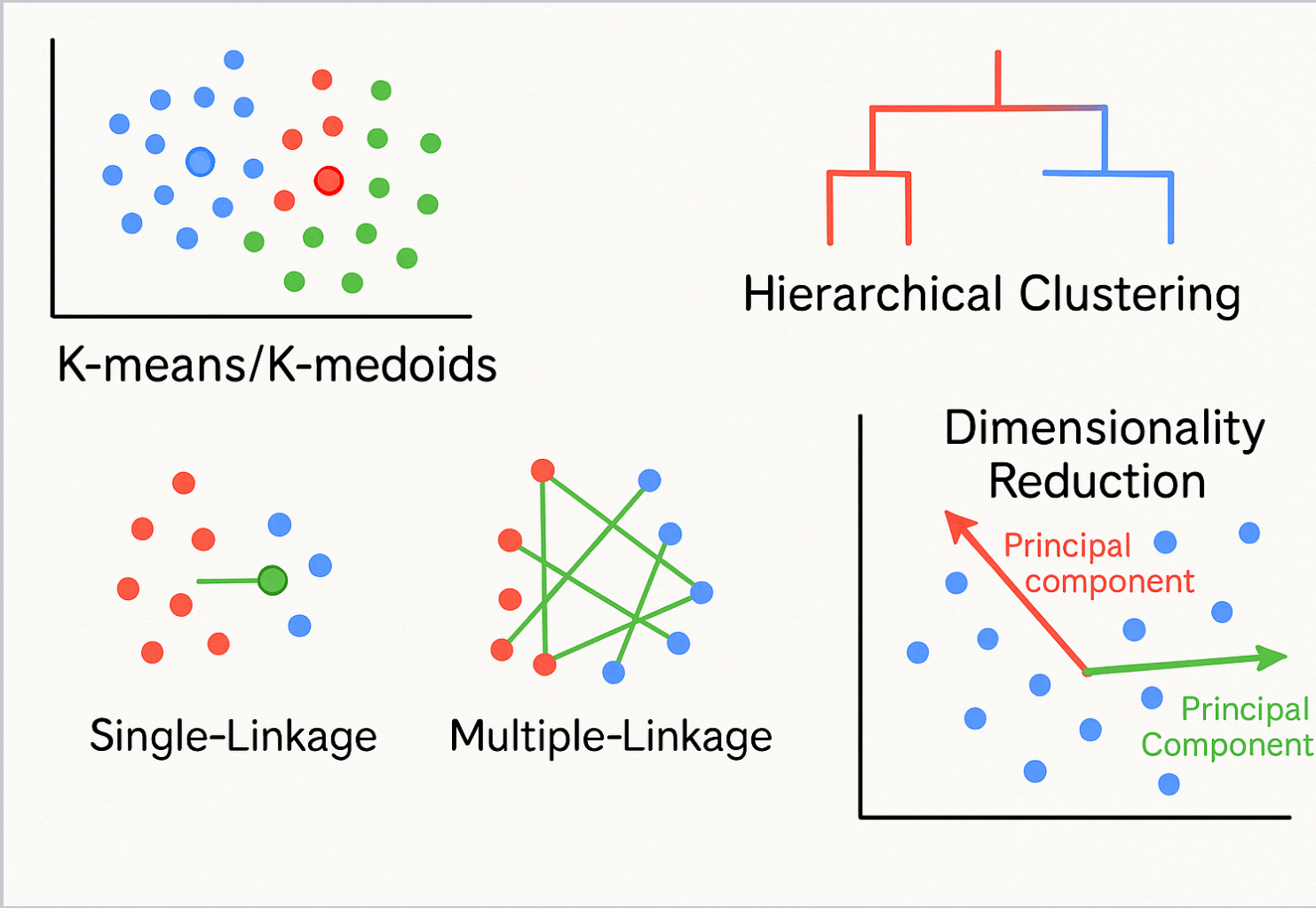
GATE – Data Science and Artificial Intelligence (DA)

Machine Learning: Unupervised Learning



GateXAIML

2026

Contents

About the Book

Artificial Intelligence and Machine Learning (AI/ML) are transforming industries across the globe — from healthcare and finance to transportation and education. From medical diagnosis systems and fraud detection to personalized recommendations and autonomous vehicles, AI/ML is shaping the way we live, work, and interact with technology.

To support this rapidly growing field, the GATE Data Science and Artificial Intelligence (DA) exam was introduced as a national-level gateway to higher studies, research, and employment opportunities in top institutions and organizations. The exam tests a candidate's proficiency in mathematics, programming, data handling, machine learning, and AI fundamentals.

This book is a compact and comprehensive guide for GATE DA aspirants. It is designed to help learners build a strong conceptual foundation while developing the problem-solving skills required for the exam. Many solved examples are included to illustrate key concepts, and each chapter features carefully crafted problems for practice.

Solutions to selected problems and topic-wise lectures will be discussed in detail on my YouTube channel (@GATEXAiml). All the concepts covered in the book will also be taught step-by-step through video tutorials, making this a complete learning resource for GATE DA preparation.

This book is designed for aspirants of the GATE DA exam focusing on **Machine Learning: Unupervised Learning**. It systematically covers theory, solved examples, and practice problems aligned with the official syllabus.

Dedicated to all my Gurus and Students.
"Knowledge grows only when shared — and it must remain free, for that is how it thrives."

Machine Learning: Unupervised Learning - Syllabus

Unsupervised Learning: clustering algorithms, k-means/k-medoid, hierarchical clustering, top-down, bottom-up: single-linkage, multiple-linkage, dimensionality reduction, principal component analysis.

STOP!

Attention!

Some examples solved in video lectures are different from those given in this book.

The procedure to solve problems and examples is well explained in the video lectures, and it is highly recommended to go through the video lectures for complete understanding.

Official Video Playlist

Watch on YouTube

Chapter 1

Unsupervised Learning

1.1 Introduction

Unsupervised learning is a type of machine learning where the model is provided with input data without corresponding output labels. The goal is to find hidden patterns, structures, or groupings in the data. Unlike supervised learning, there are no target variables, and the model must infer relationships on its own.

Key Idea

Given data $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$, the aim of unsupervised learning is to find:

- Clusters or groups of similar instances
- Low-dimensional representations (feature extraction)
- Anomalies or outliers

1.1.1 Applications

Unsupervised learning is widely used in:

- Market segmentation: Grouping customers with similar buying patterns
- Anomaly detection: Fraud detection, network intrusion
- Data compression: Dimensionality reduction using PCA
- Recommender systems: Discovering latent interests or preferences

Example 1:

A company has sales data for 1000 customers but no labels. Using clustering algorithms, they identify 3 distinct customer segments:

- High-value, frequent buyers

- Low-value, infrequent buyers
- Occasional buyers with seasonal trends

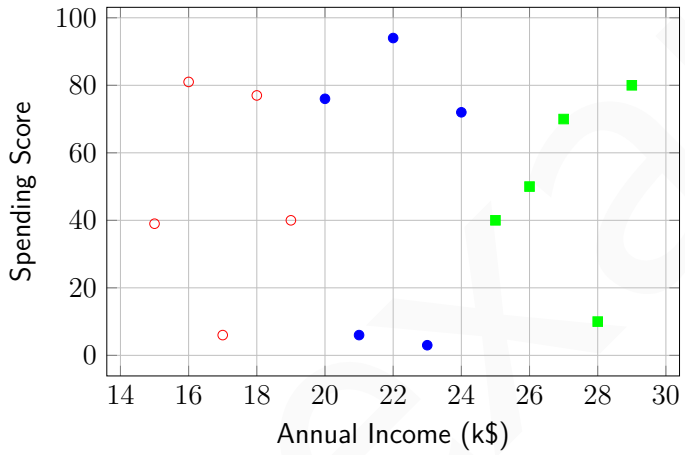
1.2 Types of Unsupervised Learning

Common Types

- **Clustering:** Partitioning data into groups of similar items (e.g., k-means, hierarchical clustering)
- **Dimensionality Reduction:** Representing data in lower dimensions while preserving structure (e.g., PCA, t-SNE)
- **Association Rules:** Discovering rules between variables in datasets (e.g., market basket analysis)

1.2.1 Clustering Example

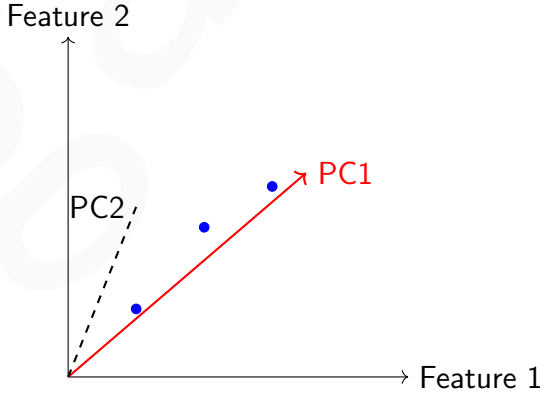
Consider 2D points representing customers' annual income and spending score. Using k-means clustering:



Explanation: The algorithm groups similar customers into clusters based on proximity in the 2D space. Each cluster can be analyzed for marketing strategies.

