

UNIT 4

Unsupervised Learning and Dimensionality Reduction

(10 Hours) — Machine Learning (CACS456) · BCA IV/VIII

⚠ Please Note

These notes are prepared for quick study and **may contain errors or omissions**. Please cross-check formulas, definitions and worked examples with your textbook, class notes and teacher before relying on them in exams.

4.1 Introduction to Clustering

Clustering is an unsupervised task: given unlabelled data, group the instances so that objects in the same cluster are *similar* to each other and *different* from those in other clusters. There is no target variable, so the algorithm must find structure by itself.

- **Applications:** customer segmentation, document / news grouping, image segmentation, anomaly detection, gene-expression analysis, social-network communities.
- **Types:** *partitional* (K-Means), *hierarchical* (agglomerative / divisive), *density-based* (DBSCAN), *model-based* (Gaussian mixtures).
- **Hard vs soft:** hard assigns each point to one cluster; soft gives membership probabilities.

4.2 K-Means Clustering

K-Means partitions N points into K clusters. Each cluster is represented by its **centroid** (mean of its points). It minimises the **within-cluster sum of squares (WCSS / inertia)**:

K-Means Objective

$$J = \sum_k \sum_{\mathbf{x} \in C_k} \|\mathbf{x} - \boldsymbol{\mu}_k\|^2 \quad (\mu_k = \text{centroid of cluster } C_k)$$

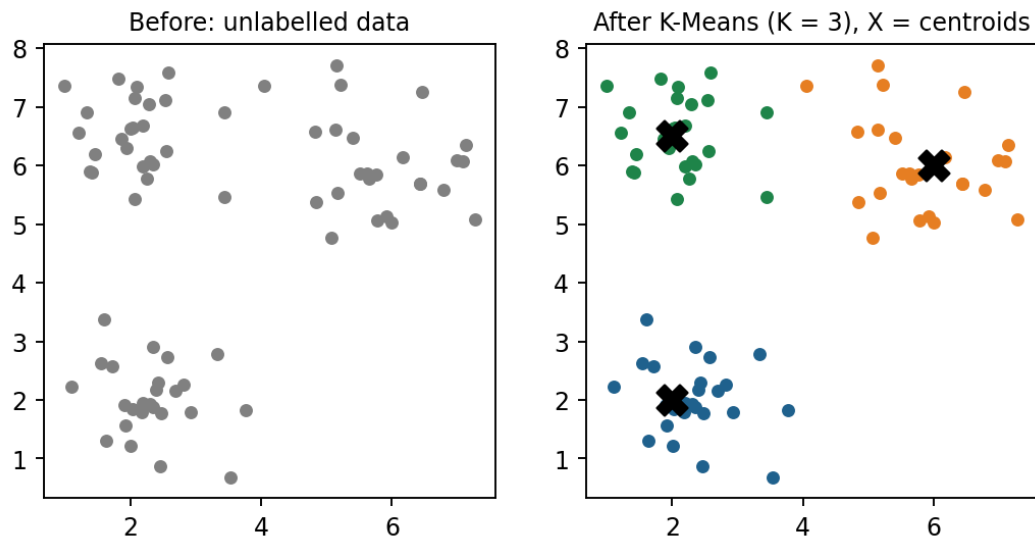


Figure 4.1: K-Means groups unlabelled data into $K = 3$ clusters around centroids.

Algorithm

- 1. Choose K and initialise K centroids (random points or k-means++).
- 2. **Assignment step:** assign every point to its nearest centroid.
- 3. **Update step:** recompute each centroid as the mean of its assigned points.
- 4. Repeat steps 2–3 until assignments stop changing (convergence) or max iterations reached.

Example 4.1

Cluster the 1-D points $\{2, 4, 10, 12, 3, 20, 30, 11, 25\}$ into $K = 2$ with initial centroids $\mu_1 = 2$ and $\mu_2 = 4$.

Solution

Iteration 1: $C_1 = \{2, 3\}$ (3 is equidistant from both centroids; ties go to C_1), $C_2 = \{4, 10, 11, 12, 20, 25, 30\}$.

New centroids: $\mu_1 = 2.5$, $\mu_2 = (4+10+12+20+30+11+25)/7 = 112/7 = 16$.

