Principles of Machine Learning

Lecture 9: Clustering and Mixture Models

Sharif University of Technology
Dept. of Aerospace Engineering

April 29, 2025

Table of Contents

1. Introduction
2. Clustering Methods
3. Mixtures of Gaussians
4. General Expectation-Maximization Algorithm
   - Gaussian Mixtures
   - Relations to K-means
   - Mixtures of Bernoulli Distributions
5. Evidence Lower Bound
   - EM Revisited
   - Independent and Identically Distributed Data
   - Parameter Priors
   - Generalized EM
   - Sequential EM
6. Summary

Table of Contents

1. Introduction
2. Clustering Methods
3. Mixtures of Gaussians
4. General Expectation-Maximization Algorithm
   - Gaussian Mixtures
   - Relations to K-means
   - Mixtures of Bernoulli Distributions
5. Evidence Lower Bound
   - EM Revisited
   - Independent and Identically Distributed Data
   - Parameter Priors
   - Generalized EM
   - Sequential EM
6. Summary

Introduction: A Natural Instinct

Motivation: Imagine hiking in the mountains and spotting a new plant. You notice others nearby—not identical, but similar enough to group as the same species. This instinct to identify clusters of similar objects is what clustering algorithms replicate in machine learning—no expert needed!

- **Clustering**: An unsupervised task that groups similar instances into clusters without labels.
- **Goal**: Maximize intra-cluster similarity, Minimize similarity between clusters (inter-cluster similarity).

Clustering vs. Classification

Classification (Supervised): Uses labeled data (e.g., iris species) with algorithms like logistic regression or SVMs.
Clustering (Unsupervised): Works with unlabeled data, finding natural groups (e.g., K-means, GMM).

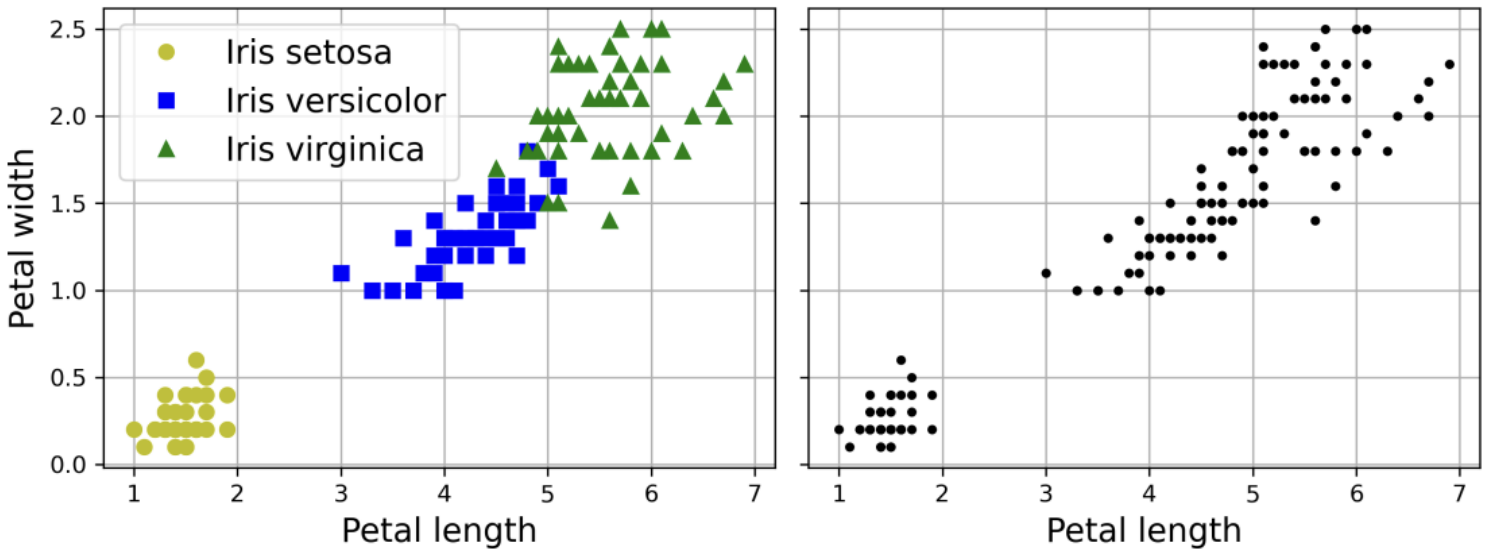


Figure: Left: Labeled iris dataset (classification). Right: Unlabeled iris dataset (clustering).