1.2.2 Dimensionality Reduction Example

Principal Component Analysis (PCA) projects high-dimensional data into lower dimensions:



Explanation: PC1 captures maximum variance along the data, allowing dimensionality reduction from 2D to 1D while preserving most information.

1.3 YouTube Links and QR Codes

Lecture	Details	YouTube Link	QR Code
38	Unsupervised Learning Overview & Learning Roadmap	`https://youtu.be/0ENj_e6zzLE`	

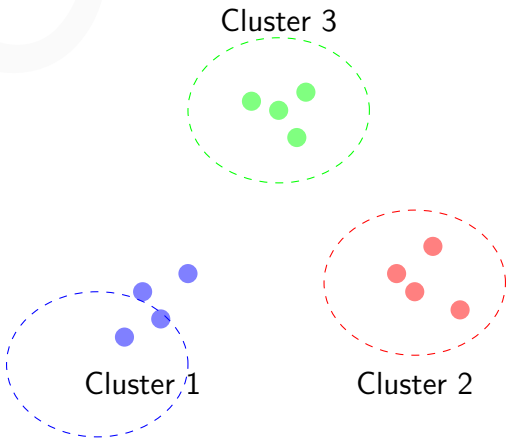
Chapter 2

Clustering Algorithms

2.1 Introduction

Clustering is an **unsupervised machine learning technique** that groups similar data points together into **clusters** based on their characteristics, **without using any labeled data**. The objective is to ensure that data points within the same cluster are more similar to each other than to those in different clusters, enabling the discovery of natural groupings and hidden patterns in complex datasets.

Goal: Discover the natural grouping or structure in unlabeled data without predefined categories.
How: Data points are assigned to clusters based on similarity or distance measures.
Similarity Measures: Euclidean distance, cosine similarity, or other metrics depending on data type and clustering method.
Output: Each group is assigned a cluster ID representing shared characteristics within the cluster.



Example of clustering in a 2D feature space

Example: Customer Purchase Data

If we have customer purchase data, clustering can group customers with similar shopping habits. These clusters can then be used for **targeted marketing**, **personalized recommendations**, or **customer segmentation**.

Types of Clustering

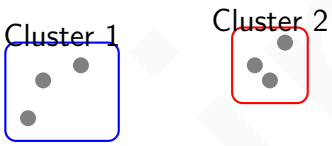
1. Hard Clustering

In **hard clustering**, each data point strictly belongs to **exactly one cluster**, no overlap is allowed. This approach assigns a clear membership.

Example: If clustering customer data into 2 segments, each customer belongs fully to either Cluster 1 or Cluster 2.

Use Cases: Market segmentation, customer grouping, document clustering.

Limitations: Cannot represent ambiguity or overlap; boundaries are crisp.



Hard Clustering: each point belongs to only one cluster

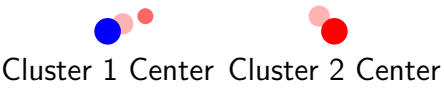
2. Soft Clustering

Soft clustering assigns each data point a **probability or degree of membership** to multiple clusters simultaneously. This allows data points to partially belong to several groups.

Example: A data point may have a 70% membership in Cluster 1 and 30% in Cluster 2.

Use Cases: Situations with overlapping boundaries, fuzzy categories, customer personas, or medical diagnosis.

Benefits: Captures ambiguity, models gradual transitions between clusters.



Soft Clustering: points can partially belong to multiple clusters

Types of Clustering Methods

Clustering methods can be classified based on how they form clusters:

1. Centroid-based Clustering (Partitioning Methods)

- Organizes data points around centroids (mean or medoid).
- **Algorithms:**
  - K-means: minimizes intra-cluster variance.
  - K-medoids: uses actual data points, robust to outliers.
- **Advantages:** Fast, scalable, easy to implement.
- **Limitations:** Requires number of clusters, sensitive to outliers, not suitable for non-spherical clusters.

2. Density-based Clustering (Model-based Methods)

- Forms clusters as high-density regions separated by low-density areas.
- **Algorithms:**
  - DBSCAN: identifies clusters and noise.
  - OPTICS: handles varying density clusters.
- **Advantages:** Detects arbitrary shapes, handles noise, cluster count not required.
- **Limitations:** Parameter selection can be difficult, less effective for varying densities (except OPTICS).

3. Connectivity-based Clustering (Hierarchical Clustering)

- Builds nested clusters based on connectivity among points.
- **Approaches:**
  - Agglomerative (Bottom-up): merge closest clusters iteratively.
  - Divisive (Top-down): split clusters iteratively.
- **Advantages:** Full hierarchy, easy visualization, cluster count not needed.
- **Limitations:** Computationally expensive, merging/splitting decisions are irreversible.

4. Distribution-based Clustering

- Assumes data comes from a mixture of probability distributions.
- **Algorithm:** Gaussian Mixture Model (GMM): assigns points based on likelihood.
- **Advantages:** Flexible cluster shapes, probabilistic memberships, handles overlapping clusters.
- **Limitations:** Requires number of components, computationally expensive, sensitive to initialization.

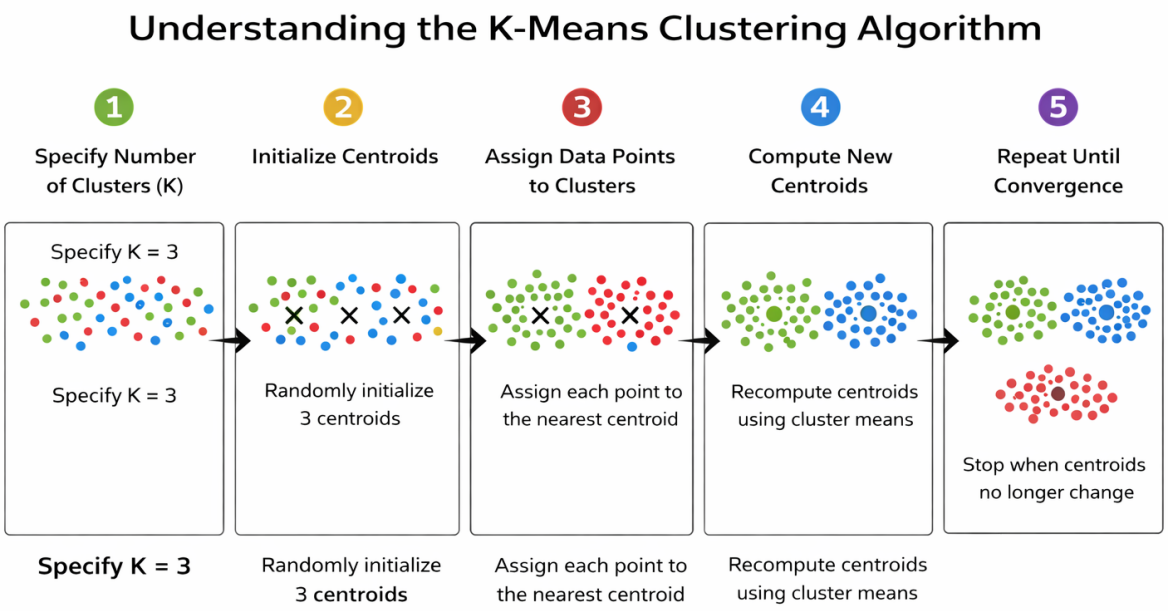
5. Fuzzy Clustering

- Allows points to belong to multiple clusters with degrees of membership.
- **Algorithm:** Fuzzy C-Means: updates fuzzy memberships iteratively.
- **Advantages:** Captures ambiguity, suitable for overlapping clusters.
- **Limitations:** Choosing fuzziness parameter can be tricky, higher computational cost.