Iteration 2: $C_1 = \{2, 3, 4\}$, $C_2 = \{10, 12, 20, 30, 11, 25\}$. $\mu_1 = 3$, $\mu_2 = 108/6 = 18$.

Iteration 3: $C_1 = \{2, 3, 4, 10\}$, $C_2 = \{12, 20, 30, 11, 25\}$. $\mu_1 = 4.75$, $\mu_2 = 98/5 = 19.6$.

Iteration 4: $C_1 = \{2, 3, 4, 10, 11, 12\}$, $C_2 = \{20, 25, 30\}$. $\mu_1 = 7$, $\mu_2 = 25$.

Iteration 5: assignments unchanged $\rightarrow$ **converged:** $C_1 = \{2, 3, 4, 10, 11, 12\}$, $C_2 = \{20, 25, 30\}$.

Choosing K and Limitations

- **Elbow method:** plot WCSS vs K and choose the “elbow” where improvement slows.
Silhouette score: measures cohesion vs separation (−1 to 1; higher is better).
- **Pros:** simple, fast ($O(N \cdot K \cdot t)$), scales to large data.
- **Cons:** K must be specified; sensitive to initial centroids (run several times or use `k-means++`); sensitive to outliers; assumes spherical, similar-sized clusters; only numeric features (K-Medoids / K-Modes alternatives).

4.3 Different Distance Functions for Clustering

Clustering relies on a notion of similarity, so the choice of distance function strongly affects results.

Distance	Formula	Use / Notes
Euclidean	$\sqrt{\sum (x_i - y_i)^2}$	Default for continuous data; spherical clusters
Manhattan (City-block)	$\sum x_i - y_i $	Less sensitive to outliers; grid-like data
Minkowski	$(\sum x_i - y_i ^p)^{1/p}$	Generalisation: $p = 1$ Manhattan, $p = 2$ Euclidean
Chebyshev	$\max x_i - y_i $	Largest coordinate difference
Cosine similarity	$x \cdot y / (\ x\ \ y\)$	Text / high-dimensional sparse data; compares direction, not magnitude
Hamming	# positions that differ	Binary / categorical strings
Mahalanobis	$\sqrt{(x-y)^T S^{-1} (x-y)}$	Accounts for correlation and scale (S = covariance matrix)
Jaccard	$1 - A \cap B / A \cup B $	Sets, binary attributes

Example 4.2

Find Euclidean and Manhattan distance between $P(1, 2, 3)$ and $Q(4, 6, 3)$.

Solution

Euclidean = $\sqrt{3^2 + 4^2 + 0^2} = \sqrt{25} = 5$

Manhattan = $3 + 4 + 0 = 7$

4.4 Hierarchical Clustering

Hierarchical clustering builds a tree of clusters (**dendrogram**) and does not need K in advance; K is chosen by cutting the dendrogram at a height.

- **Agglomerative (bottom-up)**: start with each point as its own cluster; repeatedly merge the two closest clusters until one remains. Most common.
- **Divisive (top-down)**: start with one cluster; recursively split.

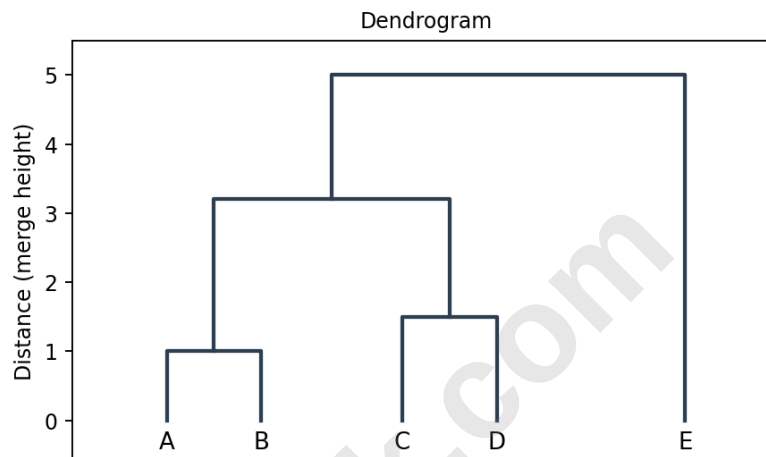


Figure 4.2: A dendrogram — the height of a join shows the distance at which clusters merged; cutting at a level gives the clusters.