Why Clustering? Real-World Applications

Applications:

- **Customer Segmentation**: Group customers by behavior for tailored marketing.
- **Data Analysis**: Break down datasets by clusters for deeper insights.
- **Dimensionality Reduction**: Use cluster affinities as new features.
- **Anomaly Detection**: Spot outliers with low cluster affinity.
- **Search Engines**: Find similar images by clustering.
- **Image Segmentation**: Simplify images by clustering pixel colors.

What's Ahead: Tools and Techniques

Preparation: This lecture equips you with key methods:

- **K-means**: Fast, hard clustering for spherical groups.
- **Hierarchical Clustering**: Nested clusters via dendrograms.
- **DBSCAN**: Density-based, handles any shape, finds outliers.
- **GMM**: Probabilistic clustering with soft assignments.
- **EM Algorithm**: A framework to optimize these models.

Goal: Understand their mechanics and when to apply them.

Table of Contents

1. Introduction
2. Clustering Methods
3. Mixtures of Gaussians
4. General Expectation-Maximization Algorithm
   - Gaussian Mixtures
   - Relations to K-means
   - Mixtures of Bernoulli Distributions
5. Evidence Lower Bound
   - EM Revisited
   - Independent and Identically Distributed Data
   - Parameter Priors
   - Generalized EM
   - Sequential EM
6. Summary

K-means Clustering: Concept and Notations

Concept: K-means partitions a dataset of N data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, each a D-dimensional vector, into K clusters. A cluster is a group where points are closer to each other than to points outside it. We minimize the total squared distance from points to their cluster centers.

Notations:

- $\mathbf{x}_n$: n-th data point, a D-dimensional vector.
- μ_k: Center (prototype) of cluster k, a D-dimensional vector, where $k = 1, \ldots, K$.
- $r_{nk} \in \{0, 1\}$: Binary indicator; $r_{nk} = 1$ if $\mathbf{x}_n$ is in cluster k, else 0 (1-of-K coding).

K-means: Cost Function Formulation

Cost Function: Measures total squared distance from each point to its cluster center:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \mu_k\|^2$$

Here, $\|\mathbf{x}_n - \mu_k\|^2$ is the squared Euclidean distance. Goal: Minimize J.
Cluster Assignment r_{nk}: Assign each point to the nearest center:

$$r_{nk} = \begin{cases} 1, & \text{if } k = \arg\min_j \|\mathbf{x}_n - \mu_j\|^2, \\ 0, & \text{otherwise.} \end{cases}$$

This means $\mathbf{x}_n$ joins cluster k if μ_k is the closest center.

K-means: EM Concept and Optimization

EM Concept: K-means uses an iterative process like Expectation-Maximization (EM):

- **E-step**: Fix μ_k, update r_{nk} (assign points).
- **M-step**: Fix r_{nk}, update μ_k (recompute centers).

Optimization: Minimize J by alternating:

1. Start with initial μ_k.
2. E-step: Assign points using r_{nk}.
3. M-step: Update μ_k (see next slide).
4. Repeat until J stops decreasing significantly.

K-means: Cluster Center Update

Derived by setting $\frac{\partial J}{\partial \mu_k} = 0$, solving:

$$2 \sum_{n=1}^{N} r_{nk}(\mathbf{x}_n - \mu_k) = 0$$

In the M-step, set μ_k to the mean of points in cluster k:

$$\mu_k = \frac{\sum_{n=1}^{N} r_{nk} \mathbf{x}_n}{\sum_{n=1}^{N} r_{nk}}$$

- Numerator: Sum of points assigned to cluster k.
- Denominator: Number of points in cluster k.
- Intuition: μ_k is the average position of all points where $r_{nk} = 1$.