2.2 Partition-based Clustering

Partition-based clustering algorithms divide the dataset into a predefined number of clusters k, optimizing a cluster-specific objective function. They are widely used because of their simplicity and efficiency, especially for large datasets. The most common algorithms are K-means and K-medoids.

2.2.1 K-means Clustering



K-means Algorithm

Step 1: Specify Number of Clusters

Let the dataset be

$$\mathcal{D} = \{x_1, x_2, \ldots, x_n\}, \quad x_i \in \mathbb{R}^d$$

Fix the number of clusters:

$$K \in \mathbb{N}$$

Step 2: Initialize Centroids Randomly
Select K distinct data points as initial centroids:

$$\mu_1^{(0)}, \mu_2^{(0)}, \ldots, \mu_K^{(0)} \in \mathcal{D}$$

Step 3: Distance Computation & Assignment Step (Cluster Membership)
For each data point x_i, compute the squared Euclidean distance to all centroids and assign it to the nearest one:

$$c_i^{(t)} = \arg\min_{k \in \{1,\ldots,K\}} \|x_i - \mu_k^{(t)}\|_2^2$$

Thus,

$$x_i \in C_k^{(t)} \quad \text{if } c_i^{(t)} = k$$

Step 4: Update Step (Centroid Recalculation)
Recompute each centroid as the mean of the data points assigned to that cluster:

$$\mu_k^{(t+1)} = \frac{1}{|C_k^{(t)}|} \sum_{x_i \in C_k^{(t)}} x_i$$

Step 5: Stopping Criterion (Convergence)
Iterate **Steps 3–4** until convergence, i.e., stop if either:

$$C_k^{(t+1)} = C_k^{(t)} \quad \forall k$$

or

$$\mu_k^{(t+1)} = \mu_k^{(t)} \quad \forall k$$

Example 2:

Given a dataset in $\mathbb{R}^2$,

$$D = \{x_1 = (1,1),\ x_2 = (1,2),\ x_3 = (4,4),\ x_4 = (5,4)\},$$

apply the **K-means clustering algorithm** with

$$K = 2$$

clusters. Perform the algorithm step by step until convergence.
—

Step 1: Specify Number of Clusters

$$K = 2$$

—

Step 2: Initialize Centroids

Choose two data points as initial centroids:

$$\mu_1^{(0)} = (1, 1), \qquad \mu_2^{(0)} = (5, 4)$$

—

Step 3: Distance Computation and Assignment

The squared Euclidean distance is given by

$$\|x - \mu\|^2 = (x_1 - \mu_1)^2 + (x_2 - \mu_2)^2$$

Point x_i	$\|x_i - \mu_1^{(0)}\|^2$	$\|x_i - \mu_2^{(0)}\|^2$	Assigned Cluster
$(1, 1)$	0	25	C_1
$(1, 2)$	1	20	C_1
$(4, 4)$	18	1	C_2
$(5, 4)$	25	0	C_2

$$C_1^{(0)} = \{(1, 1), (1, 2)\}, \qquad C_2^{(0)} = \{(4, 4), (5, 4)\}$$

—

Step 4: Update Step (Centroid Recalculation)

Compute the new centroids as the mean of points in each cluster.

$$\mu_1^{(1)} = \frac{(1, 1) + (1, 2)}{2} = (1, 1.5)$$

$$\mu_2^{(1)} = \frac{(4, 4) + (5, 4)}{2} = (4.5, 4)$$

—

Step 5: Reassignment Using Updated Centroids

Point x_i	$\|x_i - (1, 1.5)\|^2$	$\|x_i - (4.5, 4)\|^2$	Assigned Cluster
$(1, 1)$	0.25	18.5	C_1
$(1, 2)$	0.25	15.5	C_1
$(4, 4)$	15.25	0.25	C_2
$(5, 4)$	18.25	0.25	C_2

Since the cluster assignments do not change, the algorithm converges.
—

Final Result

Clusters

$$C_1 = \{(1, 1), (1, 2)\}, \qquad C_2 = \{(4, 4), (5, 4)\}$$

Centroids

$$\mu_1 = (1, 1.5), \qquad \mu_2 = (4.5, 4)$$

■

Advantages and Limitations of K-means

- **Advantages:** Simple, scalable, efficient for large datasets.
- **Limitations:** Sensitive to initialization, outliers, assumes spherical clusters, requires predefined k.

2.2.2 K-medoids Clustering

K-medoids Clustering: Concept

K-medoids clustering is similar to K-means but uses actual data points as cluster centers (medoids) instead of the mean. This makes it robust to outliers.

K-medoids Algorithm

Steps:

1. Initialize k medoids randomly from data points.
2. Assign each point to the nearest medoid.
3. For each cluster, try swapping medoid with another point in cluster; if total cost decreases, update medoid.

4. Repeat step 3 until no improvement.

K-medoids Algorithm

Step 1: Specify Number of Clusters

Let the dataset be

$$\mathcal{D} = \{x_1, x_2, \ldots, x_n\}, \quad x_i \in \mathbb{R}^d$$

Fix the number of clusters:

$$K \in \mathbb{N}$$

—

Step 2: Initialize Medoids Randomly

Select K distinct data points as initial medoids:

$$m_1^{(0)}, m_2^{(0)}, \ldots, m_K^{(0)} \in \mathcal{D}$$

Each m_k is a representative data point of cluster C_k.

—

Step 3: Distance Computation & Assignment Step

Assign each data point to the nearest medoid using a distance metric:

$$c_i^{(t)} = \arg\min_{k \in \{1,\ldots,K\}} \text{dist}(x_i, m_k^{(t)})$$

Thus,

$$x_i \in C_k^{(t)} \quad \text{if } c_i^{(t)} = k$$

—

Step 4: Update Step (Medoid Recalculation)

For each cluster $C_k^{(t)}$, choose the new medoid as the data point that minimizes the total distance to all other points in that cluster:

$$m_k^{(t+1)} = \arg\min_{x_j \in C_k^{(t)}} \sum_{x_i \in C_k^{(t)}} \text{dist}(x_i, x_j)$$

—

Step 5: Stopping Criterion (Convergence)

Iterate **Steps 3–4** until convergence, i.e., stop if:

$$m_k^{(t+1)} = m_k^{(t)} \quad \forall k$$

or cluster assignments do not change.

—

Objective Function

K-medoids minimizes the total within-cluster dissimilarity:

$$\boxed{\sum_{k=1}^{K} \sum_{x_i \in C_k} \text{dist}(x_i, m_k)}$$

Unlike K-means, the distance need not be squared and can be any valid metric.

Example 3:

Given Dataset

$$\mathcal{D} = \{1, 2, 4, 10\}, \quad K = 2$$

—

Initialization

Choose initial medoids:

$$m_1^{(0)} = 1, \qquad m_2^{(0)} = 10$$

—

Assignment Step

Using absolute distance:

Point	dist to 1	dist to 10	Cluster
1	0	9	C_1
2	1	8	C_1
4	3	6	C_1
10	9	0	C_2

$$C_1 = \{1, 2, 4\}, \quad C_2 = \{10\}$$

—

Medoid Update

Compute total distances within C_1:

$$\begin{aligned}
\text{cost}(1) &= |1-1| + |2-1| + |4-1| = 4 \\
\text{cost}(2) &= |1-2| + |2-2| + |4-2| = 3 \\
\text{cost}(4) &= |1-4| + |2-4| + |4-4| = 5
\end{aligned}$$

Thus, the new medoid is:

$$m_1^{(1)} = 2$$

For cluster C_2:

$$m_2^{(1)} = 10$$

—

Convergence

Medoids change from $(1, 10)$ to $(2, 10)$. Reassignment does not change the clusters; hence the algorithm converges.

—

Final Result

Clusters

$$C_1 = \{1, 2, 4\}, \quad C_2 = \{10\}$$

Medoids

$$m_1 = 2, \quad m_2 = 10$$

■

Comparison: K-means vs K-medoids

- K-means uses mean, sensitive to outliers; K-medoids uses medoid, robust to outliers.
- K-means is faster (simple arithmetic mean), K-medoids is computationally expensive (pairwise distances).
- Both require pre-defined k.