Linkage Criteria (distance between clusters)

Linkage	Distance between clusters A and B	Behaviour
Single	Minimum distance between any two points	Can create long “chains”; handles non-spherical shapes
Complete	Maximum distance between any two points	Compact, similar-sized clusters
Average	Mean of all pairwise distances	Compromise between single and complete
Centroid / Ward	Distance of centroids / minimum increase in variance	Ward gives compact, balanced clusters

Example 4.3 — Single-linkage on 1-D points

Points: $A = 1$, $B = 2$, $C = 6$, $D = 8$, $E = 15$. Build the agglomerative clustering (single linkage).

Solution

Closest pairs: $d(A,B) = 1 \rightarrow$ merge $\{A,B\}$.

Next: $d(C,D) = 2 \rightarrow$ merge $\{C,D\}$.

Then: $d(\{A,B\},\{C,D\}) = \min = |2 - 6| = 4 \rightarrow$ merge $\{A,B,C,D\}$.

Finally: $d(\{A,B,C,D\}, E) = |8 - 15| = 7 \rightarrow$ merge all. Merge heights: 1, 2, 4, 7.

- **Pros:** no K needed, dendrogram gives rich view, works with any distance.
- **Cons:** $O(N^2)$ memory and $O(N^3)$ time (naïve), merges are irreversible, sensitive to noise.

4.5 Supervised Learning after Clustering

Clustering can be combined with supervised learning, because the discovered clusters carry useful information, especially when labels are scarce or expensive.

- **Cluster-then-label (semi-supervised):** cluster all data, label a few points per cluster, and propagate the majority label to the rest of the cluster; then train a classifier on the enlarged labelled set.
- **Cluster ID as a feature:** add the cluster number (or distance to each centroid) as new input features to a classifier / regressor.
- **One model per cluster:** train a separate supervised model for each cluster (e.g. separate regression for each customer segment) — often more accurate than one global model.
- **Pre-processing:** cluster centroids can serve as prototypes to reduce training-set size (e.g. for k -NN) or to initialise RBF networks.
- **Evaluation:** if true labels exist, compare clusters with them via purity or adjusted Rand index.

4.6 Dimensionality Reduction Techniques

Real datasets may have hundreds or thousands of features. **Dimensionality reduction** maps the data to fewer dimensions while preserving as much useful information as possible. It helps fight the **curse of dimensionality** (distances become meaningless, data becomes sparse), reduces overfitting, speeds up learning, saves storage and enables 2-D / 3-D visualisation.

Approach	Idea	Examples
Feature selection	Keep a subset of the original features	Filter (correlation, chi-square), wrapper (forward / backward selection), embedded (Lasso)

Approach	Idea	Examples
Feature extraction	Create new features as combinations of originals	PCA, LDA (supervised), SVD, ICA, Autoencoders
Manifold learning	Preserve local structure of non-linear data	t-SNE, Isomap, UMAP

4.7 Principal Component Analysis (PCA)

PCA is a linear, unsupervised technique that finds new orthogonal axes — **principal components** — along which the data variance is maximal. PC1 captures the most variance, PC2 the most of what remains (perpendicular to PC1), and so on. Keeping the first k components gives a k-dimensional representation.

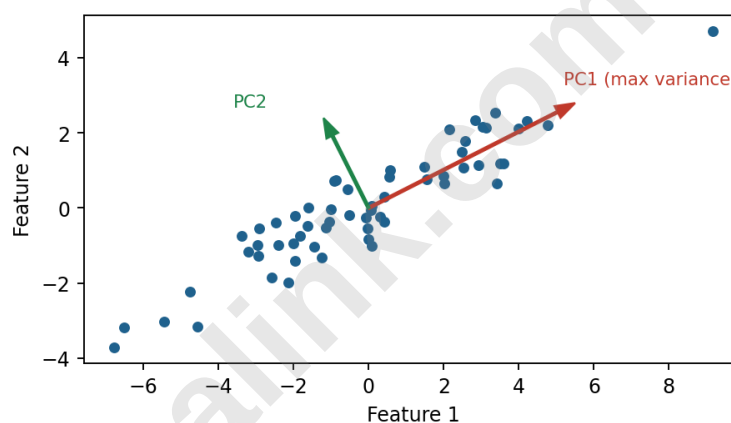


Figure 4.3: PC1 points along the direction of greatest spread; PC2 is orthogonal to it.