K-means: EM Algorithm

Algorithm 1: K-means EM Algorithm

Input : Data $\{\mathbf{x}_n\}_{n=1}^N$, initial centers $\{\mu_k\}_{k=1}^K$
Output: Final centers $\{\mu_k\}$ and assignments $\{r_{nk}\}$
for $n \leftarrow 1$ **to** N **do**
  $r_{n1} \leftarrow \cdots \leftarrow r_{nK} \leftarrow 0$
repeat
  E-step: Update assignments
  for $n \leftarrow 1$ **to** N **do**
    $k \leftarrow \arg\min_j \|\mathbf{x}_n - \mu_j\|^2$
    $r_{n1}, \ldots, r_{nK} \leftarrow 0$
    $r_{nk} \leftarrow 1$
  M-step: Update centers
  for $k \leftarrow 1$ **to** K **do**
    $\mu_k \leftarrow \dfrac{\sum_{n=1}^N r_{nk}\,\mathbf{x}_n}{\sum_{n=1}^N r_{nk}}$
until *no assignments change*
return $\{\mu_k\}, \{r_{nk}\}$

K-means Example: Old Faithful (Explanation)

Data: 272 standardized eruption records (duration vs waiting time).

K-means Example: Old Faithful (Explanation)

Data: 272 standardized eruption records (duration vs waiting time).
Algorithm Steps on Old Faithful Data (K=2):

- **Initialization**: Centers μ_1, μ_2 placed arbitrarily (Fig 1a).
- **E-step (Assignments)**:
  - Points colored by nearest center (Fig 1b, d, f, h).
  - Decision boundary = perpendicular bisector between centers (magenta line).
- **M-step (Update Centers)**:
  - Centers move to mean of assigned points (Fig 1c, e, g, i).
- **Convergence**: By iteration (i), assignments stabilize (no further changes).
- Cost function J decreases monotonically (Fig 15.2 in text).

Key Observations:

- Hard assignments (1 cluster per point).
- Sensitive to initialization.

K-means Visualization: Old Faithful Iterations

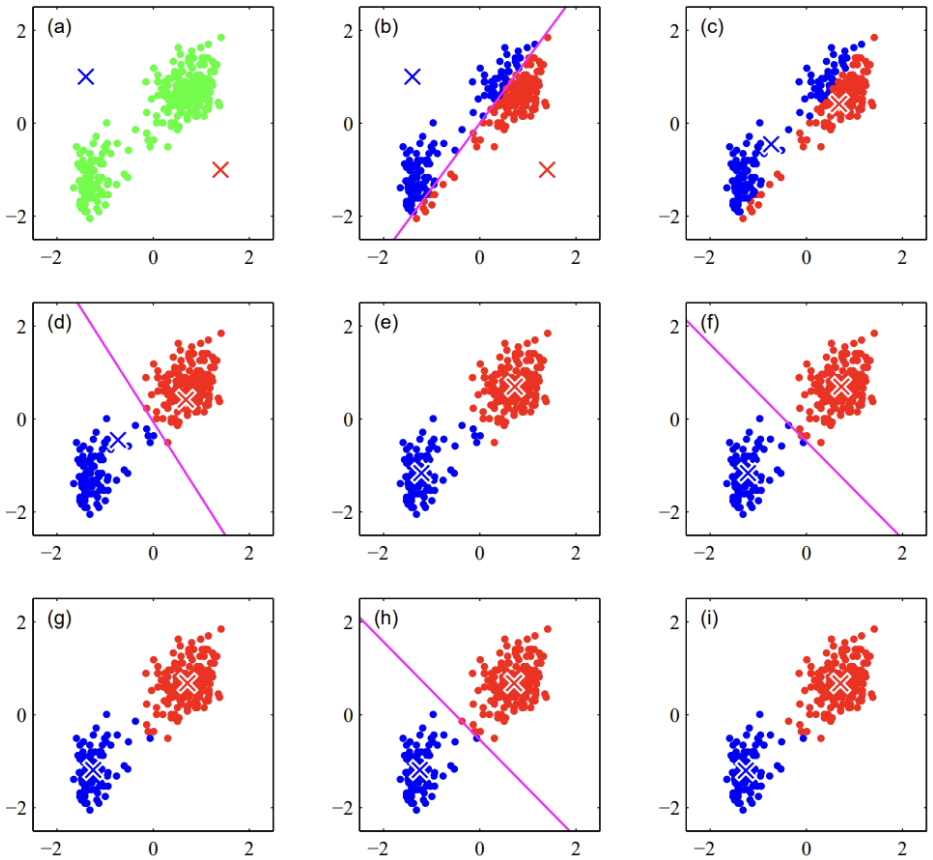


Figure: **Iterations (a)–(i)**: E-step (color assignments) and M-step (center updates). Final convergence at (i).