2.3 Hierarchical Clustering

Hierarchical clustering is an unsupervised learning technique that seeks to build a hierarchy of clusters. Unlike K-means, it does not require a pre-specified number of clusters. Hierarchical clustering can be agglomerative (bottom-up) or divisive (top-down).

2.3.1 Agglomerative (Bottom-Up) Approach

Agglomerative clustering starts with each point as a singleton cluster and iteratively merges the two closest clusters until a stopping criterion is met (e.g., all points in one cluster or a threshold distance).

Agglomerative (Bottom-Up) Approach

Algorithm: Bottom-Up (Agglomerative) Hierarchical Clustering

1. **Initialization:** Start with each data point as its own cluster:

$$C_i = \{x_i\}, \quad i = 1, 2, \ldots, n$$

2. **Compute Distance Matrix:** Compute all pairwise distances between clusters using a chosen metric (e.g., Euclidean distance):

$$d(C_i, C_j) = \min / \max / \text{average}_{x \in C_i, y \in C_j} d(x, y)$$

depending on the linkage criterion (single, complete, or average).

3. **Merge Clusters:** Identify the pair of clusters with the minimum distance according to the chosen linkage criterion and merge them.
4. **Update Distances:** Recompute distances between the new cluster and all remaining clusters using the linkage criterion.
5. **Stopping Criterion:** Repeat Steps 3–4 until one of the following conditions is met:
   - Desired number of clusters is reached.
   - Only a single cluster remains (complete hierarchy formed).

The key step is defining the inter-cluster distance. Two popular methods are:

2.3.1.1 Single-Linkage Method

Single-Linkage Method

Definition: The distance between two clusters A and B is the minimum distance between any two points in the clusters:

$$D_{\text{single}}(A, B) = \min_{x \in A, y \in B} \|x - y\|$$

Characteristics:

- Can produce elongated, chain-like clusters.
- Sensitive to noise and outliers.

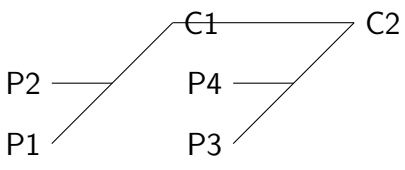
Example 4:

Consider points on a 2D plane: $P_1 = (1, 1), P_2 = (2, 1), P_3 = (5, 5), P_4 = (6, 5)$. Perform single-linkage clustering.

Solution:

1. Compute all pairwise distances: $P_1 - P_2 = 1$, $P_1 - P_3 \approx 5.66$, $P_1 - P_4 \approx 6.40$, $P_2 - P_3 \approx 5.0$, $P_2 - P_4 \approx 5.83$, $P_3 - P_4 = 1$.
2. Merge closest pair P_1 and $P_2 \rightarrow$ Cluster $C_1 = \{P_1, P_2\}$. Merge closest remaining pair P_3 and $P_4 \rightarrow$ Cluster $C_2 = \{P_3, P_4\}$.
3. Compute inter-cluster distance $D_{\text{single}}(C_1, C_2) = \min\{\|P_1 - P_3\|, \|P_1 - P_4\|, \|P_2 - P_3\|, \|P_2 - P_4\|\} = 5.0$.
4. Merge C_1 and C_2 if required.

Dendrogram:



2.3.1.2 Complete/Multiple-Linkage Method

Complete-Linkage Method

Definition: The distance between two clusters A and B is the maximum distance between any two points in the clusters:

$$D_{\text{complete}}(A, B) = \max_{x \in A, y \in B} \|x - y\|$$

Characteristics:

- Produces compact, spherical clusters.
- Less sensitive to outliers compared to single-linkage.

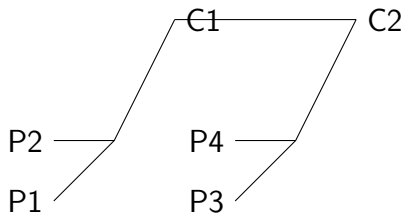
Example 5:

Using the same points as above: $P_1 = (1, 1), P_2 = (2, 1), P_3 = (5, 5), P_4 = (6, 5)$, perform complete-linkage clustering.

Solution:

1. Merge closest pair P_1 and $P_2 \rightarrow$ Cluster $C_1 = \{P_1, P_2\}$, and P_3 and $P_4 \rightarrow C_2 = \{P_3, P_4\}$.
2. Compute inter-cluster distance $D_{\text{complete}}(C_1, C_2) = \max\{\|P_1 - P_3\|, \|P_1 - P_4\|, \|P_2 - P_3\|, \|P_2 - P_4\|\} = 6.40$.
3. Merge only if threshold allows.

Dendrogram: (Shows complete linkage produces slightly higher linkage height)



Example 6:

Example: Bottom-Up (Agglomerative) Hierarchical Clustering
Consider the one-dimensional dataset:

$$D = \{x_1 = 1,\ x_2 = 2,\ x_3 = 5,\ x_4 = 6\}$$

with Euclidean distance

$$d(x_i, x_j) = |x_i - x_j|.$$

Initially, each point is a separate cluster:

$$C_1 = \{1\},\ C_2 = \{2\},\ C_3 = \{5\},\ C_4 = \{6\}.$$

Step 1: Compute Pairwise Distance Matrix

	1	2	5	6
1	0	1	4	5
2	1	0	3	4
5	4	3	0	1
6	5	4	1	0

Step 2: Single Linkage Clustering

1. Merge the two clusters with minimum distance. Minimum distance $= 1$, merge C_1 and C_2:

$$C_{12} = \{1, 2\}$$

   Remaining clusters: C_{12}, C_3, C_4

2. Next minimum distance $= 1$, merge C_3 and C_4:

$$C_{34} = \{5, 6\}$$

3. Distance between C_{12} and C_{34} (single linkage):

$$d_{\text{single}}(C_{12}, C_{34}) = \min\{3, 4, 4, 5\} = 3$$

   Merge to form final cluster $C_{1234} = \{1, 2, 5, 6\}$

Step 3: Complete Linkage Clustering

1. Merge C_1 and C_2, $C_{12} = \{1, 2\}$
2. Merge C_3 and C_4, $C_{34} = \{5, 6\}$
3. Distance between C_{12} and C_{34} (complete linkage):

$$d_{\text{complete}}(C_{12}, C_{34}) = \max\{4, 5, 3, 4\} = 5$$

   Merge to form final cluster $C_{1234} = \{1, 2, 5, 6\}$

Step 4: Average Linkage Clustering

1. Merge C_1 and C_2, $C_{12} = \{1, 2\}$
2. Merge C_3 and C_4, $C_{34} = \{5, 6\}$

3. Distance between C_{12} and C_{34} (average linkage):

$$d_{\text{avg}}(C_{12}, C_{34}) = \frac{4+5+3+4}{4} = 4$$

Merge to form final cluster $C_{1234} = \{1, 2, 5, 6\}$

Step 5: Observations

- Single linkage produces **chained, elongated clusters**.
- Complete linkage produces **compact clusters**.
- Average linkage gives a **balanced clustering**.

2.3.2 Divisive (Top-Down) Approach

Divisive (Top-Down) Clustering

Definition: Divisive clustering starts with all points in a single cluster and recursively splits clusters into smaller sub-clusters until each point forms its own cluster or a stopping criterion is satisfied.
Procedure:

1. Start with the full dataset as a single cluster.
2. Select a cluster to split based on a criterion (e.g., maximize intra-cluster distance or minimize inter-cluster similarity).
3. Split the selected cluster into two sub-clusters. This can be done using methods such as:
   - K-means or K-medoids on the cluster to partition it into two groups.
   - Principal Component Analysis (PCA) to find the direction of maximum variance and split along it.
4. Repeat steps 2–3 recursively on the new clusters until stopping criteria are met (e.g., desired number of clusters, minimal cluster size, or distance threshold).