Algorithm

- 1. **Standardise** the data: subtract the mean of each feature (and divide by σ if scales differ).
- 2. Compute the **covariance matrix** $C = (1/(N-1)) X^T X$ ($d \times d$).
- 3. Compute the **eigenvalues** λ and **eigenvectors** $\mathbf{v}$ of C : $C \mathbf{v} = \lambda \mathbf{v}$.
- 4. Sort eigenvectors by decreasing eigenvalue; pick the top k as the projection matrix W ($d \times k$).
- 5. **Project**: $Z = X W$ ($N \times k$). Eigenvalue $\lambda_i = \text{variance along } PC_i$.

Explained Variance

Explained variance ratio of PC_i $= \lambda_i / \sum_j \lambda_j$

Choose k so that the cumulative ratio is $\geq 90\text{--}95\%$, or use the scree-plot elbow.

Example 4.4

Data (2 features): (2, 1), (3, 5), (4, 3), (5, 6), (6, 7), (7, 8). Find the covariance matrix and the first principal component direction.

Solution

Means: $\bar{x} = 27/6 = 4.5$, $\bar{y} = 30/6 = 5$.

Deviations (dx, dy): (-2.5, -4), (-1.5, 0), (-0.5, -2), (0.5, 1), (1.5, 2), (2.5, 3)

$S_{xx} = 6.25 + 2.25 + 0.25 + 0.25 + 2.25 + 6.25 = 17.5 \rightarrow \text{var}(x) = 17.5/5 = 3.5$

$S_{yy} = 16 + 0 + 4 + 1 + 4 + 9 = 34 \rightarrow \text{var}(y) = 6.8$

$S_{xy} = 10 + 0 + 1 + 0.5 + 3 + 7.5 = 22 \rightarrow \text{cov}(x, y) = 4.4$

Covariance matrix $C = [[3.5, 4.4], [4.4, 6.8]]$

Characteristic equation: $\lambda^2 - 10.3\lambda + (3.5 \times 6.8 - 4.4^2) = \lambda^2 - 10.3\lambda + 4.44 = 0 \rightarrow \lambda = (10.3 \pm \sqrt{(106.09 - 17.76)})/2 = (10.3 \pm 9.40)/2 \rightarrow \lambda_1 \approx 9.85, \lambda_2 \approx 0.45$

PC1 explains $9.85/10.3 \approx 95.6\%$ of the variance.

Eigenvector for λ_1 : $(C - \lambda_1 I)v = 0 \rightarrow (3.5 - 9.85)v_1 + 4.4v_2 = 0 \rightarrow v_2 = 1.44 v_1 \rightarrow \text{normalised } \mathbf{v}_1 \approx (0.57, 0.82)$.

- **Pros:** removes correlated features, reduces noise, speeds up learning, good for visualisation.
- **Cons:** linear only; components are hard to interpret; sensitive to feature scaling; assumes high variance = important information.
- **PCA vs LDA:** PCA is unsupervised (maximise variance); LDA is supervised (maximise class separability).

Key Takeaways — Unit 4

- ✓ Clustering groups unlabelled data; K-Means alternates assignment and centroid update to minimise WCSS.
- ✓ Choice of distance (Euclidean, Manhattan, cosine, etc.) shapes the clusters; scale features first.
- ✓ Hierarchical clustering yields a dendrogram; linkage (single, complete, average, Ward) defines cluster distance.
- ✓ Clusters can aid supervised learning via cluster-then-label, extra features or per-cluster models.
- ✓ Dimensionality reduction fights the curse of dimensionality: selection vs extraction.
- ✓ PCA: standardise $\rightarrow$ covariance $\rightarrow$ eigen-decomposition $\rightarrow$ project onto top-k eigenvectors.

— End of Unit 4 —

These notes may contain errors — please verify with your textbook and teacher.