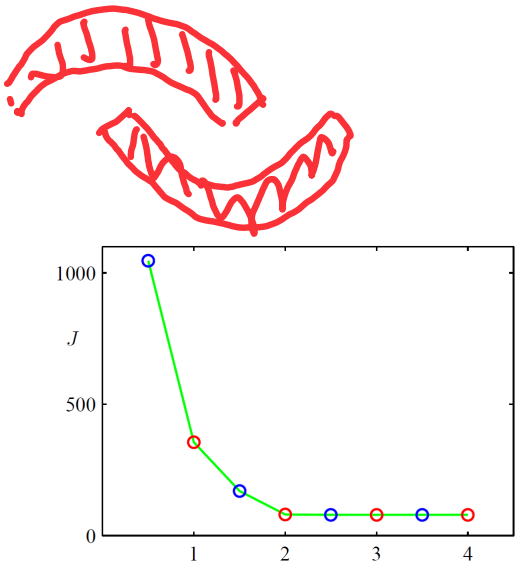


Figure: cost function after each E step (blue points) and M step (red points)

K-means: Sequential Update

Sequential Update: Instead of updating all cluster centers simultaneously, we can adjust them one data point at a time, which is practical for large datasets or streaming data where storing everything isn't feasible. This leads to:

$$\mu_k^{\text{new}} = \mu_k^{\text{old}} + \frac{1}{N_k}(\mathbf{x}_n - \mu_k^{\text{old}})$$

- N_k: Number of points in cluster k so far.
- Process: For each $\mathbf{x}_n$, assign to nearest μ_k, update μ_k.
- Benefit: Online learning, no need to store all data.

K-means Application: Image Segmentation

Image Segmentation: Cluster pixels by RGB values (3D points).

- Run K-means with K clusters.
- Replace each pixel's color with its μ_k.

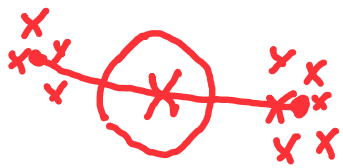
Useful for compression (vector quantization) or stylization.

Figure: Image segmentation using K-means.

Hierarchical Clustering: Concepts

Concept: Hierarchical Clustering builds a hierarchy of clusters without needing to specify the number of clusters beforehand. It produces a dendrogram, a tree-like structure showing how data points are grouped based on similarity.

- **Advantages**:
  - No need to predefine the number of clusters.
  - Provides a visual representation via the dendrogram.
- **Limitations**:
  - Assumes nested clusters, which may not always apply.
  - Computationally intensive for large datasets.



Hierarchical Clustering: Algorithm

Algorithm 2: Hierarchical Clustering Algorithm

Input : Dataset of n observations, dissimilarity measure
Output: Dendrogram
Step 1: Start with n clusters, each containing one observation.
Step 2: For $i = n$ downto 2:
a. Find the two most similar clusters based on linkage.
b. Merge them into a single cluster.
c. Update dissimilarities with the new cluster.
Step 3: Record merges to form the dendrogram.

Note: Similarity depends on the chosen linkage method and dissimilarity measure (e.g., Euclidean distance).

Hierarchical Clustering: Dendrogram Interpretation

Dendrogram: A tree-like diagram illustrating cluster merges.

- **Leaves**: Individual data points.
- **Branches**: Clusters; height shows dissimilarity (vertical distance).
- **Key**: Similarity is vertical, not horizontal proximity.