Distance Measure: For splitting, a common choice is the diameter of the cluster:

$$\text{diameter}(C) = \max_{x_i, x_j \in C} \|x_i - x_j\|$$

Clusters with larger diameter are often chosen first to split.
Characteristics:

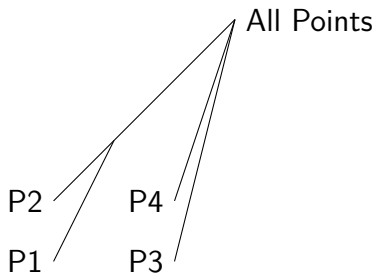
- Produces a hierarchy of clusters similar to agglomerative clustering but in reverse order.
- Computationally more expensive than agglomerative methods for large datasets.
- Flexible in choosing splitting criteria and distance measures.

Example 7:

Consider points $P_1 = (1, 1), P_2 = (2, 1), P_3 = (5, 5), P_4 = (6, 5)$.

1. Start with cluster $C = \{P_1, P_2, P_3, P_4\}$.
2. Compute the pairwise distances and find the two points with maximum dissimilarity. Here, $\|P_1 - P_4\| = 6.40$.
3. Split the cluster into $C_1 = \{P_1, P_2\}$ and $C_2 = \{P_3, P_4\}$.
4. Recursively apply the same procedure if more splits are required.

Dendrogram:



Comparison:

- Agglomerative is bottom-up, divisive is top-down.
- Agglomerative is more commonly used in practice.
- Choice of linkage affects cluster shape and sensitivity to noise.

Example 8:

Example: Divisive (Top-Down) Hierarchical Clustering
Consider the one-dimensional dataset:

$$D = \{1, 2, 5, 6\}$$

with Euclidean distance:

$$d(x_i, x_j) = |x_i - x_j|.$$

Step 1: Start with a single cluster

$$C^{(0)} = \{1, 2, 5, 6\}$$

Step 2: Identify cluster to split
- Only one cluster exists. - Compute cluster diameter:

$$\text{diameter}(C^{(0)}) = \max(D) - \min(D) = 6 - 1 = 5$$

Step 3: Split the cluster
- Use k-means with $K = 2$ to divide the cluster:

$$C_1 = \{1, 2\}, \quad C_2 = \{5, 6\}$$

- Compute cluster centroids:

$$\mu_1 = \frac{1+2}{2} = 1.5, \quad \mu_2 = \frac{5+6}{2} = 5.5$$

Step 4: Update clusters

$$C_1 = \{1, 2\}, \quad C_2 = \{5, 6\}$$

Step 5: Stopping criterion
- Desired number of clusters ($K = 2$) is reached, so **stop**.

Final Result

Cluster	Points	Centroid
C_1	1, 2	1.5
C_2	5, 6	5.5

Observation: Divisive clustering works *top-down*, splitting large clusters recursively. Linkage criteria are **not used**, unlike in agglomerative clustering.

2.4 Problems

Problem 251 *Which of the following statements about K-means clustering is correct?*

(A) K-means always finds the global optimum of the clustering objective

(B) K-means minimizes the sum of squared Euclidean distances between points and their assigned centroids

(C) K-means can naturally handle non-spherical clusters

(D) K-means is insensitive to outliers

Problem 252 *Which of the following factors influence the result of K-means clustering?*

(A) Choice of initial centroids

(B) Number of clusters k

(C) Scaling of features

(D) Distance metric used

Problem 253 *Which statement is true about K-medoids compared to K-means?*

(A) K-medoids selects cluster centers that are actual data points

(B) *K-medoids is more sensitive to outliers than K-means*

(C) *K-medoids minimizes the sum of squared Euclidean distances*

(D) *K-medoids always converges faster than K-means*

Problem 254 *Which statements are correct regarding the convergence of K-means algorithm?*

(A) *K-means converges in a finite number of iterations*

(B) *The objective function never increases across iterations*

(C) *K-means guarantees a globally optimal solution*

(D) *Convergence depends on initial centroid positions*

Problem 255 *What happens if all data points in K-means are equidistant from each other?*

(A) *The algorithm produces a unique clustering*

(B) *The clustering depends on initialization of centroids*

(C) *The algorithm will never converge*

(D) *All points will be assigned to a single cluster*

Problem 256 *Which statements are true about the K-means objective function?*

(A) *It decreases monotonically after each iteration*

(B) *It can have multiple local minima*

(C) *It is equivalent to maximizing inter-cluster distances*

(D) *It is convex in cluster assignments*

Problem 257 *Outliers in K-means clustering can cause:*

(A) *Centroids to shift toward the outliers*

(B) *Outliers to form separate clusters automatically*

(C) *No effect on clustering*

(D) *Faster convergence*

Problem 258 *Which property is unique to K-medoids compared to K-means?*

(A) *Uses mean of cluster points*

(B) *Minimizes sum of squared distances*

(C) *Medoid is an actual data point*

(D) *Faster computation*

Problem 259 *Consider the following points in 2D space:* $(1,2),(2,1),(3,4),(5,6),(8,8)$. *If K-means clustering is performed with* $k=2$ *and initial centroids* $(1,2)$ *and* $(5,6)$*, after the first iteration, which points belong to cluster 1 (centroid* $(1,2)$*)?*

(A) $(1,2),(2,1),(3,4)$

(B) $(1,2),(2,1)$

(C) $(1,2),(3,4)$

(D) $(1,2),(5,6)$

Problem 260 *A dataset has three clusters with points:* $C_1=\{(1,1),(1,2)\}$, $C_2=\{(5,5),(6,5)\}$, $C_3=\{(9,1),(8,2)\}$. *Using K-medoids with* $k=3$*, which of the following can be valid medoids after convergence?*

(A) $(1,1),(5,5),(8,2)$

(B) $(1,2),(6,5),(9,1)$

(C) $(1,1),(6,5),(9,1)$

(D) All of the above

Problem 261 *For K-means with* $k=2$ *on the points* $(1,1),(2,2),(3,3),(10,10)$*, which of the following statements is true about cluster centroids after one iteration (assuming initial centroids* $(1,1)$ *and* $(10,10)$*)?*

(A) Centroids remain unchanged

(B) Centroid 1 moves to $(2,4)$

(C) Centroid 1 moves to $(2,2)$*, Centroid 2 moves to* $(6.5,6.5)$

(D) Centroid 1 moves to $(2,2)$*, Centroid 2 moves to* $(10,10)$

Problem 262 *Points* $(1,1),(1,3),(3,1),(8,8),(9,9)$. *Apply K-means with* $k=2$. *After convergence, the centroid of the cluster containing* $(8,8)$ *is* _____.

Problem 263 *In complete-linkage hierarchical clustering, which of the following statements is true?*

(A) Maximum inter-cluster distance is used for merging.

(B) Minimum inter-cluster distance is used for merging.

(C) Average distance between all points in clusters is used.

(D) Clusters are merged randomly.

Problem 264 *Divisive hierarchical clustering uses which of the following for splitting clusters?*

(A) Largest intra-cluster distance

(B) Smallest inter-cluster distance

(C) *Random split*

(D) *Maximum number of points*

Problem 265 *Which of the following is true about complete-linkage agglomerative clustering?*

(A) *Produces compact, spherical clusters*

(B) *Sensitive to outliers*

(C) *Can produce chain-like clusters*

(D) *Merges clusters based on minimum distance*

Problem 266 *In hierarchical clustering, which of the following statements are correct? (Select all that apply)*

(A) *Agglomerative clustering is bottom-up*

(B) *Divisive clustering is top-down*

(C) *Complete-linkage merges clusters based on minimum distance*

(D) *Single-linkage produces elongated clusters*

Problem 267 *In hierarchical clustering, the dendrogram height represents:*

(A) *Distance at which clusters merge*

(B) *Number of points in cluster*

(C) *Number of clusters at that level*

(D) *Variance of points in the cluster*

Problem 268 *Which of the following is true about the computational complexity of hierarchical clustering?*

(A) *Agglomerative clustering is* $O(n^2 \log n)$ *with naive implementation*

(B) *Divisive clustering is always* $O(n^3)$

(C) *Both are* $O(n)$

(D) *Agglomerative clustering is* $O(n)$

Problem 269 *Divisive hierarchical clustering is preferred over agglomerative when:*

(A) *A small number of very large clusters is needed*

(B) *Many tiny clusters are required*

(C) *Data is extremely large*

(D) *There are no outliers*

Problem 270 *Consider a dataset with points $A(0,0)$, $B(1,0)$, $C(4,0)$, $D(5,0)$. Using agglomerative clustering with complete linkage, what is the first cluster merge?*

(A) *Merge* (A, B)

(B) *Merge* (C, D)

(C) *Merge* (A, C)

(D) *Merge* (B, D)

Problem 271 *Consider points $P(0,0)$, $Q(0,2)$, $R(2,0)$, $S(2,2)$. Using complete-linkage agglomerative clustering, what is the height of the dendrogram at which P and R merge?*

(A) *1*

(B) *2*

(C) $\sqrt{2}$

(D) $2\sqrt{2}$

Problem 272 *For a divisive clustering algorithm on 1D points $\{1, 2, 3, 10, 11, 12\}$, the first split should ideally separate:*

(A) $\{1, 2, 3\}$ *and* $\{10, 11, 12\}$

(B) $\{1, 2\}$ *and* $\{3, 10, 11, 12\}$

(C) $\{1, 2, 3, 10\}$ *and* $\{11, 12\}$

(D) $\{1\}$ *and* $\{2, 3, 10, 11, 12\}$

2.5 Try it Yourself

Exercise 1 *A dataset has 4 points: $(1,1), (1,3), (4,1), (4,3)$. You run K-means with $k = 2$ and initial centroids at $(1,1)$ and $(4,3)$. After one iteration, compute the new centroid of the first cluster.*

Exercise 2 *Given points $(2,0), (4,0), (0,2), (0,4)$ and $k = 2$, run K-means with initial centroids $(2,0)$ and $(0,4)$. Compute the sum of squared distances (objective function) after the first assignment step.*

Exercise 3 *A 1-dimensional dataset: $X = \{1, 2, 5, 6, 8\}$. Compute the new centroids after running one iteration of K-means with $k = 2$ and initial centroids $c_1 = 2, c_2 = 6$.*

Exercise 4 *Dataset: $(0,0), (0,1), (1,0), (1,1), (5,5), (6,5), (5,6), (6,6)$. Using K-medoids with $k = 2$, determine the medoid of the cluster containing $(0,0)$ after the first iteration.*

Exercise 5 *For points $(1,2), (2,1), (2,2), (3,3)$, $k = 2$, initial centroids $(1,2)$ and $(3,3)$. After one iteration of K-means, compute the distance of each point to its assigned centroid and sum them.*

Exercise 6 *A 1D dataset $X = \{1, 3, 4, 8, 9\}$ is clustered using K-means with $k = 2$, initial centroids 2 and 8. Compute the total within-cluster variance after the first iteration.*

Exercise 7 *Points* $(0,0),(0,2),(2,0),(2,2),(5,5),(6,5),(5,6)$ *are clustered using K-means with* $k=2$*, initial centroids* $(0,0)$ *and* $(5,5)$*. Compute the new centroid of the cluster containing* $(5,5)$ *after first assignment.*

Exercise 8 *A dataset has points* $(1,1),(2,1),(1,2),(2,2),(8,8),(9,8),(8,9),(9,9)$*. Using K-medoids with* $k=2$*, compute the medoid of the cluster that contains* $(9,9)$ *after the first iteration.*

Exercise 9 *Consider a dataset:* $(1,1),(1,4),(4,1),(4,4),(8,8)$*. Using K-means with* $k=3$ *and initial centroids* $(1,1),(4,4),(8,8)$*, compute the centroid of the cluster containing* $(1,4)$ *after the first iteration.*

Exercise 10 *Dataset:* $(2,3),(3,2),(3,3),(6,6),(7,6),(6,7)$*. Using K-means with* $k=2$*, initial centroids* $(2,3)$ *and* $(6,6)$*, compute the Euclidean distance between the new centroids after the first iteration.*

Exercise 11 *Consider points* $A(0,0)$*,* $B(2,0)$*,* $C(5,0)$*. Using complete-linkage agglomerative clustering, compute the distance at which the first merge occurs.*

Exercise 12 *Points* $X(1,2)$*,* $Y(4,6)$*,* $Z(5,8)$ *are clustered using complete linkage. What is the height of the dendrogram when* Y *and* Z *merge?*

Exercise 13 *For 1D points* $1,3,7,8$*, using complete-linkage agglomerative clustering, at what distance do the last two clusters merge?*

Exercise 14 *Consider points* $P(0,0)$*,* $Q(0,3)$*,* $R(4,0)$*,* $S(4,3)$*. Compute the distance at which the first merge occurs using complete-linkage clustering.*

Exercise 15 *For points* $A(1,1)$*,* $B(1,4)$*,* $C(5,1)$*,* $D(5,4)$*, find the complete-linkage distance between clusters* $\{A,B\}$ *and* $\{C,D\}$*.*

Exercise 16 *Consider points* $X=\{2,5,10,12\}$*. Using divisive clustering, determine the first split by calculating the largest intra-cluster distance.*

Exercise 17 *Points* $A(1,0)$*,* $B(2,0)$*,* $C(3,0)$*,* $D(10,0)$*. In complete-linkage clustering, what is the dendrogram height when clusters* $\{A,B,C\}$ *and* $\{D\}$ *merge?*

Exercise 18 *Given points* $P(0,0)$*,* $Q(1,1)$*,* $R(2,2)$*,* $S(5,5)$*, compute the distance at which the last merge occurs using complete-linkage agglomerative clustering.*

Exercise 19 *For points* $1,2,4,7,8$*, find the height of the dendrogram when the clusters* $\{1,2\}$ *and* $\{4,7,8\}$ *merge using complete linkage.*

Exercise 20 *Points* $A(0,0)$*,* $B(0,4)$*,* $C(3,0)$*,* $D(3,4)$ *are clustered using complete linkage. Calculate the dendrogram height at which clusters* $\{A,B\}$ *and* $\{C,D\}$ *merge.*

Exercise 21 *Consider points* $(1,1)$*,* $(1,2)$*,* $(2,1)$*,* $(5,5)$*,* $(5,6)$*,* $(6,5)$*. Agglomerative clustering with complete linkage is applied. Which cluster merges last?*

(A) $\{(1,1),(1,2),(2,1)\}$ *and* $\{(5,5),(5,6),(6,5)\}$

(B) $(1,1)$ *and* $(1,2)$

(C) $(5,5)$ *and* $(5,6)$

(D) $(5, 6)$ *and* $(6, 5)$

Exercise 22 *Consider 1D points* $\{2, 3, 4, 10, 11\}$. *Using complete-linkage agglomerative clustering, the order of merges is:*

(A) $(2, 3)$, $(3, 4)$, $(10, 11)$, $(2, 3, 4, 10, 11)$

(B) $(2, 3)$, $(3, 4)$, $(10, 11)$, $(2, 3, 4), (10, 11)$

(C) $(2, 3)$, $(4, 10)$, $(3, 4)$, $(10, 11)$

(D) $(2, 4)$, $(3, 10)$, $(4, 10)$, $(10, 11)$

2.6 YouTube Links and QR Codes

Lecture	Details	YouTube Link	QR Code
39	K-Means Clustering Explained: Partition-Based Clustering	https://youtu.be/de2BJXkc6vs	
40	K-Medoids Clustering Explained: Robust Partition-Based Algorithm	https://youtu.be/NJmF4_xt2ko	
41	Problem Solving on K-Means & K-Medoids (Problems 251–262)	https://youtu.be/U9b8OZ1JXy0	
42	Hierarchical Clustering Explained: Agglomerative & Divisive	https://youtu.be/nJUmCbnq7D0	

43	Hierarchical Clustering Solutions to Problems 263–272 & Divisive	https://youtu.be/ASFGi6uEwmI	

Chapter 3

Dimensionality Reduction

3.1 Principal Component Analysis (PCA)

3.1.1 Introduction and Motivation

Dimensionality reduction is crucial in machine learning when dealing with high-dimensional datasets. High dimensions lead to several challenges:

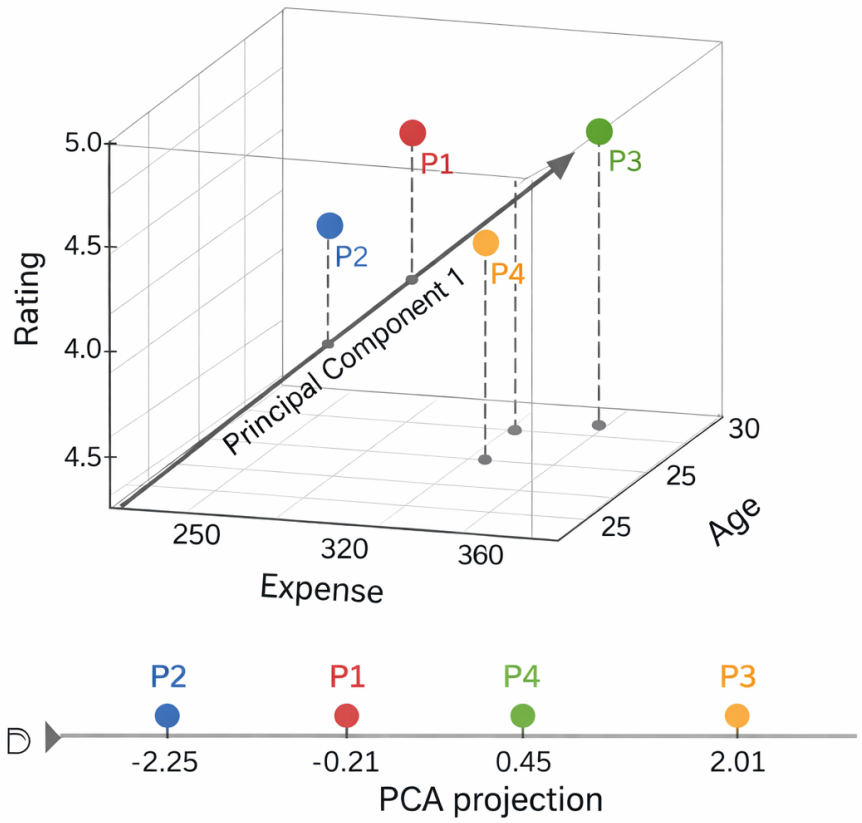
- **Curse of dimensionality:** Distances become less meaningful as dimensionality increases.
- **Visualization:** Difficult to visualize data beyond 3 dimensions.
- **Computational cost:** More features mean more computation in ML models.
- **Redundancy:** Many features are correlated and carry overlapping information.

PCA Concept

Idea: PCA transforms the original correlated features into a smaller set of uncorrelated variables called **principal components (PCs)** while retaining most of the variance in the data.
Benefits:

- Reduces dimensionality and computational cost.
- Removes feature correlations.
- Preserves maximum variance.

3.1.2 Visualization of PCA Projection



3.1.3 Mathematical Formulation

Setup Assuming population data:
Let there be n data points and d features.

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \in \mathbb{R}^{n \times d}$$

Here,

$$i = 1, \ldots, n \quad \text{(samples)}, \qquad j = 1, \ldots, d \quad \text{(features)}$$

—

Step 1: Data Standardization (Zero Mean, Unit Variance)
Mean of feature j:

$$\mu_j = \frac{1}{n} \sum_{i=1}^{n} x_{ij}$$

Standard deviation of feature j:

$$\sigma_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \mu_j)^2}$$

Standardized data:

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

—

Step 2: Standardized Data Matrix

$$Z = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1d} \\ z_{21} & z_{22} & \cdots & z_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ z_{n1} & z_{n2} & \cdots & z_{nd} \end{bmatrix} \in \mathbb{R}^{n \times d}$$

—

Step 3: Covariance Matrix

$$C = \frac{1}{n} Z^T Z \quad \in \mathbb{R}^{d \times d}$$

Properties:

$$C = C^T \quad \text{(symmetric)}$$

$$\lambda_i \geq 0 \quad \text{(positive semi-definite)}$$

—

Step 4: Eigenvalues and Eigenvectors

Solve:

$$|C - \lambda I| = 0$$

$$Cv_i = \lambda_i v_i$$

Eigenvalues ordered as:

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$$

Interpretation:

$$\lambda_i \rightarrow \text{variance along direction } v_i$$

$$v_i \rightarrow \text{principal axis}$$

—

Step 5: Selection of Principal Components

Choose top k eigenvectors corresponding to largest eigenvalues.

$$W = [v_1 \; v_2 \; \cdots \; v_k] \in \mathbb{R}^{d \times k}$$

Explained variance ratio:

$$\text{Variance explained by } \lambda_i = \frac{\lambda_i}{\sum_{j=1}^{d} \lambda_j}$$

—

Step 6: Projection (Dimensionality Reduction)

$$Y = ZW$$

$$Y \in \mathbb{R}^{n \times k}$$

This gives the representation of data in the new principal component space.

Key Notes

- Principal components are orthogonal.
- First PC captures maximum variance.
- Total variance in data = sum of eigenvalues.
- Fraction of variance captured by i-th PC: $\lambda_i / \sum_j \lambda_j$

Example 9:

Given Data

$$X = \begin{bmatrix} 2 & 0 \\ 0 & 1 \\ 3 & 1 \\ 4 & 3 \end{bmatrix} \in \mathbb{R}^{4 \times 2}$$

Here, $n = 4$ samples and $d = 2$ features.

—

Step 1: Compute Mean and Standard Deviation

Mean of each feature:

$$\mu_1 = \frac{2+0+3+4}{4} = 2.25, \quad \mu_2 = \frac{0+1+1+3}{4} = 1.25$$

Standard deviation:

$$\sigma_1 = \sqrt{\frac{1}{4}\left[(2-2.25)^2 + (0-2.25)^2 + (3-2.25)^2 + (4-2.25)^2\right]} = 1.479$$

$$\sigma_2 = \sqrt{\frac{1}{4}\left[(0-1.25)^2 + (1-1.25)^2 + (1-1.25)^2 + (3-1.25)^2\right]} = 1.090$$

—

Step 2: Standardization

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

$$Z = \begin{bmatrix} -0.17 & -1.15 \\ -1.52 & -0.23 \\ 0.51 & -0.23 \\ 1.18 & 1.61 \end{bmatrix} \in \mathbb{R}^{4\times 2}$$

—

Step 3: Covariance Matrix

$$C = \frac{1}{n} Z^T Z$$

$$C = \begin{bmatrix} 1.00 & 0.69 \\ 0.69 & 1.00 \end{bmatrix}$$

—

Step 4: Eigenvalues and Eigenvectors

$$|C - \lambda I| = 0$$

$$\lambda_1 = 1.69, \quad \lambda_2 = 0.31$$

Corresponding eigenvectors:

$$v_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad v_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

—

Step 5: Selection of Principal Component

$$\frac{\lambda_1}{\lambda_1 + \lambda_2} = \frac{1.69}{2} = 84.5\%$$

Select $k = 1$ principal component.

$$W = v_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

—

Step 6: Projection

$$Y = ZW$$

$$Y = \begin{bmatrix} -0.93 \\ -1.23 \\ 0.20 \\ 1.96 \end{bmatrix} \in \mathbb{R}^{4\times 1}$$

—

Final Result

The original 2D data is reduced to 1D while preserving maximum variance.

Example 10:

Given Data

$$X = \begin{bmatrix} 2 & 2 & 3 \\ 3 & 3 & 4 \\ 4 & 4 & 5 \\ 5 & 5 & 6 \end{bmatrix} \in \mathbb{R}^{4\times 3}$$

The three features X_1, X_2, X_3 are highly correlated.

—

Step 1: Standardization (Mean Centering)

Compute the mean of each feature:

$$\mu = \begin{bmatrix} 3.5 & 3.5 & 4.5 \end{bmatrix}$$

Centered data matrix:

$$Z = X - \mu = \begin{bmatrix} -1.5 & -1.5 & -1.5 \\ -0.5 & -0.5 & -0.5 \\ 0.5 & 0.5 & 0.5 \\ 1.5 & 1.5 & 1.5 \end{bmatrix}$$

—

Step 2: Covariance Matrix

$$C = \frac{1}{n} Z^T Z$$

Since all features move together, the covariance matrix has large off-diagonal entries:

$$C = \begin{bmatrix} \sigma^2 & \sigma^2 & \sigma^2 \\ \sigma^2 & \sigma^2 & \sigma^2 \\ \sigma^2 & \sigma^2 & \sigma^2 \end{bmatrix}$$

This indicates strong positive correlation among X_1, X_2, X_3.