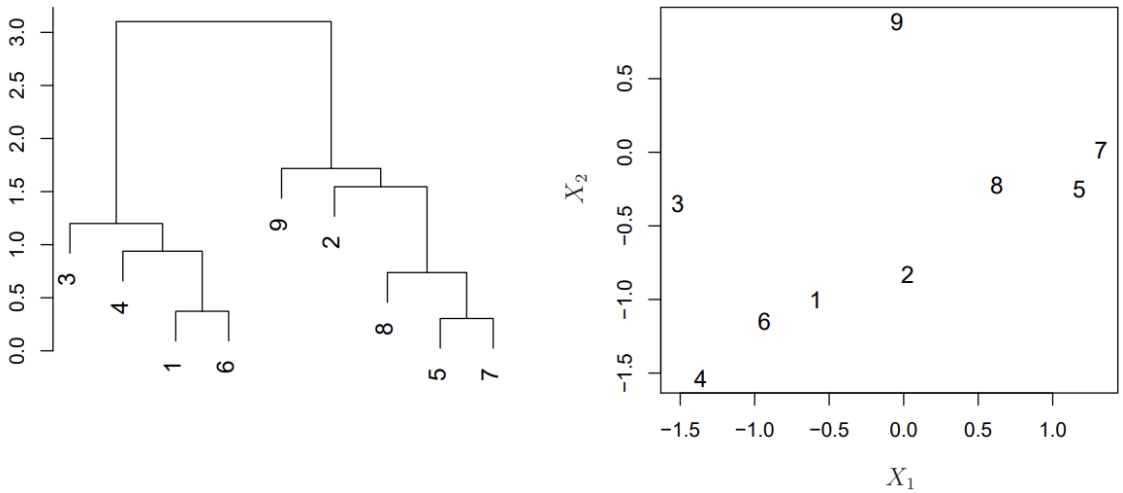


Figure: Dendrogram of nine observations. Low merges (e.g., 5 and 7) indicate high similarity.

Hierarchical Clustering: Figures - Obtaining Clusters

Method: Cut the dendrogram horizontally to form clusters.

- **Cut Height**: Higher cuts yield fewer, larger clusters; lower cuts yield more, smaller ones.
- **Example**: Cut at height 9 gives 2 clusters; at height 5 gives 3.

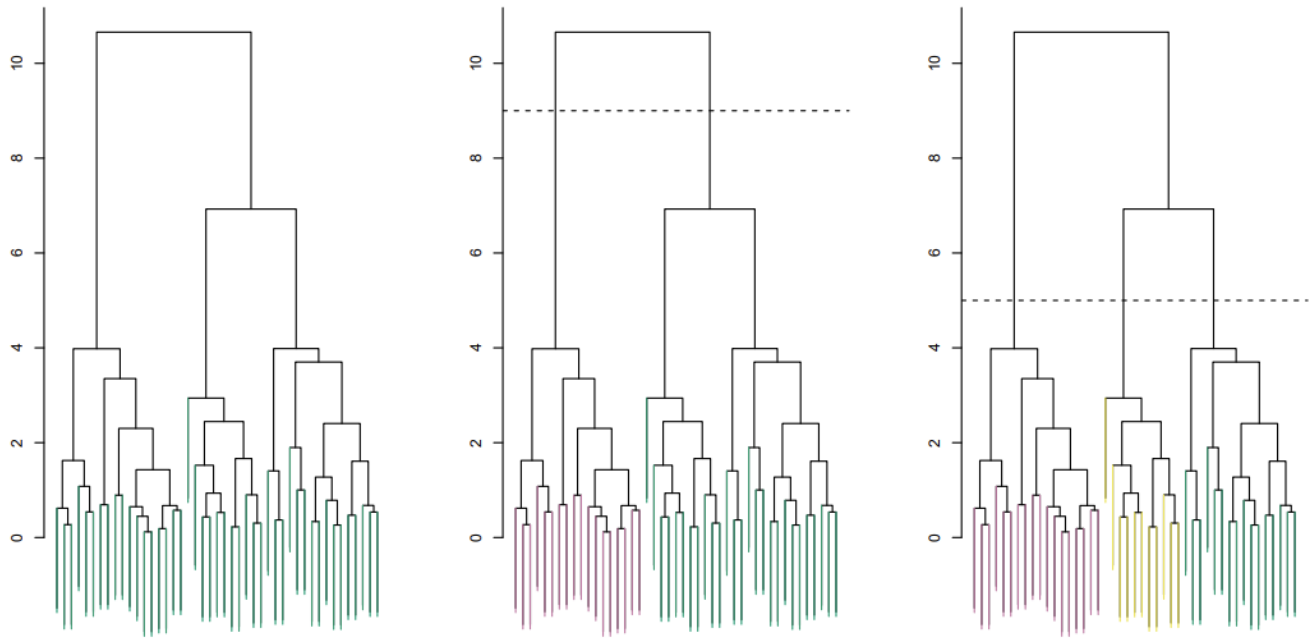


Figure: Dendrogram cuts: Original (left), cut at 9 (center, 2 clusters), cut at 5 (right, 3 clusters).