—

Step 3: Eigenvalue Decomposition

Solve:

$$Cv = \lambda v$$

The dominant eigenvector is:

$$v_1 = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

This direction captures maximum variance.

Remaining eigenvectors are orthogonal to v_1 and correspond to very small eigenvalues, capturing negligible variance.

—

Interpretation

$$\lambda_1 \gg \lambda_2, \lambda_3$$

Most of the variance lies along the direction $(1, 1, 1)$.

Therefore, the original 3D data can be projected onto a single principal component:

$$Y = Zv_1$$

This reduces the data from $\mathbb{R}^3$ to $\mathbb{R}^1$ while preserving most of the information.

—

Conclusion

PCA effectively reduces the dimensionality from 3 to 1 due to high correlation among features.

3.2 Problems

Problem 273 *Let X be a dataset of size 100×5 and Σ be its covariance matrix. If one eigenvalue of Σ is zero, it implies:*

(A) There is redundancy among features

(B) Covariance matrix is singular

(C) *All features are independent*

(D) *PCA cannot be applied*

Problem 274 *For a 3D dataset with features X_1, X_2, X_3, PCA finds eigenvalues $\lambda_1 = 4$, $\lambda_2 = 1$, $\lambda_3 = 0.5$. What fraction of variance is captured by first two PCs?*

(A) *4/5*

(B) *5/5*

(C) *4/5.5*

(D) *5/5.5*

Problem 275 *Which of the following statements about PCA is/are correct? (Multiple may be correct)*

(A) *PCA maximizes variance along new axes*

(B) *PCA ensures class separation*

(C) *PCA axes are orthogonal*

(D) *PCA can reduce dimensionality with minimal loss of information*

Problem 276 *A dataset has highly correlated features. Applying PCA will:*

(A) *Remove correlation between features*

(B) *Increase dimensionality*

(C) *Reduce noise*

(D) *Leave features unchanged*

Problem 277 *Consider the standardized dataset X and covariance matrix Σ. The principal component direction v satisfies:*

(A) $\Sigma v = \lambda v$

(B) $X^T X v = 0$

(C) $\Sigma v = 0$

(D) $Xv = \lambda v$

Problem 278 *Which of the following is true for PCA? (Multiple answers possible)*

(A) *PCA is sensitive to scaling of features*

(B) *PCA works only on numerical features*

(C) *PCA maximizes class separation*

(D) *PCA projections minimize reconstruction error*

Problem 279 *Given a 2D dataset, the covariance matrix is:*

$$\Sigma = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$$

The eigenvalues are:

(A) 4, 0

(B) 2, 2

(C) 0, 0

(D) 1, 3

Problem 280 *A dataset has zero mean and covariance matrix with eigenvectors v_1, v_2, v_3. PCA projection onto top 2 eigenvectors gives:*

(A) Dimensionality reduced to 2

(B) Original variance preserved completely

(C) Projection is along correlated axes

(D) Cannot be done if eigenvalues are equal

Problem 281 *Which statements are correct about eigenvectors in PCA? (Multiple answers)*

(A) Orthogonal

(B) Magnitude always 1

(C) Represent directions of maximum variance

(D) Only first eigenvector matters

Problem 282 *Which preprocessing step is necessary before PCA on heterogeneous features?*

(A) Standardization

(B) Sampling

(C) Mean subtraction

(D) None of the above

Problem 283 *A 3D dataset projected onto 2 PCs. Which is true? (Multiple answers)*

(A) Variance along projected axes <= original variance

(B) Reconstruction possible without loss

(C) Dimensionality reduced

(D) PCs are orthogonal

Problem 284 *A dataset of 2D points along line $y = 2x$. PCA gives first PC direction as:*

(A) $[1, 2]^T/\sqrt{5}$

(B) $[2, 1]^T/\sqrt{5}$

(C) $[1, -2]^T/\sqrt{5}$

(D) $[-1, 2]^T/\sqrt{5}$

Problem 285 *Which statements about PCA are false? (Multiple answers)*

(A) PCA assumes linearity

(B) PCA removes mean

(C) PCA maximizes class separation

(D) PCA eigenvectors are orthogonal

Problem 286 *Which statements about PCA eigenvalues and eigenvectors are true? (Multiple answers)*

(A) Eigenvectors give directions of PCs

(B) Sum of eigenvalues = total variance

(C) Eigenvectors are unit vectors

(D) First PC always along X_1 axis

Problem 287 *PCA applied on 4×3 dataset. Sum of eigenvalues = 10. Largest eigenvalue = 6. Variance captured by first PC (%) =* _____

Problem 288 *Given the two-dimensional data points*

$$(5, 15), (15, 5)$$

perform ***Principal Component Analysis (PCA)*** *and determine the* ***projection of the point*** $(5, 15)$ ***onto the first principal component****.*

3.3 Try it Yourself

Exercise 23 *Consider a 2D dataset with points $(1, 2), (2, 3), (3, 4), (4, 5)$. The first principal component (PC1) is along the line $y = x$. Which of the following statements is correct?*

(A) PC1 captures all the variance

(B) PC2 captures zero variance

(C) PC1 and PC2 are not orthogonal

(D) PC2 captures more variance than PC1

Exercise 24 *A 2D dataset has covariance matrix* $\Sigma = \begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix}$. *Compute the largest eigenvalue.* _____

Exercise 25 *For the above dataset, project the point* $(2, 3)$ *onto the first principal component.* _____

Exercise 26 *A dataset has eigenvalues* $5, 2, 1$. *Compute the fraction of total variance captured by the first PC.* _____

Exercise 27 *Standardized 3D data has covariance matrix* $\begin{bmatrix} 1 & 0.8 & 0.6 \\ 0.8 & 1 & 0.7 \\ 0.6 & 0.7 & 1 \end{bmatrix}$. *Compute the largest eigenvalue.* _____

Exercise 28 *2D points* $(1, 2), (2, 1), (3, 4), (4, 3)$ *projected onto PC1 give coordinate of first point =* _____

Exercise 29 *PCA applied on a* 4×3 *dataset. Sum of eigenvalues = 10. Largest eigenvalue = 6. Variance captured by first PC (%) =* _____

Exercise 30 *A 2D dataset* $(1, 1), (2, 2), (3, 3)$ *is projected onto the first PC. Projected coordinate of second point =* _____

Exercise 31 *Covariance matrix* $\begin{bmatrix} 4 & 2 \\ 2 & 3 \end{bmatrix}$. *Compute second eigenvalue.* _____

Exercise 32 *3D points* $(1, 0, 0), (0, 1, 0), (0, 0, 1)$. *Fraction of variance along first PC =* _____

Exercise 33 *Dataset with covariance matrix* $\begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}$. *Compute eigenvalues.* _____

Exercise 34 *2D points* $(0, 0), (1, 2), (2, 1)$. *PCA projection onto first PC for first point =* _____

Exercise 35 *Total variance of a dataset = 10, first PC eigenvalue = 6. Percentage variance retained =* _____

3.4 YouTube Links and QR Codes

Lecture	Details	YouTube Link	QR Code
44	Principal Component Analysis (PCA) Explained from Scratch	https://youtu.be/G0eHyFDm5ks	
45	Principal Component Analysis Solutions to Problems 273-288	https://youtu.be/y9k72lJBHcg	

Chapter 4

Solutions

Problems Covered	YouTube Link	QR Code
Solutions to Problems 251–262 (Lecture 41)	https://youtu.be/U9b8OZ1JXy0	
Solutions to Problems 263–272(Lecture 43)	https://youtu.be/ASFGi6uEwmI	
Solutions to Problems 273–288 (Lecture 45)	https://youtu.be/y9k72lJBHcg	

Bibliography

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006. Available at: https://link.springer.com/book/9780387310732

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016. Available at: https://www.deeplearningbook.org

[3] T. M. Mitchell, *Machine Learning*, McGraw-Hill, 1997. Available at: https://www.cs.cmu.edu/~tom/mlbook.html

[4] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed., O'Reilly Media, 2019. Available at: https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/

[5] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., Springer, 2009. Available at: https://web.stanford.edu/~hastie/ElemStatLearn/

GateXAIML

Free GATE resources for Data Science & AI and CSE

Website: www.gatexaiml.in

Email: contact@gatexaiml.in